М.Гуссенс, Ф.Миттельбах, А.Самарин

ПУТЕВОДИТЕЛЬ ПО ПАКЕТУ LATEX И ЕГО РАСШИРЕНИЮ LATEX2E:

Справочное руководство по пакету LATEX2e—современной версии LATEX'a, — ставшему в настоящее время стандартом de facto. Благодаря гибкости, простоте использования и профессиональному полиграфическому качеству, LATEX, разработанный на базе издательской системы TEX Дональда Кнута, широко применяется при подготовке изданий как по точным, так и по гуманитарным наукам. Авторам — известным специалистам в этой тематике (Швейцария, ФРГ, Швейцария) — удалось в полном объеме представить инструментарий LATEX'a: NFSS2, AMS-LATEX, epic, eepic, *MakeIndex* и BibTeX, а также богатую библиотеку пакетов (более 150) по плавающим объектам, графике, таблицам, языку PostScript и многоязыковой поддержке.

Книга предназначена для тех, кто хочет знать, как расширить возможности lATEX'a, чтобы уметь красиво оформлять издания: для профессиональных полиграфистов, авторов, разработчиков и программистов.

Предисловие редактора перевода 5
Предисловие 11
1 Введение 19
1.1 Краткая история TeX’a и LaTeX’a 19
1.1.1 Вначале был TeX 19
1.1.2 Потом Лесли Лэмпорт придумал LaTeX 20
1.1.3 С LaTeX’ом в 2000 год? 21
1.2 LaTeX и его составляющие 21
1.2.1 Как работает LaTeX? 21
1.2.2 Выходные процессоры (драйверы dvi) 24
1.3 Концепция общей разметки 25
1.3.1 Что такое общая разметка? 25
1.3.2 Преимущества общей разметки 26
1.3.3 Разделение содержания и формы 27
1.4 Необходимость локальной разметки 27
1.4.1 Недостатки локальной разметки 27
1.4.2 Когда использовать локальную разметку 28
2 Структура документа, подготовленного в LaTeX’e 29
2.1 Структура исходного файла 29
2.1.1 Обработка опций и пакетов 31
2.1.2 Разделение исходного файла на части 34
2.1.3 Комбинирование нескольких файлов 35
2.2 Логическая структура 35
2.3 Команды секционирования 36
2.3.1 Нумерация заголовков 38
2.3.2 Форматирование заголовков 42
2.3.3 Изменение стандартных заголовков 48
2.4 Структура оглавления 49
2.4.1 Набор оглавления 50

2.4.2 Ввод информации в файлы оглавления 53
2.4.3 Определение нового файла, аналогичного .toc 54
2.4.4 Сложные оглавления 55
2.5 Управление ссылками 58
2.5.1 varioref — более гибкие ссылки 60
2.5.2 Ссылки на внешние документы 64
3 Основные средства форматирования 65
3.1 Фразы и абзацы 66
3.1.1 letterspace — изменение межбуквенных интервалов 66
3.1.2 ulem — выделение посредством подчеркивания 67
3.1.3 xspace — гибкий пробел после макро 68
3.1.4 Выравнивание внутри абзаца 69
3.1.5 doublespace —изменение интерлиньяжа 70
3.1.6 picinpar — набор абзацев с прямоугольными окнами 71
3.1.7 shapepar — набор абзацев необычной формы 72
3.2 Структуры перечня 74
3.2.1 Модификация стандартных перечней 74
3.2.2 Создание собственных перечней 78
3.3 Подражание машинописному шрифту 84
3.3.1 alltt—окружение типа verbatim 84
3.3.2 verbatim — стиль для литературного текста 84
3.3.3 moreverb—дополнительные окружения типа verbatim 85
3.4 Примечания: подстрочные, на полях, выносные 88
3.4.1 Создание сносок 88
3.4.2 Примечания на полях 92
3.4.3 Выносные примечания 93
3.5 Использование многоколонного набора 94
3.5.1 multicol — гибкий способ работы с многоколонным документом 94
3.5.2 Набор текста в колонках 95
3.5.3 Создание окружения multicols 96
3.5.4 Плавающие объекты и сноски в multicol 98
3.5.5 ftnright — сноски в правой колонке при двухколонном окружении 98
3.6 Простое управление версиями 100
4 Макет полосы набора 101
4.1 Геометрические параметры макета полосы набора 102
4.2 Изменение макета 105
4.2.1 Пакеты для создания макета полосы набора 107
4.2.2 Горизонтальное расположение полос набора при печати 109
4.3 Стили полосы 110
4.3.1 Написание новых стилей полосы 112
4.3.2 Создание стиля полосы при помощи fancyheadings 114
4.4 Явное форматирование 117
5 Таблицы 120
5.1 Сравнение окружений tabbing и tabular 121

5.2 Использование окружения tabbing 122
5.2.1 Окружение program 123
5.3 array — расширение окружений tabular 124
5.3.1 Примеры команд преамбулы 124
5.3.2 Стилевые параметры 129
5.3.3 Определение новых спецификаторов колонок 131
5.3.4 Некоторые особенности реализации пакета array 132
5.3.5 tabularx — автоматическое вычисление ширины колонок 133
5.3.6 delarray — определение вида ограничителей для окружения array 137
5.4 Многостраничные таблицы 138
5.4.1 supertab —верстка многостраничных таблиц 138
5.4.2 longtable —усложненные многостраничные таблицы 142
5.4.3 Завершающее сравнение окружений supertabular и longtable 147
5.5 Дополнительные штрихи 147
5.5.1 dcolumn — управление выравниванием в колонках таблицы 147
5.5.2 hhline — комбинирование горизонтальных и вертикальных от резков 151
5.6 Приложения 152
5.6.1 Переносы в узких колонках 152
5.6.2 Сноски в таблицах 153
5.6.3 Таблицы с широкими графами 154
5.6.4 Колонки, занимающие несколько строк таблицы 155
5.6.5 Таблицы внутри таблиц 157
5.6.6 Еще два примера 160
6 Плавающие объекты 162
6.1 Параметры плавающих объектов 162
6.2 Улучшенное размещение плавающих объектов 166
6.3 float — создание новых видов плавающих объектов 169
6.3.1 Разместить плавающий объект «здесь» ! 171
6.4 Другие виды плавающих окружений 173
6.4.1 floatfig — узкие плавающие рисунки «в оборку» 173
6.4.2 wrapfig — неплавающие рисунки «в оборку» 174
6.4.3 subfigure — рисунки внутри рисунков 176
6.4.4 endfloat — размещение рисунков и таблиц в конце документа 176
6.5 Создание своих названий 178
7 Переключение шрифтов 180
7.1 Введение в NFSS 180
7.2 Характеристики шрифтов 182
7.2.1 Моноширинные и пропорциональные шрифты 183
7.2.2 Шрифты с засечками и без засечек 184
7.2.3 Семейства шрифтов и их атрибуты 184
7.2.4 Схемы кодирования 189
7.3 Переключение шрифтов в тексте 190
7.3.1 Стандартные шрифтовые команды NFSS 191

7.3.2 Комбинирование стандартных команд управления шрифтами 196
7.3.3 Сравнение командного и декларативного способов переключения шрифтов 197
7.3.4 Доступ ко всем литерам шрифта 199
7.3.5 Изменение значений по умолчанию для атрибутов текстовых шрифтов 200
7.3.6 Шрифтовые команды LaTeX 2.09 202
7.4 Переключение шрифтов в формулах 202
7.4.1 Специальные идентификаторы математических алфавитов 203
7.4.2 Текстовые шрифтовые команды при наборе математических формул 206
7.4.3 Версии математических формул 207
7.5 Стандартные пакеты 208
7.5.1 Добавление новых текстовых шрифтов 209
7.5.2 Подключение новых математических шрифтов 212
7.5.3 slides — получение демонстрационных слайдов 214
7.5.4 Обработка ранее созданных документов 214
7.5.5 Специальные пакеты для NFSS 215
7.6 Низкоуровневый интерфейс 217
7.6.1 Установка индивидуальных шрифтовых атрибутов 218
7.6.2 Установка значений для нескольких шрифтовых атрибутов 223
7.6.3 Автоматические подстановки шрифтов 224
7.6.4 Использование низкоуровневых команд в документе 225
7.7 Подключение новых шрифтов 225
7.7.1 Общая схема 225
7.7.2 Объявление новых семейств шрифтов и групп начертаний шрифтов 226
7.7.3 Параметры управления загрузкой шрифтов 235
7.7.4 Ввод определений новых схем кодирования 235
7.7.5 Внутренняя организация файла 236
7.7.6 Объявление новых шрифтов для математических формул 238
7.7.7 Порядок записи деклараций 243
7.8 Предупреждения и сообщения об ошибках 244
8 Высшая математика 252
8.1 Создание AMS-LaTeX’a 252
8.2 Шрифты и символы в формулах 253
8.2.1 Команды для математических шрифтов 253
8.2.2 Математические символы 254
8.3 Составные символы, ограничители и операторы 260
8.3.1 Кратные интегралы 260
8.3.2 Стрелки сверху и снизу 260
8.3.3 Многоточия 261
8.3.4 Двойные акценты 261
8.3.5 Акценты как верхние индексы 262

8.3.6 Акценты в виде точек 262
8.3.7 Корни 262
8.3.8 Формулы в рамке 262
8.3.9 Растяжимые стрелки 263
8.3.10 Команды \overset, \underset и \sideset 263
8.3.11 Команда \smash 264
8.3.12 Команда \text 264
8.3.13 Названия новых операций 265
8.3.14 Команда \rnod и ее аналоги 266
8.3.15 Дроби и родственные конструкции 266
8.3.16 Непрерывные дроби 268
8.3.17 Ог-г-г-громные ограничители 268
8.4 Окружения типа матрицы и коммутативные диаграммы 269
8.4.1 Окружение cases 269
8.4.2 Окружения типа matrix 269
8.4.3 Команда \substack 271
8.4.4 Коммутативные диаграммы 271
8.5 Выравнивание многострочных формул 272
8.5.1 Несколько формул без выравнивания 273
8.5.2 Несколько формул с выравниванием 274
8.5.3 Разбитые на части формулы без выравнивания 275
8.5.4 Разбитые на части формулы с выравниванием 275
8.5.5 Окружения выравнивания для набора отдельных частей выключных формул 276
8.5.6 Вертикальные пробелы и разрывы страниц при наборе формул 277
8.5.7 Команда \intertext 277
8.6 Разное 278
8.6.1 Нумерация формул 278
8.6.2 Установка счетчика формул 279
8.6.3 Подчиненная нумерация формул 279
8.6.4 Тонкая настройка в математическом режиме 280
8.6.5 На что еще обратить внимание 280
8.6.6 Опции к пакету amsmath и отдельные его составляющие 281
8.6.7 Классы документов AMS-LaTeX’a 283
8.7 Примеры многострочных формул 283
8.7.1 Окружение split 283
8.7.2 Окружение multline 286
8.7.3 Окружение gather 287
8.7.4 Окружение align 287
8.7.5 Использование окружений align и split внутри gather 288
8.7.6 Использование окружений alignat 289
8.8 Расширения для окружения theorem 290
8.8.1 Как определять новые окружения типа теоремы 291
8.8.2 Примеры определений и использования теорем 293

8.8.3 Некоторые специальные вопросы 294
8.9 Параметры, задающие математические стили 294
8.9.1 Как управлять размерами символов 294
8.9.2 Параметры математических стилей в LaTeX’e 296
9 LaTeX в многоязычной среде 298
9.1 TEX и языки, отличные от английского 298
9.1.1 Механизм виртуального шрифта 301
9.2 Babel — LaTeXвладеет многими языками 302
9.2.1 Среда пользователя 302
9.2.2 Опция german 303
9.2.3 Структура языковых стилевых файлов системы babel 305
9.3 Следование типографским нормам 312
9.3.1 Традиционные французские типографские нормы 312
9.3.2 Команды пакета french 313
9.3.3 Структура пакета french 314
10 Графика в LaTeX’e 315
10.1 Орнаменты 317
10.1.1 Министраницы в рамке 317
10.1.2 Рамки с тенью 317
10.1.3 Декоративные рамки 318
10.2 Окружение picture 320
10.2.1 Аппроксимации Безье 320
10.2.2 Как совмещать несколько рамок 321
10.2.3 Как рисовать бинарные и тернарные деревья 322
10.2.4 Как рисовать гистограммы 324
10.2.5 Примеры окружения barenv 326
10.2.6 Как рисовать произвольные кривые 327
10.2.7 Другие пакеты 333
10.3 epic — усовершенствования к окружению picture 333
10.3.1 Описание команд 335
10.4 Расширение пакета epic 340
10.4.1 Расширения LaTeX’a при помощи eepic 341
10.4.2 Расширения пакета epic при помощи eepic 341
10.4.3 Новые команды в пакете eepic 342
10.4.4 Совместимость 343
10.4.5 Примеры 344
10.5 Пакеты, основанные на epic 344
10.5.1 Как рисовать двудольные графы 344
10.5.2 Как рисовать деревья 348
11 Как использовать PostScript **350**
11.1 Язык PostScript 350
11.1.1 Несколько слов о языке 350
11.1.2 Что такое инкапсулированный PostScript ? 352
11.2 dvips—преобразование dvi в PostScript 354

11.3 Совмещение текста и графики в формате PostScript 355
11.3.1 Простые рисунки 358
11.3.2 Черновые рисунки 359
11.3.3 Более сложная организация рисунков 359
11.4 Как повернуть материал 361
11.4.1 Как повернуть табличный материал 363
11.4.2 Как повернуть рисунок 364
11.4.3 Как повернуть только подпись к рисунку 365
11.5 Отчеркивания на полях 365
11.5.1 Среда пользователя 367
11.5.2 Параметры команд пакета changebar 369
11.5.3 Недостатки и неточности 369
11.6 Обрамление и затенение 370
11.7 Цветной вывод 371
11.8 Наложение текста на выводимую страницу 371
11.9 Еще одно обращение к NFSS 371
11.9.1 Как называются эти тысячи шрифтов 371
11.9.2 Система PSNFSS 372
11.9.3 Как использовать Pi-шрифты из PostScript'a 375
11.9.4 Общие команды в пакете pifont 376
11.9.5 Шрифт Symbol 377
11.9.6 Как самостоятельно установить новые PostScript-шрифты 378
11.9.7 Как заменить все TeX'овские шрифты PostScript-шрифтами 379
11.10 DCPS — корковская кодировка с PostScript-шрифтами 380
12 Создание указателей 385
12.1 Синтаксис описания элементов указателя 387
12.1.1 Простые описания элементов указателя 387
12.1.2 Формирование подчиненных элементов 388
12.1.3 Диапазоны страниц и перекрестные ссылки 389
12.1.4 Управление формой представления указателя 390
12.1.5 Печать специальных символов 391
12.1.6 Некоторые дополнительные замечания 391
12.1.7 Согласованность элементов указателя 393
12.2 Подготовка указателя 394
12.2.1 Формирование полуфабриката указателя 394
12.2.2 Получение форматированного указателя 394
12.3 Запуск программы MakeIndex 395
12.3.1 Опции программы MakeIndex 395
12.3.2 Сообщения об ошибках 398
12.4 Изменение вида указателя 400
12.4.1 Пример стилевых файлов указателя 400
12.4.2 Указатель, формируемый отдельно 404
12.4.3 Изменение специальных символов 404
12.4.4 Изменение выходного формата указателя 405

12.4.5 Обработка нетрадиционных номеров страниц 406
12.4.6 Глоссарий 408
12.5 Изменение макета указателя 409
12.5.1 Формирование нескольких указателей 410
12.5.2 Модификация команд формирования указателей 410
13 Создание списка литературы 416
13.1 Создание библиографических ссылок 417
13.1.1 Придание ссылкам требуемого вида 418
13.1.2 Выбор формата меток 420
13.2 Совместное использование BibTeX'a и LaTeX’a 420
13.2.1 Список стилевых файлов для BibTeX'a 422
13.2.2 Примеры BibTeX"овских стилей 424
13.3 Документы с несколькими списками литературы 431
13.3.1 Пакет chapterbib 431
13.3.2 Пакет bibunits 434
13.4 Средства управления библиографическими базами данных 438
13.5 Форматbib-файлов 443
13.5.1 Общая структура записей в BibTeX"овской базе данных 443
13.5.2 Текстовая часть поля 445
13.5.3 BibTeX и аббревиатуры 450
13.5.4 BibTeX'овская преамбула 451
13.5.5 Перекрестные ссылки 452
13.5.6 Дополнительные замечания 453
13.6 Формат записей базы данных (подробное описание) 453
13.7 Как устроены BibTeX'овские стили 458
13.7.1 Общее представление о BibTeX'овских стилевых файлах 458
13.7.2 Команды BibTeX'овского стилевого файла 459
13.7.3 Встроенные функции 459
13.7.4 Стилевой файл-документация btxbst.doc 459
13.8 Модификация стилевых файлов 463
13.8.1 Добавление нового поля 464
13.8.2 Поддержка языков, отличных от английского 466
13.9 Пакет makebst для модификации библиографических стилей 469
13.9.1 Работа с программой makebst 470
14 Средства документирования макропакетов 472
14.1 Макропакеты с документацией 472
14.2 Пользовательский интерфейс пакета doc 473
14.2.1 Основные понятия 473
14.2.2 Описание новых макрокоманд и окружений 474
14.2.3 Перекрестные ссылки между используемыми макрокомандами 475
14.2.4 Заключительные процедуры создания указателя 475
14.2.5 Дополнительные возможности 475
14.2.6 Управляющий файл 481
14.2.7 Простой пример файла, документированного при помощи пакета 482

doc
14.3 Утилита DOCSTRIP 485
14.3.1 Команды, используемые в пакетных файлах 485
14.3.2 Условное включение кода 487
14.4 Пример инсталляционной процедуры 487
A LaTeX: основы программирования **492**
A.1 Разметка и форматирование 492
A.1.1 Определение новых команд 493
A.1.2 Определение новых окружений 496
A.1.3 Создание счетчиков и изменение их текущих значений 499
A.1.4 Управление параметрами расстояния 502
A.2 Разметка страниц: различные типы боксов 507
A.2.1 LR-боксы 508
A.2.2 Боксы-абзацы 510
A.2.3 Боксы-линейки 514
A.2.4 Работа с боксами 515
A.3 Структура пакетов и классов в LaTeX2e 517
A.3.1 Команды идентификации 519
A.3.2 Начальный командный код 520
A.3.3 Декларация опций 520
A.3.4 Исполнение опций 522
A.3.5 Загрузка макропакетов 523
A.3.6 Основной командный код 524
A.3.7 Команды, специально предназначенные для макропакетов и классов 524
A.3.8 Команды, специально предназначенные для классов 525
A.4 calc — макропакет для арифметических вычислений 527
A.5 ifthen — усовершенствованный условный переход 529
B Техническое обеспечение и группы пользователей **534**
8.1 Главные сайты TeX'a в Internet'e 534
8.2 Mail-серверы 538
8.3 Группы пользователей TeX'a 538
Список литературы 541
Именной указатель **555**
Предметный указатель 557
Список таблиц **591**
Список иллюстраций **594**
Оригинал-макет The LaTeX Companion 597

Предисловие редактора перевода

Этой книгой издательство «Мир» открывает новую серию «Библиотека издательских TEXнологий», в которой в основном будут представлены книги по системам на базе TEX'а и по родственным им пакетам. Так как потенциальные читатели этих книг по большей части не столько профессиональные издатели и полиграфисты, сколько либо авторы, самостоятельно готовящие свои работы на компьютере, либо разработчики издательских систем на базе TEX'а, либо программисты, мы намерены освещать в этой серии и такие вопросы, как художественное оформление изданий, шрифтовой дизайн, основы технического редактирования и др. Нашей целью является повысить культуру разработчиков и пользователей TEX'а в области издательских TEXнологий и издательских технологий. Пожелаем же друг другу удачи.

Настало время, когда российскому читателю уже не нужно объяснять, что такое издательские системы на базе TEX'а и почему именно в этих «старомодных» пакетах продолжают работать все ведущие научно-технические издательства мира. Проблема теперь в другом: как разобраться в море пакетов, окружений, макрокоманд, форматов, схем кодировок и прочих новых образований вокруг TEX'а и найти именно то, чего так остро не хватает именно вам в вашей работе. Перед вами книга, цель которой помочь узнать о многом в этой области и выбрать нужное. Это действительно заботливый и опытный гид, которому полностью можно довериться. Авторы книги являются разработчиками ряда описанных в книге пакетов; их собственные пакеты, а также и многие другие они имели возможность опробовать на практике: в CERN'е (Швейцария) и в университете г. Майнца (ФРГ). Общение с сотнями пользователей TEX'а, с одной стороны, и с издательскими работниками, с другой, позволило им глубоко вникнуть в проблемы, возникающие у потребителей TEXнических продуктов.

Как сам TEX — замечательное творение Дональда Кнута, так и созданный на его основе LATEX Лесли Лэмпорта прочно вошли в обиход в нашей стране,

причем не только среди научных работников, но и среди профессиональных полиграфистов. Неизменно пополняются ряды разработчиков, адаптирующих этот мощный аппарат оформления англоязычной печатной продукции под отечественные полиграфические нормы. Для того, чтобы читатель мог получить представление о том, как обстоят дела с русификацией, мы провели опрос по электронной почте среди наших разработчиков. Результат этого опроса представлен в табл. 1. Список заведомо неполный: есть и другие серьезные разработки (например, русификация В. К. Малышева ВаКоМа TEX, которую можно скачать, в частности, со странички *CyrTUG*'а `http://www.cemi.rssi.ru/cyrtug`), но, чтя авторское право, мы не сочли возможным помещать какую-либо информацию о пакетах, если сами разработчики того не пожелали. Существуют и «безымянные герои»: почти в каждом научном институте имеется своя русификация, в которой учтены нужды сотрудников. Иначе говоря, ситуация у нас сейчас примерно такая, какой она была к моменту написания английского оригинала книги на западе: русификаций достаточно, но о совместимости пакетов разработчики пока не задумываются. В самом свежем (на сегодняшний день) релизе LaTeX 2_ε, имеющемся на CTAN (от 1 декабря 1998 г), предусмотрена поддержка кириллицы (в частности, с пакетом babel), в чем можно убедиться, заглянув в директорию `tex-archive/macros/latex/required/cyrillic`. Не стоит, однако, обольщаться по поводу того, что теперь установился стандарт на кириллицу. Скорее, это лишь первые шаги.

Хочется надеяться, что издание данной книги на русском языке в какой-то мере поможет сделать и последующие шаги. Здесь как нельзя более уместно напомнить кредо медиков: не навреди. Ничего не меняйте внутри системы и, создавая свои макро и подпакеты, всегда помните о совместимости. Ведь Дональд Кнут, дав своему детищу статус «public domain», руководствовался благородным стремлением снабдить ученых единой универсальной системой обмена научной информацией, безукоризненно работающей на всех платформах.

Теперь несколько слов о том, как готовился русский перевод и оригинал-макет книги. Авторы любезно предоставили нам исходный TEX-файл (с исправлениями на начало 1998 г.) своей книги со всеми необходимыми стилями и макрокомандами, за что мы им чрезвычайно признательны. В этот исходник переводчики бережно вбили русский текст, стараясь сохранить все служебные команды оригинала. Тот, кто хоть однажды сталкивался с такой технологией подготовки перевода, прекрасно понимает, что в этом месте начинается самое интересное: у нас другой формат полосы набора, другие правила полиграфического оформления (ГОСТы, ОСТы и гигиенические требования), не говоря уже о шрифтах, грамматике, переносах и алфавите, наконец. Итак, нужно переделывать все. Сразу признаемся, все переделать не удалось. Мы не рискнули перенести на российскую почву программу varioref, обманным путем добились поворота в нужном направлении от пакета rotating, столь же невежливо обошлись с changebar. В русском издании не используется прием `\finallongpage`, так как такие вольности в наших типографиях не приветствуются, и по тем же причинам нет ссылок на полях (как это имеет место в оригинале) на основополагающую книгу Лесли Лэмпорта LaTeX

1	**Название**	SWconv	TTFtoGF	T2 enc
2	**Автор(ы)**	Васильев К. А. <vka@geocities.com> <vka@lt1.phys.msu.su>	Васильев К. А. <vka@geocities.com> <vka@lt1.phys.msu.su>	Werner Lemberg <wl@gnu.org>, Волович В. В. <TeX@vvv.vsu.ru>
3	**Copyright**	1998, Васильев К. А.	1996–1998, Васильев К. А.	LaTeX's licence
4	**Год выпуска**	1998	1997	1997
5	**Дата последнего обновления**	11 апреля 1998	27 апреля 1998, версия 1.34	Декабрь 1998
6	**Операционная(ые) система(ы)**	MS Windows 95, 98, NT	MS Windows 3.x, 95, 98, NT, OS/2 (Win-OS/2)	не зависит
7	**Пакет(ы), взятый(ые) за основу**	–	–	–
8	**Занимаемый объем**	250Kb	300Kb	менее 150 kb
9	**Требуемые ресурсы**	2Mb оперативной памяти	2Mb оперативной памяти	–
10	**Используемые шрифты**	–	TrueType, Adobe Type 1	любые
11	**Документация на русском языке**	в комплекте поставки	нет	в комплекте поставки
12	**Где можно получить (купить)**	http://www.geocities.com/CapeCanaveral/Lab/7032 http://vk.da.ru	http://www.geocities.com/CapeCanaveral/Lab/7032 http://vk.da.ru	CTAN:macros/latex/contrib/supported/t2
13	**Уровень отлаженности**	готовая версия	готовая версия	готовая версия
14	**Статус**	free ware	free ware	free ware
15	**Краткое описание**	Утилита предназначена для сопряжения сред набора семейства Scientific Word 3.0 с кириллическим TeX'ом. Есть возможность прозрачного присоединения для обработки документа из интегрированной среды	Утилита предназначена для преобразования масштабируемых шрифтов TrueType в растровые форматы TFM/PK, используемые в TeX'е. При использовании ATM 4.0 возможна обработка шрифтов Adobe Type 1. Утилита предоставляет удобный графический интерфейс пользователя.	Пакет осуществляет поддержку «кириллизации» как для макропакетов, базирующихся на plain TeX'е, так и для LaTeX'а; задействованы все основные кириллические кодировки (20 на данный момент) и шрифты (по умолчанию LH). Предусмотрена возможность переключения на любые другие шрифты, в том числе на Type1 и TrueType)

Таблица 1. Пакеты, поддерживающие русификацию

1	**Название**	Русский TEX 97	–	ViTEX
2	**Автор(ы)**	Знаменский С. В., Воронин А. В., Знаменская Л. Н.	Ганелин П., Львовский С., Шень А. и др.	Виноградов М. М.
3	**Copyright**	Знаменский С. В., Воронин А. В., Знаменская Л. Н. РФФИ (гранты 95-07-19400в, 98-07-90179)	–	Виноградов М. М.
4	**Год выпуска**	1997	1993	1991–1998
5	**Дата последнего обновления**	24 декабря 1998	1997 (DOS), 1998 (Unix)	1999
6	**Операционная(ые) система(ы)**	MSDOS, окно DOS под WINDOWS-95, под WINDOWS NT	DOS, UNIX	DOS, Windows 3, 95, 98, NT
7	**Пакет(ы), взятый(ые) за основу**	emTeX (1995-1998), dvips (1995-1997), bm2font (1997), mfpic (1997), disp189a (1996), az (1997), MultiEdit (демо-версия)	emTeX (DOS), teTeX (Linux)	–
8	**Занимаемый объем**	от 2.2 до 60 Mb	12 дискет 1.44 (DOS), < 2 Mb (Unix, только добавления)	5–50 Mb
9	**Требуемые ресурсы**	от PC 286	–	–
10	**Используемые шрифты**	CM с расширением RF	несколько исправленные шрифты Н. Глонти	разработки автора
11	**Документация на русском языке**	в комплекте поставки	в комплекте поставки	в комплекте поставки
12	**Где можно получить (купить)**	`ftp://tex.mi.ras.ru/pub/Russian_TeX/Russian_TeX_97` `ftp://ftp.chg.ru/pub/TeX/znamensk`	`ftp://ftp.multimedia.ru` (DOS) `ftp://mccme.ru/users/shen` (Unix), справки: `shen@mccme.ru`	у автора
13	**Уровень отлаженности**	готовая версия	готовая версия	готовая версия
14	**Статус**	public domain	public domain	коммерческий
15	**Краткое описание**	Многовариантность и простота установки и доустановки, поддержка устаревших форматных файлов, автоматическая настройка на обрабатываемый файл, возможность сосуществования с другими установками TEX'а на той же машине.	Не использует виртуальных или postscript-шрифтов; DOS-вариант для экономии места переопределяет cmr-шрифты, добавляя в них русские символы (что нехорошо), UNIX-вариант этого не делает. Можно использовать в командах и русские буквы.	Удобная многофункциональная оболочка на основе редактора MultiEdit, возможность работы с любыми TEX'овскими форматами, компиляция и просмотр отдельных фрагментов текста, в том числе в параллельном окне для Windows 95, 98, использование нестандартных шрифтов.

Таблица 1. *Продолжение*

1	**Название**	NCC-TEX	Кодировки T2A-D, X2	Пакет LH fonts	RT формат
2	**Автор(ы)**	Роженко А. И.	Бердников А. С., Колодин М. Ю., Лапко О. Г., Янишевский А. В.	Бердников А. С., Лапко О. Г., Ходулев А. Б.	Бердников А. С., Лапко О. Г., Янишевский А. В.
3	**Copyright**	на русификацию и макро NCC	*CyrTUG* + авторы	*CyrTUG* + авторы	авторы (+ авторы таблиц переносов)
4	**Год выпуска**	1995	1997	1994	1998
5	**Дата последнего обновления**	26 октября 1998	1998	1998	1999
6	**Операционная(ые) система(ы)**	DOS, Windows	–	–	–
7	**Пакет(ы), взятый(ые) за основу**	emTeX (30.06.1997), $\mathcal{AMS}$-TEX-2.1, LATEX 2.09, LATEX 2_ε (01.06.1998)	–	шрифты WNCYR ($\mathcal{AMS}$-TEX); EC; Cyrillic Я. Хараламбуса.	RT fonts (Янишевский А. В.)
8	**Занимаемый объем**	от 8 Мбайт	–	1–2mb	400 kb
9	**Требуемые ресурсы**	от PC AT-286	–	–	–
10	**Используемые шрифты**	Самарина-Глонти модифицированные Беклемишевым и Котельниковым	пакет LH font family	–	пакет LH font family 3.2 и выше
11	**Документация на русском языке**	новой документации нет	статья в трудах CyrTUG 1997	в комплекте поставки	–
12	**Где можно получить (купить)**	на серверах `sgi.sscc.ru` и `nmsf.sscc.ru`	–	`CTAN:fonts/cyrillic/lh`, `http:..www.cemi.rssi.ru/cyrtug`	–
13	**Уровень отлаженности**	для LATEX2.09 готовая версия, для LATEX 2_ε — альфа	готовая версия	готово (для T2* — бета)	альфа
14	**Статус**	public domain	public domain	public domain	public domain
15	**Краткое описание**	Многофункциональная система, адаптированная для использования в России. Поддерживает все стандартные форматы и NCC-LATEX, имеется простая оболочка.	Стандарт кодировки для кириллических языков в LATEX 2_ε. Содержит все символы существующих письменностей на основе кириллицы; работает в многоязычных документах.	Шрифты, для языков, использующих кириллицу. Поддержка популярных кодировок русского языка и кодировок T2* стандарта LATEX 2_ε.	Русификация стандартных форматов без и с использованием babel; в унифицированном виде с поддержкой популярных русских кодировок без применения inputenc/fontenc и активных символов.

Таблица 1. *Окончание*

A Document Preparation System [46]: они внесены в основной текст и выделены полужирным шрифтом так: [$\mathcal{L}$ **4**], где 4 — номер страницы книги. Не пришлось воспользоваться и такой замечательной программой, как BibTeX: ее русификация — это отдельная серьезная работа (необходима адаптация к ГОСТам и правилам русской пунктуации). Большие трудности вызвало огромное количество шрифтов. К шрифтовому изобилию оригинала, которое само по себе составляет немалую проблему (и авторы об этом пишут), добавилась в полном объеме кириллица. Добавление кириллицы несколько сбило с толку программу *MakeIndex*, так что окончательная доводка указателя проводилась вручную.

И все же бóльшая часть пакетов, приведенных в книге, успешно прошла испытание российской действительностью, и здесь самое место выразить глубокую благодарность нашему TeXническому редактору Ольге Лапко. Она бережно сохранила стиль английского оригинала, не нарушив при этом строгие нормы нашей полиграфии, и адаптировала весь TeXнический аппарат так, чтобы этот стиль мог быть реализован. В частности была задействована опция russian пакета babel: была сделана активным символом литера " и на ее основе введены тире и кавычки «елочки», сделаны отбивки перед «высокими» знаками препинания: ? ! : ;.

Не обошлось без проблем и при переводе. Почти всюду в тексте, наряду со ссылкой на работу Лесли Лэмпорта, вы встретите ссылку на книгу С. М. Львовского «Набор и верстка в пакете LaTeX», например так: [$\mathcal{L}$ **176–8**], [$\mathcal{Л}$**300–4**]. К сожалению, книга Лесли Лэмпорта не переведена на русский язык и поэтому основной массе россиян недоступна. Книга же С. М. Львовского вышла большим тиражом и есть у очень многих. Если какой-то вопрос в ней оказался неосвещенным, парная ссылка не добавлена. Ряд примеров в книге решено было оставить на языке оригинала, поскольку часто речь идет о специфике набора и оформления именно англоязычных текстов. Некоторая чехарда с терминологией объясняется тем, что здесь сошлись несколько дисциплин со своими традициями: *page* это и *страница*, и *полоса* набора; *column* это и *столбец*, и *колонка*; *строка* имеет два английских эквивалента — *row* и *line*. Так что мы не стали вводить третий и перевели английский термин *string* как *цепочка литер*, а не как *строковая переменная* (да простят нас программисты).

Книгу перевели: Маховая О. А. — предисловие, гл. 1–4, приложение В; Тюменцев Ю. В. — гл. 5–7, 12; Чистяков В. В. — гл. 8–11; Третьяков Н. В. — гл. 13–14, приложение А, список литературы. Весь коллектив, работавший над книгой, старался максимально точно передать как внутреннюю, так и внешнюю сторону содержания книги, хотя это не всегда удавалось. Поэтому вся ответственность за вкравшиеся опечатки и неточности ложится на нас, и, в первую очередь, на редактора книги. Отзывы и замечания можно направлять в издательство «Мир» по адресу `irina@mir.msk.su` и на страничку `http://www.cemi.rssi.ru/cyrtug` группы пользователей кириллического TeX'а *CyrTUG*.

И. Маховая

Посвящаем
нашим женам —
Альбине, Кристел и Татьяне
и сыновьям —
Алексею, Андрею, Арно, Николаю и Роману
за их понимание и долготерпение.

Предисловие

LaTeX — это система общего назначения для набора текстов, которая использует TeX в качестве средства форматирования. Настоящее руководство представляет собой подробный путеводитель по ослепительным и не очень бросающимся в глаза красотам LaTeX'а. Как таковой, это исчерпывающий трактат по темам, которые недостаточно раскрыты в книге Лесли Лэмпорта LaTeX: *A Document Preparation System* (в дальнейшем при ссылках — LaTeX book) [46]. Расширения основного LaTeX'а, описанного в указанной книге, обсуждаются в нашем руководстве, так что оба этих издания представляют полный обзор возможностей системы LaTeX.

Благодаря своей гибкости, простоте использования и профессиональному полиграфическому качеству, LaTeX в настоящее время применяется при подготовке изданий почти по всем областям точных и гуманитарных наук. В отличие от многих текстовых процессоров LaTeX (и лежащий в его основе TeX как средство форматирования) распространяется бесплатно и не зависит ни от конкретной архитектуры компьютера, ни от операционной системы. Поскольку исходными файлами LaTeX'а служат простые текстовые файлы, их можно пересылать вместе с соответствующим программным обеспечением с одной вычислительной платформы на другую по всему свету (по электронным сетям или обычной почтой). Получатель сможет напечатать копию, идентичную той, которая была у отправителя, независимо от используемого ими вычислительного оборудования. Таким образом, члены групп, географически отдаленные друг от друга, в разных странах и даже континентах, могут теперь работать вместе над созданием сложных документов, разные части которых подготавливаются разными лицами и затем объединяются воедино без особых проблем. Более того, использование электронных рукописей должно, в принципе, ускорить процесс публикации работ в издательствах.

LaTeX'у научиться нетрудно, и новичок уже после прочтения нескольких глав книги LaTeX book Лесли Лэмпорта — основного руководства по LaTeX'у,— сможет извлекать для себя пользу от этой системы. Немного попрактиковавшись, вы, вероятно, захотите решать более сложные проблемы, решение которых нельзя

получить непосредственно из упомянутой книги. Если вы один из тех пользователей, кто хотел бы знать, как расширить возможности LaTeX'а, чтобы получать сколь угодно красивые документы, не становясь при этом (LA)TeX гуру[1], то эта книга для вас.

Вас проведут, шаг за шагом, по различным важным областям LaTeX'а и покажут, как они между собой связаны. Во всех деталях описаны структура документа, основные средства форматирования и макет полосы набора. В удобной форме представлена богатая библиотека пакетов по плавающим объектам, графике, таблицам, PostScript и многоязыковой поддержке. Эта книга представляет собой первое издание, в которое в полном объеме вошел важный инструментарий LaTeX'а: новейшее описание второй версии Новой схемы выбора шрифтов (New Font Selection Scheme, NFSS2), математическое расширение $\mathcal{AMS}$-LaTeX, расширения epic и eepic окружения `picture` LaTeX'а и программы получения указателей и списков литературы (и управления ими) *MakeIndex* и BibTeX. И, наконец, картину завершает обзор способов определения новых команд и окружений, длин, боксов, общих перечней и т. д., равно как и способов, облегчающих работу с ними.

Мы все трое в течение нескольких лет были вовлечены в работу по поддержке и разработке приложений LaTeX'а для различных профессиональных нужд в разных странах. Нам приходилось раскрывать секреты LaTeX'а для самых разнообразных аудиторий и прислушиваться к мнениям различных групп пользователей во время дискуссий по текстовым процессорам в электронных группах новостей и в TeX-конференциях. Это позволило нам аккумулировать разные точки зрения по широкому спектру тем, которые, как нам кажется, вам рано или поздно понадобятся, если вы хотите в полной мере эксплуатировать богатство и мощь системы LaTeX. Заметим, однако, что эта книга не заменяет LaTeX book, а только лишь сопровождает ее. Предполагается, что вы знакомы с первой частью этой книги; во всяком случае, данное пособие следует рассматривать как наиболее полное описание команд LaTeX'а.

Чтобы сделать представленную здесь информацию еще более полной и полезной, приглашаем наших читателей присылать их комментарии, предложения и замечания любому из авторов. Мы будем рады исправить любые оставшиеся опечатки и ошибки в последующих изданиях и всегда открыты для предложений по усовершенствованию старых разработок или включению новых, которые по каким-то причинам выпали из нашего поля зрения.

LaTeX 2_ε — новый релиз пакета LaTeX

На протяжении многих лет для LaTeX'а разрабатывались новые расширения с одним печальным итогом: на разных сайтах вошли в употребление несовместимые форматы LaTeX'а. Таким образом, получая документы из разных мест, специа-

[1] (LA)TeX гуру — это тот, кто знает наизусть все об устройстве как LaTeX'а, так и TeX'а. В этой книге лого (LA)TeX будет употребляться в тех случаях, когда надо сослаться и на TeX, и на LaTeX.

лист, обслуживающий сайт, был вынужден поддерживать LaTeX (с NFSS или без), SliTeX, $\mathcal{AMS}$-LaTeX, и т. д. Более того, при взгляде на исходный файл не всегда было понятно, для какого формата этот документ был подготовлен.

Чтобы положить конец этому неудовлетворительному положению дел, в конце 1993 г. был объявлен новый релиз LaTeX'а, который привел все подобные расширения к единому формату и приостановил распространение взаимно несовместимых диалектов LaTeX 2.09. В LaTeX 2_ε новая схема выбора шрифтов должна была стать стандартом, а стилевые файлы типа amstex (бывший формат $\mathcal{AMS}$-LaTeX'а) или slides (бывший SliTeX) стали бы пакетами-расширениями, работающими с одним и тем же базовым форматом. Введение нового релиза также позволило добавить небольшое количество часто недостающих свойств (типа расширенной версии `\newcommand`). Все эти новые возможности описаны в нашей книге, *что позволит вам применять новый релиз LaTeX'а в полном объеме.*

Чтобы легче было различать исходники, подготовленные в старом LaTeX 2.09 и в новом релизе с использованием новых средств, первая команда в подготовленном в LaTeX'е документе `\documentstyle` была заменена на `\documentclass`, так что программное обеспечение автоматически сможет отличить старый исходник и переключиться на режим совместимости с ним, если необходимо.

Проект LaTeX3

В настоящее время LaTeX подвергается переработке и процесс этот координируется одним из авторов (Франком Миттельбахом), Крисом Роули и Райнером Шопфом. Эта инициатива носит название Проект LaTeX3 [51]. Большое количество средств, описанных в настоящей книге как расширения базового LaTeX'а, будут доступны и в этой новой системе либо как часть ядра, либо как один из пакетов-расширений. Мы направили половину нашего гонорара (в качестве финансовой поддержки) непосредственно на Проект LaTeX3. Так что когда вы покупаете эту книгу, вы не только получаете удобное, полное и свежее руководство по многим важным и полезным пакетам, работающим с современным LaTeX'ом, но также активно способствуете тому, чтобы LaTeX стал более мощным и удобным в будущем.[2]

Благодарности

Прежде всего мы хотим поблагодарить нашего редактора Питера Гордона из Эддисон-Уэсли, который не только способствовал выходу книги в свет, но и постоянно помогал нам не сбиться с правильного пути. Его предложения и идеи

[2] Имеется ввиду оригинал на английском языке, разумеется. Покупая перевод на русском языке, вы поддерживаете издательство «Мир» и его сотрудников, вносящих посильный вклад в русификацию LaTeX'а.— *Прим. ред.*

улучшили наше руководство как по содержанию, так и по форме. Мы также хотим выразить благодарность Марше Финли из Бюро подготовки рукописей за высокоэффективную помощь в практической подготовке нашей рукописи. Элен Голдстейн, младший редактор Эддисон-Уэсли, всегда была готова отвечать на наши вопросы.

Мы чрезвычайно признательны всем тем нашим коллегам из мирового сообщества (LA)TEX'а, кто разрабатывал пакеты, не только описанные в этой книге, но и сотни других, помогающих пользователям готовить свои документы лучше и быстрее. Без постоянных усилий всех этих энтузиастов LaTeX не стал бы таким замечательным и гибким средством сегодня. Нам хочется надеяться, что мы восстановили справедливость по отношению к ним, когда, при первом описании того или иного пакета, упоминали (как только информация становилась нам доступна) истинных авторов и (или) других лиц, внесших существенный вклад.

Мы все в неоплатном долгу перед Йоханнесом Брамсом, Дэвидом Карлайлом, Майклом Даунзом, Себастьяном Ратцем и Райнером Шопфом за их тщательное прочтение рукописи. Их многочисленные комментарии, предложения, исправления и советы существенно повлияли на качество текста. Роджер Вулнуф прочел корректуру первой версии рукописи, Сильвио Леви — главу о NFSS. Наконец, мы хотим выразить нашу признательность институту CERN, предоставившему нам возможность подготовить электронную версию рукописи на вычислительной технике института.

Как читать эту книгу

Названия глав достаточно ясно отражают тему, к которой они адресуются в каждом случае. В принципе, все главы написаны так, что их можно читать более или менее независимо; если необходимо, даются ссылки на другие разделы книги, где можно найти дополнительную информацию.

Глава 1 Краткое введение в систему LaTeX.

Глава 2 Обсуждаются глобальная и локальная разметки документа.

Глава 3 Описываются основные команды набора LaTeX'а.

Глава 4 Объясняется, какие имеются средства для глобального задания макета полосы набора при помощи стиля полосы.

Глава 5 Показано, как представить материал в виде строк и столбцов при помощи расширенных окружений `tabular` и `array` и их многостраничных эквивалентов `supertabular` и `longtable`.

Глава 6 Дается общая трактовка плавающих объектов.

Глава 7 Подробно обсуждается Новая схема выбора шрифтов (NFSS2) LaTeX'а и представлены различные ее команды для пользователя. Показа-

но, как добавлять новые шрифты в математическом и текстовом режимах.

Глава 8 Дается обзор пакета amstex, который добавляет много мощных полезных команд для набора математических формул.

Глава 9 Взгляд на проблему использования LaTeX'а в многоязычном или отличном от английского языка окружении. Описываются система babel и другие, ориентированные на языковые особенности, пакеты.

Глава 10 Обращение к графике, не зависящей от устройства, показывает, как epic, eepic и другие пакеты расширяют возможности основного окружения `picture` LaTeX'а.

Глава 11 Показано, не только как язык описания страниц PostScript может превратить LaTeX в объемную графическую утилиту, но и как предоставить пользователю возможность посредством NFSS выбирать шрифт среди сотен шрифтовых семейств, доступных как PostScript Type 1.

Глава 12 Обсуждаются проблемы, связанные с подготовкой указателя. Подробно описывается программа *MakeIndex*.

Глава 13 Рассказывается, как родственная LaTeX'у программа BibTeX пытается решить проблемы, связанные с поддержанием библиографической базы данных. Обсуждаются различные существующие стили для списков литературы, представлен в деталях используемый в стилевых файлах формат языка BibTeX, помогающий пользователю совершенствовать существующий стиль.

Глава 14 Показано, как документировать файлы LaTeX'а при помощи пакета doc и сопровождающей его программы DOCSTRIP.

Приложение A

Впервые приводится обзор методов работы с основными программными структурами LaTeX'а. При помощи пакета calc вводятся расширения над полем арифметических операций; обсуждаются расширенные управляющие структуры, добавленные в LaTeX 2_ε.[3]

Приложение B

Объясняется, как получить файлы, описанные в данной книге: из различных TeX-архивов и из групп пользователей TeX'а.

Чтобы сделать примеры максимально независимыми от основного TeX'а, широко использовались пакеты calc и ifthen, описанные в приложениях A.4 и A.5. Если вы хотите подробно разобраться в функционировании как можно большего

[3] В LaTeX 2.09 программные структуры типа if-then-else (если-то-иначе) доступны в пакете ifthen, который расширен и усовершенствован в LaTeX 2_ε.

количества примеров этой книги, следует изучить расширения LaTeX'а, введенные в эти пакеты.

Во многих примерах использовались новые свойства (возможности) LaTeX 2_ε; в особенности это касается изменения шрифтов в тексте, которые проводились в стиле LaTeX 2_ε, т. е. при помощи команд, приведенных в табл. 7.2 на с. 198. Как правило, сокращенные формы типа {\bf слово} не использовались, так как такие команды определяются стилем и могут быть доступны или недоступны для всех классов документов.

Несмотря на то что бóльшая часть книги почти наверняка годится для LaTeX 2.09 (пакет LaTeX 2_ε был готов, когда 90% книги было написано), мы предполагаем, что вы обновите свою версию как можно скорее, и таким образом мировое сообщество пользователей LaTeX'а снова заговорит на одном и том же языке. Как было сказано выше, LaTeX 2_ε способен распознать и обработать старые документы, написанные в LaTeX 2.09. Однако пакеты, написанные или адаптированные под LaTeX 2_ε, в старой системе работать не будут.

Соглашения о полиграфическом оформлении

Как уже говорилось при обсуждении связей между содержанием и формой или между общей и локальной разметкой, существенным моментом является представление материала сразу же в таком виде, чтобы продемонстрировать эти функции в рамках данного текста. Итак, ниже представлены соглашения о полиграфическом оформлении настоящей книги.

Названия команд и окружений LaTeX'а набраны машинописным шрифтом (например, \caption, enumerate, \begin{tabular}), а названия пакетов и файлов класса — рубленым шрифтом (без засечек) (например, article).

Примеры синтаксиса конструкций LaTeX'а заключены в рамочку; аргументы команд даны курсивом.

\commandname{*arg1*}{*arg2*}{*arg3*}

Строки, содержащие примеры с командами LaTeX'а, набраны с втяжкой машинописным шрифтом и несколько мельче основного шрифта книги.

```
\chapter{Название главы}
\section{Название раздела}
Некий текст...
```

Когда необходимо показать результат работы последовательности команд, исходный текст и вывод располагаются рядом:

В правой колонке представлен исходный текст, предназначенный для обработки LaTeX'ом. В левой колонке приводится результат вывода на печать или экран.

```
В правой колонке представлен исходный
текст, предназначенный для обработки
\LaTeX'ом. В левой колонке приводится
результат вывода на печать или экран.
```

Для бóльших по ширине примеров, когда исходный текст и вывод невозможно расположить рядом, принято следующее соглашение об оформлении:

Исходный текст

```
Это длинная строка, исходное представление которой и ее вывод нельзя
элегантным образом расположить рядом.
```

Это длинная строка, исходное представление которой и ее вывод нельзя элегантным образом расположить рядом.

Текст на выводе

Перекрестные ссылки на номера страниц, где данный предмет обсуждается в книге LaTeX book Лесли Лэмпорта, даны полужирным шрифтом так: [**Л 4**]. Они соответствуют номерам страниц первого издания, описывающего работу LaTeX 2.09; все нововведения LaTeX 2_ε представлены в настоящей книге[4].

Команды, набираемые пользователем с компьютерного терминала, представлены машинописным шрифтом и подчеркнуты: `Это вводит пользователь`.

Использование всех представленных в книге пакетов

В книге описано более 150 пакетов и опций, которые расширяют или модифицируют основные возможности LaTeX'а. Чтобы продемонстрировать их действие, мы (в принципе) должны загрузить их все одновременно. По различным причинам это на практике нереализуемо, если вообще возможно. В самом деле, многие пакеты, вроде program, используют огромное количество счетчиков, тогда как TeX позволяет всего 256. Так что если вы достигли этого предела, следует сократить количество одновременно загружаемых файлов. При подготовке оригинал-макета книги мы придерживались иной стратегии: ряд примеров создавались как отдельные файлы и затем включались в книгу в виде Encapsulated PostScript. Более того, мы использовали пакет hackalloc. Он переопределяет локализацию примитивов, так что *вся* локализация становится локальной по группам. Это означает, что, загружая пакеты только когда они нужны внутри группы, счетчики и значения переменных должны быть делокализованы, когда вы выходите из группы. Эта процедура, однако, может иметь некие побочные эффекты, и ее следует использовать с большой осторожностью. Тем не менее большинство пакетов мы использовали одновременно, и по этой причине должны были несколько раз в течение подготовки книги перекомпилировать TeX. Один из файлов протокола (log-файл), получен-

[4] Так как для русскоязычного читателя более доступна книга С. М. Львовского «Набор и верстка в пакете LaTeX» [45], мы, по возможности, добавили ссылки и на это издание в таком же оформлении: [**Л 4**].— *Прим. ред.*

ных на последних стадиях подготовки оригинал-макета, показал такую картину:

```
Here is how much of TeX's memory you used:
 9692 strings out of 16716
 118315 string characters out of 133654
 236569 words of memory out of 262141
 8131 multiletter control sequences out of 9500
 81058 words of font info for 228 fonts, out of 90000 for 255
 20 hyphenation exceptions out of 607
 34i,23n,41p,509b,1403s stack positions out of 300i,40n,60p,3000b,4000s
Output written on companion.dvi (555 pages, 2008780 bytes).
```

Как видите, мы практически исчерпали шрифтовые ресурсы (которые не могут быть увеличены в дальнейшем[5]), так как большое количество шрифтов демонстрировалось в гл. 7, а использование цепочек литер (strings characters), основной памяти (main memory) и управляющих последовательностей (control sequences) много выше, чем, вероятно, в любой из когда-либо обрабатываемых LaTeX'ом работ. Поэтому удивительно, что вся книга произведена за единственный прогон LaTeX'а со всеми пакетами, работающими вместе, чтобы произвести примеры.

Даже если вы не исчерпали ресурсы, о которых шла речь выше, между различными пакетами возможны иные коллизии. К примеру, некоторые расширения типа french делают отдельные литеры активными (т. е. некоторые литеры ведут себя как управляющие последовательности). Проблемы возникают, когда к таким литерам в дальнейшем обращаются другие пакеты. Это означает, что не все описанные в книге пакеты могут использоваться вместе. Иногда коллизий можно избежать, если такой опасный пакет загрузить как одну из последних деклараций \usepackage. Кроме того, некоторые пакеты делают активной литеру @ (например, amstex) и это может привести к ужасным последствиям, если применяется другой пакет, использующий литеру @.

Согласно правилу большого пальца, если вы наблюдаете несколько необычное поведение при добавлении к существующему перечню пакетов нового, который раньше, казалось, работал нормально с другими, это может оказаться проблемой совместимости. Попробуйте загрузить новый файл в конце, и, если это не поможет, удаляйте другие файлы по одному. Таким способом вы сможете обнаружить файл или файлы, виновные в возникшей коллизии.

[5] При подготовке русского перевода количество шрифтов возросло на полный комплект кириллицы, так что файл протокола выглядел еще более устрашающим.— *Прим. ред.*

ГЛАВА 1

Введение

LaTeX — не просто система для набора математических текстов. Область его применения — это служебные записки, деловые и частные письма, информационные бюллетени, статьи по точным и гуманитарным наукам, справочники, словари и монографии по всем отраслям знаний. В настоящее время существуют версии LaTeX'а практически для каждой крупной вычислительной машины, рабочей станции или персонального компьютера. Чтобы было понятно, почему так случилось, в первом разделе этой главы рассказывается о происхождении TeX'а и LaTeX'а и ставится вопрос об их будущем. Второй раздел посвящен общему обзору системы LaTeX. Это поможет читателю четко уяснить роль различных составляющих и файлов, создаваемых LaTeX'ом. Предметом следующих разделов является описание важных различий между общей разметкой документа и явными указаниями технического редактора. Общий подход имеет то преимущество, что он позволяет отделить содержание документа от его формы и еще раз подтверждает, что документы следует по возможности строить на принципе общей разметки. Когда же, для большей ясности, возникает необходимость в использовании непосредственных указаний технического редактора, эти указания должны быть классифицированы в категории, определения которых группируются в начале документа, а затем применяются в тех или иных конкретных местах. Такая практика гарантирует последовательный подход и простоту использования.

1.1 Краткая история TeX'а и LaTeX'а

1.1.1 Вначале был TeX

В мае 1977 г. профессор Станфордского университета Дональд Кнут [29] начал работать над издательской системой, известной ныне как TeX и METAFONT [30–34]. В предисловии к книге *The TeXbook* [30] Кнут пишет о TeX'е как о «новой системе набора, предназначенной для создания красивых книг и особенно книг, которые

содержат много математики. Приготовив рукопись в формате TEX, вы тем самым точно объясните компьютеру, как преобразовать рукопись в страницы, типографское качество которых сравнимо с работой лучших в мире наборщиков ... ».

TEX стал популярным среди многих тысяч ученых, потому что с его помощью любые тексты можно преобразовать в статьи, доклады, заявки, книги, поэтические сборники и другие форматы способом, который полностью определяется автором, благодаря богатому языку команд. Программа METAFONT делает возможным дизайн шрифтов, используемых при печати готовых страниц.

TEX особенно полезен, когда документ содержит математические формулы и когда желательно, чтобы документ выглядел, как книга. Кроме того, это независимая от компьютерного устройства система, работающая на большом количестве платформ, от персоналок до больших вычислительных машин; она ведет себя одинаково на всех машинах — факт в высшей степени важный для научно-технического сообщества. С этим свойством связана печать dvi-файлов, так что документ может быть распечатан на чем угодно, будь то CRT-экран, точечно-матричный или лазерный принтер со средним разрешением или профессиональный фотонаборный автомат с высоким разрешением.

Благодаря этим качествам, а также тому, что это свободно распространяемый продукт, TEX стал *de facto* стандартом во многих академических отделениях и исследовательских лабораториях. Вместе с тем он заменил в профессиональном издательском мире наборные процессы. Он используется на всех мыслимых компьютерных платформах: IBM PC-совместимых персональных компьютерах и Macintosh'ах, UNIX и VMS-рабочих станциях и таких суперкомпьютерах, как Cray. Кроме того, отличные программы визуализации делают возможной работу на большинстве рабочих станций и других графических дисплеях.

Более 10 лет назад в предисловии к книге «TEX и METAFONT. Новые направления в наборе» [28] Гордон Белл написал, что «TEX Дона Кнута — наиболее выдающееся изобретение нашего века в издательском деле. Он вводит стандартный язык компьютерной типографии и по своей значимости может стоять в одном ряду с изобретением печатного станка Гутенберга».

Недавно Дональд Кнут официально объявил, что в интересах стабильности TEX не будет подвергаться дальнейшему усовершенствованию [38].

1.1.2 Потом Лесли Лэмпорт придумал LATEX

В начале 80-х годов Лесли Лэмпорт начал работать над издательской системой LATEX, в основе которой лежал TEX. Эта система укрупняет уровень абстракции команд plain TEX'а и дает возможность пользователю сосредоточиться на структуре документа больше, чем на деталях форматирования. Несколько команд высокого уровня позволяют с легкостью создавать большинство документов. Вы не должны беспокоиться об оформлении — это задача компьютерного технического редактора, который заботится о создании стилевых файлов для каждого конкретного случая.

Функции LaTeX'а вместе с несколькими вспомогательными программами включают в себя создание указателей, библиографий, перекрестных ссылок, оглавлений и вставку рисунков — эти возможности в TeX'е отсутствуют.

1.1.3 С LaTeX'ом в 2000 год?

С тех пор, как количество пользователей TeX'а и особенно LaTeX'а за последние несколько лет возросло до нескольких тысяч, LaTeX получил распространение в областях, для которых он не всегда приспособлен (книги по юриспруденции, критические издания классиков, поэзия, издания с параллельным расположением текстов на разных языках, информационные бюллетени и некоторые другие). В последних выпусках *TUGboat*, журнала Группы пользователей TeX'а, помещены статьи о недостатках TeX'а, LaTeX'а и взаимодействующих с ними программ [15, 16, 49, 53, 55].

После встречи с Лесли Лэмпортом на конференции TeX Users Group в 1989 г. в Станфорде Франк Миттельбах, Крис Роули и Райнер Шопф начали работать над преобразованием и расширением LaTeX'а, так называемым проектом LaTeX3 Project [51]. Главная идея заключается в создании оптимальной и эффективной основы с базовыми командами, укомплектованной различными программами, которые найдут применение в специальных областях (таких, как таблицы, рисунки и математические формулы). Новая система обеспечит возможность полной перестройки стилевого файла в пользовательском интерфейсе, упрощая совершенствование и поддержку собственных стилей.

В марте 1992 г. на конференции немецкой группы DANTE в Гамбурге была учреждена группа NTS для «Новой типографской системы», чтобы обсудить и согласовать те области, в которых TeX будет расширяться и обретет все необходимое для создания «шедевров издательского искусства» [76].

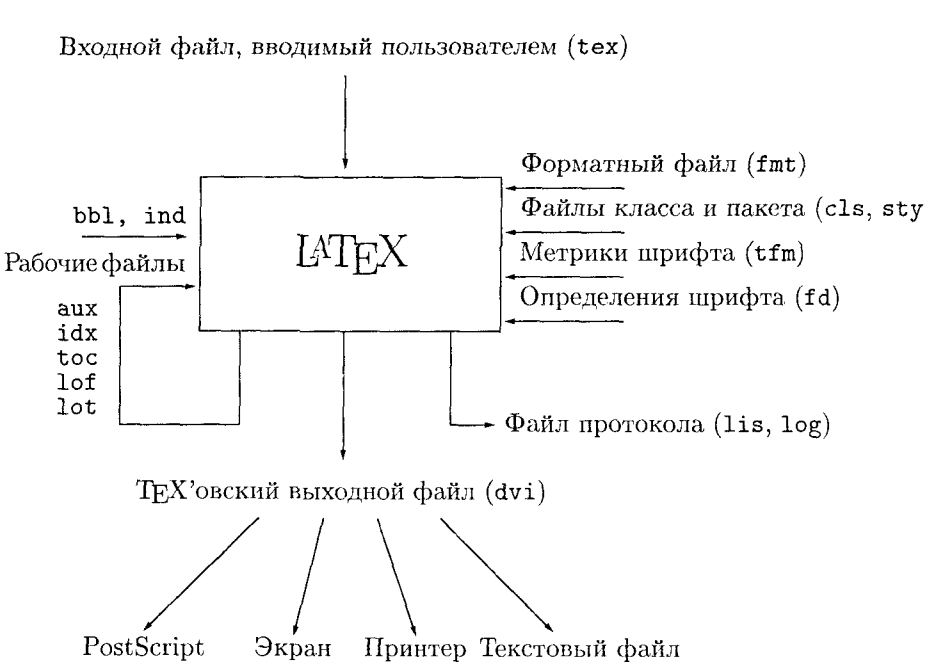
1.2 LaTeX и его составляющие

В этом разделе представлены основные принципы среды LaTeX и дано краткое описание различных файлов и программ, о которых должен знать просвещенный пользователь LaTeX'а. Подробнее см. статью Йоахима Шрода «The Components of TeX» [87].

1.2.1 Как работает LaTeX?

LaTeX читает и записывает несколько файлов, и вы должны иметь четкое представление о том, зачем они нужны. На рис. 1.1 ниже показана схема движения информации при работе в LaTeX'е и дается список различных необходимых файлов.

Самым главным в LaTeX'е является входной файл. Это текстовый файл, подготовленный с помощью текстового редактора и имеющий расширение `.tex`. Файлы, содержащие структуру и определения разметки (расширения `.cls`, `.sty`),



Пояснение	Расширение файла
Входной файл, подготовленный в LaTeX'е	`.tex`, `.ltx`
Выходной файл, отформатированный TeX'ом	`.dvi`
Файл протокола работы TeX'а	`.log`, `.texlog`, `.lis`, `.list`
Исходный файл METAFONT'а	`.mf`
Файл *определения шрифта*	`.fd`
Файл *начертания шрифта*	`.pk`
Файл метрик шрифта	`.tfm`
Файл цепочек литер	`.pool`, `.poo`, `.pol`
Форматный файл	`.fmt`
Файл LaTeX'а макета & структуры	`.clo`, `.cls`, `.dtx`, `.sty`
Дополнительный файл LaTeX'а	`.aux`
Файл *оглавления*	`.toc`
Файл *списка иллюстраций*	`.lof`
Файл списка таблиц	`.lot`
Файлы, относящиеся к BibTeX'у	`.bbl`, `.bib`, `.blg`, `.bst`
Файлы, относящиеся к указателю и *MakeIndex*'у	`.idx`, `.ilg`, `.ind`, `.ist`

Рис. 1.1. Перечень основных файлов, необходимых для работы TeX'а и LaTeX'а

обычно находятся в нескольких стандартных директориях. LaTeX разбит на 5 классов документов, которые называются article (статья), report (доклад), book (книга), slides (слайды) и letter (письмо). Эти основные классы документов могут быть обобщены путем определения одной или более опций классов или путем использования дополнительных пакетов, подобных тем, которые описаны в этой книге. Размеры шрифтов, используемых в TeX'е, называются метриками шрифтов (*font metrics*) и закодированы в файлах с расширением `.tfm` (для метрик шрифтов TeX'а). Для каждого используемого в документе шрифта должен быть `.tfm`-файл, описывающий высоту, глубину и ширину каждого символа. TeX пользуется этой информацией в своем алгоритме разбивки на строки при создании абзацев. TeX переносит слова автоматически, используя независимый от языка алгоритм Лианга [44]. При генерации форматного файла (расширение `.fmt`) могут быть определены различные способы переноса слов для каждого языка. Формат LaTeX (называемый обычно `lplain.fmt` или `latex.fmt`) содержит главным образом все команды LaTeX'а в предварительно откомпилированной форме и `.tfm`-информацию для нескольких загружаемых шрифтов. Соотношение между названиями внутренних и внешних шрифтов находится в файлах, определяющих шрифт (расширение `.fd`).

LaTeX образует группу выходных файлов. Один из них (расширение `.dvi`) содержит изображение отформатированного текста, в котором определены вид и расположение на странице каждого символа. Эти `.dvi`-файлы определяют только названия шрифтов и не содержат их изображений. TeX размещает боксы литер с разрешением более высоким, чем 1000 точек на дюйм, так что выходной файл, сгенерированный TeX'ом, фактически является *не зависящим от устройства* (*device independent*), почему он и называется `.dvi`-файлом. Чтобы увидеть результат, нужно преобразовать `.dvi`-файл с помощью `dvi`-драйвера в желаемый формат (например, PostScript).

При каждом сеансе работы LaTeX генерирует файл протокола работы, как правило с расширением `.log` или `.lis`, но может иметь и другие расширения (или не иметь их вовсе) в зависимости от операционной системы. Этот файл содержит информацию, бóльшая часть которой появляется также и на экране. Сюда входят имена прочитанных файлов, номера обработанных страниц, предупреждения и сообщения об ошибках и другие существенные данные.

Другие выходные файлы LaTeX'а содержат информацию о перекрестных ссылках (расширение `.aux`), оглавлениях (расширение `.toc`), списках иллюстраций (расширение `.lof`) и таблиц (расширение `.lot`). Они используются в последующей работе LaTeX'а при создании специальных служебных деталей документа.

Файл с расширением `.idx` содержит все термины, включенные в указатель. Они могут быть упорядочены программой сортировки, подобной программе *MakeIndex*, которую написали Пехон Чжень и Майкл Харрисон. *MakeIndex* читает `.idx`-файл, содержащий термины, включенные в указатель, и соответствующие номера страниц, сортирует эти термины, объединяет их и записывает их во входной файл LaTeX'а (расширение `.ind`). Информация об оформительских особенностях указателя может быть определена в файле с расширением `.ist`.

Сообщения, создаваемые программой MakeIndex, записываются в `.ilg`-файл (подробнее см. гл. 12).

BibTeX (см. гл. 13) — это программа, которую написал Орен Паташник для подготовки списков литературы. BibTeX управляет базами данных ссылок, собранных в `.bib`-файлах. LaTeX записывает информацию о ссылках, встречающихся в документе, в файл или файлы с расширением `.aux`; последние затем читаются BibTeX'ом, который создает упорядоченный по алфавиту список литературы в `.bbl`-файле. `.bbl`-файл используется в дальнейшей работе LaTeX'а. Способ сортировки и формат ссылок на литературу определяются библиографическими стилями, *bibliography styles*, установленными в файлах с расширением `.bst`. *Сообщения*, сгенерированные BibTeX'ом, записываются в `.blg`-файл.

1.2.2 Выходные процессоры (драйверы dvi)

Как только документ будет успешно обработан TeX'ом, вы, вероятно, захотите *просмотреть результат. Есть несколько возможностей, а именно:*

- Получить высококачественную (больше, чем 1000 точек на дюйм) распечатку. В этом случае `.dvi` -файл переводится в точки на пленку с помощью драйвера или PostScript'а.
- Получить распечатку среднего качества (300 точек на дюйм). Могут быть использованы различные `dvi` -драйверы, пригодные для данного разрешения. Все чаще и чаще для этого используется PostScript.
- Содержание (текст и графика) может быть просмотрено на графическом экране компьютера, при этом используются утилиты, подобные `xdvi` (или X Window System). И опять превьюеры PostScript'а могут использовать последний язык для просмотра информации в документе.
- Содержание можно увидеть на «немом» терминале. В этом случае графика не будет показана, однако текст будет иметь достаточно узкий формат, чтобы выглядеть красиво на экране с шириной колонки менее 80. Этот вид документации часто используется для описаний узловых компьютерных программ с кодом.

Из всего, что было сказано выше, становится ясно, что язык PostScript играет важную роль в процессе визуализации документов. Исходники METAFONT (`.mf`) существуют для всех LaTeX'овских шрифтов и шрифтов Computer Modern, следовательно, bitmap-образы (`.pk`) могут быть сгенерированы для любого типа принтера. PostScript 1-аналоги шрифтов Computer Modern, LaTeX'овских, шрифтов $\mathcal{AMS}$ (Американского математического общества) и эйлеровских шрифтов также существуют, они являются коммерческими и их можно получить у Blue Sky Research и Y&Y. Более того, Adobe и другие разработчики шрифтов предлагают широкий выбор шрифтов, которые могут быть использованы в LaTeX'е

(см. разд. 11.9). Это означает, что вы сами можете решить, какой способ представления информации в вашем документе является лучшим и наиболее привлекательным.

1.3 Концепция общей разметки

1.3.1 Что такое общая разметка?

Изначально разметкой назывались комментарии технического редактора к рукописи, которые поясняли наборщику, как отформатировать документ. Они состояли из рукописных замечаний, например: *«Набрать этот заголовок курсивным шрифтом Helvetica 12 пунктов, в тексте шрифт 10 пунктов, выключная строка на шпон 22 пик, слева отступы 1 ет, справа без отступов».*

С появлением компьютеров стало возможным закодировать эти замечания в электронном виде, используя специальную систему кодирования. Каждый фотонаборный автомат имел свой собственный «язык», называемый языком разметки, по аналогии со старой ручной системой. Пока многие полиграфические предприятия обеспечивали также «клавиатурный» сервис, источник разметки в набираемых документах был всегда одним и тем же, и проблема совместимости не возникала. Когда же авторы и клиенты полиграфических фирм начали набирать свои собственные рукописи, возникли проблемы. Они могли набирать, только если хорошо знали формат разметки, принятый в типографии. Если же они набирали в своей собственной системе, было вполне вероятно, что формат разметки их документа будет несовместим с типографским.

Ситуация только ухудшилась, когда стали использоваться компьютеры для подготовки документов. Так же, как и в случае с фотонаборными автоматами, документы были закодированы с помощью *специфических* команд разметки. Это были команды форматирования нижнего уровня, такие как «возврат каретки», «отцентрировать следующий текст» и «перейти на следующую страницу». Документ, содержащий следующую специфическую разметку (SCRIPT [96]):

```
.pa ;.sp 2 ;.ce ;.bd
Title of chapter
.sp
```

может быть конвертирован в другую наборную систему с большим трудом. Другой пример специфической разметки (plain TEX) выглядит следующим образом:

```
\vfill\eject\begingroup\bf\obeylines\vskip 20pt
\hfil TITLE OF CHAPTER
\vskip 10pt\endgroup\bigskip
```

Началось движение по разработке стандартного языка разметки, который мог бы быть принят всеми наборными машинами в качестве исходного. Предполагалось, что задачей наборщика станет перевод этого языка на язык его собственной фотонаборной машины. Этот язык является языком общей разметки. Общая разметка

означает добавление информации в текст с указанием логических компонентов документа, таких как абзацы, заголовки и сноски. Форматирование (визуальное представление) вместе с его составляющими отделяется от его функции (позиции) в (иерархической) структуре текста.

LaTeX является в большой степени примером языка общей разметки (generic markup language — GML). Благодаря введенному механизму класса визуальный стиль различных элементов документа описан в одном месте за пределами самого документа.

Основываясь на исследовательской работе по языку GML Чарльза Голдфарба, а также принимая во внимание идеи системы Scribe Брайена Рейда, Международная организация стандартов разработала стандарт SGML (Standard Generalized Markup Language — стандартный язык общей разметки) (ISO 8879), который был опубликован в 1986 г. SGML — это язык разметки для представления документов в формате, пригодном для взаимного обмена. SGML предназначен для «публикаций в широком смысле этого слова, начиная от обычных и заканчивая мультимедийными с базами данных. SGML также может быть использован при обработке служебных документов, когда требуются удобочитаемость и взаимный обмен между издательскими системами» [99]. Для ознакомления с языком SGML см. [62, 83]. SGML является метаязыком. Это означает, что он точно определяет те правила, с помощью которых можно создать бесконечное число языков разметки. SGML сам по себе не имеет дела с форматированием размеченных документов, т. е. в нем *нет* команд разметки, таких как «новая страница», «новая строка» или «линейка». Наоборот, эти команды заложены в определенных составляющих форматирования. Например, в неком стиле заголовок первого уровня может либо начинаться с новой страницы, либо отделяться линейкой от тела текста.

Как практическое средство набора, созданное ученым, LaTeX сочетает в себе и уравновешивает преимущества разметки на высоком уровне в духе SGML со спецификой локальной разметки. Механизм классификации файлов позволяет получать один и тот же документ в различных вариантах, в зависимости от стиля макетирования, при этом достаточно возможностей для получения высшего качества печати.

1.3.2 Преимущества общей разметки

Использование последовательной разметки на протяжении всего документа помогает читателю понимать различную информацию, связывая ее визуально с тем или иным компонентом. Это также позволяет использовать документ для того, чтобы представить информацию on-line, а также упрощает автоматический поиск информации с помощью определенных ключевых слов.

Следует также иметь в виду, что полиграфия — творческое занятие, требующее опыта и мастерства, что редко можно встретить у человека, который мало занимался созданием оригинал-макетов. Поэтому лучше предоставить разработку нового стиля специалистам по дизайну, а случайным пользователям желательно

ограничиться *небольшими* и *последовательными* изменениями уже существующего стиля. Особенно нужно позаботиться о том, чтобы не нарушить едва различимый видимый баланс между отдельными компонентами документа.

1.3.3 Разделение содержания и формы

Чтобы иметь гарантию того, что все логические элементы текста получат одинаковую полиграфическую обработку по всему документу, вы должны дать определение новым элементам документа в общем виде в преамбуле. Это позволит вам быть уверенным в том, что одни и те же детали документа будут представлены одинаково.

Например, при создании руководства пользователя вы можете захотеть, чтобы каждая команда была выделена определенным шрифтом и автоматически попадала в указатель. Или можете захотеть, чтобы описание какой-либо команды с ее параметрами всегда было на затененном фоне. Вы можете даже подумать о создании определенной формы для табличного материала `tabular`. Следовательно, лучше всего определить стиль для окружения `tabular` в преамбуле документа. То же самое касается перечней, заголовков и т. д. Определения и возможные поздние изменения нужно собрать в одном месте в преамбуле, и они будут распределены автоматически по всему документу.

Другое преимущество использования общих команд заключается в том, что очень просто обратиться к другому стилю, нужно просто выбрать другой класс в команде `\documentclass` или определить дополнительные опции в командах `\documentclass` и `\usepackage` (см. разд. 2.1).

1.4 Необходимость локальной разметки

Несмотря на все описанные выше преимущества общей разметки, при окончательной подготовке документа вам иногда нужно будет подавить некоторые команды TEX'а.

Примером может служить воспроизведение данных в окружении `tabular`, где ясность разметки данных в таблице может быть существенна для лучшего понимания. Более того, когда окончательная версия документа готова, часто оказывается, что автор должен вставить строку, и страница разрывается в каких-либо стратегических местах.

1.4.1 Недостатки локальной разметки

Как уже говорилось, использование определенной разметки внутри текста документа — путь не очень хороший. Гораздо лучше определить новое окружение LATEX'а, например, `Ctab` для окружения `tabular` с центрированием, если это требуется постоянно. Точно так же следует ограничить локальные изменения шрифта внутри данного окружения, а общие принципы использования данного шриф-

та должны быть формализованы посредством определения новых окружения и (или) команды. Другими словами, любое изменение в представлении результатов вашего материала заканчивается большим количеством ручной работы — вы должны найти и исправить каждую деталь разметки.

Не следует также прибегать к общим структурным командам для достижения данного визуального эффекта. Например, команды секционирования, такие как `\paragraph` *не* должны использоваться для получения нескольких первых слов в абзаце полужирным шрифтом. Команды секционирования имеют структурную функцию внутри документа (см. разд. 2.3.1), и использование их при локальной разметке может быть чревато сюрпризами, когда вы используете разные инструменты в одних и тех же структурах классов — например, некоторым из ваших абзацев предшествует нумерация. Лучше, чтобы достичь желаемого эффекта, определить команду `\Boldtext` или `\Boldpar`.

1.4.2 Когда использовать локальную разметку

При окончательной подготовке документа вы обнаружите, что часто возникает необходимость локального вмешательства на микроуровне, чтобы получить определенный визуальный эффект. Все же всегда предпочтительнее спрятать как можно больше локальной разметки в общую. Вы можете сделать это, например, создав в общих командах средства для проверки пробелов на странице, для подсчета ширины отдельных строк текста и т. д.

ГЛАВА 2

Структура документа, подготовленного в LaTeX'e

Как объяснялось ранее, необходимо разделять форму и структуру документа. В этой главе мы покажем, как этот общий принцип воплощен в LaTeX'e.

Первый раздел этой главы показывает, как классы документов, пакеты, опции и команды преамбулы могут повлиять на структуру и макет документа[1]. Логическая разбивка документа на разделы объяснена в целом, предваряя более детальный рассказ о том, как команды разбивки (секционирования) и их аргументы определяют иерархическую структуру, как генерируют номера заголовков и обеспечивают появление верхних и нижних колонтитулов. На примерах показаны разные способы набора заголовков разделов. Показано также, как управлять информацией, записываемой в оглавлении, и как оформить оглавление, равно как и списки таблиц и рисунков. В последнем разделе представлены команды LaTeX'a управления перекрестными ссылками и их возможности.

2.1 Структура исходного файла

Вы можете использовать LaTeX в разных целях, например, для написания статьи или письма или для получения демонстрационных слайдов. Ясно, что для разных документов требуются разные логические структуры, т. е. разные команды и окружения. Мы говорим, что документ принадлежит к *классу* документов, имеющих одинаковую общую структуру (но не обязательно одинаковое полигра-

[1] По сравнению с пакетом LaTeX 2.09, в пакете LaTeX 2_ε общая структура исходного файла преобразована и увеличена. Старые документы будут обработаны автоматически с использованием режима совместимости.

```
\documentclass[twocolumn,a4paper]{article}
\usepackage{multicol}
\usepackage[german,french]{babel}
\addtolength{\textheight}{2cm}
\begin{document}
```

Рис. 2.1. Пример преамбулы документа

Класс этого документа — article. Макет соответствует требованию сформатировать в 2 колонки twocolumn и опции a4paper (формат листа бумаги A4). Первая декларация \usepackage говорит LaTeX'у, что этот документ содержит команды и структуры, обеспечиваемые пакетом multicol. Мы также используем пакет babel с опциями german (поддержка немецкого языка) и french (поддержка французского языка). Наконец, для этого документа высота тела текста, задаваемая по умолчанию, была увеличена на 2 см.

фическое исполнение). Вы задаете класс, к которому относится ваш документ, с помощью запуска LaTeX'овского файла с командой \documentclass, где обязательный параметр дает название *класса документа*. Класс документа определяет возможные логические команды и окружения (например, \chapter в классе report) так же, как и форматирование по умолчанию для этих элементов. Факультативный аргумент позволяет вам модифицировать форматирование этих элементов, снабжая списком *опций класса*. Например, опция 11pt распознается большинством классов документов и дает указание LaTeX'у выбрать шрифт величиной 11 пунктов в качестве основного.

Многие команды LaTeX'а, описанные в этой книге, не являются специфическими для одного класса, а могут использоваться в нескольких классах. Собрание этих команд называется пакетом, и вы информируете LaTeX об использовании определенных пакетов в вашем документе путем размещения одной или нескольких команд \usepackage после команды \documentclass.

Подобно команде \documentclass, команда \usepackage имеет обязательный аргумент, являющийся названием пакета, и факультативный аргумент, который может содержать список *опций пакета*, модифицирующих работу пакета.

Классы документов и пакеты реализованы во внешних файлах с расширениями соответственно .cls и .sty[2]. Код для опций класса иногда сохраняется в файлах (в этом случае с расширением .clo), но обычно определяется непосредственно в файле класса или пакета. Для более подробной информации об этих опциях см. приложение A. Как бы то ни было в случае с опциями, имя файла может отличаться от имени опции, например, опция 11pt может относиться

[2] Стилевые файлы в пакете LaTeX 2.09 имели расширение .sty. Так как почти все такие файлы LaTeX'а 2.09 могут быть использованы без изменений в качестве пакетов в LaTeX 2_ε, для пакетов в LaTeX 2_ε было выбрано расширением .sty.

к файлу `art11.clo`, если она используется в классе article, и к файлу `bk11.clo` внутри класса book.

Команды, заключенные между командами `\documentclass` и `\begin{document}`,—это так называемая *преамбула документа*. Все параметры стиля должны быть определены в этих пределах, в пакете, классе или непосредственно в вашем документе *перед* командой `\begin{document}`, которая устанавливает приоритеты для некоторых глобальных параметров; пример показан на рис. 2.1 на развороте.

Как правило, нестандартный пакет LaTeX'овских файлов содержит общие модификации или улучшения[3], по сравнению со стандартным LaTeX'ом, в то время как команды в преамбуле определяют изменения текущего документа.

Итак, если вы хотите изменить внешний вид вашего документа, у вас есть несколько возможностей:

- изменить стандартный набор параметров в файле класса с помощью опций этого класса;
- добавить один или более пакетов в ваш документ и использовать их;
- изменить стандартный набор параметров в файле пакета с помощью опций этого пакета;
- определить свои собственные пакеты, содержащие особый набор параметров, и загрузить их с помощью команды `\usepackage` после пакета или класса, которые должны быть модифицированы (это объясняется в следующем разделе) и
- сделать последние урегулирования внутри преамбулы.

А если вы хотите проникнуть глубже в недра LaTeX'а, вы, конечно, можете разработать свои собственные общие пакеты и управлять ими с помощью опций. Дополнительную информацию об этом вы найдете в приложении А.

2.1.1 Обработка опций и пакетов

Алгоритм, используемый в LaTeX2_ε для обработки опций в командах `\documentclass` и `\usepackage`, является более мощным, чем механизм, управляющий опциями команды `\documentstyle` в LaTeX'е 2.09. Существует четкое разграничение между декларированными опциями класса или пакета и файлами пакетов для общих целей[4]. Последние должны быть определены с помощью

[3] Многие из них стали стандартом de facto и описаны в этой книге. Это, однако, не означает, что пакеты, не описанные в этой книге, менее важны или полезны и что они хуже качеством и не должны быть использованы. Мы просто сосредоточились на нескольких наиболее известных пакетах, а для остальных решили объяснить, какие функции возможны в данной области.

[4] В LaTeX'е 2.09 опции команды `\documentstyle` представляли собой смесь декларированных (непосредственно осуществляемых) и недекларированных опций (которые реализуются в файле с расширением `.sty` после того, как закончена работа команды `\documentstyle`). В LaTeX2_ε это сейчас четко разграничено.

команды \usepackage. Эти опции принадлежат целому документу (если они используются в команде \documentclass), либо отдельным пакетам (если они определены в команде \usepackage).

Вы можете определить опции в команде \usepackage, если эти опции декларированы пакетом. В противном случае вы получите сообщение об ошибке, информирующее вас о том, что данная опция неизвестна. Опции команды \documentclass обрабатываются несколько по-другому. Если определенная опция не декларирована классом, она будет принята за «глобальную опцию».

Все опции команды \documentclass (декларированные и глобальные) автоматически передаются в качестве опций класса всем декларациям команды \usepackage. Таким образом, если файл пакета, загружаемый с декларацией команды \usepackage, распознает (т. е. декларирует) некоторые из опций класса, он может производить присваивающие действия; в противном случае класс опций при обработке пакета будет проигнорирован. Поскольку все опции должны быть определены внутри файла класса или пакета, их действие контролируется классом или пакетом (можно установить внутренние средства для прочтения внешнего файла). По этой причине их порядок в факультативном аргументе команд \documentclass или \usepackage нерелевантен.

Если вы хотите использовать несколько пакетов с одинаковыми опциями (например, никакими), можно загрузить их все одной командой \usepackage путем определения имен пакета, отделяя их запятой в обязательном аргументе. Например,

```
\usepackage[german]{babel}
\usepackage[german]{varioref}
\usepackage{multicol}
\usepackage{epic}
```

можно записать в таком виде:

```
\usepackage[german]{babel,varioref}
\usepackage{multicol,epic}
```

Определив german как глобальную опцию, получаем сокращенный вариант:

```
\documentclass[german]{book}
\usepackage{babel,varioref,multicol,epic}
```

после чего команда german будет передана во все определенные пакеты и, следовательно, будет обработана теми пакетами, которые ее декларировали.

В конечном счете, когда мы доходим до команды \begin{document}, все глобальные опции проверены, чтобы убедиться, что они были использованы в каком-либо пакете; если нет, вы получите предупреждающее сообщение. Причиной обычно является орфографическая ошибка в названии опции или удаление команды \usepackage, загружающей пакет, использующий эту опцию.

Поэтому если вы хотите внести некоторые изменения в класс документов или пакет (например, изменить параметры или переопределить некоторые команды), вы должны поместить соответствующий код в отдельный файл с расширением `.sty`. Затем загрузите этот файл командой `\usepackage` после пакета, работу которого вы хотите модифицировать (или класса, если изменения касаются классов). Такой файл локального пакета должен содержать одну специальную декларацию в начале, указывающую дистрибутив или релиз LaTeX2ε, а именно:

`\NeedsTeXFormat{`*format*`}[`*release*`]`

В качестве формата нужно указать `LaTeX2e`. Если определен факультативный аргумент *release*, он должен содержать дату обновления вашего дистрибутива LaTeX2ε в последовательности год — месяц — число (YYYY/MM/DD). Например, команда

```
\NeedsTeXFormat{LaTeX2e}[1994/02/01]
```

задает версию LaTeX2ε, обновленную 1 февраля 1994 г. (LaTeX2ε обновляется дважды в год в определенные числа.) Цель этой команды — обеспечить защиту от устаревших версий LaTeX2ε. Если, например, в вашем пакете используется команда, которая в версии от 01.02.1994 имела недостатки, а затем была исправлена в версии от 01.08.1994, то факультативный аргумент `\NeedsTeXFormat` предупредит вас о том, что ваш пакет используется в старой версии LaTeX2ε. Более новая дата обновления принимается без предупреждения.

Другой способ модифицировать макет заключается в установке соответствующего кода непосредственно в преамбуле вашего документа. Однако есть одно важное TeXническое различие [$\mathcal{L}$ **150**], [$\mathcal{A}$ **268**] между командами внутри файлов класса или пакета и командами в преамбуле: у LaTeX'а есть много *внутренних* команд, которые вы не можете использовать в своем документе без специальных мер предосторожности. Имена этих внутренних команд содержат знак @. Его нужно переименовать, так как знаки, не являющиеся буквами, обычно не присутствуют в имени LaTeX'овских команд. (За исключением двух десятков команд, чьи имена представляют собой символ бэкслеш (\) и следующий за ним один небуквенный знак, все остальные команды состоят из символа \, сопровождающегося одной или несколькими буквами.) Теперь, когда команды в файле пакета исполняются, LaTeX рассматривает знак @ как букву. Это позволяет пакетным файлам беспрепятственно работать с этими внутренними командами, в то время как они не могут быть использованы в этом качестве в преамбуле документа.

Несмотря на это, вы можете работать с внутренними командами в преамбуле, ограничив область, где @ будет считаться буквой внутри команд `\makeatletter` и `\makeatother`. В качестве примера рассмотрим команду, которая приказывает LaTeX'у установить новый счетчик, тогда как другой счетчик продолжает наращивать свои значения (внутренняя команда `\@addtoreset`). Таким образом, при использовании класса `article` вы можете нумеровать уравнения внутри разделов с помощью кода, указанного ниже в вашей преамбуле.

```
\documentclass{article}
...
\makeatletter   % '@' теперь обычная "буква" для TeX'а
\@addtoreset{equation}{section}
\makeatother    % '@' запоминается как "небуква" для TeX'а
\begin{document}
   ......
```

Как объяснялось ранее, внутри файла пакета знак @ считается обычной буквой и может быть использован в имени команды. Поэтому вы должны быть осторожны и *никогда* не использовать команды `\makeatletter` и `\makeatother` внутри пакетных файлов.

2.1.2 Разделение исходного файла на части

Исходные документы LaTeX'а можно удобным образом разбить на несколько частей с помощью команд `\include`. Более того, документы могут быть частично переформатированы путем определения аргументов команды `\includeonly` [$\mathcal{L}$ **76,188**] (см. также [40, с. 78]) только для тех файлов LaTeX'а, которые нужно заново обработать. Для остальных файлов, указанных в `\include`, нумеруемая информация (страница, глава, таблица, рисунок, уравнение и т. д.) будет считана из соответствующих файлов с расширением `.aux`, которые получаются после предыдущей обработки. Например, в случае, показанном на рис. 2.2, пользователь хочет заново обработать только файлы `chap1.tex` или `appen1.tex`.

Имейте в виду, что LaTeX только выдает предупреждение типа «Нет файла xxx.tex», когда он не может найти файл, определенный в декларации `\include`, *но не выдает сообщение об ошибке и продолжает работу.*

Если информация в файлах с расширением `.aux` отвечает новым требованиям, можно обработать только часть документа и получить все счетчики, перекрестные ссылки и страницы в правильном виде в переформатированной ча-

```
\documentclass{book}       % класс документа ``book''
\includeonly{chap1,appen1} % включить только chap1 и appen1
\begin{document}
\include{chap1}            % ввести chap1.tex
\include{chap2}            % ввести chap2.tex
\include{chap3}            % ввести chap3.tex
\include{appen1}           % ввести appen1.tex
\include{appen2}           % ввести appen2.tex
\end{document}
```

Рис. 2.2. Структурирование документа, подготовленного в LaTeX'е

сти. Однако, если один из счетчиков (включая номера страниц для перекрестных ссылок) меняется в заново обработанной части, документ целиком должен быть перекомпилирован, чтобы получить правильные указатель, оглавление и библиографические ссылки.

Таким образом, в тех случаях, когда целесообразно разбить большой документ на мелкие части и работать с отдельными файлами с помощью текстового редактора, в промежуточной стадии работы над одной или более главами частичное переформатирование должно осуществляться с большой осторожностью. Когда же требуется последняя и полная правильная копия, единственный безопасный путь — это новая обработка целого документа. Если документ слишком большой, чтобы обработать его за один сеанс, его можно разбить на части и обработать частями. Однако в этом случае обработку следует производить в *правильной последовательности*, чтобы убедиться в правильности перекрестных ссылок и номеров страниц.

2.1.3 Комбинирование нескольких файлов

Посылая кому-нибудь документ в LaTeX'е, вам может понадобиться послать локальные или нестандартные пакетные файлы (например, ваши частные модификации некоторых пакетов) с исходным файлом. В таких случаях часто бывает полезно поместить всю информацию, требующуюся для обработки документа, в один файл.

Для этого LaTeX предлагает окружение `filecontents`. Это окружение берет один аргумент, имя файла; его тело будет состоять из содержимого этого файла. Оно может появляться только перед декларацией `\documentclass`.

Если LaTeX столкнется с таким окружением, он проверит, может ли он найти упомянутое имя файла. Если нет, он запишет тело окружения дословно в файл текущей директории и сообщит вам об этом действии. С другой стороны, если файл с таким именем будет обнаружен LaTeX'ом, он проинформирует вас о том, что он проигнорировал этот пример окружения `filecontents`, так как этот файл уже предоставлен.

Для получения списка всех или почти всех файлов, использованных в вашем документе, задайте команду `\listfiles` в преамбуле.

2.2 Логическая структура

Стандартные классы LaTeX'а содержат команды и окружения для определения разных иерархических структурных единиц документа [$\mathcal{L}$ **23,157**], [$\mathcal{M}$ **152**] (т. е. главы, разделы, приложения). Каждая такая команда определяет уровень вложенности внутри иерархии и каждую структурную единицу, принадлежащую тому или иному уровню.

Типичный документ (например, статья) состоит из заглавия, нескольких разделов, подразделяющихся, возможно, на более мелкие разделы, и списка литера-

```
\documentclass{article}  % стандартный класс ''article''
\begin{document}
\maketitle
\section{...}
\section{...}
   \subsection{...}
        \subsubsection{...}
\section{...}
\begin{thebibliography}  ...  \end{thebibliography}
\end{document}
```

Рис. 2.3. Иерархическая структура простого LaTeX'овского документа
Этот пример показывает вложенность структуры документа в LaTeX'e. В случае класса article команда `\chapter` недоступна.

туры. Для описания такой структуры используются команда, генерирующая заголовки `\maketitle`, команды секционирования `\section` и `\subsection` и окружение `thebibliography` (рис. 2.3). Команды должны быть правильно вложены. Например, команда `\subsection` должна применяться только после соответствующей команды `\section`.

Более длинные работы (такие как доклады, руководства и книги), имеющие более сложный титул, разбитые на главы и части, обеспечивают перекрестной информацией (оглавлением, списком иллюстраций, списком таблиц и указателем) и иногда имеют приложения. В таком документе вы можете легко различить *вступительную часть*, *тело* и *заключительную часть* (рис. 2.4).

Во вступительной части обычно используется так называемый вариант «со звездочкой» [$\mathcal{L}$ **157**], [$\mathcal{A}$ **153**] команды секционирования `\section`. Этот вариант подавляет нумерацию заголовков. Разделы с фиксированными именами, такие как Введение, Указатель и Предисловие, обычно не нумеруются. В стандартных классах команды `\tableofcontents`, `\listoftables`, `\listoffigures` и окружения `theindex` и `thebibliography` вызывают изнутри команду (`\section` или `\chapter`), используя свой вариант со звездочкой.

2.3 Команды секционирования

Стандартный LaTeX обеспечивает работу команд секционирования, показанных в табл. 2.1. Команда `\chapter` определяет нулевой уровень иерархической структуры документа, команда `\section` — первый уровень и так далее, до факультативной команды `\part`, которая определяет уровень -1 (или нулевой в классах, где не определена команда `\chapter`). Не все из этих команд определены во всех клас-

```
\documentclass{book} % стандартный класс ''book''
\begin{document}
%--------------------- вступительная часть документа
\maketitle
   \section*{...}    % название раздела, например, ''Предисловие''
\tableofcontents     % оглавление
\listoffigures       % список иллюстраций
\listoftables        % список таблиц
%--------------------- тело документа
\part{...}
\chapter{...}
   \section{...}
\chapter{...}
\part{...}
%--------------------- заключительная часть документа
\appendix            % приложение, разбитое на главы
\chapter{...}
\chapter{...}
\begin{thebibliography}  \end{thebibliography}
\begin{theindex}         \end{theindex}
\end{document}
```

Рис. 2.4. Иерархическая структура сложного LaTeX'овского документа
Показанный здесь более сложный пример подразделяет документ на вступительную часть, тело и заключительную часть. Каждый из них содержит более мелкие элементы документа, такие как оглавление во вступительной части, глава, разделы и подразделы в теле и приложения, указатель и библиографию в заключительной части.

сах документов: в классе article нет команды \chapter, а класс letter вообще не поддерживает команды секционирования. В пакетах можно также определять дополнительные команды секционирования, используя либо дополнительные уровни, либо варианты уже существующих уровней.

Обычно команды секционирования автоматически выполняют одно или несколько из изложенных ниже полиграфических действий:

- создают нумерацию заголовков, отражающую иерархический порядок;
- накапливают заголовки для размещения их в оглавлении (в файле с расширением .toc);
- сохраняют содержание заголовков для возможного использования в верхнем и/или нижнем колонтитулах;
- форматируют заголовки.

\part (book и report)	уровень -1	\part (article)	уровень 0
\chapter	уровень 0	\section	уровень 1
\subsection	уровень 2	\subsubsection	уровень 3
\paragraph	уровень 4	\subparagraph	уровень 5

Таблица 2.1. Стандартные команды секционирования LaTeX'а

вид	*нумерация*	.toc	*в/н колонтитул*
\section{*title*}	да	*title*	*title*
\section[*toc_entry*]{*title*}	да	*toc_entry*	*toc_entry*
\section*{*title*}	нет	нет	нет

Таблица 2.2. Синтаксис команд секционирования

Все команды секционирования имеют общий синтаксис [$\mathcal{L}$ **23,157**], [$\mathcal{A}$ **272–3**], как показано в табл. 2.2. Команда, помеченная звездочкой (например, \section*{...}), подавляет нумерацию заголовка. Факультативный аргумент используется, когда текст в оглавлении и в верхнем и/или нижнем колонтитулах отличается от напечатанного заголовка.

Остальная часть этого раздела посвящена обсуждению того, как усовершенствовать появление заглавий разделов. Вы узнаете, как определять команду, подобную команде \section, которая имеет упомянутый выше синтаксис, создает вхождение в оглавление, если это нужно, но руководствуется своими строгими правилами относительно текста заголовков или использует *курсивный* шрифт обычного размера вместо **полужирного** шрифта увеличенного размера.

Вначале приводится несколько примеров, показывающих, как можно менять нумерацию заголовков. Дальнейшие примеры показывают, как вводить информацию о заголовках в оглавление. И наконец, обсуждаются изменения, касающиеся общего вывода заголовков, и демонстрируются возможности LaTeX'а в области их определения.

2.3.1 Нумерация заголовков

Для поддержки нумерации LaTeX использует счетчик для каждой единицы секционирования и создает нумерацию заголовков из этих счетчиков.

Возможные желаемые изменения, касающиеся нумерации заголовков, чаще всего связаны с изменением уровня вложенности, для которого должен быть создан номер. Это управляется счетчиком под названием `secnumdepth` [$\mathcal{L}$ **157,160**], [$\mathcal{A}$ **272–3**], который отвечает за высший уровень с пронумерованными заголовками. Например, некоторые документы не содержат пронумерованных заголовков. Вместо того чтобы каждый раз использовать вариант «со звездочкой» команд

```
\newcounter{part}                        % (-1) части
\newcounter{chapter}                     % (0)  главы
\newcounter{section}[chapter]            % (1)  разделы
\newcounter{subsection}[section]         % (2)  подразделы
\newcounter{subsubsection}[subsection]% (3)  подподразделы
\newcounter{paragraph}[subsubsection] % (4)  пункты
\newcounter{subparagraph}[paragraph]  % (5)  подпункты
```

Рис. 2.5. Нумерация заголовков раздела

секционирования, удобнее счетчик secnumdepth установить в -2 в преамбуле документа. Преимущества этого метода заключаются в том, что может быть создано вхождение в оглавление и аргументы из команд секционирования могут создавать информацию в колонтитулах. Как было сказано выше, эти черты в отмеченном звездочкой варианте подавляются.

Чтобы пронумеровать все заголовки вплоть до \subparagraph или до низшего уровня в данном классе, достаточно написать следующее:

```
\setcounter{secnumdepth}{10}
```

В конечном счете использование команды \addtocounter дает легкий способ нумерации большего или меньшего числа уровней, позволяя не беспокоиться по поводу уровневых номеров соответствующих команд секционирования. Например, если вам понадобится еще один уровень нумерации, вы можете просто добавить

```
\addtocounter{secnumdepth}{1}
```

в преамбуле вашего документа.

Каждая команда секционирования имеет связанный с ней счетчик, который по договоренности называется так же, как команда секционирования (например, команда \subsection осуществляется вместе со счетчиком subsection). Этот счетчик содержит текущий номер для данной команды секционирования. Так, в классе report команды \chapter, \section, \subsection и т. д. представляют иерархическую структуру документа, а счетчик (например, subsection) следит за нумерацией подразделов (\subsection), используемых внутри текущего раздела (\section). Обычно когда работает счетчик на данном иерархическом уровне, все счетчики низшего уровня (т. е. те, у которых номера больше) перезагружаются. Так, например, файл класса report содержит определения, показанные на рис. 2.5.

Эти команды взаимодействуют с разными счетчиками. Счетчик первого уровня (раздела) перезагружается, когда работает счетчик нулевого уровня (главы), и так же счетчик второго уровня (подраздела) перезагружается, когда работает счетчик первого уровня (раздела). Тот же механизм действует по отношению к

команде \subparagraph. Обратите внимание, что в стандартных классах счетчик частей (part) полностью отделен от других счетчиков, и на него не влияют команды секционирования низшего уровня. Это означает, что главы в классах book или report или разделы в классе article будут пронумерованы последовательно, даже если вторгается команда \part. Изменить это просто, вам нужно только заменить соответствующее определение счетчика глав, например:

```
\newcounter{chapter}[part]
```

Поведение уже существующего счетчика может быть изменено с помощью команды \@addtoreset, например:

```
\@addtoreset{chapter}{part}
```

Помните, что эта инструкция может быть использована только внутри некого вспомогательного файла или в преамбуле документа между командами \makeatletter и \makeatother как объяснялось в разд. 2.1.1.

Каждый счетчик в LaTeX'e, включая счетчики секционирования, имеет связанную с ним команду, созданную путем присоединения к имени счетчика префикса \the [$\mathcal{L}$ 92], который генерирует полиграфическое представление данного счетчика. В случае команд секционирования это представление используется для того, чтобы создать полный номер, связанный с командами, так, как показано в следующих определениях:

```
\renewcommand{\thechapter}{\arabic{chapter}}
\renewcommand{\thesection}{\thechapter.\arabic{section}}
\renewcommand{\thesubsection}{\thesection.\arabic{subsection}}
```

В приведенном выше примере команда \thesubsection создает нумерацию арабскими цифрами счетчика subsection с помощью команды \thesection и точки. Этот вид рекурсивного определения упрощает модификации представления счетчика, так как изменения нужно сделать только в одном месте. Если, например, вы хотите пронумеровать главы, используя прописные буквы, можете переопределить команду \thechapter:

```
\renewcommand{\thechapter}{\Alph{chapter}}
\section{Раздел выглядит иначе}
Благодаря определениям по умолчанию,
меняются не только номера глав, но команды
секционирования низшего уровня также выдают
такое представление в своей нумерации.
```

D.7 Раздел выглядит иначе

Благодаря определениям по умолчанию, меняются не только номера глав, но команды секционирования низшего уровня также выдают такое представление в своей нумерации.

Таким образом, меняя команды представления счетчика, можно изменить номер, демонстрируемый командой секционирования. Однако изображение номера не может быть изменено произвольно с помощью этого метода. Допустим, вы хо-

тите создать заголовок раздела с номером в рамочке. Судя по приведенному выше примеру, есть простой способ — переопределить команду \thesection, например:

```
\renewcommand{\thesection}{\fbox{\thechapter.\arabic{section}}}
```

Но это неправильно, что становится очевидным при попытке сделать ссылку на этот раздел.

4.7 Ошибка

Ссылка на раздел в этом формате приводит к забавному результату, который можно увидеть, взглянув на разд. 4.7. Мы получили ссылку в рамочке.

```
\renewcommand{\thesection}
    {\fbox{\thechapter.\arabic{section}}}
\section{Ошибка}\label{wrong}
Ссылка на раздел в этом формате приводит
к забавному результату, который можно
увидеть, взглянув на разд.~\ref{wrong}.
Мы получили ссылку в рамочке.
```

Иными словами, команды изображения счетчика также используются LaTeX'овским механизмом перекрестных ссылок (команды \label, \ref см. разд. 2.5). Таким образом, мы можем делать только небольшие изменения в командах изображения счетчика, так, чтобы их использование в команде \ref все еще имело смысл. Чтобы создать рамочку вокруг номера заголовка и при этом не нарушить действие команды \ref, нужно переопределить внутреннюю команду LaTeX'а \@seccntformat, которая отвечает за ту часть названия раздела, где находится номер. Определение по умолчанию \@seccntformat печатает команду \the изображения счетчика разделов (т. е. в приведенном выше примере он использует команду \thesection), за которой следует фиксированный горизонтальный пробел величиной 1 em. Чтобы уладить дело, перепишите приведенный выше пример следующим образом:

4.7 Поправка

Создание ссылки на раздел с использованием этого определения дает правильный результат — ссылку на разд. 4.7.

```
\makeatletter
\renewcommand{\@seccntformat}[1]{\fbox
  {\csname the#1\endcsname}\hspace{0.5em}}
\makeatother
\section{Поправка}\label{sec:OK}
Создание ссылки на раздел с использованием
этого определения дает правильный результат
"--- ссылку на разд.~\ref{sec:OK}.
```

Как видим, рамочка вокруг номера в заголовке раздела теперь определена командой \@seccntformat, и в результате ссылки получаются правильно[5]. Также имейте в виду, что мы сократили пробел между рамочкой и текстом до 0.5 em

[5] Команда \@seccntformat берет в качестве аргумента идентификатор уровня раздела, который присоединяется к префиксу \the, чтобы сгенерировать необходимую форму изображения с помощью команд \csname, \endcsname. В нашем примере команда \@seccntformat вызывается вместе с аргументом раздела (section), и таким образом генерируется замененный текст \fbox{\csname thesection\endcsname\hspace{0.5em}}. О команде \csname см. более подробно в [30].

(вместо заданного 1 em). Определение `\@seccntformat` применяется ко всем заголовкам, определенным с помощью команды `\@startsection`, которая описывается в следующем разделе. Таким образом, если вы хотите использовать разные определения команды `\@seccntformat` для разных заголовков, вы должны поставить соответствующий код в каждое определение заголовка.

2.3.2 Форматирование заголовков

В LaTeX'e есть общая команда `\@startsection`, которая может быть использована для определения широкого спектра вида заголовков. Чтобы определить или изменить команду секционирования, нужно выяснить, может ли это сделать команда `\@startsection`. Если этим путем нельзя достичь желаемого результата, можно использовать команду `\secdef` для порождения самых разных форматов секционирования.

Заголовки в основном делятся на 2 группы: заверстанные «в разрез» и «в подбор». Заголовком «в разрез» называется заголовок, отделенный вертикальным пробелом от предшествующего и последующего текста. Большинство заголовков в этой книге выглядит именно так.

Заголовок, заверстанный «в подбор», отделен по вертикали от предыдущего текста, но последующий текст располагается на той же строке, что и сам заголовок, и лишь отделяется от него горизонтальным пробелом.

Заголовки, заверстанные «в подбор». Настоящий пример показывает, как выглядит заголовок, заверстанный «в подбор». Текст абзаца, следующий за заголовком, продолжается на той же строке, что и заголовок.

```
\paragraph{Заголовки, заверстанные
"<в подбор">.}
Настоящий пример показывает, как выглядит
заголовок, заверстанный "<в подбор">.
Текст абзаца, следующий за заголовком,
продолжается на той же строке, что и
заголовок.
```

Команда `\@startsection`

Общая команда `\@startsection` позволяет определить оба типа заголовков. Ее синтаксис и описание аргумента выглядят следующим образом:

`\@startsection{`*name*`}{`*level*`}{`*indent*`}{`*beforeskip*`}{`*afterskip*`}{`*style*`}`

name Это имя определяемой команды секционирования без предшествующего бэкслеша — например, чтобы определить команду `\section`, вводится слово `section`.

level Это номер, обозначающий уровень команды секционирования. Этот уровень используется для того, чтобы решить, получает ли номер команда секционирования (если уровень меньше или равен `secnumdepth`, см. разд. 2.3.1), либо появляется в оглавлении (если значение меньше или равно `tocdepth`, см. разд. 2.4.1). Следовательно, он отражает позицию в иерархии команд

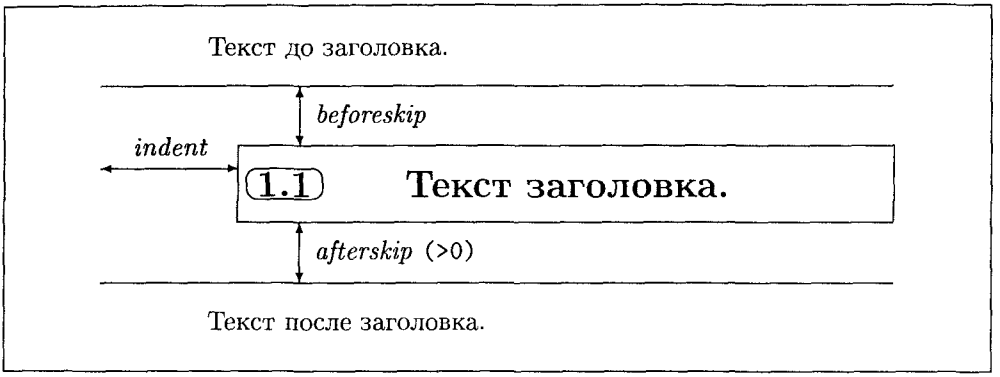


Рис. 2.6. Макет заголовка «в разрез»

секционирования, где наиболее удаленная команда секционирования имеет нулевой уровень[6].

indent Отступ от заголовка до левого поля. Если значение отрицательное, заголовок будет начинаться во внешнем поле. Если вы сделаете его положительным, отступ всех строк заголовка будет равен этому значению.

beforeskip Абсолютное значение этого параметра определяет пробел, который нужно оставить перед заголовком. Если параметр отрицательный, то абзацный отступ после заголовка подавляется. Это непостоянная величина, т. е. ее можно сжимать и растягивать. Имейте в виду, что LaTeX начинает новый абзац перед заголовком, поэтому к пробелу спереди добавляется значение `\parskip`.

afterskip Это пробел, который нужно оставить после заголовка. Это вертикальный пробел после заголовка «в разрез» или горизонтальный после заголовка «в подбор». Знак *afterskip* указывает на то, какой это заголовок: «в разрез» (*afterskip* ≥ 0) или «в подбор» (*afterskip* < 0). Имейте в виду, что в первом случае начинается новый абзац, поэтому к пробелу после заголовка добавляется значение `\parskip`. Неприятной стороной этого параметра является то, что невозможно определить заголовок «в разрез» с эффективным последующим пробелом меньше, чем `\parskip`, используя команду `\@startsection`. Когда вы пытаетесь компенсировать положительное значение `\parskip`, используя отрицательное *afterskip*, получается заголовок «в подбор».

style Это стиль текста заголовка. Этот аргумент может брать любую инструкцию, которая влияет на набор текста, например, команды `\Large`, `\bfseries` или `\raggedright` (см. примеры ниже).

Рис. 2.6 и 2.7 ниже показывают эти параметры графически для заголовков «в разрез» и заголовков «в подбор» соответственно.

[6] В классах book и report команда `\part` обычно имеет уровень -1 (см. рис. 2.5).

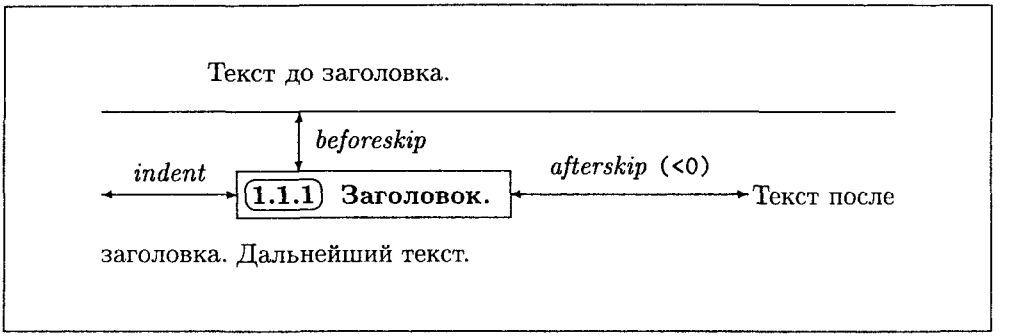


Рис. 2.7. Макет заголовка «в подбор»

Теперь мы покажем, как эти аргументы используются на практике для определения новых команд секционирования. Предположим, что вы хотите изменить команду `\section` в классе report, чтобы текст выглядел приблизительно так:

... предыдущий текст.

4.6.3 Пример заголовка «в разрез»

Заголовок набран курсивным шрифтом обычного размера и отделен от предыдущего текста на высоту одной строки набора. Последующий текст отделен на половину высоты строки набора.

```
\ldots\ предыдущий текст.
\subsection{Пример
заголовка "<в разрез">}
Заголовок набран курсивным шрифтом
обычного размера и отделен от
предыдущего текста на высоту одной
строки набора. Последующий текст отделен
на половину высоты строки набора.
```

В этом случае нужно сделать следующее переопределение:

```
\renewcommand{\subsection}{\@startsection
   {subsection}%                        % имя
   {2}%                                 % уровень
   {0mm}%                               % отступ
   {-\baselineskip}%                    % отбивка ''до''
   {0.5\baselineskip}%                  % отбивка ''после''
   {\normalfont\normalsize\itshape}}% % стиль
```

Первый аргумент — `subsection` — имя команды секционирования. В иерархии секционирования `subsection` находится на втором уровне. Третий аргумент — `0mm`, поскольку заголовок должен начинаться от левого поля. Абсолютное значение четвертого аргумента указывает, что перед заголовком должен быть оставлен пробел, равный высоте строки набора, и если параметр отрицательный, то абзацный отступ после заголовка должен быть подавлен. Абсолютное значение пятого аргумента (*afterskip*) указывает, что после заголовка должен быть оставлен пробел, равный половине высоты строки набора, и если параметр положительный, то должен быть создан заголовок «в разрез». И наконец, согласно шестому пара-

метру, заголовок должен быть набран курсивным шрифтом такого же размера, что и шрифт документа.

Необходимо заметить, что внутри определения каждый необязательный пробел был подавлен путем постановки знака процента после каждой правой фигурной скобки во втором аргументе команды `\renewcommand`. Помимо того что необязательные пробелы в определении занимают память, они часто вторгаются в документ и дают забавные результаты.

Другое оформление текста, которое часто встречается в художественной литературе, получается посредством следующего определения:

```
\renewcommand{\section}{\@startsection
    {section}%                          % имя
    {1}%                                % уровень
    {1em}%                              % отступ
    {\baselineskip}%                    % отбивка ''до''
    {-\fontdimen2\font                  % отбивка ''после''
      plus -\fontdimen3\font
      minus -\fontdimen4\font}%
    {\normalfont\normalsize\scshape}}% % стиль
```

Так определяется заголовок, заверстанный «в подбор» капителью. Определение пробела для горизонтальной отбивки «после» следует пояснить: это значение сжимаемого пробела между словами, взятое из текущего шрифта, уничтоженное для того, чтобы создать заголовок, заверстанный «в подбор». Подробности о `\fontdimen` можно найти в разд. 7.7.2. Результат будет примерно таким:

... предыдущий текст.

ЧЕЛОВЕК начал убегать от грузовика. Он видел, что его преследуют

```
\ldots\ предыдущий текст.
\section{Человек}
начал убегать от грузовика.
Он видел, что его преследуют
```

Конечно, чтобы текст выглядел таким образом, нужно отключить нумерацию заголовков, установив значение `secnumdepth` равным -1.

Какие команды могут быть использованы для установки стилей текстов заголовков с применением аргумента *style* команды `\@startsection`? Кроме директив изменения шрифтов (см. гл. 7) здесь можно использовать несколько инструкций. Команда `\centering` создает центрированный заголовок «в разрез», а команда `\raggedright` выравнивает текст по левому краю. Возможно использование команды `\raggedleft`, но она может дать странные результаты. Вы можете также использовать `\hrule\medskip`, `\newpage` или подобные команды, которые производят локальные изменения. Примеры на рис. 2.8 на следующей странице показывают результаты разных возможных определений для аргумента *style*.

В стандартных классах документов LaTeX'а слова в длинных заголовках можно переносить. Если это нежелательно, переносы можно отменить на локальном

4.6.3 Это заголовок раздела

Стиль определяется командами \centering\itshape

4.6.4 Это заголовок раздела

Стиль определяется командами \raggedright\itshape

4.6.5 Это заголовок раздела

Стиль определяется командами \raggedleft\itshape

4.6.6 Это заголовок раздела

Стиль определяется командами \hrule\medskip\itshape

```
\Csubsection{Это заголовок раздела}
Стиль определяется командами
\verb!\centering\itshape!

\Lsubsection{Это заголовок раздела}
Стиль определяется командами
\verb!\raggedright\itshape!

\Rsubsection{Это заголовок раздела}
Стиль определяется командами
\verb!\raggedleft\itshape!

\Hsubsection{Это заголовок раздела}
Стиль определяется командами
\verb!\hrule\medskip\itshape!
```

Рис. 2.8. Изменение стиля заголовка

Меняя аргумент *style* команды \@startsection, можно достичь разных эффектов.

уровне путем определения и использования команды \nohyphens[7] в части *style* команды \@startsection. Другая проблема заключается в том, что, когда пользователь пытается сказать TeX'у, как разбить заголовок на несколько строк, используя символ «~» или команду \\, могут возникнуть побочные эффекты при форматировании оглавления. В этом случае самым простым решением является повтор текста заголовка без особой разметки в факультативном параметре команды секционирования.

И наконец, несколько слов о подавлении абзацного отступа для первого абзаца после заголовка «в разрез». Стандартные классы документов LaTeX'а, следуя английской (американской) типографской традиции, в этом случае подавляют абзацный отступ. Все первые абзацы после заголовка «в разрез» могут иметь абзацный отступ, если использовать пакет indentfirst (Дэвид Карлайл).

Команда \secdef

Команды секционирования высшего уровня \part и \chapter создают свои заголовки без команды \@startsection. Подобным образом вы можете создать собственные команды секционирования без ограничений. Однако вы должны следовать некоторым соглашениям, чтобы позволить LaTeX'у осуществить все необходимые типографские действия при их выполнении.

Команда \secdef может помочь вам при определении таких команд, обеспечивая простой интерфейс трех возможных форм заголовков разделов, как показано на рис. 2.9 на следующей странице в случае команды \part.

[7] \newcommand{\nohyphens}{\hyphenpenalty=10000\exhyphenpenalty=10000\relax}

Определение `\newcommand{\part}{\secdef\cmda\cmdb}` *the* приводит к следующим действиям:

`\part{`*title*`}`	вызовет	`\cmda[`*title*`]{`*title*`}`
`\part[`*toc_entry*`]{`*title*`}`	вызовет	`\cmda[`*toc_entry*`]{`*title*`}`
`\part*{`*title*`}`	вызовет	`\cmdb{`*title*`}`

Рис. 2.9. Соглашения относительно команды `\secdef`

Команды, которые вам понадобятся, являются (пере)определением команды `\part` и определением команд с метками `\cmda` или `\cmdb` соответственно. Имейте в виду, что `\cmda` имеет факультативный аргумент, содержащий текст, который должен быть включен в файл оглавления `.toc`, тогда как второй (обязательный) аргумент команды, так же, как и единственный аргумент команды `\cmdb`, задает текст заголовка, который должен быть напечатан. В схематичном примере ниже используется расширенный вариант команды LaTeX2ε `\newcommand`. Она допускает применение факультативного аргумента; см. также разд. A.1.1.

```
\renewcommand{\part}{ ... \secdef \cmda \cmdb }
\newcommand{\cmda}[2][default]{ ... }
\newcommand{\cmdb}[1]{ ... }
```

Очевидным примером является упрощенный вариант команды `\appendix`. Это переопределение команды `\section` для создания заголовков приложений (обращаясь либо к команде `\Appendix`, либо к команде `\sAppendix`) путем изменения представления счетчика `section` и присвоения ему значения 0. Модифицированная команда `\section` также начинает новую страницу, устанавливает специальный формат для первой страницы (см. гл. 4), запрещает появление плавающих объектов в верхней части первой страницы и подавляет отступ первого абзаца раздела.

```
\renewcommand{\appendix}{%
   \renewcommand{\section}{%                    Переопределение \section
        \newpage\thispagestyle{plain}%
        \secdef\Appendix\sAppendix}%
   \setcounter{section}{0}%
   \renewcommand{\thesection}{\Alph{section}}%
}
```

В следующем ниже определении вы можете увидеть, как команда `\Appendix` продвигает счетчик `section`, используя команду `\refstepcounter` (последняя также перезагружает все второстепенные счетчики и определяет «текущую ссылку», см. разд. 2.5). Она записывает строку в файл `.toc` с командой `\addcontentsline`,

осуществляет форматирование заголовка и запоминает заголовок для помещения его в верхний и/или нижний колонтитул, вызывая команду \sectionmark.

```
\newcommand{\Appendix}[2][?]{%   Сложный вариант
    \refstepcounter{section}%
    \addcontentsline{toc}{appendix}%
        {\protect\numberline{\appendixname~\thesection} #1}%
    {\flushright\large\bfseries\appendixname\ \thesection\par
     \nohyphens\centering#2\par}%
    \sectionmark{#1}\vspace{\baselineskip}}
```

Команда \sAppendix (помеченная звездочкой) только осуществляет форматирование.

```
\newcommand{\sAppendix}[1]{%          Упрощенный (со звездочкой) вариант
        {\flushright\large\bfseries\appendixname\par
         \nohyphens\centering#1\par}%
        \vspace{\baselineskip}}
```

Применение этих определений приводит к такому тексту на печати:

Приложение А
Список всех команд

Далее следует текст первого раздела приложения. Дальнейший текст приложения. Еще немного текста приложения.

```
\appendix
\section{Список всех команд}

Далее следует текст первого раздела
приложения.
Дальнейший текст приложения.
Еще немного текста  приложения.
```

Не забывайте, что приведенный выше пример показывает только упрощенную версию переопределенной команды \section. Кроме того, среди прочего, мы не приняли во внимание счетчик secnumdepth, который содержит порог нумерации. Вы должны также знать заранее код, касающийся разных типов форматов документа, таких, как набор в одну и 2 колонки или односторонняя и двухсторонняя печать (см. гл. 4).

2.3.3 Изменение стандартных заголовков

Некоторые из стандартных команд заголовков создают фиксированные тексты, например, команда \chapter создает строку «Chapter» перед текстом. Так же некоторые окружения генерируют заголовки с фиксированными текстами. Например, по указанию окружение abstract помещает слово «Abstract» перед неким текстом пользователя. В ранних версиях LaTeX'а эти ограничения были закодированы внутри системы, поэтому было довольно трудно их изменить. В настоящей версии LaTeX'а эти ограничения устанавливаются согласованием команд (см. табл. 2.3 на развороте), поэтому можно с легкостью установить их для по-

Команда	*Текст по умолчанию*
\contentsname	Contents
\listfigurename	List of Figures
\listtablename	List of Tables
\bibname	Bibliography
\refname	References
\indexname	Index
\chaptername	Chapter
\appendixname	Appendix
\partname	Part
\abstractname	Abstract

Таблица 2.3. Команды для стандартных заголовков разделов

лучения любимых имен. Это показано в следующем ниже примере, где название «Abstract», определенное классом article, заменено словом «Резюме».

Резюме

Эта книга описывает, как модифицировать вид документов, созданных LaTeX'ом.

```
\renewcommand{\abstractname}{Резюме}
\begin{abstract}
Эта книга описывает, как модифицировать
вид документов, созданных
 \LaTeX'ом.
\end{abstract}
```

Файлы стандартных классов LaTeX'а определяют еще несколько ограничений. См. разд. 9.2, особенно табл. 9.2, чтобы получить полный список и обсуждение системы Babel, которая осуществляет перевод этих ограничений более чем на 20 языков.

2.4 Структура оглавления

Оглавление [$\mathcal{L}$ **70,158**], [$\mathcal{A}$**158, 273, 279–81**] — это специальный список с названиями разделов и указанием номеров страниц, с которых начинается каждый раздел. Этот список может быть достаточно сложным, если в него включено много разделов нескольких уровней вложенности, и он должен быть тщательно отформатирован, так как является главным ориентиром для читателя.

Существуют аналогичные оглавлению списки, содержащие ссылки на информацию о «плавающих» объектах документа, а именно *список таблиц* и *список рисунков*. Структура этих списков проще, поскольку их содержимое — заголовки «плавающих» объектов — одного уровня.

Стандартный LaTeX может автоматически создать эти три списка [$\mathcal{L}$ **186**], [$\mathcal{A}$**158,168**]. По умолчанию LaTeX вводит текст, сгенерированный одним из аргументов команд секционирования в файл с расширением `.toc`. Подобным образом

LaTeX поддерживает еще 2 файла, один для списка иллюстраций (`.lof`) и один для списка таблиц (`.lot`), которые содержат текст, определенный как аргумент команды `\caption` для рисунков и таблиц.

Информация, записанная в этих файлах во время предыдущей обработки [$\mathcal{L}$ 158] LaTeX'ом, прочитывается и печатается (обычно в начале документа) во время последующей обработки LaTeX'ом путем вызова команд `\tableofcontents`, `\listoffigures` и `\listoftables`.

Чтобы сгенерировать эти списки перекрестных ссылок, всегда необходимо обработать документ LaTeX'ом по крайней мере дважды — один раз для получения относящейся к нему информации, а второй раз для того, чтобы снова прочесть информацию и поместить ее в нужное место документа. Поскольку при второй обработке появляется дополнительный материал, который нужно напечатать, перекрестные ссылки могут измениться, поэтому необходимо произвести дальнейшую обработку LaTeX'ом. Это одна из причин традиционного использования разных систем нумерации страниц для рукописи и основного текста: во времена ручного набора любое дополнение приводило к значительному удорожанию окончательной продукции.

В следующих разделах будет обсуждаться вопрос о том, как набирать и генерировать эти списки. Также будет показано, как можно ввести информацию непосредственно в один из этих стандартных файлов, или даже, как открывать дополнительный файл и записывать в него информацию под полным контролем пользователя.

2.4.1 Набор оглавления

Как было сказано выше, оглавления состоят из вхождений разных типов, относящихся к структурным единицам, которые они представляют. Кроме этих стандартных вхождений, эти списки могут содержать некоторые команды. Стандартное вхождение определяется командой:

`\contentsline{`*type*`}{`*text*`}{`*page*`}`

Параметры таковы:

type тип вводимой информации, например, `section` или `figure`;

text текущий текст согласно определению в аргументе команды секционирования или команды `\caption`; и

page номер страницы.

Фрагмент кода, который генерирует оглавление, относящееся к части книги, показан на рис. 2.10 на следующей странице.

Имейте в виду, что номера разделов вводятся как параметр команды `\numberline`, чтобы обеспечить форматирование с правильным отступом. Пользователь также может создать оглавление вручную с помощью команды `\contentsline`.

Исходный текст

```
\contentsline {section}
  {\numberline {2.4}Структура оглавления}{49}
\contentsline {subsection}
  {\numberline {2.4.1}Набор оглавления}{50}
\contentsline {subsection}
  {\numberline {2.4.2}Ввод информации в файлы оглавления}{53}
```

Текст на выводе

2.4 Структура оглавления 49
2.4.1 Набор оглавления 50
2.4.2 Ввод информации в файлы оглавления 53

Рис. 2.10. Создание оглавления

Для форматирования вхождений в файлы оглавлений стандартный LaTeX использует следующую команду:

`\@dottedtocline{`*level*`}{`*indent*`}{`*numwidth*`}{`*text*`}{`*page*`}`

Два последних параметра совпадают с теми, которые определяются командой `\contentsline`, так как последние обычно вызываются командой `\@dottedtocline`. Другие параметры таковы:

level Уровень вложенности вхождения. Параметр позволяет пользователю контролировать, сколько уровней вложенности будет представлено. Уровни большие, чем значение счетчика `tocdepth`, в оглавлении не появятся.

indent Это абсолютный отступ от левого поля.

numwidth Ширина бокса, содержащего номер, если *текст* имеет команду `\numberline`. Это также величина дополнительного отступа, прибавляемого ко второй и последующим строкам многострочного вхождения.

В дополнение к этому команда`\@dottedtocline` использует следующие форматирующие параметры, которые задают внешний вид всех вхождений.

`\@pnumwidth` Ширина бокса, в котором помещен номер страницы.

`\@tocrmarg` Отступ от правого поля для всех строк, кроме последней, в многострочных вхождениях. Величина с размерностью, но изменяемой с помощью команды `\renewcommand`!

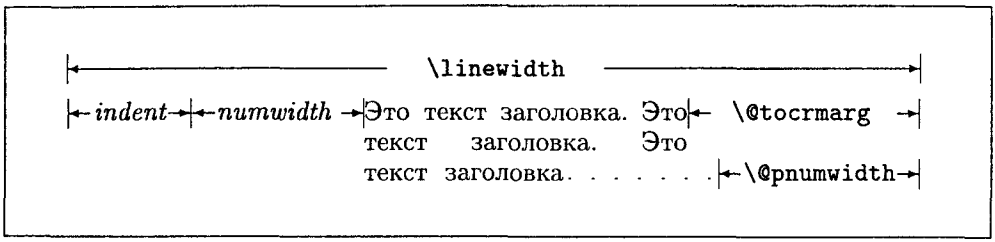


Рис. 2.11. Параметры, определяющие макет оглавления

`\@dotsep` Пробел между точками в `mu` (математические единицы)[8]. Это обычное число (как `1.7` или 2). Делая это число достаточно большим, вы можете избавиться от всех точек сразу. Также изменяется при помощи команды `\renewcommand`!

Наглядно описанные эффекты представлены на рис. 2.11. Поле, обозначенное *numwidth*, содержит номер раздела, выровненный по левому краю, если таковой имеется. Вы можете достичь правильного отступа для вложенных вхождений, меняя установочные значения *indent* и *numwidth*.

Команда `\contentsline` применяется, чтобы взять ее первый аргумент *type* и затем использовать его для вызова соответствующей команды `\l@`*type*, которая производит сам набор. Отдельная команда для каждого из типов должна быть определена в стилевом файле. Например, в классе report можно найти следующие определения:

```
\newcommand{\l@section}{\@dottedtocline{1}{1.5em}{2.3em}}
\newcommand{\l@subsection}{\@dottedtocline{2}{3.8em}{3.2em}}
\newcommand{\l@subsubsection}{\@dottedtocline{3}{7.0em}{4.1em}}
\newcommand{\l@paragraph}{\@dottedtocline{4}{10em}{5em}}
\newcommand{\l@subparagraph}{\@dottedtocline{5}{12em}{6em}}
\newcommand{\l@figure}{\@dottedtocline{1}{1.5em}{2.3em}}
\newcommand{\l@table}{\l@figure}
```

Определив `\l@`*type*, чтобы вызвать `\@dottedtocline`, и указав три аргумента (*level*, *indent* и *numwidth*), остальные аргументы *text* и *page* команды `\contentsline` мы будем брать командой `\@dottedtocline` как четвертый и пятый аргументы.

Имейте в виду, что некоторые уровни разделов выстраивают свои вхождения в оглавление более сложным путем, поэтому в стандартных классах документов есть определения для `\l@part` и `\l@chapter`, которые не используют `\@dottedtocline`. Обычно они используют ряд специальных команд форматиро-

[8] В одном `em` содержится 18 `mu`; `em` взят из `\fontdimen2` шрифта математических символов `symbols`. Более подробно о `\fontdimen` см. разд. 7.7.1.

вания, возможно, допускающих набор более крупным шрифтом и без отточий. Подобный пример приведен ниже:

Исходный текст

```
\contentsline{part}{\numberline{II}Part}{1}
\contentsline{chapter}{\numberline{1}Chapter}{2}
\contentsline{section}{\numberline{1.1}Section}{3}
\contentsline{subsection}{\numberline{1.1.1}Subsection}{4}
\contentsline{subsection}{\numberline{}Subsection without number}{5}
\contentsline{subsection}{Unnumbered subsection}{6}
```

Текст на выводе

II Part 1

1 Chapter **2**

1.1 Section 3

1.1.1 Subsection 4

Subsection without number 5

Unnumbered subsection 6

Уровень, вплоть до которого информация о заголовках отображается в оглавлении, контролируется счетчиком `tocdepth` [$\mathcal{L}$ **160**]. Он может быть изменен, например, следующим образом:

```
\setcounter{tocdepth}{2}
```

В этом случае будет показана информация о заголовках до второго уровня (например, часть, глава и раздел).

2.4.2 Ввод информации в файлы оглавления

LaTeX предлагает две команды для ввода информации непосредственно в файл оглавления [$\mathcal{L}$ **159**], [$\mathcal{A}$ **280,281**]:

`\addtocontents{`*file*`}{`*text*`}` `\addcontentsline{`*file*`}{`*type*`}{`*text*`}`

file Расширение файла оглавления, обычно `toc`, `lof` или `lot`.

type Тип вхождения. Для файла с расширением `.toc` *type* обычно такой же, как для заголовков, согласно тому, в каком формате вхождение должно быть напечатано. Для файлов `.lof` или `.lot` формат задается в соответствии с `figure` или `table`.

text Информация, которая должна быть записана в *file*. Команды LaTeX'а следует защитить командой `\protect`, чтобы избежать нежелательных последствий.

Команда \addtocontents не содержит параметра *type* и предназначена для того, чтобы ввести информацию о форматировании, задаваемую пользователем. Например, если вы хотите создать дополнительный пробел в середине оглавления, нужно использовать следующую команду:

```
\addtocontents{toc}{\protect\vspace{2ex}}
```

Инструкция \addcontentsline обычно вызывается *автоматически* командами секционирования документа или командами \caption. Если вхождение содержит пронумерованный текст, нужно использовать команду \numberline, чтобы отделить номер раздела (*number*) от остального текста вхождения (*heading*) в параметре *text*:

```
\protect\numberline{number}heading
```

Например, команда \caption внутри окружения figure сохраняет текст подрисуночной подписи следующим образом:

```
\addcontentsline{lof}{figure}%
    {\protect\numberline{\thefigure}captioned text}
```

Иногда команда \addcontentsline используется в документе для того, чтобы дополнить действия стандартного LaTeX'а. Например, в случае варианта «со звездочкой» команд секционирования в файле .toc не записывается никакая информация. Поэтому если вы хотите, чтобы заголовок без номера (вариант «со звездочкой») фигурировал в файле .toc, вы можете написать нечто вроде:

```
\chapter*{Foreword}
\addcontentsline{toc}{chapter}{\numberline{}Foreword}
```

Это создает вхождение на уровне «глава» с отступом в оглавлении, где вместо номера главы будет оставлен пробел. Если ликвидировать команду \numberline, слово «Foreword» сдвинется влево.

2.4.3 Определение нового файла, аналогичного .toc

Если вы хотите сделать список, охватывающий все примеры в книге, вам нужно создать новый файл оглавления, а затем использовать описанные выше средства. Прежде всего нужно определить две новые команды: команда \listofexamples прочтет информацию, записанную в файле с расширением .xmp (см. ниже) и поместит ее в том месте документа, где вызывается команда. Вторая команда, \ecaption, присоединяет название примера к каждому окружению и записывает его аргумент в файл оглавления с расширением .xmp. Команда \listofexamples вызывает \@starttoc{*xxx*}, которая читает внешний файл (с расширением *xxx*) и затем снова открывает его для записи. Эта команда также используется командами \tableofcontents, \listoffigures и \listoftables.

Дополнительному файлу может быть присвоено расширение xmp. Команда, подобная \chapter*{List of examples}, может быть поставлена непосредственно перед командой \listofexamples для создания заголовка, и, если это нужно, команда \addcontentsline может сигнализировать читателю о наличии этого списка, введя его в .toc-файл.

Текущая печать индивидуальных вхождений в .xmp-файл управляется командой \l@example. В приведенном ниже примере названия оформляются как абзацы, за которыми следуют набранные курсивом номера страниц.

```
\newcommand{\listofexamples}{\@starttoc{xmp}}
\newcommand{\ecaption}[1]{\addcontentsline{xmp}{example}{#1}}
\newcommand{\l@example}[2]{\par\noindent#1 {\itshape #2}}
```

Вы можете также заглянуть в разд. 6.3, где в случае «плавающих» объектов команда \listof сгенерирует список этих объектов так, как определяет ее аргумент.

2.4.4 Сложные оглавления

Пакет minitoc, первоначально написанный Найгелом Уордом и Даном Джурафски и полностью переработанный Жан-Пьером Друкбером, создает мини-оглавление («minitoc») в начале каждой главы в классах book или report.

Мини-оглавление будет появляться в начале главы после команды \chapter. Параметры, которые управляют этим пакетом, показаны в табл. 2.4.

Для каждого мини-оглавления будет создан вспомогательный файл с расширением .mtc<N>, где <N> — номер главы[9].

По умолчанию эти мини-оглавления содержат только ссылки на разделы и подразделы. Счетчик minitocdepth, подобный tocdepth, позволяет пользователю изменить эту ситуацию.

Так как мини-оглавление располагается на первых страницах главы, он меняет нумерацию страниц. Поэтому обычно требуется обработать документ трижды, чтобы получить правильную информацию в мини-оглавлении.

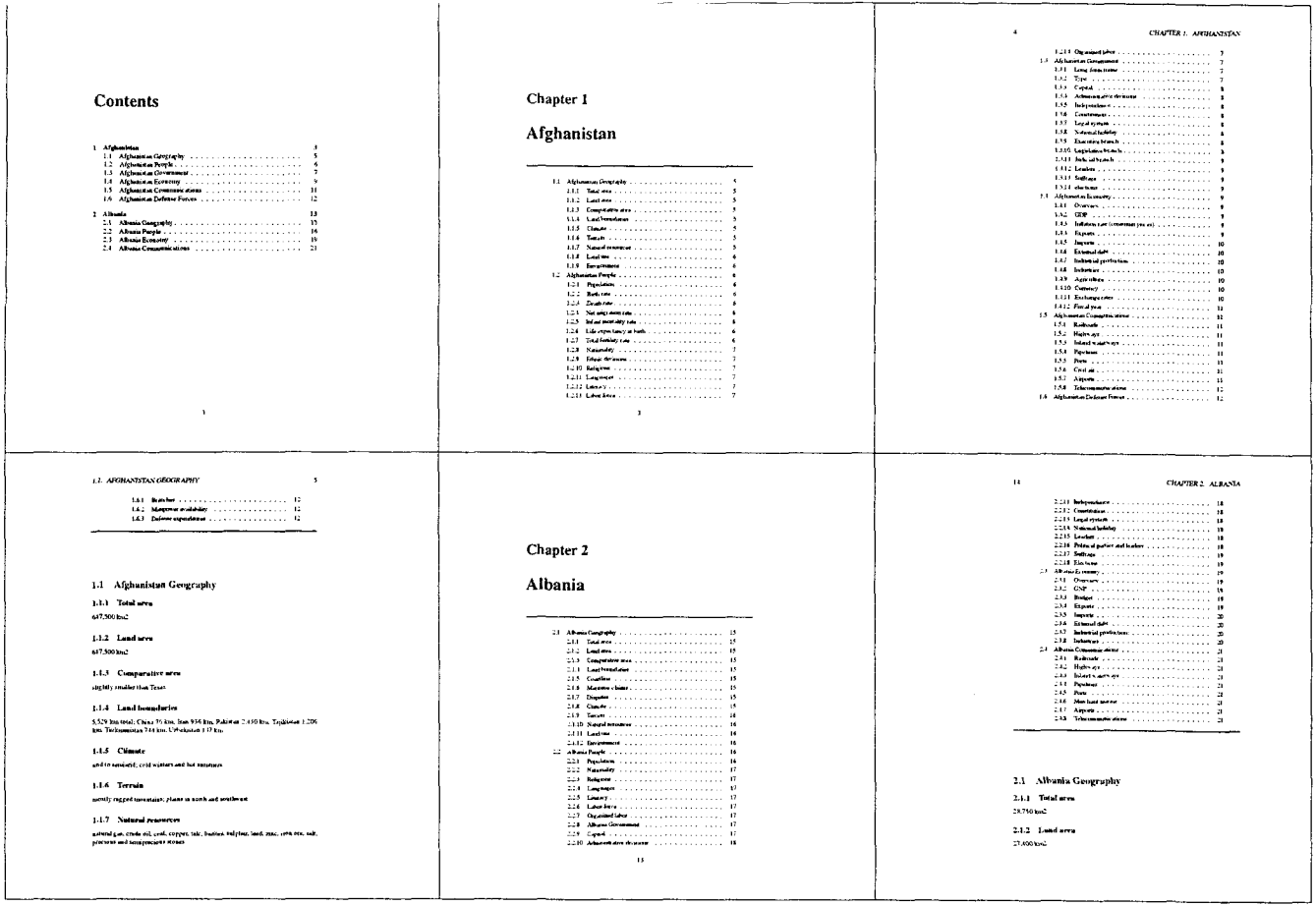
Чтобы отменить команды \minitoc, просто замените пакет minitoc на minitocoff в вашей команде \usepackage. Это послужит гарантией того, что все команды \minitoc будут проигнорированы.

Пример применения пакета minitoc дается на рис. 2.12, где мы установили глобальный счетчик tocdepth в единицу, так что в оглавлении документа будут показаны только заголовки разделов (вы можете увидеть результат в первой картинке на рис. 2.13). Счетчик глубины уровней для мини-оглавлений, minitocdepth, равен двум, поэтому заголовки разделов и подразделов появляются в каждой такой таблице (как видно на остальных пяти картинках на рис. 2.13). Текст главы начинается сразу же после конца мини-оглавления.

[9] Специальная версия этого пакета сделана для операционных систем, в которых расширение файлов ограничено тремя знаками, например, MS-DOS или MS-Windows.

```
\documentclass{book}
\usepackage{times}
\usepackage{minitoc}
\setcounter{tocdepth}{1}            % глубина уровней оглавления
\setlength{\mtcindent}{24pt}        % отступ для мини-оглавлений по умолчанию
\renewcommand{\mtcfont}{\small\rm} % шрифт для мини-оглавлений по умолчанию
\setcounter{minitocdepth}{2}        % глубина уровней мини-оглавления
\begin{document}
\dominitoc                          % создание мини оглавлений
\tableofcontents                    % создание общего оглавления
\chapter{Afghanistan}
\minitoc                            % мини-оглавление после названия гл. 1


\section{Afghanistan Geography}

\subsection{Total area}
    647,500 km2
\subsection{Land area}
    647,500 km2



        ..... продолжение гл.~1



\chapter{Albania}
\minitoc                           % мини-оглавление после названия гл. 2
\section{Albania Geography}

\subsection{Total area}
    28,750 km2
\subsection{Land area}
    27,400 km2



        ..... продолжение гл.~2
```

Рис. 2.12. Мини-оглавление — пример ввода



Contents

1 Afghanistan 5
1.1 Afghanistan Geography . . . 5
1.2 Afghanistan People . . . 6
1.3 Afghanistan Government . . . 7
1.4 Afghanistan Economy . . . 9
1.5 Afghanistan Communications . . . 11
1.6 Afghanistan Defense Forces . . . 12

2 Albania 13
2.1 Albania Geography . . . 15
2.2 Albania People . . . 16
2.3 Albania Economy . . . 18
2.4 Albania Communications . . . 21

Chapter 1

Afghanistan

1.1 Afghanistan Geography . . . 5
1.1.1 Total area . . . 5
1.1.2 Land area . . . 5
1.1.3 Comparative area . . . 5
1.1.4 Land boundaries . . . 5
1.1.5 Climate . . . 5
1.1.6 Terrain . . . 5
1.1.7 Natural resources . . . 5
1.1.8 Land use . . . 6
1.1.9 Environment . . . 6
1.2 Afghanistan People . . . 6
1.2.1 Population . . . 6
1.2.2 Birth rate . . . 6
1.2.3 Death rate . . . 6
1.2.4 Net migration rate . . . 6
1.2.5 Infant mortality rate . . . 6
1.2.6 Life expectancy at birth . . . 6
1.2.7 Total fertility rate . . . 6
1.2.8 Nationality . . . 7
1.2.9 Ethnic divisions . . . 7
1.2.10 Religions . . . 7
1.2.11 Languages . . . 7
1.2.12 Literacy . . . 7
1.2.13 Labor force . . . 7

1.2.14 Organized labor . . . 7
1.3 Afghanistan Government . . . 7
1.3.1 Long form name . . . 7
1.3.2 Type . . . 7
1.3.3 Capital . . . 8
1.3.4 Administrative divisions . . . 8
1.3.5 Independence . . . 8
1.3.6 Constitution . . . 8
1.3.7 Legal system . . . 8
1.3.8 National holiday . . . 8
1.3.9 Executive branch . . . 8
1.3.10 Legislative branch . . . 8
1.3.11 Judicial branch . . . 9
1.3.12 Leaders . . . 9
1.3.13 Suffrage . . . 9
1.3.14 elections . . . 9
1.4 Afghanistan Economy . . . 9
1.4.1 Overview . . . 9
1.4.2 GDP . . . 9
1.4.3 Inflation rate (consumer prices) . . . 9
1.4.4 Exports . . . 9
1.4.5 Imports . . . 10
1.4.6 External debt . . . 10
1.4.7 Industrial production . . . 10
1.4.8 Industries . . . 10
1.4.9 Agriculture . . . 10
1.4.10 Currency . . . 10
1.4.11 Exchange rates . . . 10
1.4.12 Fiscal year . . . 11
1.5 Afghanistan Communications . . . 11
1.5.1 Railroads . . . 11
1.5.2 Highways . . . 11
1.5.3 Inland waterways . . . 11
1.5.4 Pipelines . . . 11
1.5.5 Ports . . . 11
1.5.6 Civil air . . . 11
1.5.7 Airports . . . 11
1.5.8 Telecommunications . . . 12
1.6 Afghanistan Defense Forces . . . 12

1.6.1 Branches . . . 12
1.6.2 Manpower availability . . . 12
1.6.3 Defense expenditures . . . 12

1.1 Afghanistan Geography

1.1.1 Total area

647,500 km2

1.1.2 Land area

647,500 km2

1.1.3 Comparative area

slightly smaller than Texas

1.1.4 Land boundaries

5,529 km total; China 76 km, Iran 936 km, Pakistan 2,430 km, Tajikistan 1,206 km, Turkmenistan 744 km, Uzbekistan 137 km

1.1.5 Climate

arid to semiarid; cold winters and hot summers

1.1.6 Terrain

mostly rugged mountains; plains in north and southwest

1.1.7 Natural resources

natural gas, crude oil, coal, copper, talc, barites, sulfur, lead, zinc, iron ore, salt, precious and semiprecious stones

Chapter 2

Albania

2.1 Albania Geography . . . 15
2.1.1 Total area . . . 15
2.1.2 Land area . . . 15
2.1.3 Comparative area . . . 15
2.1.4 Land boundaries . . . 15
2.1.5 Coastline . . . 15
2.1.6 Maritime claims . . . 15
2.1.7 Disputes . . . 15
2.1.8 Climate . . . 15
2.1.9 Terrain . . . 16
2.1.10 Natural resources . . . 16
2.1.11 Land use . . . 16
2.1.12 Environment . . . 16
2.2 Albania People . . . 16
2.2.1 Population . . . 16
2.2.2 Nationality . . . 17
2.2.3 Religions . . . 17
2.2.4 Languages . . . 17
2.2.5 Literacy . . . 17
2.2.6 Labor force . . . 17
2.2.7 Organized labor . . . 17
2.2.8 Albanian Government . . . 17
2.2.9 Capital . . . 17
2.2.10 Administrative divisions . . . 18

2.2.11 Independence . . . 18
2.2.12 Constitution . . . 18
2.2.13 Legal system . . . 18
2.2.14 National holiday . . . 18
2.2.15 Executive . . . 18
2.2.16 Political parties and leaders . . . 18
2.2.17 Suffrage . . . 19
2.2.18 Elections . . . 19
2.3 Albania Economy . . . 19
2.3.1 Overview . . . 19
2.3.2 GNP . . . 19
2.3.3 Budget . . . 19
2.3.4 Exports . . . 19
2.3.5 Imports . . . 20
2.3.6 External debt . . . 20
2.3.7 Industrial production . . . 20
2.3.8 Industries . . . 20
2.4 Albania Communications . . . 21
2.4.1 Railroads . . . 21
2.4.2 Highways . . . 21
2.4.3 Inland waterways . . . 21
2.4.4 Pipelines . . . 21
2.4.5 Ports . . . 21
2.4.6 Merchant marine . . . 21
2.4.7 Airports . . . 21
2.4.8 Telecommunications . . . 21

2.1 Albania Geography

2.1.1 Total area

28,748 km2

2.1.2 Land area

27,400 km2

Рис. 2.13. Мини-оглавление — пример вывода

`\dominitoc`	должен быть помещен перед `\tableofcontents`, чтобы инициализировать систему minitoc (обязательный).
`\faketableofcontents`	эта команда замещает `\tableofcontents`, если вы хотите, чтобы мини-оглавление было, а общего оглавления не было.
`\minitoc`	эта команда должна располагаться справа после каждой команды `\chapter`,в которой мини-оглавление требуется.
`minitocdepth`	счетчик LaTeX'a, который указывает, сколько уровней заголовков нужно представить в мини-оглавлении (значение по умолчанию 2).
`\mtcindent`	длина левого/правого отступа в мини-оглавлении (по умолчанию 24 pt).
`\mtcfont`	команда, *определяющая шрифт для вхождений* мини-оглавления (по умолчанию это светлый прямой «small»).

Таблица 2.4. Перечень параметров minitoc

2.5 Управление ссылками

В LaTeX'e есть команды, позволяющие легко управлять ссылками в документе. В целом он поддерживает перекрестные ссылки (внутренние ссылки между элементами в пределах документа), библиографические ссылки (ссылки на внешние документы) и указатель отдельных слов или выражений (предметный и именной). Средства для создания указателей будут обсуждаться в гл. 12, а библиографические ссылки — в гл. 13.

Чтобы сделать возможными перекрестные ссылки между элементами внутри документа, вы должны присвоить «ключ», состоящий из цепочки литер, цифр и знаков препинания, данному структурному элементу и затем использовать этот ключ везде для ссылки на этот элемент.

`\label{`*key_string*`}` `\ref{`*key_string*`}` `\pageref{`*key_string*`}`

Команда `\label` присваивает ключ *key_string* текущему элементу документа [$\mathcal{L}$ **71,186**], [$\mathcal{A}$ **26**]. Команда `\ref` печатает строку, идентифицирующую данный элемент, такой, как раздел, уравнение или номер рисунка, в зависимости от типа структурного элемента, который был активным при выполнении команды `\label`. Команда `\pageref` печатает номер страницы, где была задана команда `\label`. Ключи, безусловно, должны быть уникальными, и в целях простоты может оказаться полезным предварить их цепочкой литер, идентифицирующей

данный структурный элемент: `sec` может представлять единицы секционирования, `fig` — рисунки и т. д.

Ссылка на этот подраздел выглядит так: «см. разд. 2.5 на с. 59».

```
Ссылка на этот подраздел\label{sec:this}
выглядит так: "<см.~разд.~\ref{sec:this}
на с.~\pageref{sec:this}">.
```

Для построения меток перекрестных ссылок текущий активный структурный элемент документа определяется следующим образом. Команды секционирования (`\chapter`, `\section`, ...), окружения `equation`, `figure`, `table` и семейство `theorem` так же, как и разные уровни окружения `enumerate` и `\footnote`, устанавливают *цепочку текущей ссылки*, которая содержит номер, сгенерированный LaTeX'ом для данного элемента. Эта цепочка ссылки обычно устанавливается в начале элемента и переустанавливается после выхода за рамки элемента.

Заметными исключениями из этого правила являются окружения `table` и `figure`, где цепочка ссылки определяется командами `\caption`. Допускается существование нескольких пар `\caption` и `\label` внутри одного окружения. Так как именно `\caption` дает распоряжение генерировать номер, соответствующая команда `\label` должна *следовать* за данной командой `\caption`, иначе будет сгенерирован неправильный номер. Это ясно показано в следующем примере, где только метки '`fig:in2`' и '`fig:in3`' расположены правильно для создания необходимых ссылок на номера рисунков. В случае '`fig:in4`' видно, что окружения (в данном случае, `center`) ограничивают возможности ссылок, так как мы получаем номер текущего раздела, а не номер рисунка.

Ссылка на предыдущий текст обозначена как '2.5'.

... тело рисунка ...

Рис. 4.14. Первый заголовок

... тело рисунка ...

Рис. 4.15. Второй заголовок

Метки: 'before' (2.5), 'fig:in1' (2.5), 'fig:in2' (4.14), 'fig:in3' (4.15), 'fig:in4' (2.5), 'after' (2.5).

```
\label{sec:before}
Ссылка на предыдущий текст обозначена как
'\ref{sec:before}'.
\begin{figure}[H]              \label{fig:in1}
   \begin{center}
      \fbox{\ldots{} тело рисунка \ldots}
      \caption{Первый заголовок}\label{fig:in2}

      \fbox{\ldots{} тело рисунка \ldots}
      \caption{Второй заголовок}\label{fig:in3}
   \end{center}                \label{fig:in4}
\end{figure}
\label{sec:after}
\raggedright
Метки: 'before' (\ref{sec:before}),
'fig:in1' (\ref{fig:in1}),
'fig:in2' (\ref{fig:in2}),
'fig:in3' (\ref{fig:in3}),
'fig:in4' (\ref{fig:in4}),
'after' (\ref{sec:after}).
```

Для каждого ключа *key_string*, декларированного с помощью `\label{`*key_string*`}`, LaTeX записывает цепочку текущей ссылки и номер страни-

цы. Таким образом, сложные команды \label (с разными ключами *key_string*) внутри одного и того же раздела создадут одинаковую цепочку ссылки, но, возможно, разные номера страниц.

2.5.1 varioref — более гибкие ссылки

Во многих случаях полезно при создании ссылок на рисунок или таблицу использовать в документе обе команды \ref и \pageref, особенно в тех случаях, когда ссылку от объекта отделяет одна страница или больше. Поэтому некоторые пользуются командой вроде

```
\newcommand{\fullref}[1]{\ref{#1} on page~\pageref{#1}}
```

чтобы уменьшить количество ключей, необходимых для создания полной ссылки. Но поскольку никто никогда не знает, где в конце концов окажется объект ссылки, этот метод может завершиться ссылкой на текущую страницу, что приведет к нарушениям, поэтому его надо избегать. Пакет varioref, написанный Франком Миттельбахом, пытается разрешить эту проблему.

Пользовательский интерфейс

\vref{*key_string*}

Команда \vref подобна \ref, когда ссылка и \label находятся на одной и той же странице. Если метка и ссылка отличаются на одну страницу, \vref создает одну из строк: «on the facing page», «on the preceding page», «on the following page»[10]; слова «on the facing page» используются, когда и метка, и ссылка попадают на двойной разворот. Если разница больше одной страницы, \vref создает как \ref, так и \pageref. Имейте в виду, что если используется специальная система нумерации страниц вместо обычных арабских цифр (например, \pagenumbering{roman}), то не будет различия между тем, одна или несколько страниц пропущено.

\vpageref[*samepage*][*otherpage*]{*key_string*}

Иногда бывает нужно сослаться только на номер страницы. В этом случае ссылка должна быть подавлена, если вы ссылаетесь на текущую страницу. Для этой цели определена команда \vpageref. Она создает те же самые цепочки, что и \vref, за исключением того, что она не начинается с \ref, и создает цепочку, сохраняемую в \reftextcurrent, если метка и ссылка попадают на одну и ту же страницу.

[10] Что означает, соответственно «на развороте», «на предыдущей странице», «на следующей странице».— *Прим. перев.*

Определение `\reftextcurrent`, позволяющее создать нечто вроде «on this page» гарантирует, что

```
... see the diagram \vpageref{ex:foo} which shows ...
```

не превратится в « ... see the diagram which shows ... », что может ввести в заблуждение.

Вы можете поместить пробел перед командой `\vpageref`; он будет проигнорирован, если команда не создает какого-либо текста вообще. Если добавляется какой-либо текст, соответствующий неразрываемый пробел автоматически помещается перед текстом.

В действительности `\vpageref` позволяет даже большее, благодаря использованию двух факультативных аргументов. Первый определяет текст, который должен быть использован, если метка и ссылка попадают на одну и ту же страницу. Это полезно, когда обе находятся близко друг к другу, так что они могут или не могут быть разделены разрывом страницы. В таком случае вы обычно будете знать, идет ссылка до или после метки, так что вы можете написать нечто вроде:

```
... see the diagram \vpageref[above]{ex:foo} which shows ...
```

Конечный текст будет таким: « ... see the diagram above which shows ... », если обе находятся на одной и той же странице, или « ... see the diagram on the page before which shows ... » (или что-то подобное, в зависимости от команд `\reftext..before` и `\reftext..after`, см. ниже), если их разделяет страница. Однако имейте в виду, что если вы используете `\vpageref` с факультативным аргументом для ссылки на рисунок или таблицу в зависимости от расположения плавающих параметров, объект может оказаться в верхней части текущей страницы, а следовательно, перед ссылкой, даже если он следует за ней в исходном файле[11].

Возможно, вы даже предпочтете написать « ... see the above diagram», когда диаграмма и ссылка попадают на одну и ту же страницу, т. е. изменить порядок слов по сравнению с нашим предыдущим примером. В действительности в некоторых языках порядок слов в этом случае меняется автоматически. Чтобы позволить такой вариант, можно использовать второй факультативный аргумент *otherpage*. Он определяет текст, предшествующий ссылке, если объект и ссылка не попадают на одну страницу. Поэтому для достижения желаемого эффекта нужно написать:

```
... see \vpageref[above diagram][diagram]{ex:foo} which shows ...
```

Если пробел, дополненный `\vpageref`, оказывается неправильным, как в

```
(\vpageref[here]{ex:foo})
```

[11] Чтобы убедиться в том, что плавающий объект всегда будет на своем месте, используйте пакет flafter, который описан в разд. 6.2.

используйте вариант с двумя факультативными аргументами и поместите весь текст вплоть до предыдущего пустого пространства (в этом случае просто открывающая круглая скобка) в оба факультативных аргумента, т. е.

```
\vpageref[(here)][(]{ex:foo})
```

Языковая поддержка

Этот пакет поддерживает опции, определенные системой Babel (см. разд. 9.2); поэтому декларация типа

```
\usepackage[german]{varioref}
```

создаст тексты, подходящие для немецкого языка. Чтобы сделать возможной дальнейшую работу, все сгенерированные цепочки текста определяются через макросы (которые будут переопределены языковыми опциями).

Обратные ссылки используют `\reftextbefore`, если метка находится на предыдущей странице, но невидима, и `\reftextfacebefore`, если она находится на данной странице (в том случае, если номер текущей страницы нечетный).

Подобным образом `\reftextafter` используется, когда метка находится на следующей странице, но нужно перевернуть страницу, и `\reftextfaceafter`, если она находится на следующей, но на развороте. Эти четыре цепочки могут быть переопределены с помощью команды `\renewcommand`.

И наконец, цепочка, относящаяся к `\reftextfaraway`, используется, когда метка и ссылка различаются более, чем на 1 страницу, или если они не пронумерованы. Эта макрокоманда немного другая, так как нужен один аргумент, символическая цепочка ссылки, чтобы можно было использовать `\pageref` в замещенном тексте. Например, если вы хотите использовать ваши макро в документах на немецком языке, вы должны написать что-то вроде:

```
\renewcommand{\reftextfaraway}[1]{auf Seite~\pageref{#1}}
```

Чтобы сделать возможными некоторые произвольные вариации в сгенерированных цепочках, вы можете использовать команду `\reftextvario` внутри цепочки макро.

`\reftextvario{`*variationa*`}{`*variationb*`}`

Эта команда берет два аргумента и выбирает один или другой для печати в зависимости от номера команд `\vref` или `\vpageref`, уже встретившихся в документе.

В качестве примера ниже даны определения по умолчанию разных макрокоманд, описанных в этом разделе.

```
\newcommand{\reftextfaceafter}
           {on the \reftextvario{facing}{next} page}
\newcommand{\reftextfacebefore}
           {on the \reftextvario{facing}{preceding} page}
\newcommand{\reftextafter}
           {on the \reftextvario{following}{next} page}
\newcommand{\reftextbefore}
           {on the \reftextvario{preceding page}{page before}}
\newcommand{\reftextcurrent}
           {on \reftextvario{this}{the current} page}
\newcommand{\reftextfaraway}[1]{on page~\pageref{#1}}
```

Итак, если вы хотите приспособить пакет к вашим собственным нуждам, просто напишите соответствующие переопределения приведенных выше команд в файл с расширением `.sty` (например, `vrflocal.sty`). Если вы также поместите `\RequirePackage{varioref}` (см. разд. А.3) в начало этого файла, то ваш пакет будет автоматически загружать varioref.

Несколько предостережений

Определяющие команды, подобные описанным выше, ставят ряд интересных проблем. Предположим, например, что сгенерированный текст вроде «на следующей странице» может быть разбит на две страницы. Если это происходит, то очень трудно найти приемлемое алгоритмическое решение, тем более что подобная ситуация может сложиться в документе, который все время меняется от одной стадии к другой (т. е. вставка одной цепочки литер, обнаружение того, что это неправильно, вставка другой цепочки при следующей обработке, которая опять приведет к тому, что первая вставка станет правильной, вставка ...). Текущая версия пакета varioref считает конец сгенерированной строки релевантным. Например,

табл. 5 на данной ⟨*page break*⟩ странице

считается правильным, если табл. 5 окажется на странице, содержащей слово «странице», а не на той, которая содержит слово «данной». Однако это не вполне удовлетворительно и в некоторых случаях может привести к зацикливанию (когда снова и снова требуется дополнительная обработка LaTeX'ом). Поэтому все подобные ситуации создадут в LaTeX'е сообщение об ошибке, так что вы можете контролировать проблему и, возможно, решите использовать в этом месте команду `\ref`.

Также будьте в курсе потенциальных проблем, которые могут возникнуть из-за использования `\reftextvario`: если вы ссылаетесь на один и тот же объект несколько раз в близко расположенных местах, изменение в словах каждый второй раз будет выглядеть странно.

Последнее предупреждение: каждое использование `\vref` генерирует на внутреннем уровне два имени макрокоманды. В результате вы можете выйти за пределы пространства, отведенного для имен, и исчерпаете основную память, если

упорно используете эту команду при инсталляции TeX'a в сокращенном варианте. Для этой цели предоставляется также команда `\fullref`. Ее можно использовать везде, где вы уверены, что метка и ссылка не окажутся на близлежащих страницах.

2.5.2 Ссылки на внешние документы

Дэвид Карлайл, используя раннюю работу Жан-Пьера Друкбера, разработал пакет xr, который предоставляет систему для внешних ссылок.

Если, например, в документе необходимо сделать ссылки на разделы другого документа, скажем `other.tex`, то вы можете указать пакет xr в основном файле и задать команду `\externaldocument{other}` в преамбуле.

Затем вы можете использовать `\ref` и `\pageref`, чтобы сослаться на все, что было определено с помощью команды `\label` либо в `other.tex`, либо в вашем основном документе. Вы можете декларировать любое число таких внешних документов.

Если какие-нибудь внешние документы или основной документ используют одну и ту же команду `\label`, то может возникнуть конфликт, так как метка определена много раз. Для разрешения этой проблемы команда `\externaldocument` имеет факультативный аргумент. Если вы напишите `\externaldocument[A-]{other}`, то все ссылки из файла `other.tex` будут иметь префикс `A-`. Так, например, если раздел в файле `other.tex` имел `\label{intro}`, то на него нужно ссылаться как `\ref{A-intro}`. Префикс не обязательно должен быть `A-`; это может быть любая выбранная цепочка литер, гарантирующая, что все метки, взятые из внешних файлов, уникальны. Однако имейте в виду, что если один из используемых вами пакетов объявляет определенные литеры активными (например, : во французском языке или " в немецком), то эти литеры нельзя использовать внутри команд `\label`. Точно так же вы не должны использовать их в факультативном аргументе `\externaldocument`.

ГЛАВА 3

Основные средства форматирования

Способ наглядного представления информации может в большой степени влиять на то, каким образом эта информация будет понята читателем. Следовательно, чтобы как можно точнее передать смысл ваших слов, очень важно выбрать наилучшее из доступных средств их представления. Нужно, однако, отметить, что формы наглядного представления должны помогать читателю в понимании текста и не должны отвлекать внимания. Таким образом, чтобы добиться логической стройности и однородности восприятия, необходимо задать единый стиль выделения структурных элементов и придерживаться его на протяжении всего документа. Это ограничение легко достигается посредством задания специальных команд или окружения для каждого элемента документа, который нуждается в особом оформлении. Эти команды и окружения группируются в отдельном файле или в преамбуле документа. Пользуясь далее исключительно этими командами, вы можете быть уверены в логической стройности представления документа.

В настоящей главе описываются различные способы оформления частей документа. В разд. 3.1 показано, как выделять короткие фрагменты текста или абзацы. Следующий раздел посвящен перечням LaTeX'а. В первую очередь обсуждаются различные параметры и команды управления стандартными перечнями LaTeX'а: `enumerate`, `itemize` и `description`. Затем вводится общее окружение `list`, и вы имеете возможность научиться получать различные способы оформления перечней, варьируя значениями управляющих параметров окружения `list`. Буквальное воспроизведение текста шрифтом пишущей машинки (verbatim) — предмет обсуждения разд. 3.3, в котором приводятся различные способы представления текстов в набранном виде (наподобие компьютерных листингов). В разд. 3.4 рассматриваются различного рода примечания: подстрочные примечания, заметки на полях и блоки примечаний в конце (выносные примечания). О пакете, позволяющем легко офор-

мить текст в виде нескольких колонок, рассказано в разд. 3.5. Глава завершается обсуждением пакета, осуществляющего некоторого рода управление версиями.

3.1 Фразы и абзацы

Отдельные фрагменты текста могут быть выделены посредством наглядного оформления в виде, отличном от основного текста. Для этого могут быть задействованы такие параметры, как начертание шрифта и его насыщенность (см. разд. 7.3.1). Можно также использовать подчеркивание или разрядку: здесь будет описано, как это сделать.

Также будут введены два способа изменения вида абзаца, и вы научитесь создавать текст с рваным правым краем и менять интерлиньяж внутри абзаца.

3.1.1 letterspace — изменение межбуквенных интервалов

Пакет letterspace Филиппа Тейлора вводит команду `\letterspace`, которая позволяет менять ширину фрагмента текста посредством варьирования расстояния между буквами и между словами (последнее называется *трэкингом*). Требуемая ширина может также определяться как функция естественной ширины бокса, содержащего текст, посредством использования параметра `\naturalwidth` (см. пример ниже).

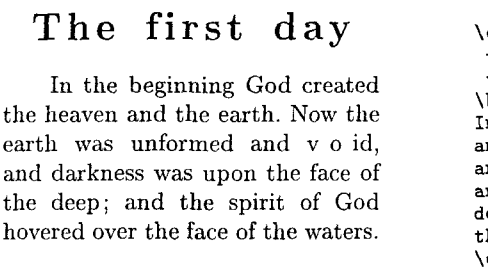
В первой строке показан слегка сжатый текст, во второй строке этот текст имеет естественную ширину, а в третьей текст разрежен на 10%. В четвертой и пятой строках текст равномерно разрежен соответственно на половину строки и на целую строку.

Time for good men
Time for good men (естественная ширина)
Time for good men
Time for good men
T i m e f o r g o o d m e n

```
\letterspace to .9\naturalwidth
                        {Time for good men}
\letterspace
 {Time for good men (естественная ширина)}
\letterspace to 1.1\naturalwidth
                        {Time for good men}
\letterspace to .5\linewidth
                        {Time for good men}
\letterspace to  \linewidth
                        {Time for good men}
```

Команду `\letterspace` следует использовать с большой осторожностью, поскольку при этом меняется уровень «серого оттенка» текста, что может оказаться для читателя критичным. Рекомендуется ограничиться ситуациями, когда нужно компенсировать недостаток заданных ширин шрифта, например, чтобы добиться специальных эффектов или чтобы уместить заданное количество букв в строку, меняя плотность набора. Это весьма уместно при создании более презентабельных заголовков, так как шрифт больших кеглей лучше смотрится с большими межбуквенными интервалами (с более свободным трэкингом). Также удобно таким

образом выделить отдельную фразу. В приведенном ниже примере мы используем команду `\letterspace` для разрядки заголовка и для выделения слова «void» в тексте, применив разрядку к его первым трем буквам.




```
\centering\mbox{\Large\textbf
 {\letterspace to 1.3\naturalwidth
 {The first day}}}
\begin{quotation}
In the beginning God created the heaven
and the earth. Now the earth was unformed
and \letterspace to1.7\naturalwidth{voi}d,
and darkness was upon the face of the
deep; and the spirit of God hovered over
the face of the waters.
\end{quotation}
```

3.1.2 ulem — выделение посредством подчеркивания

Вместо явного задания другого шрифта (вроде `\bfseries` или `\itshape`) LaTeX предпочитает [$\mathcal{L}$ 16–18], [$\mathcal{N}$94,100] пользоваться для шрифтового выделения командой `\emph` или декларацией `\em`[1]. В пакете ulem Дональда Арсено команда `\emph` переопределяется таким образом, что вместо курсива появляется подчеркивание. При этом внутри подчеркнутого текста допускается разрыв при переходе на другую строку и даже примитивный перенос слов. Каждое слово набирается в подчеркнутом боксе, поэтому автоматический перенос недопустим, однако принудительная расстановка знаков переноса (`\-`) возможна. Между словами тоже сохраняется подчеркивание, т. е. интервалы между словами становятся подчеркнутыми. Поскольку интервалы ограничены словами, могут возникнуть затруднения с синтаксическими пробелами (типа `"2.3pt"`). Для обработки подобных пробелов предприняты некоторые усилия. Если возникли проблемы, попробуйте заключить провинившуюся команду в фигурные скобки, так как все, что внутри фигурных скобок, находится внутри `\mbox`. Таким образом, фигурные скобки будут предотвращать растяжение и нежелательный разрыв строк для текста, который в них заключен. Обратите внимание, что этим пакетом не всегда корректно трактуются вложенные конструкции (см. приведенные ниже усилия получить правильные межсловные интервалы посредством помещения отдельных слов в фигурные скобки внутри выделяемого `\emph` выражения).

No, I did not act in the movie The Persecution and Assassination of Jean-Paul Marat, as performed by the Inmates of the Asylum of Charenton under the Direction of the Marquis de Sade! But I did see it.

```
No, I did \emph{not} act in the movie
\emph{\emph{The} \emph{Persecution} \emph{and}
\emph{Assassination} \emph{of}
\emph{Jean-Paul} \emph{Marat}, as performed
by the Inmates of the Asylum of Charenton
under the Direc\-tion of the Marquis de~Sade!}
But I \emph{did} see it.
```

[1] Относительно новых команд шрифтового выделения в LaTeX 2$_\varepsilon$ см. гл. 7.

Можно отключить действие этой команды посредством \normalem и \ULforem. Следующий пример демонстрирует другие возможности выделения:

Обычное подчеркивание (under-line), подчеркивание волной (under-wave), перечеркивание прямой (~~strike out~~), зачеркивание (~~cross out, X out~~).

```
Обычное подчеркивание (\uline{under-line}),\\
подчеркивание волной (\uwave{under-wave}), \\
перечеркивание прямой (\sout{strike out}), \\
зачеркивание (\xout{cross out, X out}).
```

3.1.3 xspace — гибкий пробел после макро

Небольшой пакет xspace Дэвида Карлайла определяет команду \xspace, предназначенную для использования в конце макро, которое написано преимущественно для употребления в тексте. Эта команда добавляет пробел, за исключением тех случаев, когда за макро следует знак препинания.

Команда\xspace избавляет вас от необходимости набирать \␣ или {} в большинстве случаев появления того или иного макро в тексте. Однако, если за одной из этих конструкций поместить команду \xspace, она не добавит пробела. Таким образом, можно быть уверенным, что добавление \xspace в конец уже существующего макро не приведет к изменениям в вашем документе. Возможные кандидаты для \xspace — это аббревиатуры типа e.g., и i.e.,.[2]

```
\newcommand{\eg}{e.g.,\xspace}
\newcommand{\ie}{i.e.,\xspace}
\newcommand{\etc}{etc.\@\xspace}
```

Обратите внимание на использование команды \@ для получения правильного интервала. Если ее поместить справа от знака препинания, то она предотвращает появление дополнительного пробела и точка не интерпретируется как точка в конце предложения[3]. Помещенная слева, эта команда побуждает LaTeX рассматривать точку как точку в конце предложения [$\mathcal{L}$ **14,154**], [$\mathcal{A}$ **91**].

Иногда команда \xspace может принять ошибочное решение и добавить пробел, когда этого не требуется. В таких случаях после макро следует поместить {}, чтобы добиться сжатия интервала.

Great Britain was unified in 1707.
Great Britain, the United States of America and Canada have close cultural links.

```
\newcommand{\USA}{United States of
                  America\xspace}
\newcommand{\GB}{Great Britain\xspace}
\GB was unified in 1707.\\ \GB, the \USA
and Canada have close cultural links.
```

[2] В настоящем издании также используются такие команды, например, «т.е. » задается следующим образом: \newcommand{\Cyrie}{т.\,е.\xspace}.

[3] По типографским правилам англоязычных стран интервал в конце предложения удваивается. — *Прим. ред.*

3.1.4 Выравнивание внутри абзаца

В некоторых документах абзацы набираются без выравнивания по правому краю (подобно представленному) [$\mathcal{L}$ **111–2**], [$\mathcal{A}$**126**]. Для этого в LaTeX'е есть окружение `flushleft`, позволяющее так оформить заключенный в скобки абзац. Однако параметры набора абзацев не универсальны, так как большинство окружений (такие как `minipage`, `tabular` и семейство `list`) и команд (типа `\parbox`, `\footnote` и `\caption`) восстанавливают выравнивание абзацев. Иначе говоря, они полагают расстояние `\rightskip` равным нулю. Чтобы внутри таких окружений и команд получить рваный правый край, можно в рамках их действия ввести команду `\setlength{\rightskip}{0pt plus 1fil}` . Внутри окружений типа `list` вместо `\rightskip` используют другую, растяжимую длину `\@rightskip`.

В следующем примере, в котором на внутреннем уровне применено окружение `minipage`, мы для получения требуемого эффекта переопределим команду `\rightskip`. Обратите внимание, что мы позволяем команде `\rightskip` ужимать максимум на 2 см, чтобы ограничить появление справа пустого (белого) пространства.

In the beginning God created the heaven and the earth. Now the earth was unformed and void, and darkness was upon the face of the deep; and the spirit of God hovered over the face of the waters.

```
\setlength{\rightskip}{0pt plus 2cm}
In the beginning God created the heaven
and the earth. Now the earth was unformed
and void, and darkness was upon the face
of the deep; and the spirit of God
hovered over the face of the waters.
```

Другие возможности оформления абзацев— это флаг вправо и центрирование посредством окружений `flushright` и `center` соответственно. В таких случаях разрыв строки обычно указывается командами `\\`, тогда как при рваном правом крае (обсуждавшееся выше окружение `flushleft`) вы можете поручить эту заботу самому LaTeX'у.

Три окружения, обсуждавшиеся выше в этом разделе, работают посредством изменения деклараций, управляющих набором абзацев в TeX'е. Эти декларации доступны также как команды LaTeX'а, согласно приведенной ниже [$\mathcal{L}$ **112**], [$\mathcal{A}$**127**] таблице соответствий.

окружение:	`center`	`flushleft`	`flushright`
команда:	`\centering`	`\raggedright`	`\raggedleft`

Эти команды не начнут новый абзац, если нет соответствующего окружения. Они могут быть использованы внутри другого окружения и в parbox, в частности, для управления выравниванием внутри p колонок окружений `array` или `tabular`. Отметим, однако, что в этом случае следует сделать соответствующее предупреждение, как это обсуждается на с. 127, где вводится соответствующая команда `\PreserveBackslash`.

Для управления межсловными интервалами в выровненном абзаце (пробелами между отдельными словами) служат несколько TeX'овских параметров; самые важные из них — `\tolerance` и `\emergencystretch`. Сделав их соответству-

ющими вашему документу, вы можете предотвратить большинство сообщений «Overfull box», не прибегая к разрыву строк вручную. Команда `\tolerance` задает величину того, как много межсловных интервалов в абзаце может отклоняться от заданного оптимального значения[4]. Это счетчик TEX'а (а не LATEX'а), и, следовательно, эта команда имеет необычный синтаксис назначения, например `\tolerance=500`. Нижние значения понуждают TEX оставаться как можно ближе к оптимуму; верхние же допускают очень свободный набор. По умолчанию часто задается 200. В случае, если TEX не в состоянии выдержать заданный интервал, в выводе будут боксы переполнения (т. е. строки, выступающие за поля, вроде этой). Увеличивая значение `\tolerance`, TEX станет также рассматривать плохие, но все еще допустимые разрывы строк, а не переадресовывать эту проблему вам для решения ее вручную. Разумные значения находятся между 50 и 9999 — не используйте выше 10000: это позволит TEX'у выводить сколь угодно дефектные строки (вроде этого безобразия).
Если вам действительно нужно разрывать строки автоматически, то лучше присвоить длине параметра `\emergencystretch` положительное значение. Если TEX не может разорвать абзац без того, чтобы не породить боксы переполнения (соблюдая указанный `\tolerance`) при положительном значении `\emergencystretch`, *он добавит эту длину как сжимаемый пробел в каждой* строке, допуская при этом разрыв строки, который до этого был отвергнут. В результате вы можете получить ряд сообщений о жидких строках (underful box messages), поскольку теперь все строки набраны свободно, но все же это выглядит лучше, чем одна расхлябанная строка в середине безукоризненного абзаца.

У LATEX'а есть две команды, влияющие на обсуждавшиеся выше параметры: задаваемая по умолчанию `\fussy` и `\sloppy`, которая допускает относительно плохие строки. Команда `\sloppy` иногда применяется LATEX'ом автоматически (в таких ситуациях, как, например, использование аргументов `\marginpar` или p колонок в окружении `tabular`), где редко удается сделать идеальный разрыв при узкой колонке.

3.1.5 doublespace — изменение интерлиньяжа

Параметр TEX'а `\baselineskip` служит для определения [$\mathcal{L}$ **94,155**], [$\mathcal{A}$**151**] *интерлиньяжа* — нормального вертикального расстояния между последовательными строками. В качестве стандартного интерлиньяжа LATEX задает интервал, примерно на 20% превышающий высоту шрифта (см. разд. 7.6.1 на с. 218). Поскольку не рекомендуется менять установленный `\baselineskip` внутри документа, LATEX предоставляет команду `\baselinestretch`, позволяющую делать глобальную замену интерлиньяжа `\baselineskip` во всех размерах шрифта.

Заметьте, что *после* команды `\renewcommand{\baselinestretch}{1.5}` интерлиньяж не увеличится тотчас же. Команды изменения размера шрифта (типа `\small`, `\Large`, etc.) должны сначала исполниться, чтобы новое значение стало действенным.

[4] Это оптимальное значение определяется шрифтом; см. разд. 7.7.1.

In the beginning God created the heaven and the earth. Now the earth was unformed and void, and darkness was upon the face of the deep; and the spirit of God hovered over the face of the waters.

```
\begin{spacing}{2}
In the beginning God created the
heaven and the earth. Now the earth
was unformed and void, and darkness
was upon the face of the deep; and
the spirit of God hovered over the
face of the waters.
\end{spacing}
```

Рис. 3.1. Абзац с увеличенным интерлиньяжем

интервал	10 pt	11 pt	12 pt
полтора	1.25	1.21	1.24
два	1.67	1.62	1.66

Таблица 3.1. Эффективные значения `\baselinestretch` для различных размеров шрифтов

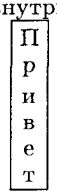
Пакет doublespace Стефена Пейджа определяет окружение `spacing`. Параметром *coef* служит значение `\baselinestretch` для текста, на которое распространяется это окружение.

`\begin{spacing}{`*coef*`}` ... *text* ... `\end{spacing}`

В примере на рис. 3.1 коэффициент «2» приводит к интерлиньяжу, превышающему «двойной интервал», принятый в некоторых изданиях. В этом случае интерлиньяж увеличивался дважды: один раз посредством `\baselineskip` (где TEX уже добавил около 20% расстояния между строками) и второй раз посредством установки `\baselinestretch`. «Двойной интервал» означает, что вертикальное расстояние между строками примерно вдвое больше размера шрифта. Поскольку `\baselinestretch` указывает отношение между требуемым расстоянием и `\baselineskip`, значения `\baselinestretch` для различных шрифтов, взятых в качестве основных для данных документов (при двух различных оптических интервалах), могут быть вычислены; они приведены в табл. 3.1.

3.1.6 picinpar — набор абзацев с прямоугольными окнами

Пакет picinpar Фридгельма Совы (который опирался на работу Алана Хёнига) позволяет оставлять внутри абзаца «окна». Основным окружением является `window`, у которого имеется два варианта: `figwindow` и `tabwindow`. Они, соответственно, обеспечивают подписи под рисунками и заголовки таблиц. Окружение `figwindow` аналогично окружению `wrapfigure`, описанному в разд. 6.4.2. Как там объясняется, следует быть осторожным при одновременном использовании `figwindow` и обычного окружения `figure`, так как последнее может проскользнуть мимо



В этом случае внутри абзаца в центре помещается набранное вертикально слово. Нетрудно понять, что также легко вставляются и таблицы при помощи окружения tabwindow. Когда абзац заканчивается, как здесь, а окно еще не поместилось целиком, то следующий абзац начинается справа от окна.

```
\begin{window}[1,c,%
 {\fbox{\shortstack{П\\р\\и\\в\\е\\т}}},{}]
В этом случае внутри абзаца в центре
помещается набранное вертикально слово.
Нетрудно понять, что также легко
вставляются и таблицы при помощи
окружения \texttt{tabwindow}.\par Когда
абзац заканчивается, как здесь, а окно
еще не поместилось целиком, то следующий
абзац начинается справа от окна.
\end{window}
```

Рис. 3.2. «Окно» в абзаце

неплавающих окружений figwindow, в результате чего образуются номера с отсутствующими рисунками.

`\begin{window}`[*nl,align,material,explanation*]

nl	Количество строк до окна;
align	Равнение окна внутри абзаца (l — по умолчанию в левый край, c — по центру и r — в правый край);
material	Материал, размещаемый в окне;
explanation	Пояснительный текст, относительно содержимого окна (например, подпись или заголовок для figwindow и tabwindow).

На рис. 3.2 показано, как поместить окно в середине абзаца.

Обратите внимание, что использование команды \shortstack позволяет расположить буквы друг под другом, начиная сверху.

Рисунок или таблицу также можно разместить внутри абзаца. Такая ситуация представлена на рис. 3.4: здесь справа помещена карта Великобритании. Окружение figwindow снабжает рисунок также подписью.

3.1.7 shapepar — набор абзацев необычной формы

Пакет shapepar Дональда Арсено вводит команду \shapepar, позволяющую набирать абзацы необычного вида. Общий размер регулируется автоматически, так что текст заполняет всю заданную фигуру. Предназначенные для такого оформления абзацы не должны содержать выключных математических формул или материала \vadjust, который вводится командами \vspace. Подобные абзацы форматируются в несколько итераций, пока его размер и форма не станут правильными. Поскольку это медленный процесс, пакет рекомендуется применять для открыток или приглашений, но не для оформления всей книги целиком!

Команду \shapepar следует помещать в начале абзаца, и тогда она действует на весь абзац:

Is this a dagger which I see before me, The handle toward my hand? Come, let me clutch thee. I have thee not, and yet I see thee still. Art thou not, fatal vision, sensible To feeling as to sight? or art thou but A dagger of the mind, a false creation, Proceeding from the heat-oppressed brain? I see thee yet, in form as palpable As this which now I draw. Thou marshall'st me the way that I was going; And such an instrument I was to use. Mine eyes are made the fools o' the other senses, Or else worth all the rest; I see thee still, And on thy blade and dudgeon gouts of blood, Which was not so before. (*Macbeth*, Act II, Scene 1).



Figure 3.3: United Kingdom

```
\begin{figwindow}[3,r,%
{\fbox
{\epsfig{file=ukmap.eps,width=27mm}}},%
                              {United Kingdom}]
Is this a dagger which I see before me,
The handle toward my hand? Come, let me
clutch thee. I have thee not, and yet I
see thee still. Art thou not, fatal
vision, sensible To feeling as to
sight? or art thou but A dagger of the
mind, a false creation, Proceeding from
the heat-oppressed brain? I see thee
yet, in form as palpable As this which
now I draw. Thou marshall'st me the
way that I was going; And such an
instrument I was to use. Mine eyes are
made the fools o' the other senses, Or
else worth all the rest; I see thee
still, And on thy blade and dudgeon
gouts of blood, Which was not so
before.
(\emph{Macbeth}, Act II, Scene 1).
\end{figwindow}
```

Рис. 3.4. Абзац с рисунком «в оборку»

`\shapepar{`*shape_spec*`}` текст абзаца

Параметр *shape_spec* описывает форму абзаца, используя особые синтаксические правила. Так что интересующимся рекомендуем обратиться непосредственно к файлам пакета. Существуют четыре заданные формы, с тремя из которых связаны специальные команды: `\diamondpar`, `\squarepar` и `\heartpar`. Результат их применения показан ниже.

◇
Infandum, regina, iubes renovare dolorem, Troiannas ut opes et lamentabile regnum cruerint Danai; quaeque ipse miserrima vidi, et quorum pars magna fui. Quis talia fando Myrmidonum Dolopumve aut duri miles Ulixi temperet a lacrimis? Et iam nox umida caelo praecipitat, suadentque cadentia sidera somnos. Sed si tantus amor casus cognoscere nostros et breviter Troiae supremum audire laborem, quamquam animus meminisse horret, luctuque refugit, incipiam.
◇

```
\diamondpar{Infandum, regina,...}
```

Infandum, regina, iubes renovare dolorem, Troianas ut opes et lamentabile regnum cruerint Danai; quaeque ipse miserrima vidi, et quorum pars magna fui. Quis talia fando Myrmidonum Dolopumve aut duri miles Ulixi temperet a lacrimis? Et iam nox umida caelo praecipitat, suadentque cadentia sidera somnos. Sed si tantus amor casus cognoscere nostros et breviter Troiae supremum audire laborem, quamquam animus meminisse horret, luctuque refugit, incipiam.

```
\squarepar{Infandum, regina,...}
```

Infandum, regina, iubes renovare dolorem, Troianas ut opes et lamentabile regnum cruerint Danai; quaeque ipse miserrima vidi, et quorum pars magna fui. Quis talia fando Myrmidonum Dolopumve aut duri miles Ulixi temperet a lacrimis? Et iam nox umida caelo praecipitat, suadentque cadentia sidera somnos. Sed si tantus amor casus cognoscere nostros et breviter Troiae supremum audire laborem, quamquam animus meminisse horret, luctuque refugit, incipiam.

♡

```
\heartpar{Infandum, regina,...}
```

Infandum, regina, iubes renovare dolorem, Troianas ut opes et lamentabile regnum cruerint Danai; quaeque ipse miserrima vidi, et quorum pars magna fui. Quis talia fando Myrmidonum Dolopumve aut duri miles Ulixi temperet a lacrimis? Et iam nox umida caelo praecipitat, suadentque cadentia sidera somnos. Sed si tantus amor casus cognoscere nostros et breviter Troiae supremum audire laborem, quamquam animus meminisse horret, luctuque refugit, incipiam.

```
\shapepar\nutshape{Infandum, ...}
```

3.2 Структуры перечня

Перечни — весьма типичная конструкция LaTeX'а, и они часто используются для построения многих LaTeX'овских окружений при оформлении тех или иных списков. Стандартные окружения LaTeX'а для перечней `enumerate`, `itemize` и `description` описываются в следующем разделе и там же показано, как их создать. Перечни общего вида обсуждаются в разд. 3.2.2.

3.2.1 Модификация стандартных перечней

Создать три стандартных LaTeX'овских окружения для перечней относительно просто: этому посвящены три соответствующих подраздела ниже. Для того чтобы сделать глобальное переопределение этих окружений, заданных по умолчанию, следует переопределить некие параметры в преамбуле, отвечающие за перечни, либо задавать их локально.

Создание окружения `enumerate`

Окружение LaTeX'а `enumerate` для нумерованных перечней [$\mathcal{L}$ **26,165–6**], [$\mathcal{A}$ **128,130**] характеризуется командами и формами представления, приведенными в табл. 3.2. В первой строке показаны названия счетчиков, отвечающих за четыре возможных уровня перечня. Во второй и третьей строках даны команды, представляющие эти счетчики и их определение по умолчанию в стандартных LaTeX'овских классах. В четвертой, пятой и шестой строках содержатся соответственно команды, вид по умолчанию и примеры нумерации реальных перечней, напечатанных в таком окружении.

Ссылки на элемент нумерованного перечня осуществляются посредством `\theenumi`, `\theenumii` и других аналогичных команд, которым предшествуют соответственно команды `\p@enumi`, `\p@enumii`, и т. д.. В последних трех строках таблицы приводятся эти команды, их определения по умолчанию и примеры по-

	Первый уровень	Второй уровень	Третий уровень	Четвертый уровень
счетчик	`enumi`	`enumii`	`enumiii`	`enumiv`
представление	`\theenumi`	`\theenumii`	`\theenumiii`	`\theenumiv`
определение по умолчанию	`\arabic{enumi}`	`\alph{enumii}`	`\roman{enumiii}`	`\Alph{enumiv}`
метка	`\labelenumi`	`\labelenumii`	`\labelenumiii`	`\labelenumiv`
вид по умолчанию	`\theenumi.`	`(\theenumii)`	`\theenumiii.`	`\theenumiv.`
пример нумерации	1., 2.	(a), (b)	i., ii.	A., B.
префикс	`\p@enumi`	`\p@enumii`	`\p@enumiii`	`\p@enumiv`
определение по умолчанию	`{}`	`\theenumi`	`\theenumi (\theenumii)`	`\p@enumiii \theenumiii`
пример ссылки	1, 2	1a, 2b	1(a)i, 2(b)ii	1(a)iA, 2(b)iiB

Таблица 3.2. Команды, управляющие окружением `enumerate`

лучающихся ссылок. Очень важно корректно использовать определения как представлений, так и команд построения ссылок, чтобы получить правильные ссылки.

Теперь, применяя только что усвоенный материал, можно получать разнообразные описания нумерованных перечней.

В нашем первом примере переопределены счетчики первого и второго уровней как прописные римские цифры и латинские буквы соответственно. Это выглядит как значение счетчика, за которым поставлена точка. Для префиксной команды ссылки `\p@enumi` использовано значение по умолчанию из табл. 3.2.

I. **Introduction**

A. **Applications**
Motivation for research and applications related to the subject.

B. **Organization**
Explain organization of the report, what is included, and what is not.

II. **Literature Survey**

III. **Proposed Research**

q1=I-A q2=I-B q3=II q4=III

```
\makeatletter
\renewcommand{\theenumi}{\Roman{enumi}}
\renewcommand{\labelenumi}{\theenumi.}
\renewcommand{\theenumii}{\Alph{enumii}}
\renewcommand{\labelenumii}{\theenumii.}
\renewcommand{\p@enumii}{\theenumi--}
\makeatother
\begin{enumerate}
\item \textbf{Introduction}
 \begin{enumerate}
  \item \textbf{Applications}  \newline
   Motivation for research and applications
   related to the subject.       \label{q1}
  \item \textbf{Organization}  \newline
   Explain organization of the report, what
   is included, and what is not. \label{q2}
 \end{enumerate}
 \item \textbf{Literature Survey} \label{q3}
 \item \textbf{Proposed Research} \label{q4}
\end{enumerate}
q1=\ref{q1} q2=\ref{q2} q3=\ref{q3} q4=\ref{q4}
```

Вы можете также украсить поле `enumerate`, добавив что-нибудь к метке. В примере ниже был выбран знак § в качестве префикса для каждой метки первого *уровня в перечне*.

§1. текст внутри перечня, еще текст внутри перечня, текст внутри перечня,

§2. текст внутри перечня, еще текст внутри перечня, текст внутри перечня,

§3. текст внутри перечня, еще текст внутри перечня, текст внутри перечня, еще текст внутри перечня.

w1=1 w2=2

```
\renewcommand{\labelenumi}{\S\theenumi.}
\begin{enumerate}
\item текст внутри перечня, еще текст
      внутри перечня, текст внутри
      перечня, \label{w1}
\item текст внутри перечня, еще текст
      внутри перечня, текст внутри
      перечня, \label{w2}
\item текст внутри перечня, еще текст внутри
      перечня, текст внутри перечня, еще
      текст внутри перечня.
\end{enumerate}
w1=\ref{w1}  w2=\ref{w2}
```

Вы можете даже захотеть использовать различные маркеры для последовательных меток. Так, в следующем примере используются литеры из PostScript-шрифта Цапфа—Дингбата. В этом случае не существует прямого способа автоматического получения правильных ссылок посредством команд `\ref`. Можно, однако, использовать окружение `dingautolist`, определяемое в пакете pifont, представляющего собой часть системы PSNFSS (см. разд. 11.9.3 на с. 375). Обратите внимание также, что для добавления чего-нибудь в команду `\setcounter` следует *использовать пакет* calc (*см.* разд. A.4).

① текст внутри перечня, еще текст внутри перечня; текст внутри перечня, еще текст внутри перечня;

② текст внутри перечня, еще текст внутри перечня; текст внутри перечня, еще текст внутри перечня;

③ текст внутри перечня, еще текст внутри перечня; текст внутри перечня, еще текст внутри перечня.

```
\newcounter{local}\renewcommand{\labelenumi}
   {\setcounter{local}{171+\value{enumi}}%
    \ding{\value{local}}}
\begin{enumerate}
\item текст внутри перечня, еще текст внутри
      перечня; текст внутри перечня, еще
      текст внутри перечня;
\item текст внутри перечня, еще текст внутри
      перечня; текст внутри перечня, еще
      текст внутри перечня;
\item текст внутри перечня, еще текст внутри
      перечня; текст внутри перечня, еще
      текст внутри перечня.
\end{enumerate}
```

И, наконец, для тех, кто не хочет самостоятельно писать эти команды, существует пакет enumerate Дэвида Карлайла, в котором переопределено окружение `enumerate` с факультативным аргументом, определяемым стилем, в котором печатается документ. Этот аргумент может содержать один из элементов `A`, `a`, `I`, `i` или `1` для набора значения счетчика, используемого (соответственно) в стилях `\Alph`, `\alph`, `\Roman`, `\roman` или `\arabic`.

Более того, в качестве аргумента можно взять любое LaTeX'овское выражение, однако, если элементы `A`, `a`, `I`, `i` или `1` не интерпретируются как описано выше, они должны быть специфицированы внутри группы `{}`.

	Первый уровень	Второй уровень	Третий уровень	Четвертый уровень
Команды	`\labelitemi`	`\labelitemii`	`\labelitemiii`	`\labelitemiv`
Определение	`\m@th\bullet`	`\bfseries--`	`\m@th\ast`	`\m@th\cdot`
Представление	•	–	*	·

Таблица 3.3. Команды, управляющие перечнем `itemize`

Указанная выше внутренняя команда `\m@th` *устанавливает локально в нуль значение* параметра `\mathsurround` (дополнительный пробел вокруг формул в тексте). К команде `\m@th` следует прибегать, когда математический режим вводится для нематематических целей, чтобы избежать дополнительных пробелов в случаях, когда `\mathsurround` было сделано положительным в файле класса документа.

Команды перекрестных ссылок `\label` и `\ref` можно использовать как в стандартном окружении `enumerate`. Заметьте, однако, что с этим пакетом команда `\ref` даст только выбранное представление значения счетчика, а не всю метку. На печати получится значение в том же стиле, что и `\item`, как это определяется одним из элементов `A`, `a`, `I`, `i` или `1` в качестве факультативного аргумента.

EX i. текст 1-го элемента уровня 1. Еще текст 1-го элемента уровня 1.

EX ii. текст 2-го элемента уровня 1.

пример a) текст 1-го элемента уровня 2. Еще текст 1-го элемента уровня 2.

пример b) текст 2-го элемента уровня 2.

A-1 текст 1-го элемента уровня 1 для списка 2.

A-2 текст 2-го элемента уровня 1 для списка 2.

Показано, как ссылаться на элементы перечня: 'i', 'iia' и '1' или, более полно, 'EX i.' и 'A-1'.

```
\begin{enumerate}[EX i.]
\item текст 1-го элемента уровня~1.
   Еще текст 1-го элемента уровня~1.%
   \label{LA}
\item текст 2-го элемента уровня~1.
 \begin{enumerate}[{пример} a)]
  \item текст 1-го элемента уровня~2.
   Еще текст 1-го элемента уровня~2.%
   \label{LB}
  \item текст 2-го элемента уровня~2.
 \end{enumerate}
\end{enumerate}
\begin{enumerate}[{A}-1]
\item текст 1-го элемента уровня~1 для
списка~2.                    \label{LC}
\item текст 2-го элемента уровня~1 для
 списка~2.
\end{enumerate}
Показано, как ссылаться на элементы
перечня: '\ref{LA}', '\ref{LB}' и
'\ref{LC}' или, более полно,
'EX~\ref{LA}.' и 'A-\ref{LC}'.
```

Создание окружения `itemize`

Для простого ненумерованного перечня `itemize` [$\mathcal{L}$ **26,165–6**], [$\mathcal{A}$ **128,129**] метки определяются командами, указанными в табл. 3.3.

Для создания перечня с другими метками следует переопределить команду порождения меток. Это можно сделать локально для одного конкретного перечня как в примере ниже, а можно — глобально, воспользовавшись в преамбуле доку-

мента переопределением \labelitemi. Следующий простой перечень представляет собой стандартный itemize с PostScript'овскими метками из шрифта Цапфа—Дингбата (см. разд. 11.9.3) для первого уровня:

- ☞ Текст 1-го элемента перечня.
- ☞ Текст 1-го предложения 2-го элемента перечня. Второе предложение.
- ☞ Это предложение в тексте 3-го элемента перечня.

```
\newenvironment{MYitemize}{%
  \renewcommand\labelitemi{\ding{43}}%
          \begin{itemize}}{\end{itemize}}
\begin{MYitemize}
\item Текст 1-го элемента перечня.
\item Текст 1-го предложения 2-го
    элемента перечня. Второе предложение.
\item Это предложение в тексте 3-го
      элемента перечня.
\end{MYitemize}
```

Создание окружения description

Окружение description можно использовать для изменения команды \descriptionlabel порождения метки [$\mathcal{L}$ 26,165–6], [$\mathcal{M}$ 132]. В следующем примере шрифт для меток меняется с полужирного на рубленый.

A. текст внутри перечня, текст внутри перечня, текст внутри перечня, еще текст внутри перечня;

B. текст внутри перечня, текст внутри перечня, текст внутри перечня, еще текст внутри перечня;

C. текст внутри перечня, текст внутри перечня, текст внутри перечня, еще текст внутри перечня.

```
\renewcommand{\descriptionlabel}[1]%
        {\hspace{\labelsep}\textsf{#1}}
\begin{description}
\item[A.] текст внутри перечня, текст
      внутри перечня, текст внутри
      перечня, еще текст внутри перечня;
\item[B.] текст внутри перечня, текст
      внутри перечня, текст внутри
      перечня, еще текст внутри перечня;
\item[C.] текст внутри перечня, текст
      внутри перечня, текст внутри
      перечня, еще текст внутри перечня.
\end{description}
```

В стандартных LaTeX'овских классах начальная точка бокса для метки в окружении description выдвинута влево на расстояние \labelsep от левого поля соответствующего окружения, так что команда \descriptionlabel в приведенном выше примере прежде всего добавляет значение \labelsep, чтобы установить метку непосредственно в край левого поля.

3.2.2 Создание собственных перечней

Для получения перечня существует окружение list [$\mathcal{L}$ 112,166–8], [$\mathcal{M}$ 288–290]:

\begin{list}{*default_label*}{*decls*} *item_list* \end{list}

Параметр *default_label* представляет собой текст, который используется в качестве метки, если команда \item выступает без факультативного аргумента. Параметр *decls* устанавливает различные геометрические параметры окружения list (см. рис. 3.5). На этом рисунке также приведены значения этих параметров по

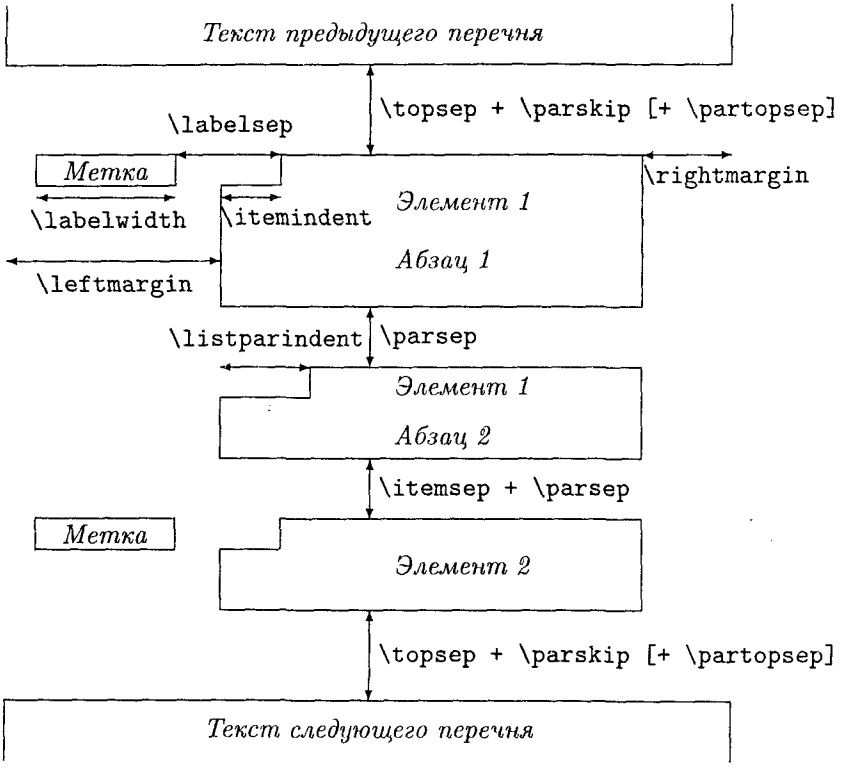


Рис. 3.5. Структура перечня в общем виде

Длины по вертикали

Все вертикальные пробелы, о которых говорится ниже, представляют собой растяжимые длины, зависящие от размера шрифта и уровня перечня.

`\topsep` пробел между первым элементом и предыдущим абзацем.

`\partopsep` дополнительный пробел, добавляемый к `\topsep`, когда окружение начинает новый абзац.

`\itemsep` пробел между последовательными элементами.

`\parsep` пробел между абзацами внутри элемента.

Длины по горизонтали

`\leftmargin` пробел между левым полем окружения (или страницы, если это верхний уровень перечня) и левым полем перечня. Он должен быть неотрицательным и зависит от уровня перечня.

`\rightmargin` аналогично `\leftmargin`, только для правого поля. Его значение обычно равно 0 pt.

`\listparindent` дополнительный отступ в начале каждого абзаца перечня, за исключением того, с которого начинается новый элемент `\item`. Может быть отрицательным, но обычно равен 0 pt.

`\itemindent` дополнительный отступ, добавляемый по горизонтали в первой строке элемента. Это имеет отношение к тому месту ссылки, с которого начальное положение метки вычисляется вычитанием значений `\labelsep` и `\labelwidth`. Обычно оно равно 0 pt.

`\labelwidth` номинальная ширина бокса, содержащего метку. Если естественная ширина метки $\leq$`\labelwidth`, то метка набирается флагом вправо внутри бокса ширины `\labelwidth`. В противном случае используется бокс естественной ширины, и тогда отступ делается в этой строке.

`\labelsep` пробел между концом бокса метки и текстом первого элемента. По умолчанию это значение равно 0.5 em.

умолчанию. Все эти параметры могут быть переопределены при помощи команд `\setlength` или `\addtolength`.

Некоторые LaTeX'овские окружения определяются при помощи `list` (например, `quote`, `quotation`, `center`, `flushleft` и `flushright`). Обратите внимание, что эти окружения имеют только один элемент и что команда `\item[]` задается окружением определений.

Рассмотрим в качестве примера окружение `quote`, чье определение дает идентичные вещи как в правом, так и в левом поле. Ниже приводится простой вариант `Quote`, который идентичен `quote`, за тем исключением, что текст взят в двойные кавычки. Обратите внимание на специальные меры предосторожности, которые предпринимаются с целью удалить ненужные пробелы после предшествующего (`\ignorespaces`) и перед последующим (`\unskip`) текстом.

... предшествующий текст.

"Some quoted text, more quoted text. Some quoted text, more quoted text."

Последующий текст ...

```
\newenvironment{Quote}% Definition of Quote
  {\begin{list}{}{%
      \setlength{\rightmargin}{\leftmargin}}%
        \item[]``\ignorespaces}
  {\unskip''\end{list}}
\ldots\ предшествующий текст.
\begin{Quote}
   Some quoted text, more quoted text.
   Some quoted text, more quoted text.
\end{Quote}
Последующий текст~\ldots
```

Общие перечни часто используются для документирования компьютерных команд или программных функций. Так, в следующих примерах участвуют окружение `entry` и его разновидности. В каждом случае название описываемой темы вводится как параметр команды `\item`.

Дескриптор: Возвращение из функции. Если попадаем на верхний уровень, интерпретатор просто останавливает работу, как если бы был достигнут конец ввода.

Ошибки: Нет.

Значения возврата:
Все аргументы производят действие возврата к вызывающей программе.

```
\begin{entry}
\item[Дескриптор]
 Возвращение из функции. Если
 попадаем на верхний уровень,
 интерпретатор просто
 останавливает работу, как если
 бы был достигнут конец ввода.
\item[Ошибки]   Нет.
\item[Значения возврата]\mbox{}\\
 Все аргументы производят действие
 возврата к вызывающей программе.
\end{entry}
```

Здесь продемонстрирована типичная проблема, возникающая в перечнях типа описаний, когда текст метки шире, чем место, отведенное под эту метку. Стандартный LaTeX непосредственно за текстом метки помещает текст раздела *описание*. Обычно это нас не устраивает, поэтому, чтобы улучшить внешний вид перечня, описательную часть перечня нужно начинать со следующей строки. Для этого нужно на той же строке разместить пустой бокс, а за ним — команду '\\'.

В предыдущем примере команда \makelabel (команда с одним аргументом, определяющим внешний вид метки \item) и два геометрических параметра (\labelwidth и \leftmargin) были переопределены, как показано ниже[5].

```
\newcommand{\entrylabel}[1]{\mbox{\textsf{#1:}}\hfil}
\newenvironment{entry}
   {\begin{list}{}%
          {\renewcommand{\makelabel}{\entrylabel}%
            \setlength{\labelwidth}{35pt}%
            \setlength{\leftmargin}{\labelwidth+\labelsep}%
          }%
   }%
   {\end{list}}
```

В оставшейся части этого раздела будут изучены разные возможности управления шириной и взаимным расположением меток и описаний. Один из вариантов — просто изменить ширину метки. Для этого в окружении имеется аргумент, устанавливающий необходимую ширину поля метки (обычно берется величина самой широкой метки). Обратите внимание, что переопределяется команда \makelabel, где вы указываете, как следует набирать метку. Поскольку переопределение находится внутри определения окружения Ventry, литера # должна быть заменена на ##, чтобы сигнализировать LaTeX'у, что вы обращаетесь к аргументу команды \makelabel, а не к аргументу внешнего окружения.

Дескриптор:	Возвращение из функции. Если попадаем на верхний уровень, интерпретатор просто останавливает работу, как если бы был достигнут конец ввода.	`\begin{Ventry}{Значения возврата}`
		`\item[Дескриптор]`
		`Возвращение из функции. Если попадаем на верхний уровень, интерпретатор просто останавливает работу, как если бы был достигнут конец ввода.`
		`\item[Ошибки] Нет.`
		`\item[Значения возврата] Все аргументы производят действие возврата к вызывающей программе.`
		`\end{Ventry}`

Ошибки:	Нет.	
Значения возврата:	Все аргументы производят действие возврата к вызывающей программе.	

Окружение Ventry определяется посредством

```
\newenvironment{Ventry}[1]%
   {\begin{list}{}{\renewcommand{\makelabel}[1]{\textsf{##1:}\hfil}%
      \settowidth{\labelwidth}{\textsf{#1:}}%
      \setlength{\leftmargin}{\labelwidth+\labelsep}}}%
   {\end{list}}
```

[5] В этом и в некоторых последующих примерах мы использовали пакет calc и развитые управляющие структуры LaTeX 2_ε (см. приложение А, разд. А.4 и А.5).

Однако с полиграфической точки зрения несколько перечней с разными по ширине полями меток на одной и той же странице недопустимы. Другой способ состоит в задании некой определенной ширины поля метки. Если метка шире, чем \labelwidth, то дополнительный пустой бокс приводит к тому, что описательная часть начинается с новой строки. Это совпадает с традиционным методом представления дисплейного материала для пособий по UNIX.

```
\newlength{\Mylen}
\newcommand{\Lentrylabel}[1]{%
  \settowidth{\Mylen}{\textsf{#1:}}%
  \ifthenelse{\lengthtest{\Mylen > \labelwidth}}%
      {\parbox[b]{\labelwidth}%          term > labelwidth
         {\makebox[0pt][l]{\textsf{#1:}}\\}}%
      {\textsf{#1:}}%                    term < labelwidth
  \hfil\relax}
\newenvironment{Lentry}
   {\renewcommand{\entrylabel}{\Lentrylabel}%
    \begin{entry}}
   {\end{entry}}
```

Как можно видеть в последней строке приведенного выше определения, окружение Lentry определяется в терминах окружения entry. Команда \entrylabel порождения меток теперь заменена командой \Lentrylabel. Последняя сначала устанавливает длину переменной \Mylen равной ширине метки, затем сравнивает эту длину с \labelwidth. Если метка у́же, чем \labelwidth, то набор происходит на той же самой строке, что и описание, в противном случае бокс становится нулевой ширины, набор текста простирается так далеко вправо, как это требуется, после чего происходит переход на следующую строку, на которой и размещается описание.

Дескриптор:
: Возвращение из функции. Если попадаем на верхний уровень, интерпретатор просто останавливает работу, как если бы был достигнут конец ввода.

Ошибки: Нет.

Значения возврата:
: Все аргументы производят действие возврата к вызывающей программе.

```
\begin{Lentry}
\item[Дескриптор]
 Возвращение из функции. Если
 попадаем на верхний уровень,
 интерпретатор просто
 останавливает работу, как если
 бы был достигнут конец ввода.
\item[Ошибки] Нет.
\item[Значения возврата] Все
 аргументы производят действие
 возврата к вызывающей
 программе.
\end{Lentry}
```

Вот еще одна возможность, использующая многострочный набор меток.

Дескриптор:	Возвращение из функции. Если попадаем на верхний уровень, интерпретатор просто останавливает работу, как если бы был достигнут конец ввода.
Ошибки:	Нет.
Значения возврата:	Все аргументы производят действие возврата к вызывающей программе.

```
\begin{Mentry}
\item[Дескриптор]
 Возвращение из функции. Если
 попадаем на верхний уровень,
 интерпретатор просто
 останавливает работу, как если
 бы был достигнут конец ввода.
\item[Ошибки]   Нет.
\item[Значения\\возврата] Все
 аргументы производят действие
 возврата к вызывающей программе.
\end{Mentry}
```

В этом примере мы снова в качестве основного используем окружение `entry`, но на этот раз команда `\Mentrylabel` заменяется командой `\entrylabel`. Идея состоит в том, чтобы разбить широкие метки на несколько строк. Следует предпринять некоторые меры предосторожности, чтобы можно было переносить каждое первое слово абзаца, в связи с чем в определение вводится команда `\hspace{0pt}`. Материал набирается внутри бокса абзаца правильной ширины `\labelwidth`, который затем выравнивается по верхней строке и левому краю в боксе, который в свою очередь помещается внутрь бокса высоты 1 ex и без глубины. Таким образом, LaTeX не понимает, что материал простирается ниже первой строки.

```
\newcommand{\Mentrylabel}[1]%
   {\raisebox{0pt}[1ex][0pt]{\makebox[\labelwidth][l]%
       {\parbox[t]{\labelwidth}{\hspace{0pt}\textsf{#1:}}}}}
\newenvironment{Mentry}%
   {\renewcommand{\entrylabel}{\Mentrylabel}\begin{entry}}%
   {\end{entry}}
```

Можно создать окружение с автоматическим приращением счетчика, включив команду `\usecounter` в декларацию окружения `list`. Эту функцию демонстрирует окружение `Notes`, производящее последовательность замечаний. В этом случае для автоматической генерации текста для меток используется первый параметр окружения `list`.

```
\newcounter{notes}
\newenvironment{Notes}%
  {\begin{list}{\textsc{Примечание} \arabic{notes}. }{\usecounter{notes}%
        \setlength{\labelsep}{0pt}\setlength{\leftmargin}{0pt}%
        \setlength{\labelwidth}{0pt}%
        \setlength{\listparindent}{0pt}}}%
  {\end{list}}
```

После декларирования счетчика `notes` по умолчанию определяется метка окружения `Notes` как слово ПРИМЕЧАНИЕ, набранное капителью, за которым следует значение счетчика `notes`, представленное арабскими цифрами с точкой.

ПРИМЕЧАНИЕ 1. Это текст первого примечания. Еще немного текста первого примечания.

ПРИМЕЧАНИЕ 2. Это текст второго примечания. Еще немного текста второго примечания.

```
\begin{Notes}
\item Это текст первого примечания.
      Еще немного текста первого примечания.
\item Это текст второго примечания.
      Еще немного текста второго примечания.
\end{Notes}
```

3.3 Подражание машинописному шрифту

Часто бывает нужно представить информацию «буквально» (verbatim), т. е. так, как она была набрана на терминале (введена с клавиатуры компьютера). Для удобства читателя, однако, бывает полезно выделить особым образом слово или фрагмент текста. Это приводит к необходимости использовать другие команды LaTeX'а внутри «буквального» текста. В настоящем разделе описываются пакеты, позволяющие облегчить эту задачу.

3.3.1 `alltt` — окружение типа `verbatim`

Пакет alltt Лесли Лэмпорта определяет окружение `alltt`. Его действие подобно окружению `verbatim`, за тем исключением, что бэкслеш '\' и фигурные скобки '{' и '}' имеют их обычный смысл. Таким образом, внутри окружения `alltt` можно использовать другие команды и окружения; см. рис. 3.6.

3.3.2 verbatim — стиль для литературного текста

Пакет verbatim Райнера Шопфа представляет собой переделанные LaTeX'овские окружения `verbatim` и `verbatim*`. Одним из его главных преимуществ является то, что он допускает сколь угодно длинные тексты типа `verbatim`. В пакете также вводится окружение `comment`, позволяющее пропустить весь закомментированный текст между командами `\begin{comment}` и `\end{comment}`. Однако в нем переопределена команда `\verb` LaTeX'а, которая успешнее обнаруживает пропущенный закрывающий ограничитель.

Этот пакет также предоставляет приемы для внедрения пользовательских расширений для определения созданных им окружений типа `verbatim`. Несколько таких расширений реализованы в пакете moreverb, описанном ниже.

```
Можно изменить шрифт, к примеру, так:
emphasized text.

Строка специальных символов # $ % ^ & ~ _

Вставить текст из файла "foo.tex", набрав
'\input{foo}'. Остерегайтесь
того, чтобы "return" начинал новую строку,
иначе, если foo.tex заканчивается на
"return", вы получите дополнительную пустую
строку, если не будете внимательны.

Можно также получить математические
формулы, набрав \(...\) или
\[...\]. Помните, что '$' даст всего
лишь знак доллара. То же верно и для других
специальных символов '^', '_' внутри
математического режима: используйте
\sp как в a^2,
используйте \sb как
в a_2.
```

```
\begin{alltt}
Можно изменить шрифт, к примеру, так:
{\em}emphasized text\/}.

Строка специальных символов # $ % ^ & ~ _

Вставить текст из файла "foo.tex", набрав
'\(\backslash\)input\{foo\}'. Остерегайтесь
того, чтобы "return" начинал новую строку,
иначе, если foo.tex заканчивается на
"return", вы получите дополнительную пустую
строку, если не будете внимательны.

Можно также получить математические
формулы, набрав \verb!\(...\)! или
\verb!\[...\]!. Помните, что '$' даст всего
лишь знак доллара. То же верно и для других
специальных символов '^', '_' внутри
математического режима: используйте
\(\backslash\)sp как в \(a\sp{2}\),
используйте \(\backslash\)sb как
в \(a\sb{2}\).
\end{alltt}
```

Рис. 3.6. Окружение `alltt`

3.3.3 moreverb — дополнительные команды и окружения типа verbatim

Пакет moreverb Ангуса Даггана основывается на обсуждавшемся выше пакете verbatim. Он предоставляет некоторые интересные предопределенные команды типа `verbatim` для написания файлов и чтения из них, равно как и окружения для получения листингов.

`\begin{verbatimwrite}{`*filename*`}`

Окружение `verbatimwrite` (изначально написанное Райнером Шопфом) пишет свое содержимое в файл *filename*. Справа представлен исходный файл (табулятор обозначен как ▷), а слева можно видеть, как это реализовано.

```
*       *       *       *
Верхний уровень
        Первый уровень
                Второй уровень
        Вложенный       табулятор
```

```
\begin{verbatimwrite}{ testtab.out}
*▷*▷*▷*
Верхний уровень
▷Первый уровень
▷▷Второй уровень
▷Вложенный▷табулятор
\end{verbatimwrite}
```

`\begin{verbatimtab}[`*tabstop*`]`

Окружение `verbatimtab` позволяет литерам *табулятора* (изображаемым посредством ▷) занять точно отведенное для табулятора число пустых литер. (Напоминаем, что в стандартном LaTeX'е литеры табулятора рассматриваются как единый пробел.) Расстояния между последовательными табуляциями могут быть заданы как факультативный аргумент. По умолчанию оно равно восьми пустым литерам.

```
12345678901234567890123456789012345678
|       one     two     three   four

|   one two three   four
```

```
\begin{verbatimtab}
12345678901234567890123456789012345678
\end{verbatimtab}

\begin{verbatimtab}
▷one▷two▷three▷four
\end{verbatimtab}

\begin{verbatimtab}[4]
▷one▷two▷three▷four
\end{verbatimtab}
```

`\verbatimtabinput[`*tabstop*`]{`*filename*`}`

Команда `\verbatimtabinput` введет файл *filename*, заданный как обязательный аргумент. Расстояние между последовательными табуляциями можно задать как факультативный аргумент *tabstop*. Обратите внимание, что в приведенном ниже примере текст расположен при помощи табулятора из четырех пустых литер, тогда как в начале раздела, в файле `testtab.out`, этот же текст был оформлен задаваемым по умолчанию значением табулятора (восемь пустых литер).

```
*   *   *   *
Верхний уровень
    Первый уровень
        Второй уровень
    Вложенный   табулятор
```

```
\verbatimtabinput[4]{testtab.out}
```

Окружение `boxedverbatim` используется для того, чтобы взять в рамку текст verbatim.

```
Окружение boxedverbatim
очерчивает рамку вокруг
окружения verbatim.
```

```
\begin{boxedverbatim}
Окружение boxedverbatim
очерчивает рамку вокруг
окружения verbatim.
\end{boxedverbatim}
```

Ниже приводится пример окружения verbatimcmd, подобного окружению alltt, описанному в разд. 3.3.1.

```
Окружение verbatimcmd можно
использовать для включения
команд в
окружение LaTeX'a verbatim.
Обратите внимание, что пробелы после
команд значимые, так что для
разделения слов следует использовать
пустые группы {}. В противном
случае вы можете обнаружить
 странные пробелы
в вашем выводе.

А вот выключная формула:
```

$$a^{b_c d}$$

```
\begin{verbatimcmd}
Окружение verbatimcmd можно
использовать для включения
{\normalfont\itshape{}команд} в
окружение \LaTeX'a verbatim.
Обратите внимание, что пробелы после
команд значимые, так что для
разделения слов следует использовать
пустые группы \{\}. В противном
случае вы можете обнаружить
{\normalfont\itshape странные} пробелы
в вашем выводе.

А вот выключная формула:
\[a\sp{b\sb{c}d}\]
\end{verbatimcmd}
```

```
\begin{listing}[step]{firstline} ... \end{listing}
\begin{listing*}[step]{firstline} ... \end{listing*}
```

Окружение listing напоминает окружение verbatim с той разницей, что здесь строки нумеруются. Вариант listing* со звездочкой представляет пустую литеру как ␣. Факультативный аргумент *step* задает шаг между нумерованными строками (по умолчанию он равен 1, и это позволяет нумеровать все строки), тогда как обязательный аргумент *firstline* присваивает номер только первой строке. Если *step* отличается от «1», то первым номером будет «*step*+*firstline*−1», например, если окружение listing было вызвано командой \begin{listing}[2]{3}, будет нумероваться каждая вторая строка, начиная с номера 2 + 3 − 1, т. е. 4.

```
    Окружение listing нумерует строки,
4   благодаря факультативному
    аргументу, задающему шаг нумерации
6   строк (строка 1 всегда нумеруется,
    если представлена), и обязательному
8   аргументу, который указывает
    начальную строку.
```

```
\begin{listing}[2]{3}
Окружение listing нумерует строки,
благодаря факультативному
аргументу, задающему шаг нумерации
строк (строка 1 всегда нумеруется,
если представлена), и обязательному
аргументу, который указывает
начальную строку.
\end{listing}
```

```
\begin{listingcont} ... \end{listingcont}
\begin{listingcont*} ... \end{listingcont*}
```

Окружение listingcont(*) (пример на следующей странице) продолжает нумерацию с того места, где она прервалась в окружении listing(*). Версия этого

окружения без звездочки вводит ширину табуляции, равную по умолчанию в восемь литер, тогда как версия со звездочкой табулятора не касается.

```
10  Окружение listingcont
    продолжает нумерацию с того
12  места, где она прервалась
    в окружении listing.
14  Оба окружения listing и
    listingcont распространяют
16  ширину табуляции "      "
    до значения по умолчанию~8.
```

```
\begin{listingcont}
Окружение listingcont
продолжает нумерацию с того
места, где она прервалась
в окружении listing.
Оба окружения listing и
listingcont распространяют
ширину табуляции "▷"
до значения по умолчанию~8.
\end{listingcont}
```

`\listinginput[`*step*`]{`*firstline*`}{`*filename*`}`

Команда `\listinginput` позволяет прочитать файл *filename* как листинг. Нумерация начинается с первой строки в *firstline* и затем продолжается с шагом *step*.

В качестве примера возьмем файл `testtab.out`, написанный при помощи окружения `verbatimwrite`, как было продемонстрировано в начале предыдущего раздела, и прочтем его (и перенумеруем) при помощи команды `\listinginput{1}{testtab.out}`.

```
1   *       *       *       *
2   Верхний уровень
3           Первый уровень
4                   Второй уровень
5           Вложенный       табулятор
```

3.4 Примечания: подстрочные, на полях, выносные

LaTeX имеет средства для набора «вставок», таких как примечания на полях, сноски, рисунки и таблицы. В настоящем разделе мы более пристально рассмотрим разного рода примечания, а плавающие объекты будут подробно описаны в гл. 6.

3.4.1 Создание сносок

Сноски (подстрочные примечания) в LaTeX'е получать обычно просто и для этого имеется мощный механизм, обеспечивающий размещение материала внизу стра-

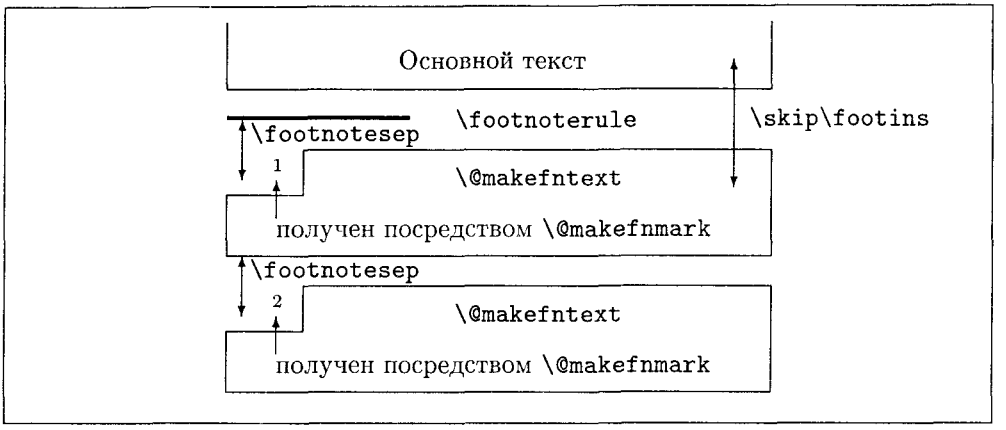


Рис. 3.7. Схематическое построение сносок

ницы[6]. Этот материал может состоять из нескольких абзацев и включать перечни, формулы, таблицы и т. д.

Для создания сносок LaTeX требует несколько параметров [$\mathcal{L}$ **91,156**], [$\mathcal{A}$ **306–8**]. Они схематично изображены на рис. 3.7.

Между сносками в основном тексте и сносками внутри окружения `minipage` делается строгое разграничение [$\mathcal{L}$ **99,195**] (см. также [40, с. 104, 181]). Первые нумеруются при помощи счетчика `footnote`, тогда как для сносок внутри `minipage` переопределяется команда `\footnote` и используется счетчик `mpfootnote`. Таким образом, ссылка на сноску порождается либо командой `\thefootnote`, либо командой `\thempfootnote` в зависимости от контекста. По умолчанию в тексте это арабские цифры[7], а внутри окружения `minipage` это строчные буквы. Чтобы оформить сноски по-другому, можете переопределить команду, задав, например, для сносок в тексте такую:

текст текст текст* текст текст
текст текст† текст

* Первая
† Вторая

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
текст текст текст\footnote{Первая}
текст текст текст текст\footnote{Вторая} текст
```

Для сносок, получаемых при помощи команды `\footnote` внутри окружения `minipage`, используется счетчик `mpfootnote`, и они помещаются под блоком, порождаемым окружением `minipage`. Тем не менее, если внутри окружения `minipage` используется команда `\footnotemark`, то она создаст такую же ссылку на сноску,

[6] Интересная и всеобъемлющая дискуссия по этому предмету прошла на страницах журнала *Cahiers GUTenberg* [2, 56], издаваемого группой франкоязычных пользователей TeX'а.

[7] Имеется в виду сквозная нумерация по всему материалу. В отечественной литературе чаще применяется своя нумерация сносок для каждой страницы, т. е. при переходе на другую страницу счетчик должен быть обнулен.— *Прим. ред.*

как если бы она была в основном тексте, т. е. обратится к счетчику `footnote` и воспользуется командой `\thefootnote` для представления сноски. Такое поведение позволяет внутри окружения `minipage` давать сноски, нумеруемые в ряду сносок основного текста и помещаемые внизу соответствующей страницы книги: надо поместить `\footnotemark` внутри `minipage`, а соответствующую `\footnotetext` — после.

Сноски внутри minipage нумеруются при помощи строчных букв[a]. Этот текст обращается к сноске внизу страницы[8].

[a] *Внутри minipage*

```
\begin{minipage}{\linewidth}
 Сноски внутри minipage нумеруются
 при помощи строчных
 букв\footnote{Внутри minipage}.
 \par Этот текст обращается к сноске
 внизу страницы\footnotemark.
\end{minipage}\footnotetext{Внизу страницы}
```

В классе article сноски имеют сквозную нумерацию по всему документу. В классах report и book сноски нумеруются внутри глав. Можно это изменить, воспользовавшись командой `\@addtoreset` (см. разд. 2.3.1). Не пытайтесь, однако, при помощи этого механизма нумеровать свои сноски внутри страницы. Поскольку LaTeX при формировании завершающих страниц смотрит вперед, ваши сноски скорее всего будут перенумерованы неправильно. Чтобы сноски нумеровались постранично, воспользуйтесь пакетом footnpag Йоахима Шрода. Для управления нумерацией сносок пакет записывает информацию в дополнительный файл «*jobname*`.fot`». Таким образом, вам следует по меньшей мере дважды прогнать LaTeX, чтобы у сносок получились правильные номера. Кроме того, при таком использовании факультативный аргумент `\footnote` больше недоступен; но это, видимо, не очень существенная потеря, поскольку при таком оформлении сносок этот аргумент не понадобится вовсе.

Команда `\@makefnmark` генерирует значок сноски при помощи команды `\@thefnmark`, которая следит за текущим значением сноски, создаваемым `\thefootnote` или поддерживаемым вами посредством факультативного аргумента команды `\footnote`. Вот определение этого значка сноски по умолчанию:

```
\renewcommand{\@makefnmark}{\mbox{$^{\@thefnmark}$}}
```

Вид стандартной сноски можно изменить, воспользовавшись параметрами, которые приведены на рис. 3.7 и описаны ниже:

`\footnotesize` Размер шрифта для сносок (см. также табл. 7.1).

`\footnotesep` Высота распорки, помещаемой перед каждой сноской. Если это значение превышает `\baselineskip`, используемый для `\footnotesize`, то дополнительное пустое пространство по вертикали будет вставлено перед каждой сноской. Относительно дополнительной информации о распорках см. разд. А.2.3.

[8] Внизу страницы

\skip\footins Команда TEX'а низкого уровня, определяющая расстояние между основным текстом и началом сносок. Ее значение можно изменить посредством команд \setlength или \addtolength, поместив \skip\footins в первый аргумент, например,

```
\addtolength{\skip\footins}{3mm}
```

\footnoterule Макро для вычерчивания линейки, отделяющей сноски от основного текста. Она помещается непосредственно после вертикального интервала \skip\footins. Она занимает нулевое вертикальное пространство, т. е. используется отрицательный пробел для компенсации той положительной толщины, которую она имеет, например:

```
\renewcommand{footnoterule}{\vspace*{-3pt}%
   \rule{.4\columnwidth}{0.4pt}\vspace*{2.6pt}}
```

Можно соорудить вычурную «линейку», например, состоящую из точек:

```
\renewcommand{footnoterule}{\vspace*{-3pt}%
   \qquad\dotfill\qquad\vspace*{2.6pt}}
```

Команда \footnote выполняет \@makefntext внутри \parbox с шириной \columnwidth. Вариант по умолчанию выглядит так:

```
\newcommand{\@makefntext}[1]%
   {\noindent\makebox[1.8em][r]{\@makefnmark}#1}
```

В более сложном варианте может быть использовано окружение list. Каждая сноска набирается как перечень, состоящий из одного элемента.

```
\renewcommand{\@makefntext}[1]{\setlength{\parindent}{0pt}%
  \begin{list}{}{\setlength{\labelwidth}{1.5em}%
    \setlength{\leftmargin}{\labelwidth}%
    \setlength{\labelsep}{3pt}\setlength{\itemsep}{0pt}%
    \setlength{\parsep}{0pt}\setlength{\topsep}{0pt}%
    \footnotesize}\item[\hfill\@makefnmark]#1%
  \end{list}}
```

LATEX не позволяет использовать внутри \footnote другую команду \footnote, как это принято для других объектов, но можно внутри первой сноски использовать команду \footnotemark и затем поместить текст в виде сноски к сноске как аргумент команды \footnotetext.

А что, если вы решили сослаться на некую сноску? Для этого можно воспользоваться обычным LATEX'овским механизмом \label и \ref, но

можно определить и свою собственную команду, чтобы иметь свои особые ссылки:

Это некий текст.[1]
... как показано в сноске (1) на с. 92,...

[1] Текст внутри сноски, на которую ссылаются.

```
\newcommand{\fnref}[1]{~(\ref{#1})}
Это некий текст.\footnote{Текст внутри сноски,
на которую ссылаются\label{fn:myfoot}.}\par
... как показано в сноске\fnref{fn:myfoot} на
с.~\pageref{fn:myfoot},...
```

Стандартный LaTeX не позволяет создавать сноски внутри табличного материала. В разд. 5.6.2 будет представлено несколько путей решения этой проблемы.

Пакет fnpara Доминика Вуястика и Криса Роули полностью меняет представление сносок[9]. В этом пакете сноски набираются[10] подряд в одном абзаце, а не друг под другом. Это подходит для текстов типа критических замечаний, содержащих много коротких сносок.[11]

3.4.2 Примечания на полях

`\marginpar[`*left-text*`]{`*right-text*`}`

Команда `\marginpar` порождает примечания на полях следующим образом: на поле выносится текст, заданный в виде аргумента, причем его первая строка совпадает с той строкой основного текста, где появилась команда `\marginpar` [$\mathcal{L}$ **61,178**], [$\mathcal{A}$ **170–1**]. Если задан только обязательный аргумент *right-text*, то при односторонней печати текст появится на правом поле, при двусторонней — на внешнем и при форматировании в виде двух колонок — на ближайшем поле. Если присутствует и факультативный аргумент, то он управляет левым полем, тогда как другой (обязательный) аргумент относится к правому полю.

При использовании примечаний на полях необходимо помнить о нескольких важных вещах. Во-первых, команда `\marginpar` не начинает абзац, иными словами, если ее поставить перед первым словом абзаца, вертикальное выравнивание не совпадет с началом абзаца. Во-вторых, если поле узкое, а слова длинные (скажем, немецкие), то первое такое слово следует предварить командой `\hspace{0pt}`, разрешающей перенос этого слова.

`\marginlabel` использует для примечаний внешнее поле.

Эти две потенциальные проблемы упрощаются введением команды `\marginlabel{`*text*`}`, которая начинается пустым боксом `\mbox{}`, печатает примечание с рваным левым краем и добавляет `\hspace{0pt}` перед аргументом.

```
\newcommand{\marginlabel}[1]
   {\mbox{}\marginpar{\raggedleft\hspace{0pt}#1}}
```

[9] Вот пример сноски, набранной подряд в одном абзаце. [10] Вот другой пример сноски, набранной подряд в одном абзаце. [11] См., например, систему EDMAC [42] на предмет того, какого рода сноски и примечания в конце текста приняты в критических работах.

По умолчанию при односторонней печати примечания размещаются на внешнем поле. Это соглашение можно изменить посредством следующих деклараций:

`\reversemarginpar`	Примечания размещаются на поле, противоположном заданному по умолчанию;
`\normalmarginpar`	примечания размещаются на поле, заданном по умолчанию.

Как показано в табл. 4.2, стиль примечаний на полях задается тремя параметрами: `\marginparwidth`, `\marginparsep` и `\marginparpush`.

3.4.3 Выносные примечания

В научных и учебных изданиях примечания обычно группируют в конце каждой главы или в конце всего документа. Такие примечания будем называть выносными. Стандартный LaTeX не поддерживает выносных примечаний, но существует несколько способов их создания.

В пакете endnotes Джона Лаваньино выносные примечания набираются так же, как подстрочные. Для этого используется внешний файл с расширением `.ent`: в нем содержится текст выносных примечаний. После прогона этот файл можно удалить, поскольку каждый раз генерируется новая версия.

Чтобы содержимое файла включить в документ, используется команда типа следующей:

```
\newpage\begingroup%2
\setlength{\parindent}{0pt}\setlength{\parskip}{2ex}
\renewcommand{\enotesize}{\normalsize}
\theendnotes\endgroup%2
```

Эта команда начинает новую страницу, открывает группу, чтобы ограничить переопределения параметров и команд, переопределяет параметры абзаца и устанавливает размер шрифта примечаний (`\enotesize`). Затем команда `\theendnotes` напечатает выносные примечания, собранные во внешнем файле, в соответствующем месте текста. Наконец, закрывается группа, так что переопределения действуют локально.

При помощи этого пакета вы можете оформить свои сноски как выносные примечания, воспользовавшись командой

```
\renewcommand{\footnote}{\endnote}
```

Пользовательский интерфейс в случае выносных примечаний весьма похож на ситуацию со сносками, достаточно лишь вместо слова «foot» подставить «end». Следующий пример иллюстрирует принцип использования выносных примечаний в случае, когда вы сохраняете текст при помощи команды `\endnote`, а затем

печатаете весь собранный текст в соответствующей точке документа, указанной пользователем.

Это просто текст.[1] И это текст.[2] Еще немного текста.[3]

Примечания

[1]Первое примечание.
[2]Второе примечание.
[3]Третье примечание.

```
Это просто текст.\endnote{Первое примечание.}
И это текст.\endnote{Второе примечание.}
Еще немного текста.\endnote{Третье примечание.}

\theendnotes   % здесь разместить
\bigskip       % выносные примечания
Дальше опять идет текст.
```

Дальше опять идет текст.

3.5 Использование многоколонного набора

В стандартном LaTeX'е предусмотрена возможность набора текста в одну или две колонки, но на одной и той же странице нельзя представить и то, и другое.

Пакет multicol (написанный Франком Миттельбахом) определяет окружение `multicols`, которое позволяет переключаться между различными многоколонными выводами на одной и той же странице. Опция twocolumn допускает расположение сносок только в правой колонке, что достигается при помощи пакета ftnright (также Франка Миттельбаха).

3.5.1 multicol — гибкий способ работы с многоколонным документом

В стандартном LaTeX'е предусмотрена возможность получать документы в виде одно- или двухколонного текста (twocolumn) [$\mathcal{L}$ **82,162**], [$\mathcal{A}$**124**]. Однако нельзя представить в двухколонном виде только часть страницы, потому что команды `\twocolumn` и `\onecolumn` начинают всегда с новой страницы. Кроме того, колонки никогда не подравниваются. В результате иногда можно получить расположение материала несколько странного вида.

В пакете multicol эти проблемы решаются посредством определения окружения `multicols` со следующими свойствами:

- Можно получать любое количество колонок (до десяти), заполняющих несколько страниц.
- Когда работа окружения завершается, колонки балансируются по длине, так что в результате они будут почти одинаковой длины.
- Это окружение можно использовать внутри других, таких как `figure` или `minipage`, при этом получится бокс, содержащий текст, разбитый на нужное количество колонок. Это означает, что теперь нет необходимости вручную форматировать подобные ситуации.

- Колонки могут быть отделены друг от друга вертикальными линейками любой заданной пользователем толщины.
- Форматирование может быть выполнено глобально или для некоторого определенного окружения.

3.5.2 Набор текста в колонках

`\begin{multicols}{`*columns*`}[`*preface*`][`*skip*`]`

Как правило, окружение начинается с задания числа требуемых колонок.

Этот текст разбит на несколько колонок. Если колонки слишком узкие, старайтесь оформить их с рваным правым краем.

```
\begin{multicols}{2}
Этот текст разбит на несколько
колонок. Если колонки слишком узкие,
старайтесь оформить их с рваным
правым краем.
\end{multicols}
```

Вас, однако, может интересовать случай, когда есть преамбула в виде материала, оформленного в одну колонку. Это достигается при помощи факультативного аргумента *preface*. LaTeX постарается выполнить указания *preface* и начнет многоколонный текст на той же странице.

Маленький совет

Этот текст разбит на несколько колонок. Если колонки слишком узкие, старайтесь оформить их с рваным правым краем.

```
\begin{multicols}{2}
      [\section*{Маленький совет}]
Этот текст разбит на несколько
колонок. Если колонки слишком узкие,
старайтесь оформить их с рваным
правым краем.
\end{multicols}
```

Если на текущей странице нет достаточного свободного пространства, окружение `multicols` начинает новую страницу. Этим управляет глобальный параметр. При использовании *preface* значение этого параметра, заданное по умолчанию, может оказаться слишком малым. В этом случае можно либо изменить значение *global* по умолчанию (см. ниже), либо отрегулировать это значение в окружении *current* посредством другого факультативного параметра *skip* следующим образом:

```
\begin{multicols}{3}
        [\section*{Указатель}]
        [7cm]
  Текст Текст Текст Текст ...
\end{multicols}
```

Если на данной странице свободно менее 7 см, будет начата новая.

3.5.3 Создание окружения multicols

Окружение multicols имеет дело с несколькими параметрами, управляющими форматированием. Их смысл разъясняется в следующих подразделах. Значения по умолчанию сведены в табл. 3.4 (длины) и табл. 3.5 (счетчики). Если не утверждается обратное, то все изменения должны быть сделаны прежде, чем начнется окружение, для которого они предназначены.

Свободное пространство

Окружение multicols прежде всего проверяет размер свободного пространства слева на странице: оно должно быть по крайней мере равно \premulticols или значению факультативного аргумента *skip*, если таковой задан. Если такого пространства нет, то возникает \newpage. Аналогичные действия предпринимаются по достижении конца окружения, только на этот раз используется параметр длины \postmulticols. До и после окружения вставляется вертикальный пробел длины \multicolsep.

Ширина колонок и разделители

Внутри окружения multicols автоматически вычисляется ширина колонок при заданном количестве колонок и текущем значении \linewidth. Между каждыми двумя колонками остается пустое пространство \columnsep.

Вертикальные линейки

Между двумя колонками помещается линейка ширины \columnseprule. Если установлено значение параметра 0pt, линейка подавляется.

Этот текст разбит на несколько колонок.

Если колонки слишком узкие, старайтесь

оформить их с рваным правым краем.

```
\setlength{\columnseprule}{0pt}
\begin{multicols}{3}
  \raggedright
  Этот текст разбит на несколько
  колонок. Если колонки слишком узкие,
  старайтесь оформить их с рваным
  правым краем.
\end{multicols}
```

Если вы выбрали ширину линейки, превосходящую ширину пространства, разделяющего колонки, то линейка напечатается поверх текста.

Форматирование колонок

По умолчанию (с установкой\flushcolumns), окружение multicols стремится уровнять все колонки по длине, вставляя нужные вертикальные пробелы в колонки. Если выбрано \raggedcolumns, то дополнительные пробелы будут помещены внизу каждой колонки.

\premulticols	50.0pt
\postmulticols	20.0pt
\multicolsep	12.0pt plus 4.0pt minus 3.0pt
\columnsep	10.0pt
\columnseprule	0.0pt

Таблица 3.4. Параметры длины, используемые в multicols

collectmore	0
unbalance	0
columnbadness	10000
finalcolumnbadness	9999
tracingmulticols	0

Таблица 3.5. Счетчики, используемые в multicols

В конце работы окружения остаток текста будет размещен так, чтобы колонки подровнялись. Если в левой колонке больше текста, следует воспользоваться счетчиком unbalance. При этом пробелы в основном будут добавляться в левую колонку. Счетчик unbalance укажет количество дополнительных строк, которые будут добавлены в левую колонку. Это делается автоматически и в конце работы multicols значение счетчика обнуляется.

Этот текст разбит на несколько колонок. Если колонки слишком узкие, старайтесь оформить их с рваным правым краем.

```
\begin{multicols}{3} \raggedright
  Этот текст разбит на несколько
колонок. Если колонки слишком узкие,
старайтесь оформить их с рваным
правым краем.
  \setcounter{unbalance}{1}
\end{multicols}
```

В дальнейшем сбалансированность колонок по длине управляется двумя счетчиками: columnbadness и finalcolumnbadness. Как только LaTeX построит боксы (типа колонки), вычисляется значение дефектности (badness), отражающее качество бокса. При этом оптимальным будет нулевое значение, а 10000 с точки зрения LaTeX'а бесконечно дефектно[12]. Если в процессе сравнения дефектности возможных решений все колонки, за исключением последней, имеют дефектность выше columnbadness, такое решение игнорируется. Когда алгоритм, наконец, находит решение и оно выглядит как дефектность последней колонки и больше finalcolumnbadness, то эта колонка с дополнительным пространством внизу колонки будет короткой. Можно прослеживать работу алгоритма, установив по-

[12] *Если в боксе происходит переполнение, то дефектность устанавливается в 100000, чтобы отметить этот особый случай.*

ложительное значение для счетчика `tracingmulticols` (более высокие значения дают более четкую информацию).

3.5.4 Плавающие объекты и сноски в multicol

Плавающие объекты (как, например, рисунки и таблицы) поддерживаются в `multicols` лишь частично. Можно использовать варианты плавающих окружений со звездочкой, т. е. таких, что располагаются на суммарную ширину всех колонок. Однако плавающие колонки и `\marginpar` не поддерживаются.

Сноски помещаются внизу страницы на полную ширину, а не под отдельной колонкой. При определенных обстоятельствах сноска может оказаться не на той же странице, что и ссылка на нее в тексте. Если такое произойдет, то `multicols` выдаст предупреждение. В этом случае надо проверить сомнительную страницу, и если текст сноски и ссылка на нее действительно оказались на разных страницах, следует решать проблему локально, поставив команду `\pagebreak` в критическом месте. Причина такого поведения кроется в том, что окружение `multicols` должно просматривать материал вперед и собирать его, и может оказаться не в состоянии использовать весь собранный позже материал. Объем обзора вперед управляется счетчиком `collectmore`.

3.5.5 ftnright — сноски в правой колонке при двухколонном окружении

Иногда при двухколонном наборе бывает удобно сгруппировать все сноски внизу правой колонки. Эта возможность реализована в пакете ftnright (Франк Миттельбах). Работа пакета продемонстрирована на рис. 3.8, представляющего собой первую страницу авторской документации к пакету ftnright. Видно, как различные примечания собираются в нижней части правой колонки.

Основная идея пакета ftnright состоит в том, чтобы все сноски из всех колонок на странице собрать вместе и поместить их внизу правой колонки. Между сносками и текстом оставляется достаточно пустого места, а сами сноски набираются более мелким шрифтом[13]. Далее, значок сноски располагается на строке, а не в виде верхнего индекса[14].

Этот пакет используется с большинством других классических файлов LaTeX'а. Пакет ftnright, разумеется, эффективен только при работе с документами, представленными в виде двухколонника, заданного опцией twocolumn в команде `\documentclass`. В большинстве случаев лучше использовать ftnright как самый последний пакет, чтобы быть уверенными, что другие опции на него не повлияют.

[13] В некоторых журналах, к сожалению, используется один и тот же шрифт для основного текста и для сносок, что мешает отличать их друг от друга.

[14] Разумеется, речь идет только о значке сноски перед ее текстом, а не о ссылке внутри основного текста, в котором маленькие значок или цифра над строкой не будут препятствовать мыслительному процессу.

Footnotes in a multi-column layout*

Frank Mittelbach

August 10, 1991

1 Introduction

The placement of footnotes in a multi-column layout always bothered me. The approach taken by LaTeX (i.e., placing the footnotes separately under each column) might be all right if nearly no footnotes are present. But it looks clumsy when both columns contain footnotes, especially when they occupy different amounts of space.

In the multi-column style option [5], I used page-wide footnotes at the bottom of the page, but again the result doesn't look very pleasant since short footnotes produce undesired gaps of white space. Of course, the main goal of this style option was a balancing algorithm for columns which would allow switching between different numbers of columns on the same page. With this feature, the natural place for footnotes seems to be the bottom of the page[1] but looking at some of the results it seems best to avoid footnotes in such a layout entirely.

Another possibility is to turn footnotes into endnotes, i.e., printing them at the end of every chapter or the end of the entire document. But I assume everyone who has ever read a book using such a layout will agree with me, that it is a pain to search back and forth, so that the reader is tempted to ignore the endnotes entirely.

When I wrote the article about "Future extensions of TeX" [6] I was again dissatisfied with the outcome of the footnotes, and since this article should show certain aspects of high quality typesetting, I decided to give the footnote problem a try and modified the LaTeX output routine for this purpose. The layout I used was inspired by the yearbook of the Gutenberg Gesellschaft Mainz [1]. Later on, I found that it is also recommended by Jan White [9]. On the layout of footnotes I also consulted books by Jan Tschichold [8] and Manfred Simoneit [7], books, I would recommend to everyone being able to read German texts.

1.1 Description of the new layout

The result of this effort is presented in this paper and the reader can judge for himself whether it was successful or not.[2] The main idea for this layout is to assemble the footnotes of all columns on a page and place them all together at the bottom of the right column. Allowing for enough space between footnotes and text, and in addition, setting the footnotes in smaller type[3] I decided that one could omit the footnote separator rule which is used in most publications prepared with TeX.[4] Furthermore, I decided to place the footnote markers[5] at the baseline instead of raising them as superscripts.[6]

All in all, I think this generates a neat layout, and surprisingly enough, the necessary changes to the LaTeX output routine are nevertheless astonishingly simple.

1.2 The use of the style option

This style option might be used together with any other style option for LaTeX which does not change the three internals changed by `ftnright.sty`.[7] In most cases, it is best to use this style option as the very last option in the `\documentstyle` command to make sure that its settings are not overwritten by other options.[8]

*. The LaTeX style option `ftnright` which is described in this article has the version number v1.0d dated 92/06/19. The documentation was last revised on 92/06/19.

1. You can not use column footnotes at the bottom, since the number of columns can differ on one page.

2. Please note, that this option only changed the placement of footnotes. Since this article also makes use of the `doc` option [4], that assigns tiny numbers to code lines sprincled throughout the text, the resulting design is not perfect.

3. The standard layout in *TUGboat* uses the same size for footnotes and text, giving the footnotes, in my opinion, much too much prominence.

4. People who prefer the rule can add it by redefining the command `\footnoterule` [2, p. 156]. Please, note, that this command should occupy no space, so that a negative space should be used to compensate for the width of the rule used.

5. The tiny numbers or symbols, e.g., the '5' in front of this footnote.

6. Of course, this is only done for the mark preceeding the footnote text and not the one used within the main text where a raised number or symbol set in smaller type will help to keep the flow of thoughts, uninterrupted.

7. These are the macros `\@startcolumn`, `\@makecol` and `\@outputdblcol` as we will see below. Of course, the option will take only effect with a document style using a twocolumn layout (like `ltugboat`) or when the user additionally specifies `twocolumn` as a document style option in the `\documentstyle` command.

8. The `ltugboat` option (which is currently set up as a style option instead of a document style option which it actually is) will overwrite

Рис. 3.8. Расположение текста и сносок при использовании пакета ftnright

3.6 Простое управление версиями

В окружении comment (предлагаемом в пакете verbatim) есть возможность игнорировать в процессе форматирования некоторые элементы. Пакет version, обсуждаемый в оставшейся части этой главы, делает шаг вперед, допуская управление версиями на элементарном уровне.

Пакет version (Стефен Беллантони) определяет окружения и команды, обеспечивающие LaTeX средствами некоторого рода управления версиями.

Чтобы воспользоваться этим средством, нужно где-то в начале документа поставить команду управления версиями окружений:

`\includeversion{`*versionname*`}`

Декларируется, что материал внутри окружений *versionname* будет обработан (набран) LaTeX'ом обычным образом.

`\excludeversion{`*versionname*`}`

Декларируется, что материал внутри окружений *versionname* будет проигнорирован (не набран) LaTeX'ом.

Аргумент *versionname* представляет собой имя, выбранное пользователем. Его следует определять в соответствии с тем, сколькими окружениями вы хотите управлять. Например:

Ответ ДА.

```
\includeversion{YES}\excludeversion{NO}
Ответ
\begin{NO}НЕТ.\end{NO}%
\begin{YES}ДА.\end{YES}
```

Пакет version также определяет окружение comment, материал которого по умолчанию игнорируется. Это поведение можно, однако, изменить, задав команду `\includeversion{comment}`.

AB
A Игнорируется по умолчанию. B

```
A\begin{comment} Обычно игнорируется.
 \end{comment}B

\includeversion{comment}
A\begin{comment} Игнорируется
  по умолчанию.
 \end{comment}B
```

ГЛАВА 4

Макет полосы набора

Текст документа обычно располагается в прямоугольной области на странице — это так называемое *тело* текста. Над телом находится *верхний колонтитул*, а под телом — *нижний колонтитул*. Колонтитулы могут состоять из одной и более строк и включать номер страницы, информацию о текущих главе, разделе, времени или дате, а также другие пометки. Пространства слева и справа от тела текста называются *полями*. Как правило, они остаются пустыми, но иногда могут быть использованы под небольшие фрагменты текста, типа замечаний или аннотаций — так называемые *заметки на полях*.

Размер, вид и расположение этих полей при выводе на экран или бумагу, а также вид и содержание колонтитулов называются *макетом полосы набора*. В этой главе объясняется, как задавать разные макеты полос. Зачастую в одном и том же документе бывают нужны разные макеты. Например, первая страница главы (спусковая полоса), на которой находится заголовок главы, компонуется иначе, чем остальные страницы этой главы.

Стандартные классы документов LaTeX'а предоставляют возможность форматирования документа для (*двусторонней*) печати. Макет полосы при односторонней печати отличается от ее макета при двусторонней печати. В первом случае поля полос с нечетными номерами те же, что и поля четных полос, тогда как во втором случае нужно позаботиться о том, чтобы границы текста с обеих сторон страницы совпадали. Вообще говоря, речь идет о *внутреннем* и *внешнем* полях. При двусторонней печати «внутреннее» означает левое поле для нечетных полос и правое поле для четных полос, тогда как при односторонней печати «внутреннее» всегда означает левое поле. В раскрытой книге нечетные полосы мы видим справа.

4.1 Геометрические параметры макета полосы набора

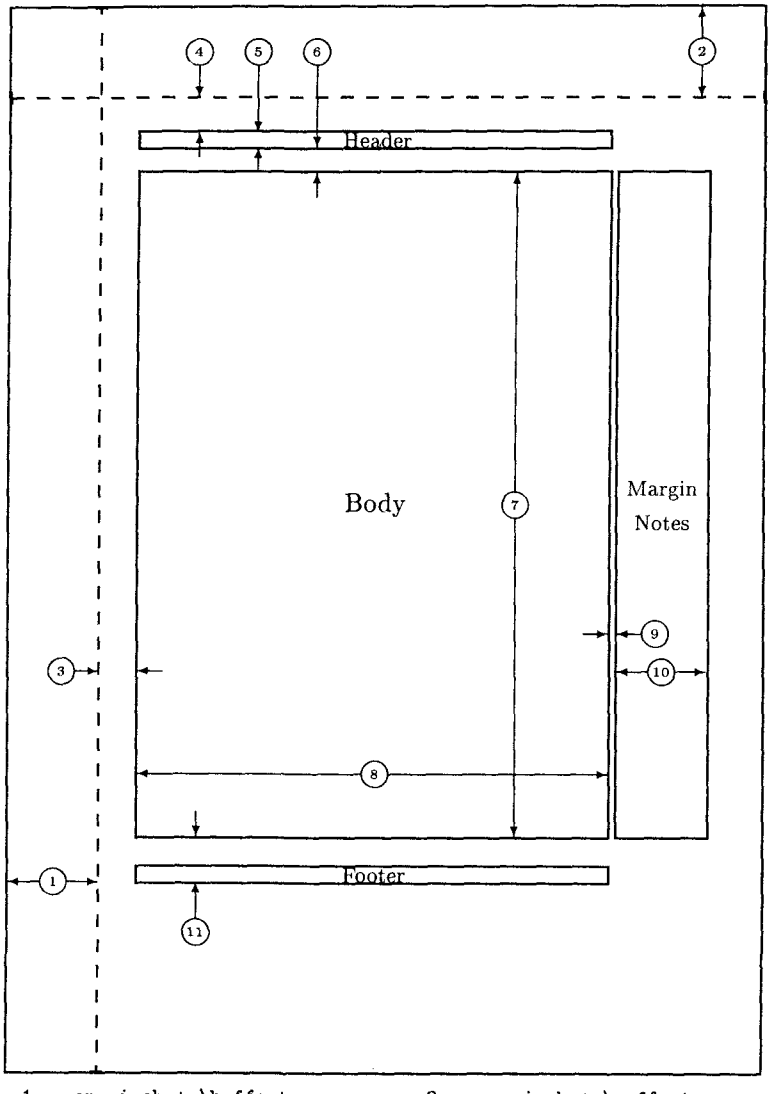
Параметры размерности, управляющие макетом полосы, показаны схематически на рис. 4.1 [$\mathcal{L}$ **163**], [$\mathcal{A}$ **149–51**].

`\textheight`	Высота тела текста (без верхнего и нижнего колонтитулов).
`\textwidth`	Ширина тела текста.
`\columnsep`	Ширина промежутка между колонками при многоколонном режиме.
`\columnseprule`	Ширина вертикальной линейки, отделяющей две соседние колонки при многоколонном режиме (по умолчанию 0pt, т. е. нет видимой линейки).
`\columnwidth`	Ширина одной колонки при многоколонном режиме. Подходящий размер вычисляется LaTeX'ом из `\textwidth` и `\columnsep`.
`\linewidth`	Длина строки текущего текста. Обычно равна `\columnwidth`, но окружения, изменяющие поля, могут дать другие значения.
`\evensidemargin`	При двусторонней печати дополнительный интервал, добавляемый слева на четных полосах.
`\oddsidemargin`	При двусторонней печати дополнительный интервал, добавляемый слева на нечетных полосах; в противном случае дополнительный интервал, добавляемый слева на всех полосах.
`\footskip`	Вертикальный интервал, отделяющий базовую линию последней строки текста от базовой линии нижнего колонтитула.
`\headheight`	Высота верхнего колонтитула.
`\headsep`	Вертикальный разделитель между верхним колонтитулом и телом текста.
`\topmargin`	Дополнительный вертикальный интервал, добавляемый над верхним колонтитулом.
`\marginparpush`	Минимальный вертикальный интервал между двумя последовательными заметками на полях (на рисунке не показан).
`\marginparsep`	Горизонтальный интервал между телом текста и заметками на полях.
`\marginparwidth`	Ширина заметок на полях.

Значения этих параметров по умолчанию представлены в табл. 4.2. В LaTeX 2_ε имеется два дополнительных параметра, описывающих физический лист бумаги:

`\paperheight`	Высота листа при печати.
`\paperwidth`	Ширина листа при печати.

По умолчанию этим параметрам в четырех стандартных классах присвоены значения размеров листа формата US-letter. Большинство других параметров макета полосы набора в классах LaTeX 2_ε задаются в терминах размеров физического




```
1    one inch + \hoffset          2    one inch + \voffset
3    \oddsidemargin = 31pt        4    \topmargin = 28pt
5    \headheight = 12pt           6    \headsep = 20pt
7    \textheight = 526pt          8    \textwidth = 372pt
9    \marginparsep = 7pt          10   \marginparwidth = 71pt
11   \footskip = 36pt                  \marginparpush = 5pt (not shown)
     \hoffset = 0pt                    \voffset = 0pt
```

Рис. 4.1. Макет полосы набора книги *The LaTeX Companion*

Рисунок, аналогичный этому, можно получить командой `\layout`, которая определяется в пакете layout Кента Макферсона. Реальные размеры получаются умножением показанных на рисунке размеров на два. [Здесь: Header — верхний колонтитул, Footer — нижний колонтитул, Body — тело текста, Margin notes — заметки на полях.— *Перев.*]

letterpaper	$8\frac{1}{2} \times 11$ дюймов		
legalpaper	$8\frac{1}{2} \times 14$ дюймов		
executivepaper	$7\frac{1}{4} \times 10\frac{1}{2}$ дюймов		
a4paper	$\approx 8\frac{1}{4} \times 11\frac{3}{4}$ дюймов	210×297	мм
a5paper	$\approx 5\frac{7}{8} \times 8\frac{1}{4}$ дюймов	148×210	мм
b5paper	$\approx 7 \times 9\frac{7}{8}$ дюймов	176×250	мм

Таблица 4.1. Опции стандартных листов бумаги в LaTeX 2_ε

листа бумаги, поэтому они автоматически меняются, как только в начале файла класса модифицируются `\paperwidth` или `\paperheight`. Их изменение в преамбуле вашего документа не даст соответствующего эффекта, поскольку к этому времени уже были вычислены значения других параметров.

Чтобы облегчить подгонку под различные форматы бумаги при печати, классы документов LaTeX 2_ε поддерживают ряд опций, устанавливающих эти параметры в соответствии с физическими размерами заданного листа бумаги, а также регулирующих другие (типа `\textheight`) зависящие от них параметры. Список опций листов бумаги, поддерживаемых стандартными классами документа, приведены в табл. 4.1. Таким образом, чтобы напечатать на листе формата A4, достаточно просто указать

```
\documentclass[a4paper]{article}
```

Дополнительные или другие опции могут быть доступны для других классов документа, но вряд ли целесообразно задавать, скажем, опцию a0paper для book, которая в результате даст строки текста невероятной длины.

Согласно стандартным соглашениям dvi-драйверы оставляют место для ссылок TeX'а на дюйм ниже и правее верхнего левого угла листа. Такие однодюймовые отступы называются *полями драйвера*. Место ссылки можно сдвинуть, переопределив длины `\hoffset` и `\voffset`. По умолчанию эти значения равны нулю. Вообще говоря, значения этих параметров никогда не следует менять. Тем не менее это предоставляет удобный способ сдвинуть всю полосу целиком (тело, верхний и нижний колонтитулы и заметки на полях) при выводе на экран или лист бумаги, не затрагивая макета полосы. Поля драйвера унаследованы от TeX'а, а в параметризации LaTeX'а макета полосы они не участвуют. Изменение `\topmargin` полностью сдвигает текст по вертикали, тогда как изменения `\oddsidemargin` и `\evensidemargin` приводят к сдвигу по горизонтали. Чтобы упростить вычисления, можно «вычесть» поля драйвера, положив `\hoffset` и `\voffset` равными -1 дюйм, и считать, что место ссылки расположено в левом верхнем углу полосы.

Обратите внимание, что некоторые dvi-драйверы вводят свои собственные сдвиги в расположение текста на полосе. Чтобы убедиться, что точка ссылки позиционирована правильно, нужно прогнать тестовый файл `testpage.tex` Лесли Лэмпорта (модифицированный Стефеном Гилди) через LaTeX с этим dvi-

параметр	двусторонняя печать			односторонняя печать		
	10pt	11pt	12pt	10pt	11pt	12pt
\oddsidemargin	44pt	36pt	21pt	63pt	54pt	39pt
\evensidemargin	82pt	74pt	59pt	63pt	54pt	39pt
\marginparwidth	107pt	100pt	85pt	90pt	83pt	68pt
\marginparsep	11pt	10pt	10pt	*то же*		
\marginparpush	5pt	5pt	7pt	*то же*		
\topmargin	27pt	27pt	27pt	*то же*		
\headheight	12pt	12pt	12pt	*то же*		
\headsep	25pt	25pt	25pt	*то же*		
\footskip	30pt	30pt	30pt	*то же*		
\textheight	43	38	36	*то же*		
	×\baselineskip					
\textwidth	345pt	360pt	390pt	*то же*		
\columnsep	10pt	10pt	10pt	*то же*		
\columnseprule	0pt	0pt	0pt	*то же*		

Таблица 4.2. Значения по умолчанию параметров макета полосы набора формата letterpaper

Эти значения идентичны для трех стандартных классов документа LaTeX'a (article, book и report). Если выбрана другая опция формата листа бумаги, эти значения могут быть иными.

драйвером. На полученной странице будет обозначено место ссылки относительно углов листа бумаги. Для LaTeX 2_ε этот файл переписал Райнер Шопф, чтобы интерактивно поддерживать опции стандартных форматов бумажных листов.

4.2 Изменение макета

Если нужно переопределить значения одного или нескольких параметров макета полосы, следует воспользоваться командой `\setlength` или `\addtolength`. Рекомендуется, однако, чтобы эти параметры изменялись в классе или пакете и/или в преамбуле документа.

Сначала для установки вертикальных расстояний рекомендуется использовать параметр `\baselineskip` TeX'a. Он представляет собой расстояние между базовыми линиями в двух последовательных строках текста внутри абзаца, набранного обычным для этого документа шрифтом. Параметр `\baselineskip` мож-

но также рассматривать как высоту строки текста. Таким образом, следующая установка всегда означает «две строки текста».

```
\normalsize                                % Установить обычный \baselineskip
\setlength{\headheight}{2\baselineskip} % Высота заголовка
```

Чтобы гарантировать точную установку \baselineskip, сначала обратимся к \normalsize для выбора размера шрифта, соответствующего основному шрифту документа.

Иногда удобно вычислять параметры макета полосы в соответствии с заданными полиграфическими правилами. Например, требование «текст должен содержать 50 строк» можно выразить при помощи приведенной ниже команды. Предполагается, что высота всех (за исключением одной) строк равна \baselineskip, а высота верхней строки тела текста есть \topskip (это параметр TEX'a \baselineskip для первой строки с заданным по умолчанию значением 10 pt). Обратите внимание, что в примерах этой главы используются пакет LATEX'a calc (который упрощает обозначения при вычислениях) и расширенные управляющие структуры LATEX 2_ε (см. приложение А, разд. А.4 и А.5).

```
\setlength{\textheight}{\baselineskip*49+\topskip}
```

Требование типа «высота тела текста должна быть 198 мм» можно принять аналогичным образом; ниже показаны вычисления. Сначала вычисляется число строк, которое содержится в теле текста требуемого размера. Для этого следует разделить один размер на другой и выделить целую часть. TEX все еще не способен произвести такого рода операцию непосредственно, поэтому сначала размеры присваиваются значениям счетчиков. Обратите внимание, что последнее присваивание производится с высокой точностью, поскольку здесь используются единицы sp на внутреннем уровне.

```
\setlength{\textheight}%                   % Из требуемого размера
          {198mm-\topskip}                 %   вычитается верхняя строка
\newcounter{tempc}                         % 1-му временному счетчику
\setcounter{tempc}{\textheight}            % присваивается 1-е значение
\newcounter{tempcc}                        % 2-му временному счетчику
\setcounter{tempcc}{\baselineskip}         % присваивается 2-е значение
\setcounter{tempc}%                        % значения счетчиков делятся
        {\value{tempc}/\value{tempcc}}
\setlength{\textheight}{\baselineskip*\value{tempc}+\topskip}
```

Можно также вычислить высоту верхнего поля \topmargin. Предположим, что мы хотим установить это поле таким образом, чтобы сверху над телом текста осталось в два раза меньше места, чем снизу. Следующие вычисления показывают, как это осуществить в случае бумаги формата А4 (ее высота равна 297 мм).

```
\setlength{\topmargin}%
  {(297mm-\textheight)/3 - 1in - \headheight -\headsep}
```

4.2.1 Пакеты для создания макета полосы набора

Так как изначально классы LaTeX'а основывались на стандартах бумажных листов, принятых в США, европейские пользователи разработали ряд пакетов, приспособивших макет полосы набора к метрической системе. Примерами таких пакетов служат a4 (порождающий чрезвычайно маленькие полосы), a4dutch Йоханнеса Брамса и Нико Попелье, очень хорошо документированный, и a4wide Жан-Франсуа Лами, который позволяет делать несколько более длинные строки. Часто под теми же названиями можно встретить локальные разработки. Для формата бумаги A5 Марио Волчко написал пакеты a5 и a5comb. Другой подход предпринял Фолкер Кульман, сделав пакет vmargin, в котором он ввел команды установки полей для всех видов бумажных листов, как в метрической системе, так и в американском стандарте. Для того чтобы воспользоваться этим пакетом, нужно сначала выбрать формат листа бумаги посредством `\setpapersize[`*orient*`]{`*size*`}`, где в качестве *size* могут выступать `Afour`, `Bfive`, `USletter` и многое другое, тогда как факультативный параметр *orient* задает ориентацию: по умолчанию `portrait` или `landscape`. Если пожелаете, можете даже задать свой собственный формат листа. Поля устанавливаются командой

```
\setmargins{leftmargin}{topmargin}{textwidth}{textheight}%
           {headheight}{headsep}{footheight}{footskip}
```

или командой

```
\setmarginsrb{leftmargin}{topmargin}{rightmargin}{bottommargin}%
             {headheight}{headsep}{footheight}{footskip}
```

Если используется бумага формата `USlegal` и нужно, чтобы все поля были равны 1 дюйму и не было ни верхних, ни нижних колонтитулов, то следует сказать:

```
\setpapersize{USlegal}
\setmarginsrb{1in}{1in}{1in}{1in}{0pt}{0mm}{0pt}{0mm}
```

Вообще говоря, если вы хотите изменить макет полосы набора, полезно иметь хоть некоторое представление о полиграфических правилах и традициях (см., например, [71]). Изучение печатной продукции на английском языке показывает, что строка должна содержать не более 10–12 слов, что соответствует 60–70 литерам в строке.[1]

Количество строк на полосе зависит от размера и начертания используемого шрифта. Ниже показан способ вычисления величины `\textheight`, которая зависит от базового размера (внутренняя переменная LaTeX'а `\@ptsize` принимает

[1] Для русского языка количество слов ограничивается 8–10, а количество знаков 50–60, потому что в русском языке в среднем более длинные слова и более широкие буквы. Подробнее об особенностях отечественной полиграфии см. [17].— *Прим. перев.*

значения 0, 1 или 2 для основных шрифтов размера 10 pt, 11 pt и 12 pt соответственно).

```
\ifthenelse{\@ptsize = 0}%      10-пунктовый шрифт как основной
  {\setlength{\textheight}{53\baselineskip}}{}
\ifthenelse{\@ptsize = 1}%      11-пунктовый шрифт как основной
  {\setlength{\textheight}{46\baselineskip}}{}
\ifthenelse{\@ptsize = 2}%      12-пунктовый шрифт как основной
  {\setlength{\textheight}{42\baselineskip}}{}
\addtolength{\textheight}{\topskip}
```

Другим важным параметром является количество белого пространства вокруг текста. Если печатный документ предполагается взять в рамку, во внутреннем поле следует оставить достаточно пустого места, чтобы иметь возможность это сделать. Если величина `\oddsidemargin` фиксирована, то вычисление `\evensidemargin` для двусторонней печати основывается на следующем соотношении:

```
width_of_paper =
  (1in+\hoffset)*2+\oddsidemargin+\textwidth+\evensidemargin
```

Двусторонняя печать включается опцией класса twoside, которая устанавливает булевский регистр `@twoside` в `true`. Вычисление горизонтальных параметров макета полосы, показывающее, как учесть размер основного шрифта и одно- или двустороннюю печать, выглядит так:

```
\ifthenelse{\@ptsize = 0}%        10-пунктовый шрифт как основой
  {\setlength{\textwidth}{5.00in}%
   \setlength{\marginparwidth}{1.00in}%
   \ifthenelse{\boolean{@twoside}}%
      {\setlength{\oddsidemargin}{0.55in}%    двусторонняя печать
       \setlength{\evensidemargin}{0.75in}%
      }
      {\setlength{\oddsidemargin}{0.55in}%    односторонняя печать
       \setlength{\evensidemargin}{0.55in}%
      }
  }{}
```

Точно так же, когда в документе много заметок на полях, имеет смысл изменить макет полосы, чтобы увеличить поля. Например, в пакете a4dutch этой цели служит команда `\WideMargins`. Это макро модифицирует геометрические параметры таким образом, что для заметок на полях резервируется ширина в 1.5 дюйма за счет уменьшения ширины тела текста.

4.2.2 Горизонтальное расположение полос набора при печати

Обычно мы считаем, что более длинная сторона полосы расположена вертикально (так называемая *вертикальная* (*portrait*) ориентация). Для некоторых видов документов, как, например, слайды или таблицы, более целесообразно использовать другую, *горизонтальную* (*landscape*) ориентацию, при которой более длинная сторона располагается по горизонтали. Современные принтеры и dvi-драйверы позволяют печатать документы и в той, и в другой ориентации.

При горизонтальной и вертикальной ориентациях требуются разные макеты полосы. Хотя следует воздерживаться от внесения изменений в геометрические параметры макета полосы после команды `\begin{document}`, все-таки можно менять эти параметры между полосами. Эта возможность эксплуатируется в пакете portland (Губера Партля) посредством команд переключения между вертикальной и горизонтальной ориентациями:

```
\portrait    \landscape
```

Первая команда (пере)устанавливает макет полосы в изначальные значения макета полосы (т. е. значения в точке `\begin{document}`). Вторая команда меняет макет полосы таким образом, что вертикальные и горизонтальные размеры меняются местами. Общие размеры тела текста остаются неизменными. В отличие от изменения макета полосы обе команды выдают `\clearpage`, завершающую текущую полосу и устанавливающую несколько внутренних размерностей LaTeX'а. Этот пакет также определяет два окружения: `portrait` и `landscape`, которые можно использовать вместо указанных выше команд. В случае, если вам нужно будет использовать высоту полосы как параметр `\paperheight`, придется сделать некоторые усовершенствования пакета. По умолчанию закладывается формат полосы A4.[2] Когда применяется dvi-драйвер, позволяющий смешивать вертикальный и горизонтальный выводы на печать при одном и том же прогоне (к примеру, dvips), стóит добавить соответствующие команды `\special` непосредственно после команд `\clearpage`, чтобы получать различные выводы на печать корректным образом.

Если при переключении горизонтального и вертикального режимов затрагивается только текст без верхних и нижних колонтитулов, можно использовать пакет lscape Дэвида Карлайла, который также определяет окружение `landscape`, поворачивающее полосу в пределах 90 градусов. В этом случае вам не придется заботиться о самостоятельном вводе команд `\special`, так как этот пакет использует в свою очередь пакет rotate (Томаш Рокицки) и требует его dvi-драйвер dvips (см. разд. 11.2).

[2] В обновленной версии пакета для LaTeX 2_ε можно непосредственно использовать параметры `\paperwidth`, предлагаемые файлами класса.

4.3 Стили полосы

В то время как размеры почти на всех полосах документа остаются одними и теми же, формат верхних и нижних колонтитулов (*стиль полосы*) может меняться на протяжении всего документа. Стиль полосы выбирается двумя командами: [$\mathcal{L}$ 161], [$\mathcal{A}$ 148,292]

`\pagestyle{`*style*`}` `\thispagestyle{`*style*`}`

Первая команда устанавливает стиль на текущей и последующих полосах в соответствии с переменной *style*; вторая команда действует аналогично, но в пределах только текущей полосы.

Стандартные стили полосы LaTeX'а следующие:

empty	Верхний и нижний колонтитулы пустые.
plain	Верхний колонтитул пуст, а нижний содержит номер полосы.
headings	Верхний колонтитул содержит информацию, определяемую классом документа, и номер полосы; нижний колонтитул пуст.
myheadings	Аналогично headings, но верхний колонтитул управляется пользователем

Три первых стиля используются в стандартных классах. Обычно для титульной полосы выдается команда `\thispagestyle{empty}`. Для первых полос основных разделов документа, задаваемых командами типа `\part` или `\chapter`, а также для `\maketitle` в стандартных классах LaTeX'а имеется команда `\thispagestyle{plain}`. Это значит, что когда вы в начале вашего документа указываете команду `pagestyle{empty}`, вы все еще будете получать номера полос там, где появляется либо команда `\chapter`, либо `\maketitle`. Чтобы получить желаемое поведение, нужно за каждой такой командой поместить команду `thispagestyle{empty}` либо переопределить стиль plain на empty, т. е. `\let\ps@plain=\ps@empty`.

В документах небольших или средних размеров нет необходимости в столь сложных переключениях стилей полосы. Обычно стиль полосы выбирается в классе документа. Но для более солидных документов, таких как книги, следует принимать во внимание полиграфические традиции. Например, материал, предшествующий основному тексту, обычно нумеруется римскими цифрами, тогда как полосы основного текста имеют арабскую нумерацию; части и главы начинаются с нечетных полос (справа на развороте) и т. д.[3]

Нумерация полос управляется счетчиком page.

[3] В традициях отечественной полиграфии таких ограничений нет: полосы всей книги нумеруются подряд арабскими цифрами, начинать каждую новую часть или главу с новой нечетной страницы желательно, но не обязательно. Тем не менее все зависит от художественного оформления книги: в ней могут быть сложные спусковые полосы, справочный материал, оформленный по-другому, всевозможные заставки, требующие отдельного стиля, и др.— *Прим. ред.*

```
\documentclass[...]{companion}  %%% файл класса устанавливает стиль
                                %%% полосы по умолчанию
...
\newcommand{\clearemptydoublepage}{\newpage{\pagestyle{empty}\cleardoublepage}}
\begin{document}
%  ================== Преамбула ============================
\title{The \LaTeX{} companion}
\cyrAuthor{...}
\pagenumbering{roman}  % нумерация полос римскими цифрами
\maketitle             % для титульной полосы используется стиль "empty"
\include{ch0}          % для полосы с копирайтом используется стиль "empty"
\clearemptydoublepage
\tableofcontents       % для первой полосы используется стиль "empty"
\clearemptydoublepage  % переход к нечетной полосе
% ==================== Тело документа ====================
\pagenumbering{arabic} % нумерация полос арабскими цифрами
\include{ch1}          % для первой полосы используется стиль "empty"
\clearemptydoublepage  % переход к нечетной полосе
\include{ch2}
\clearemptydoublepage  % переход к нечетной полосе
...
%  ================== Приложения ===========================
...                    % особый стиль полосы для указателей
```

Рис. 4.2. Стили полосы, использованные в *The LaTeX Companion*

`\pagenumbering{`*style*`}`

Эта команда переустанавливает счетчик в единицу и переопределяет команду `\thepage` в `\`*style*`{page}`. Имеются заготовки стилей счетчиков полос: `Alph`, `alph`, `Roman`, `roman` и `arabic` (см. разд. A.1.3).

`\clearpage` `\cleardoublepage`

Обе эти команды завершают текущие абзац и полосу. При двусторонней печати `\cleardoublepage` также позволяет убедиться, что следующая полоса нечетная (правая). В качестве примера на рис. 4.2 схематически представлены структура данной книги и стили типичных полос. Обратите внимание на команду `\clearemptydoublepage`, которая порождает абсолютно пустую полосу без номера, когда таковая требуется.

4.3.1 Написание новых стилей полосы

Команды форматирования, связанные с каждым аргументом *style* команды \pagestyle, управляются посредством определения соответствующих макро \ps@*style*. Эти макро, в свою очередь, определяют внутренние команды LaTeX'a, которые форматируют верхние и нижние колонтитулы.

\@oddhead При двусторонней печати порождает верхний колонтитул для нечетных полос, в противном случае порождает верхние колонтитулы для всех полос.

\@oddfoot При двусторонней печати порождает нижний колонтитул для нечетных полос, в противном случае порождает нижние колонтитулы для всех полос.

\@evenhead При двусторонней печати порождает верхний колонтитул для четных полос.

\@evenfoot При двусторонней печати порождает нижний колонтитул для четных полос.

Определение стиля полосы plain, который производит только номер полосы по центру нижнего колонтитула, эквивалентно следующему:

```
\newcommand{\ps@plain}{%
   \renewcommand{\@oddhead}{}%          пуст
   \renewcommand{\@evenhead}{}%         пуст
   \renewcommand{\@evenfoot}{\hfil\textrm{\thepage}\hfil}%
   \renewcommand{\@oddfoot}{\@evenfoot}}
```

Использование названий разделов в качестве верхних или нижних колонтитулов представляет собой некоторую проблему. Команды секционирования (разбиения на разделы) позволяют захватить название раздела или его часть, обратившись к командам \chaptermark, \sectionmark и т. д. Эти команды выполняются автоматически посредством соответствующих команд колонтитула. Они имеют один аргумент, в который получают текст колонтитула или его краткую форму из факультативного аргумента команды колонтитула. Например, в классе book эти команды определяются примерно так:

```
\renewcommand{\chaptermark}[1]{\markboth{\chaptername\
                                    \thechapter. #1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection. #1}}
```

Для главы сначала идет слово "Chapter" (или его эквивалент на соответствующем языке, см. табл. 9.2 в разд. 9.2), затем ее номер (сохраненный в счетчике chapter) и название главы (или его краткий вариант), которое будет сохранено в \chaptermark. Для раздела (параграфа) сначала указывается его номер (сохраненный в счетчике section), за которым следует название (или краткий вариант), которое будет сохранено в \sectionmark.

Как правило, верхний или нижний колонтитул содержит информацию о самом последнем названии второго уровня (например, раздела или параграфа),

появившемся на текущей полосе. Вообще говоря, вы не можете просто сохранять и вводить названия разделов в колонтитулы. В силу асинхронности алгоритма TEX'а разбиения на страницы, невозможно заранее узнать, какая из команд секционирования появится последней, прежде чем произойдет переход на другую страницу. TEX решает эту проблему при помощи механизма *markers*: пользователь расставляет маркеры внутри текста и, прежде чем отправить текущую страницу в `.dvi`-файл, программа вывода TEX'а определяет, какой из маркеров текущей страницы был первым, а какай последним.

`\markboth{`*left_head*`}{`*right_head*`}` `\markright{`*right_head*`}`

Эти две команды LATEX'а проставляют маркер в заданном месте текста [$\mathcal{L}$ **161–2**], [$\mathcal{A}$ **293–5**]. Первая команда проставляет пару маркеров: левый и правый. Она обычно немедленно вызывает следующую команду секционирования. Вторая команда также проставляет пару маркеров, но она меняет только правый, оставляя другой нетронутым. Эта команда вызывается сразу после команды секционирования.

`\leftmark` `\rightmark`

Эти две команды содержат текущие значения левых и правых маркеров, которые были определены программой вывода LATEX'а из разных команд `\markboth` и `\markright` для готовой к выводу полосы. Команда `\leftmark` содержит аргумент *left_head* команды `\markboth`, последней на текущей полосе. Команда `\rightmark` содержит аргумент *right_head* либо команды `\markright`, либо `\markboth`, первой на текущей полосе, если таковая существует; в противном случае это самый последний из определенных ранее.

Команды маркировки работают весьма неплохо для правых маркеров, «нумерованных внутри» левых маркеров (например, когда левый маркер изменяется командой `\chapter`, а правый — командой `\section`). Однако, если команде `\markboth` предшествует какая-нибудь другая команда расстановки маркеров на той же самой полосе, результат получится несколько странный — см. получившуюся полосу `L5 R3.2` на рис. 4.3. На этом рисунке схематически показано, как получаются левый и правый маркеры на полосах, готовых к выводу на печать.

Как показано в табл. 4.3, в стиле полосы `headings` команды секционирования автоматически расставляют верхние колонтитулы, используя `\markboth` и `\markright`.

Стандартный стиль `myheadings` аналогичен `headings`, но предоставляет возможность создавать колонтитул, определяя описанные выше команды `\markboth` и `\markright`. Он также обеспечивает способ управления названиями разделов, вытянутыми из других разделов, типа оглавления, списка иллюстраций, указателя. На самом деле команды (`\tableofcontents`, `\listoffigures` и `\listoftables`), а также окружения (`thebibliography` и `theindex`) используют команду `\chapter*`, которая не обращается к `\chaptermark`, но выдает команду `\@mkboth`. Стиль полосы `headings` определяет `\@mkboth` как `\markboth`, тогда как

```
                                    пара маркеров  маркеры принтера(???)
                                                     левый    правый
\markboth{L1}{}                     {L1}{}
\newpage%    ---разрыв полосы---                      L1
\markright{R1.1}                    {L1}{R1.1}
\markboth{L2}{}                     {L2}{}
\markright{R2.1}                    {L2}{R2.1}
\newpage%    ---разрыв полосы---                      L2       R1.1
\markright{R2.2}                    {L2}{R2.2}
\markright{R2.3}                    {L2}{R2.3}
\markright{R2.4}                    {L2}{R2.4}
\newpage%    ---разрыв полосы---                      L2       R2.2
\markboth{L3}{}                     {L3}{}
\markright{R3.1}                    {L3}{R3.1}
\newpage%    ---разрыв полосы---                      L3
\newpage%    ---разрыв полосы---                      L3       R3.1
\markright{R3.2}                    {L3}{R3.2}
\markboth{L4}{}                     {L4}{}
\markboth{L5}{}                     {L5}{}
\newpage%    ---разрыв полосы---                      L5       R3.2
\markright{R5.1}                    {L5}{R5.1}
\end{document}                                       L5       R5.1
```

Рис. 4.3. Схематическое представление работы механизма LaTeX'а расстановки маркёров

стиль полосы myheadings определяет `\@mkboth`, чтобы ничего не предпринимать и предоставить решение пользователю.

4.3.2 Создание стиля полосы при помощи fancyheadings

Пакет Пита Ван Остpюма fancyheadings позволяет легко создавать верхние и нижние колонтитулы, предоставляя следующие функции:

- три части верхних и нижних колонтитулов;
- линейки в верхних и нижних колонтитулах;
- колонтитулы, более широкие чем `\textwidth`;
- многострочные колонтитулы;
- разные колонтитулы для четных и нечетных полос;
- отдельные колонтитулы для начальных полос глав.

Чтобы применить этот стиль полосы, нужна команда `\pagestyle{fancy}`; она должна выдаваться после каждого изменения `\textwidth`.

Стиль печати	Команда	Класс документа	
		book, report	article
двусторонняя	\markboth[a]	\chapter	\section
	\markright	\section	\subsection
односторонняя	\markright	\chapter	\section

[a] Определяется пустой правый маркер (см. рис. 4.3)

Таблица 4.3. Стиль полосы, определяемый командами LaTeX'a

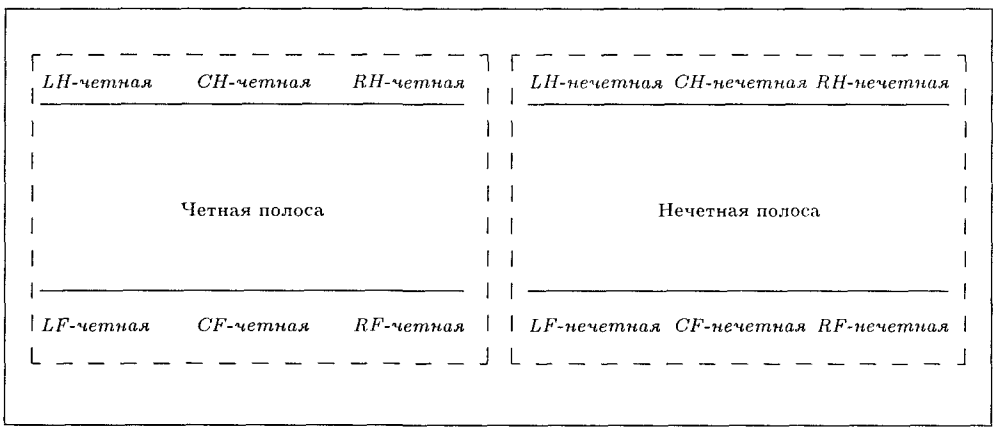


Рис. 4.4. Параметры макета полосы в пакете fancyheadings

Следующие команды дают информацию для шести сегментов верхнего и нижнего колонтитулов при макете полосы fancy, показанном на рис. 4.4.[4]

\lhead[*LH-четная*]{*LH-нечетная*}	\lfoot[*LF-четная*]{*LF-нечетная*}
\chead[*CH-четная*]{*CH-нечетная*}	\cfoot[*CF-четная*]{*CF-нечетная*}
\rhead[*RH-четная*]{*RH-нечетная*}	\rfoot[*RF-четная*]{*RF-нечетная*}

Информация в L-сегментах будет выровнена по левому краю, в C-сегментах — центрирована, а в R-сегментах — выровнена по правому краю.

Толщина линеек под верхним и над нижним колонтитулами управляется параметрами длины \headrulewidth (по умолчанию 0.4 pt) и \footrulewidth (по умолчанию 0 pt). Толщина 0 pt делает линейку невидимой. Если переопределить команды \headrule и/или \footrule, то можно делать более сложные изменения.

[4] Здесь буква H соответствует верхнему колонтитулу (header), F — нижнему (footer); буквы L, C и R означают равнение соответственно по левому краю (left), по центру (center) и по правому краю (right).— *Прим. перев.*

Верхние и нижние колонтитулы набираются в боксах ширины \headwidth, по умолчанию равной значению \textwidth. Бокс может быть сделан шире (или у́же) посредством команды \setlength, определяющей \headwidth. Колонтитулы могут выходить за тело полосы в ту же сторону, что и заметки на полях. Например, чтобы колонтитулы перекрывали и заметки на полях, нужно добавить к \headwidth две величины: \marginparsep и \marginparwidth (см. рис. 4.6).

Каждый из шести сегментов устанавливается в соответствующем парбоксе (parbox), так что есть возможность использовать многострочный материал посредством \\. Дополнительные интервалы добавляются командами \vspace. Заметим, что в этом случае длина \headheight и, возможно, \footskip могут увеличиться.

Некоторые команды LaTeX'a, типа \chapter, используют команду \thispagestyle для автоматического переключения на стиль полосы plain, отменяя, таким образом, действие текущего стиля полосы. Для создания таких полос используется стиль fancyplain, который для обычных полос устанавливает fancy и вдобавок переопределяет стиль полосы plain, чтобы также использовать стиль fancy со следующими модификациями. Толщина линеек определяется посредством \plainheadrulewidth и \plainfootrulewidth; обе по умолчанию равны 0pt. При помощи команды \fancyplain все шесть сегментов могут определяться отдельно для обычных полос и полос типа plain:

\fancyplain{*plain_value*}{*normal_value*}

Эта команда может использоваться внутри как факультативных, так и обязательных аргументов определенных выше команд \..head и \..foot. Например,

```
\lhead[\fancyplain{F1}{F2}]{\fancyplain{F3}{F4}}
```

демонстрирует спецификацию сегмента \lhead левого верхнего колонтитула при двусторонней печати. В этом случае F1 представляет собой спецификацию для простой четной полосы (plain); F2 — для обычной четной полосы; F3 — для простой нечетной полосы (plain); F4 — для обычной нечетной полосы.

По умолчанию устанавливаются следующие параметры:

```
\headrulewidth        0.4pt        \footrulewidth        0pt
\plainheadrulewidth     0pt        \plainfootrulewidth   0pt
```

По умолчанию в макете устанавливается наклонный шрифт для информации в верхнем колонтитуле, нижний колонтитул содержит только номер полосы в прямом начертании.

```
\lhead[\fancyplain{}{\sl\rightmark}]{\fancyplain{}{\sl\leftmark}}
\rhead[\fancyplain{}{\sl\leftmark}]{\fancyplain{}{\sl\rightmark}}
\cfoot{\rm\thepage}
\chead{}\lfoot{}\rfoot{}
```

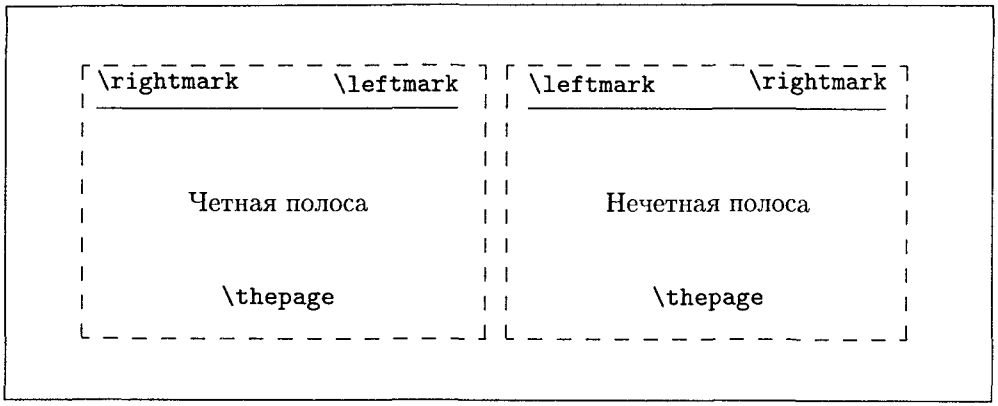


Рис. 4.5. Макет, устанавливаемый по умолчанию в пакете fancyheadings

У простых полос верхний колонтитул пуст, а в нижнем содержится только номер полосы по центру, тогда как для обычных полос нижний колонтитул состоит из номера полосы по центру, а верхний — из маркеров (см. рис. 4.5).

Для демонстрации более изощренных возможностей пакета fancyheadings рассмотрим макет, в котором в правом нижнем углу полосы в двух строках располагаются названия раздела и подраздела. В этом случае следует написать нечто вроде

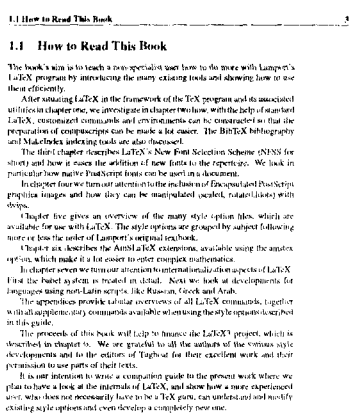
```
\documentclass{book}
\usepackage{fancyheadings}
\pagestyle{fancy}
\renewcommand{\sectionmark}[1]{\markboth{#1}{}}
\renewcommand{\subsectionmark}[1]{\markright{#1}}
\rfoot{\leftmark\\\rightmark}
..... % остальные команды преамбулы
\begin{document}
```

В качестве последнего примера на рис. 4.6 приводятся определения стиля полосы, аналогичного использованному Лесли Лэмпортом в его книге LaTeX book.

4.4 Явное форматирование

Как упоминалось во введении, на окончательной стадии макетирования солидных изданий часто бывает нужно провести форматирование вручную, чтобы избежать некоторых неудачных разрывов полос. Для этой цели LaTeX предлагает команды \pagebreak, \nopagebreak, \newpage и \clearpage наряду с декларацией \samepage [$\mathcal{L}$ **90,190**], [$\mathcal{A}$ **119**], хотя последняя в полном объеме работает только в LaTeX 2_ε. (Воспользовавшись декларацией \samepage вместе с подходящим

```
\documentclass{book}
\usepackage{fancyheadings}
\pagestyle{fancyplain}
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
%       запоминается название главы
\renewcommand{\chaptermark}[1]%
                  {\markboth{#1}{}}
%       номер раздела и название
\renewcommand{\sectionmark}[1]%
             {\markright{\thesection\ #1}}
\lhead[\fancyplain{}{\bfseries\thepage}]%
      {\fancyplain{}{\bfseries\rightmark}}
\rhead[\fancyplain{}{\bfseries\leftmark}]%
      {\fancyplain{}{\bfseries\thepage}}
\cfoot{}
```


1.1 How to Read This Book 3

1.1 How to Read This Book

The book's aim is to teach a non-specialist user how to do more with Lamport's LaTeX program by introducing the many existing tools and showing how to use them efficiently.

After situating LaTeX in the framework of the TeX program and its associated utilities in chapter one, we investigate in chapter two how, with the help of standard LaTeX, customized commands and environments can be constructed so that the preparation of compuscripts can be made a lot easier. The BibTeX bibliography and MakeIndex indexing tools are also discussed.

The third chapter describes LaTeX's New Font Selection Scheme (NFSS) for short) and how it eases the addition of new fonts to the repertoire. We look in particular how native PostScript fonts can be used in a document.

In chapter four we turn our attention to the inclusion of Encapsulated PostScript graphics images and how they can be manipulated (scaled, rotated, cloned) with ease.

Chapter five gives an overview of the many style option files, which are available for use with LaTeX. The style options are grouped by subject following more or less the order of Lamport's original textbook.

Chapter six describes the AmsLaTeX extensions, available using the amstex option, which make it a lot easier to enter complex mathematics.

In chapter seven we turn our attention to internationalization aspects of LaTeX. First the Babel system is treated in detail. Next we look at developments for languages using non-Latin scripts like Russian, Greek and Arab.

The appendices provide tabular overviews of all LaTeX commands, together with all supplementary commands available when using the style options described in this guide.

The proceeds of this book will help to finance the LaTeX3 project, which is described in chapter six. We are grateful to all the authors of the various style developments and to the editors of TugboaT for their excellent work and their permission to use parts of their texts.

It is our intention to write a companion guide to the present work where we plan to have a look at the internals of LaTeX, and show how a more experienced user, who does not necessarily have to be a TeX guru, can understand and modify existing style options and even develop a completely new one.

Рис. 4.6. Установка колонтитулов в книге LaTeX book

количеством команд `\nobreak`, вы можете потребовать, чтобы некоторый фрагмент документа был неразрывным. К сожалению, результат получается не всегда удовлетворительным; в частности, LaTeX никогда не сделает полосу длиннее заданной номинальной длины (`\textheight`), а скорее перенесет все, что вышло за пределы `\samepage`, на следующую полосу. Команда LaTeX 2ε `\enlargethispage*`, описанная ниже, предлагает альтернативный вариант.)

В книжном деле общепринято оставлять некоторые полосы более короткими или длинными (обычно на развороте), чтобы избежать неудачных разрывов впоследствии. Это означает, что номинальная длина полосы сокращается или увеличивается на некоторую величину, скажем на `\baselineskip`. Для реализации этого на практике LaTeX 2ε предлагает команду

`\enlargethispage{`*size*`}`

Если, к примеру, мы хотим увеличить или уменьшить размер некоторых полос на одну дополнительную строку текста, нам следует определить

```
\newcommand{\longpage}{\enlargethispage{\baselineskip}}
\newcommand{\shortpage}{\enlargethispage{-\baselineskip}}
```

и использовать эти команды между двумя абзацами на проблемной полосе.[5]

Команда `\enlargethispage` увеличивает `\textheight` для текущей полосы, не меняя при этом параметров форматирования. Это значит, что если действует

[5] Эта книга изобилует неразрывными примерами, и авторам тоже пришлось прибегнуть к указанному приему. (По нормам российской полиграфии полоса может быть короче, но не длиннее.— *Ред.*)

команда \flushbottom, то текст должен заполнить проблемную полосу на длину \textheight, сужая или расширяя, при необходимости, вертикальные интервалы на данной полосе. Таким образом, приведенные выше определения добавляют или удаляют точно одну строку текста на полосе, не затрагивая расположения других строк. Это очень важно — сохранить равномерность представления.

\enlargethispage*{*size*}

Группа команд \enlargethispage* также удлиняет или укорачивает полосу, но в то же время полученная в результате окончательная полоса будет сжата на столько, на сколько возможно (т. е. в зависимости от наличия свободного пространства на полосе). Это бывает полезно, когда вы хотите уместить некоторый фрагмент вашего документа на одной полосе, даже если полоса получится чуть длиннее. (В противном случае используйте только окружение minipage.) Прием состоит в том, чтобы затребовать большой объем дополнительного пространства и затем разместить явный разрыв полосы в том месте, где вы хотите, чтобы этот разрыв произошел, например:

```
\enlargethispage*{100cm}        % абсурдное требование
\begin{center}
 \begin{tabular}{llll}          % несколько более длинная
  ....                          % таблица
\end{tabular}
\end{center}
\pagebreak                      % вынужденный разрыв полосы
```

Из сказанного выше понятно, что обе команды должны использоваться только на окончательном этапе подготовки оригинала-макета, поскольку любое более позднее вторжение в документ (добавление или удаление единственного слова, если так случилось) может сделать ваше форматирование вручную бессмысленным, и, как результат,— уродливо оформленные полосы.

ГЛАВА 5

Таблицы

Довольно часто данные представляют в виде таблиц — это наиболее эффективная форма их представления. Для организации материала в табличной форме TEX предоставляет мощные базисные средства (примитивы). Эти примитивы, однако, относятся к средствам низшего уровня, при применении которых все действия по форматированию требуемого выхода целиком ложатся на пользователя. В связи с этим на основе указанных примитивов разработан ряд макропакетов, взаимодействие с которыми осуществляется при помощи интерфейса, более дружественного к пользователю, с применением средств, родственных командным языкам высокого уровня.

В LATEX'е наиболее удобное средство организации таблиц — окружения `tabular` и `array`, хотя в ряде случаев может быть полезным и окружение `tabbing`.

В последующих разделах после краткого рассмотрения окружения `tabbing` описываются расширенные варианты базовых окружений `tabular` и `array`, содержащихся в LATEX'е. Эти расширенные варианты реализуются пакетом array. Данный пакет обеспечивает ряд дополнительных возможностей, особенно в части более гибкого размещения абзацев; он обладает улучшенным управлением промежутками между колонками и строками в таблицах, а также возможностью определения новых спецификаторов в преамбуле таблицы. Комбинируя средства пакетов array и tabularx, можно сравнительно просто создавать таблицы сложной структуры.

Далее рассматриваются проблемы верстки таблиц, размещающихся на нескольких страницах (многостраничные таблицы). Описываются два окружения, `supertabular` и `longtable`, которые позволяют либо автоматически переносить остающуюся часть таблицы на следующую страницу, когда текущая страница исчерпана, либо выполнять операцию такого переноса принудительно, когда пользователь дает LATEX'у команду перехода к новой странице.

Следующие разделы данной главы посвящены пакетам, предназначенным для совместного использования со средствами, упомянутыми выше. В частности, па-

кет delarray автоматически вычисляет высоту ограничителей вокруг табличного материала, пакет dcolumn упрощает выравнивание чисел по десятичному разделителю, а пакет hhline определяет способ взаимодействия с горизонтальными или вертикальными линейками.

В заключительном разделе дается ряд практических советов и демонстрируется ряд интересных сложных примеров верстки таблиц средствами, рассмотренными в данной главе.

Читателям, интересующимся подготовкой математических текстов, следует ознакомиться также с гл. 8 (расширенные возможности организации материала математического характера), в особенности с разд. 8.5, где обсуждается выравнивание многострочных математических выражений, а также изучить примеры, начиная со с. 283. Те же читатели, которые имеют возможность воспользоваться выходными устройствами, воспринимающими язык PostScript, найдут много дополнительных возможностей обработки табличного материала в разд. 11.4.1.

5.1 Сравнение окружений tabbing и tabular

В LaTeX'е содержится два семейства окружений, позволяющих организовать материал в виде колонок; одно из них — окружение `tabbing`, второе — окружения `tabular` и `array`. Ниже перечислены их основные различия.

- Окружение `tabbing` является менее общим, чем окружение `tabular`. С его помощью материал можно набрать только в виде отдельного абзаца, тогда как окружение `tabular` может быть помещено в любом месте текста, а также и в материале математического характера.
- Окружение `tabbing` может быть разделено между несколькими страницами, в то время как окружение `tabular` в его стандартном варианте на это не рассчитано.
- При работе с окружением `tabbing` пользователь должен задавать все позиции табуляции в явном виде. Окружение `tabular` в LaTeX'е может определять ширину колонок автоматически.
- Изменять характер организации материала проще в окружении `tabbing`, так что с его помощью несложно представлять тексты компьютерных программ.
- Окружения `tabbing` не могут быть вложенными, окружения `tabular` — могут, вследствие чего с их помощью можно реализовывать таблицы сложной структуры.

В ряде случаев одно или несколько окружений `tabbing` или `tabular` удобно ввести внутрь плавающего объекта (см. гл. 6), подобного окружению `table`. Это будет полезно, если имеется потребность сгруппировать несколько таблиц или организовать таблицу, обтекаемую текстом. Следует быть внимательным в употреблении окружений `tabular` и `table`, поскольку первое из них обеспечивает организацию материала в виде колонок, а второе представляет собой логический элемент документа, показывающий, что составные части, содержащиеся

в нем, должны перемещаться при верстке документа совместно. Например, одно окружение `table` может включать несколько окружений `tabular`. Табличный материал, занимающий более одной страницы, может быть сверстан при помощи окружений `longtable` и `supertabular`, описываемых в разд. 5.4.

5.2 Использование окружения tabbing

В этом разделе рассматриваются некоторые малоизвестные свойства окружения `tabbing` [$\mathcal{L}$ **62–3,179–82**], [$\mathcal{A}$ **182–8**]. Первое, что следует принять во внимание,— форматирование здесь полностью контролируется пользователем. В частности, переход к заданной позиции табуляции всегда приводит к перемещению в требуемую горизонтальную позицию независимо от того, где размещена в данный момент текущая точка. Это значит, что текущую точку можно смещать назад и набирать поверх предыдущего текста.

Надо также помнить, что обычные команды LaTeX'а для получения диакритических знаков (акцентов) — `\'` , `\‘` и `\=` в окружении `tabbing` переопределены. Вместо них в пределах окружения `tabbing` необходимо пользоваться командами `\a'` , `\a‘` и `\a=` соответственно. Команда `\-` , сигнализирующая в обычном тексте о возможной точке переноса слова, также переопределена, однако это не имеет особого значения, поскольку строки в окружении `tabbing` никогда не разрываются.

Область действия команд в строках таблицы обычно ограничивается областью между позициями табуляции (т. е. графой таблицы).

Стилевой параметр `\tabbingsep`, используемый совместно с командой `\'`, позволяет набираемый текст прижать вправо к следующей позиции табуляции, отступив от нее на заданное расстояние. Его значение по умолчанию устанавливается равным значению параметра `\labelsep`, принимаемому обычно 5 pt.

Среди общепринятых способов определения позиции табуляции — использование набираемой строки с командами `\=` для установки позиций табуляции в ней, а также явное задание перехода к следующей позиции табуляции при помощи команды `\>`.

Следующий пример показывает неправильный выбор шаблона для первой строки таблицы, поскольку первая колонка второй строки шире заданной в этом шаблоне. В последней строке позиция табуляции переопределяется, здесь же демонстрируются различные применения обратной кавычки.

confused embrouillé *confused mind* esprit trouble
confusiondéconfiture

conjecturehypothèse (from Greek)

```
\begin{tabbing}
\textbf{confused}  \= embrouill\a'e \=
 \em confused mind \=
                    esprit trouble \kill
\textbf{confused} \> embrouill\a'e  \>
 \em confused mind\> esprit trouble \\
\textbf{confusion}\>
                  d\a'econfiture\\[3mm]
\textbf{conjecture}\=hypoth\a'ese
                       \‘(from Greek)\\
\end{tabbing}
```

Окружение tabbing очень полезно для табличного представления таких данных, для которых требуемая ширина колонок постоянна и известна. Это положение иллюстрируется фрагментом таблицы А.1.

pc Pica = 12pt
cc Cicero = 12dd
cm Centimeter = 10mm

```
\newcommand{\Rivpt}{\rule{.4pt}{4pt}}
\begin{tabbing}
dd\quad \= \hspace{.65\linewidth} \= \kill
pc      \> Pica = 12pt
    \> \makebox[1pc]{\Rivpt\hrulefill\Rivpt} \\
cc      \> Cicero  = 12dd
    \> \makebox[1cc]{\Rivpt\hrulefill\Rivpt} \\
cm      \> Centimeter = 10mm
    \> \makebox[1cm]{\Rivpt\hrulefill\Rivpt} \\
\end{tabbing}
```

5.2.1 Окружение program

Пакет program, разработанный Мартином Уордом, представляет собой достаточно сложный пример того, что может быть сделано при помощи окружения tabbing. Этот пакет помогает распечатывать тексты компьютерных программ в структурированном виде, причем наилучшим образом он отвечает особенностям языка Pascal. Данный пакет определяет окружения program и programbox, команды для ключевых слов и выделенных (полужирных) букв. Последние из этих команд, которые используются для представления фрагментов программ, имеют один аргумент, формирующий нижний индекс.

Внутри окружения program существенны переводы строки. Каждая строка — в математическом режиме, поэтому пробелы во входном файле несущественны. Команда \\ вызывает разрыв строки при выводе на печать.

Окружение programbox набирает программу в боксе LaTeX'a. Это полезно для фиксации фрагмента программы на одной странице или для набора небольших программ при изготовлении гранок.

Ниже полностью приведена программа на языке Algol 68. Она представляет собой пример нотации, используемой для процедур и функций. Здесь надо обратить внимание на применение команды \mbox для помещения текстовых фрагментов в выходной документ, а также команды \rcomment для формирования комментария, сдвинутого к правому краю документа.

A fast exponentiation procedure:
begin for
  $i := 1$ **to** 10 **step** 1 **do**
  expt$(2, i)$;
  newline() **od**
where A comment flush to the right margin
proc *expt*(x, n) $\equiv$
  $z := 1$;
  do if $n = 0$ **then exit fi**;
    do if *odd*(n) **then exit fi**;
      $n := n/2$; $x := x * x$ **od**;
    $n := n - 1$; $z := z * x$ **od**;
  print(z).
end

```
\begin{programbox}
\mbox{A fast exponentiation procedure:}
\BEGIN %
  \FOR i:=1 \TO 10 \STEP 1 \DO
     |expt|(2,i); \\ |newline|() \OD
\WHERE  \rcomment{A comment flush
                    to the right margin}
\PROC |expt|(x,n) \BODY
       z:=1;
       \DO \IF n=0 \THEN \EXIT \FI;
         \DO \IF |odd|(n) \THEN \EXIT \FI;
            n:=n/2; x:=x*x \OD;
         n:=n-1; z:=z*x \OD;
        |print|(z) \ENDPROC
\END
\end{programbox}
```

5.3 array — расширение окружений tabular

В общем случае, когда требуется формировать таблицы заранее неизвестного уровня сложности, обычно проще воспользоваться окружениями типа tabular, определенными в LaTeX'е. Эти окружения упорядочивают набираемый материал по горизонтали — в строках, и по вертикали — в колонках (столбцах).

\begin{array}[*pos*]{*cols*}	*строки*	\end{array}
\begin{tabular}[*pos*]{*cols*}	*строки*	\end{tabular}
\begin{tabular*}{*width*}[*pos*]{*cols*}	*строки*	\end{tabular*}

В течение ряда лет было выполнено несколько расширений семейства окружений tabular — они описываются в LaTeX book [$\mathcal{L}$ **63–5,182–5**], [$\mathcal{M}$**188–208**]. Оставшаяся часть этой главы посвящена изучению дополнительных функциональных возможностей, предоставляемых пакетом array, разработанным Франком Миттельбахом с участием Дэвида Карлайла.

В табл. 5.1 показаны различные опции для аргумента *cols* преамбулы окружений семейства tabular.

5.3.1 Примеры команд преамбулы

Если имеется потребность использовать специальный шрифт, например, \bfseries, в колонке, выровненной по левому краю, можно написать в преамбуле таблицы >{\bfseries}l. Теперь нет необходимости начинать каждый элемент колонки командой \bfseries.

A	B	*C*
100	**10**	*1*

```
\begin{tabular}{| >{\large}c | >{\large\bfseries}l
                | >{\large\itshape}c |}
\hline A & B & C\\\hline 100 & 10 & 1 \\\hline
\end{tabular}
```

Обратите внимание на использование декларации \extrarowheight во втором примере, представленном ниже. Она добавляет промежуток по вертикали, равный 4pt, над каждой строкой таблицы[1].

A	B	*C*
100	**10**	*1*

```
\setlength{\extrarowheight}{4pt}
\begin{tabular}{| >{\large}c | >{\large\bfseries}l
                | >{\large\itshape}c |}
\hline A & B & C\\\hline 100 & 10 & 1 \\\hline
\end{tabular}
```

Ниже на примерах схематически показано влияние на вид таблицы опций, вставляемых в преамбулу таблицы и влияющих на характер размещения абзацев

[1] Фактически влияние параметра \extrarowheight будет заметно, если только сумма его значения и произведения значений параметров \baselineskip и \arraystretch (см. разд. 5.3.2) больше, чем фактическая высота графы таблицы или, более точно, в случае p, m или b, высота *первой строки* этой графы.

Опции, оставшиеся неизменными	
`l`	Колонка выравнивается по левому краю.
`c`	Колонка центрируется.
`r`	Колонка выравнивается по правому краю.
`p{`*width*`}`	Эквивалентно команде `\parbox[t]{`*width*`}`.
`@{`*decl.*`}`	Подавляет межколонные промежутки и взамен между колонками вставляет *decl.*
Измененная опция	
`|`	Вставляет вертикальную линию. Расстояние между двумя колонками увеличивается на ширину этой линии в противоположность тому, как это имеет место в первоначальном LaTeX'овском определении.
Новые опции	
`m{`*width*`}`	Определяет колонку ширины *width*. Каждая графа (элемент) колонки, центрируется по вертикали. Это напоминает в известной степени команду `\parbox{`*width*`}`.
`b{`*width*`}`	Совпадает с командой `\parbox[b]{`*width*`}`.
`>{`*decl.*`}`	Можно использовать перед опциями `l`, `r`, `c`, `p`, `m` или `b`. Вставляет *decl.* непосредственно перед элементом колонки.
`<{`*decl.*`}`	Можно использовать после опций `l`, `r`, `c`, `p{..}`, `m{..}` или `b{..}`. Вставляет *decl.* сразу после элемента колонки.
`!{`*decl.*`}`	Можно использовать в любом месте преамбулы таблицы; соответствует опции `|`. Разница между этими двумя опциями в том, что вместо вертикальной линии вставляется *decl.*, так что эта опция, в противоположность `@{...}`, не подавляет промежуток между колонками.

Таблица 5.1. Опции, помещаемые в преамбулу пакета array

в колонках: `p` (бокс, содержащий абзац, выравнивается по верхней строке), `m` (бокс центрируется по высоте) и `b` (бокс выравнивается по нижней строке).

1 1 1 1 1 1 1 1 1 1 1 1	2 2 2 2 2 2 2 2	3 3 3 3

```
\begin{tabular}{|p{1cm}|p{1cm}|p{1cm}|}
\hline 1 1 1 1 1 1 1 1 1 1 1 1 &
       2 2 2 2 2 2 2 2         & 3 3 3 3 \\
       \hline
\end{tabular}
```

1 1 1 1 1 1 1 1 1 1 1 1	2 2 2 2 2 2 2 2	3 3 3 3

```
\begin{tabular}{|m{1cm}|m{1cm}|m{1cm}|}
\hline 1 1 1 1 1 1 1 1 1 1 1 1 &
       2 2 2 2 2 2 2 2         & 3 3 3 3 \\
       \hline
\end{tabular}
```

1 1 1 1 1 1 1 1 1 1 1 1	2 2 2 2 2 2 2 2	3 3 3 3

```
\begin{tabular}{|b{1cm}|b{1cm}|b{1cm}|}
\hline 1 1 1 1 1 1 1 1 1 1 1 1 &
       2 2 2 2 2 2 2 2         & 3 3 3 3 \\
       \hline
\end{tabular}
```

В колонках, соответствующих опциям p, m или b, для параметра \parindent по умолчанию принято значение 0pt. Оно может быть изменено командой \setlength, например, >{\setlength{\parindent}{1cm}}p{...}.

1 2 3 4 5 6	1 2 3 4 5 6 7 8
7 8 9 0 1 2 3 4	9 0 1 2 3 4 5 6
5 6 7 8 9 0	7 8 9 0

```
\begin{tabular}
 {|>{\setlength{\parindent}{5mm}}p{2cm}|p{2cm}|}
\hline
  1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 &
  1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 \\
\hline
\end{tabular}
```

Опция < была первоначально введена для следующего применения: запись >{$}c<{$} в окружении tabular в математическом режиме порождает колонку. Использование преамбулы такого вида в окружении array приводит к колонке в LR-режиме, так как дополнительные знаки $ упраздняют уже существующие знаки $.

$10!^{10!}$	большое число
10^{-999}	маленькое число

```
\setlength{\extrarowheight}{2pt}
\[  \begin{array}{|l|>{$}l<{$}|}    \hline
      10!^{10!}        & большое число   \\
      10^{-999}        & маленькое число \\
    \hline  \end{array}   \]
```

Имеется важное различие между r@{\hspace{5mm}}l и r!{\hspace{5mm}}l . В первом случае межколонный промежуток *в точности равен* 5мм. Во втором варианте промежуток в 5мм *добавляется* к промежутку между колонками (в зависимости от окружения используется \arraycolsep или \tabcolsep).

LEFT BOX	RIGHT BOX
LEFT BOX	RIGHT BOX

```
\begin{tabular}{r@{\hspace{5mm}}l}
  \fbox{LEFT BOX} & \fbox{RIGHT BOX}
\end{tabular}
\par \vspace{\baselineskip} \par
\begin{tabular}{r!{\hspace{5mm}}l}
  \fbox{LEFT BOX} & \fbox{RIGHT BOX}
\end{tabular}
```

Вертикальные линейки переменной толщины

Вертикальные линейки переменной толщины могут порождаться либо при помощи декларации !{*decl*}, либо при помощи базовой TEX'овской команды \vrule с аргументом, задающим толщину. Эта команда используется в связи с тем, что она автоматически учитывает высоту колонки, тогда как команда LATEX'а \rule требует явного задания этой высоты.

A	B	C
100	10	1

```
\begin{tabular}{|c!{\vrule width 3pt}c|c|}
\hline
 A   & B & C  \\ \hline
100 & 10 & 1  \\ \hline
\end{tabular}
```

Верстка узких колонок

В случае когда таблица должна содержать узкую колонку, надо не только обеспечить, чтобы первое слово переносилось (см. с. 83), но и принять во внимание, что тексты с такими короткими строками проще набирать в режиме `ragged right`, т. е. с невыровненным правым краем. Этого можно добиться, если набираемому материалу будет предшествовать команда `\raggedright` (см. разд. 3.1.4). Поскольку рассматриваемое окружение переопределяет команду разрыва строки `\\`, необходимо сохранить (запомнить) команду окончания строки `\\` окружений `tabular` и `array`. С этой целью определим команду `\PreserveBackslash`, которая восстановит правильное значение команды `\\`.

```
\newcommand{\PreserveBackslash}[1]{\let\temp=\\#1\let\\=\temp}
```

Как видно из следующего ниже примера, теперь можно набирать текст внутри окружения `tabular` прижатым влево, вправо или центрированным, кроме того, имеется возможность контролировать разрывы строк. Видно также, что первое слово в каждой из колонок перенесено корректно, хотя в случае третьей колонки, содержащей текст на голландском языке, пришлось немного помочь TEX'у, указав вручную возможные точки переносов. Заметим также, что акцентированное французское слово «Possibilités» подверглось переносу, тогда как второе аналогичное слово «espérances» осталось неперенесенным. Это произошло вследствие того, что в первом случае точка переноса лежит до акцентированной буквы, а во втором — после такой буквы.

Super-conscious-ness is a long word	Possi-bilités et espérances	Mogelijk-heden en hoop
Ragged left text in column one	Centered text in column two	Ragged right text in column three

```
\let\PBS=\PreserveBackslash %shorthand
\begin{tabular}%
      {|>{\PBS\raggedleft\hspace{0pt}}
                                  p{14mm}%
       |>{\PBS\centering\hspace{0pt}}
                                  p{14mm}%
       |>{\PBS\raggedright\hspace{0pt}}
                                  p{14mm}|}
\hline
  Superconsciousness is a long word      &
  Possibilit\'es et esp\'erances         &
  Moge\-lijk\-heden en hoop              \\
  \hline
  Ragged left text in column one         &
  Centered text    in column two         &
  Ragged right text in  column three     \\
  \hline
\end{tabular}
```

Управление промежутками между колонками таблицы

На нескольких примерах рассмотрим, каким образом окружение `tabular` управляет шириной колонок, расстоянием между содержимым и разделителями колонок, а также видом линеек.

Когда вертикальных линеек нет, использование спецификаторов @{} исключает пробелы по горизонтали между колонками. Содержимое таблицы, рассматриваемой в примерах, описывается следующим образом:

```
\begin{tabular}{...} %%%%%% <---- Изменяемая часть
  BOXES & BOXES  \\   BOXES & BOXES  \\
\end{tabular}
```

Таблицы, показанные ниже, получены при различных видах преамбулы, добавляемой к данному описанию. Вид преамбулы, соответствующей каждой из таблиц, показан справа от этих таблиц.

BOXES BOXES BOXES BOXES	`{ll}`
BOXES BOXES BOXES BOXES	`{@{}ll@{}}`
BOXESBOXES BOXESBOXES	`{@{}l@{}l@{}}`

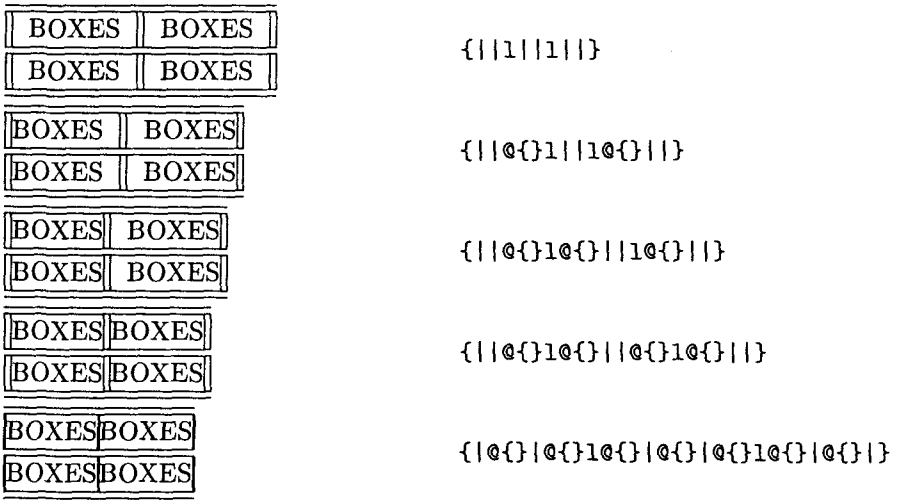
Если в формируемой таблице есть вертикальные линейки-разделители, имеется возможность управлять величиной промежутка до и после этих разделителей. Чтобы показать, какое влияние на вид таблицы оказывает наличие горизонтальных линеек, введем их в следующем примере.

```
\begin{tabular}{...} %%%%%% <---- Изменяемая часть
                      \hline
  BOXES & BOXES  \\  \hline
  BOXES & BOXES  \\  \hline
\end{tabular}
```

BOXES BOXES BOXES BOXES	`{|l|l|}`
BOXES BOXES BOXES BOXES	`{|@{}l|l@{}|}`
BOXES BOXES BOXES BOXES	`{|@{}l@{}|l@{}|}`
BOXESBOXES BOXESBOXES	`{|@{}l@{}|@{}l@{}|}`

Увеличим сложность примеров, использующих окружение tabular, добавив в них двойные линейки.

```
\begin{tabular}{...} %%%%%% <---- Изменяемая часть
  \hline\hline
  BOXES & BOXES  \\   \hline\hline
  BOXES & BOXES  \\   \hline\hline
\end{tabular}
```



В общем случае, если необходимо устранить все промежутки между колонками, более прямой путь — переопределить длины \arraycolsep и \tabcolsep. Более того, последний пример показывает неверное использование опции @{...}, так как для управления промежутком между двойными вертикальными (и аналогичными горизонтальными) линейками-разделителями, следует переопределить параметр \doublerulesep.

5.3.2 Стилевые параметры

Результатами работы окружений типа tabular [$\mathcal{L}$ 185], [$\mathcal{A}$ 199] можно управлять при помощи различных стилевых параметров. Значения этих параметров можно менять при помощи команд \setlength и \addtolength, причем делать это можно в любом месте документа. Область их действия может быть как глобальной, так и локальной. В последнем случае данная область должна быть ограничена другим окружением или при помощи скобок {}.

\arraycolsep Половина ширины промежутка между колонками по горизонтали в окружении array (значение по умолчанию равно 5 pt).

\tabcolsep Половина ширины промежутка между колонками по горизонтали в окружении tabular (значение по умолчанию равно 6 pt).

\arrayrulewidth Ширина вертикальной линейки, разделяющей колонки (если в преамбуле окружения задана опция |), или линеек, создаваемых командами \hline, \cline или \vline (значение по умолчанию равно 0.4 pt). При использовании пакета array значение этого параметра принимается во внимание при определении ширины таблицы (стандартный LaTeX формирует линейки таким образом, чтобы они не влияли на окончательную ширину таблицы).

\doublerulesep Ширина промежутка между строками, порождаемого парой опций || в преамбуле окружения или же двумя последовательными командами \hline (значение по умолчанию равно 2 pt).

\arraystretch Коэффициент, на который умножается нормальное значение межстрочного промежутка; например, значение 1.5 приведет к увеличению промежутка между строками на 50% . Значение его устанавливается командой \renewcommand (значение по умолчанию равно 1 pt).

\extrarowheight Дополнительный параметр, введенный в пакете array. Его значение добавляется к нормальной высоте каждой строки таблицы, в то время как глубина остается неизменной. Это важно для таблиц с горизонтальными строками, поскольку часто требуется точная подстройка расстояния между этими строками и содержимым таблицы (значение по умолчанию равно 0 pt).

Окружение TabularC, приведенное ниже, показывает, как эти параметры используются для формирования таблицы с заданным числом колонок равной ширины, при полной ширине таблицы, равной \linewidth. Здесь используется пакет calc, рассматриваемый в разд. А.4, а также команда \PreserveBackslash, определенная на с. 127. Окружение TabularC содержит в качестве аргумента требуемое число колонок. Это число (обозначим его через x) используется для вычисления фактической ширины колонок таблицы путем вычитания из полной ширины таблицы (она равна \linewidth) $2x$ промежутков между колонками и $(x+1)$ толщин линеек-разделителей. Полученное значение делится на x, чтобы получить ширину одной колонки. Содержимое колонки центрируется, затем восстанавливается команда \\, а также разрешается выполнить перенос первого слова.

```
\let\PBS=\PreserveBackslash
\newlength{\tmplength}
\newenvironment{TabularC}[1]
  {%
   \setlength{\tmplength}{\linewidth/(#1)
                          - \tabcolsep*2
                          - \arrayrulewidth*(#1+1)/(#1)}%
   \par\begin{tabular*}{\linewidth}%
                       {*{#1}{|>{\PBS\centering\hspace{0pt}}%
                        p{\the\tmplength}}|}%
  }
  {\end{tabular*}\par}
```

Один из возможных вариантов использования этого определения дает следующий результат:

Material in column one	column two	This is column three
Column one again	and column two	This is column three
Once more column one	column two	Last time column three

```
\begin{TabularC}{3}
\hline
Material in column one & column two
    & This is column three    \\ \hline
Column one  again      & and column two
    & This is column three    \\ \hline
Once more column one   & column two
    & Last time column three  \\ \hline
\end{TabularC}
```

Несложно построить окружения и для других конфигураций колонок. Если с этими командами и параметрами вы чувствуете себя несколько неуютно, можно воспользоваться окружением tabularx, которое поможет вам выполнить все эти вычисления (см. для сравнения разд. 5.3.5 и рис. 5.1).

5.3.3 Определение новых спецификаторов колонок

Если в таблицу надо включить колонку, отличающуюся по виду от стандартных вариантов, было бы удобно иметь возможность записать:

>{*some declarations*}c<{*some more decls*}

Однако, если использовать колонки приходится часто, такая форма их записи становится слишком многословной. В связи с этим для повторяющегося использования заданного некоторым образом спецификатора определена следующая команда для определения нового типа колонки:

\newcolumntype{*col*}[*narg*]{*decl*}

Здесь *col* — однобуквенный спецификатор, описывающий в преамбуле новый тип колонки, *narg* — необязательный параметр, задающий число аргументов в спецификаторе, *decl* — собственно описание, например:

\newcolumntype{x}{>{*some declarations*}c<{*some more decls*}}

Вновь определенный здесь спецификатор колонки x может быть использован теперь в качестве аргумента преамбул всех окружений типа array и tabular, когда возникает необходимость организовать колонку, описываемую спецификатором данного вида.

Весьма часто в окружении tabular или array могут потребоваться колонки в математическом режиме или LR-режиме. В такой ситуации можно записать следующие определения:

```
\newcolumntype{C}{>{$}c<{$}}
\newcolumntype{L}{>{$}l<{$}}
\newcolumntype{R}{>{$}r<{$}}
```

После того как эти определения записаны, можно использовать спецификатор C для получения центрированного LR-режима в окружении array, или же центрированного математического режима в окружении tabular.

Отметим, что команда \newcolumntype имеет тот же самый первый необязательный аргумент, что и команда \newcommand, определяющий число аргументов в спецификаторе колонки. Однако текущая реализация LaTeX'a не поддерживает расширенного синтаксиса команды \newcommand, введенного в LaTeX 2_ε.

Существенно другое использование команды \newcolumntype опирается на тот факт, что заменяемый текст в \newcolumntype может ссылаться более чем на одну колонку. Предположим, что документ содержит несколько окружений tabular, в которых преамбула одна и та же, и есть необходимость поэкспериментировать с различными преамбулами. В этом случае можно записать:

```
\newcolumntype{Z}{clr}
\begin{tabular}{Z}
  ...
\end{tabular}
```

Заменяющий текст в команде \newcolumntype может быть любым из базисных средств (примитивов) окружения array, или же некоторой новой буквой-спецификатором, определенной в другой команде \newcolumntype.

Чтобы отобразить на терминале список всех активных в данный момент определений \newcolumntype, следует использовать в преамбуле команду \showcols.

5.3.4 Некоторые особенности реализации пакета array

Сообщения об ошибках, порождаемые в процессе грамматического разбора спецификации колонки, относятся к аргументу преамбулы *после* того, как он переписан командой \newcolumntype, а не в момент входа в преамбулу.

На использование команды \extracolsep наложено ограничение, заключающееся в том, что может быть по крайней мере одна команда \extracolsep на @-выражение или !-выражение. Команда эта должна записываться не как часть макроопределения, а вводиться непосредственно в @-выражение. Таким образом, команда \newcommand{\ef}{\extracolsep{\fill}} ... @{\ef} в пакете array работать не будет. Вместо нее в данном случае можно рекомендовать такой вариант: \newcolumntype{e}{@{\extracolsep{\fill}}}.

5.3.5 tabularx — автоматическое вычисление ширины колонок

Пакет tabularx (автор — Дэвид Карлайл) реализует вариант окружения `tabular*`, в котором ширина некоторых колонок может быть определена автоматически в зависимости от общей ширины этой таблицы. Колонки, ширину которых следует установить автоматически, обозначаются в преамбуле опцией (спецификатором) `X`. Для получения правильной ширины колонки такая спецификация преобразуется к виду `p{`*some value*`}`.

Следующая ниже таблица, содержащая заглавия комедий Шекспира, была задана как `\begin{tabularx}{100mm}{|X|X|X|}`.

The Two Gentlemen of Verona	The Taming of the Shrew	The Comedy of Errors
Love's Labour's Lost	A Midsummer Night's Dream	The Merchant of Venice
The Merry Wives of Windsor	Much Ado About Nothing	As You Like It
Twelfth Night	Troilus and Cressida	Measure for Measure
All's Well That Ends Well	Pericles Prince of Tyre	The Winter's Tale
Cymbeline	The Tempest	

Если изменить вид ее представления на

```
\begin{tabularx}{\linewidth}{|X|X|X|}
```

то получим такую таблицу:

The Two Gentlemen of Verona	The Taming of the Shrew	The Comedy of Errors
Love's Labour's Lost	A Midsummer Night's Dream	The Merchant of Venice
The Merry Wives of Windsor	Much Ado About Nothing	As You Like It
Twelfth Night	Troilus and Cressida	Measure for Measure
All's Well That Ends Well	Pericles Prince of Tyre	The Winter's Tale
Cymbeline	The Tempest	

Команды, используемые для набора X-колонок

По умолчанию X-спецификация преобразуется в спецификацию p{*some value*}. Для узких колонок часто требуется специальный формат, который может быть получен использованием >-синтаксиса. Таким способом можно задавать спецификации вроде >{\small}X. Другой формат, полезный при наборе узких колонок — формирование текста с невыровненным правым краем. Однако LaTeX'овское макроопределение \raggedright переопределяет команду \\ способом, противоречащим ее использованию в окружениях array и tabular (см. также обсуждение команды \PreserveBackslash на с. 127). Исходя из этого, в пакете tabularx определена команда \arraybackslash. Ее разрешается использовать после деклараций \raggedright, \raggedleft и \centering. В результате в преамбуле окружения tabularx можно записать >{\raggedright\arraybackslash}X. Эта спецификация запоминается при помощи команды \newcolumntype{Y}{>{\small\raggedright\arraybackslash}X}. Теперь можно использовать спецификатор Y в качестве аргумента преамбулы окружения tabularx.

X-колонки формируются путем использования p-спецификации, которая отвечает команде \parbox[t]. Может возникнуть потребность проведения такого формирования на основе какой-либо другой спецификации, например, m-спецификации, отвечающей команде \parbox[c]. Используя >-синтаксис, изменить тип спецификации колонки нельзя, для этой цели необходимы другие средства. Можно определить (как макро) команду \tabularxcolumn с одним аргументом, которая расширяет преамбулу окружения tabular таким образом, чтобы допускать использование спецификатора X. Когда данная команда исполняется, значение ее аргумента определяет фактическую ширину колонки.

Определение команды \tabularxcolumn по умолчанию имеет вид:

```
\newcommand{\tabularxcolumn}[1]{p{#1}}.
```

Возможно также альтернативное определение:

```
\renewcommand{\tabularxcolumn}[1]{>{\small}m{#1}}
```

Ширина колонок

Как правило, все X-колонки в отдельной таблице формируются имеющими одинаковую ширину. Тем не менее в окружении tabularx возможно формирование и колонок с разной шириной. Преамбула вида {>{\setlength{\hsize}{.5\hsize}}X>{\setlength{\hsize}{1.5\hsize}}X} например, задает две колонки, из которых вторая втрое шире первой. Однако, используя данный метод, необходимо следовать таким двум правилам:

- Сумма ширин всех X-колонок должна оставаться неизменной. В приведенном примере новые ширины добавились к ширинам двух стандартных X-колонок.

- Не допускаются элементы, порождаемые командой \multicolumn, пересекающие некоторую X-колонку.

Superconsciousness is a long word	Mogelijkheden en hoop
Some text in column one	A somewhat longer text in column two

```
\tracingtabularx
\begin{tabularx}{\linewidth}%
 {|>{\setlength{\hsize}{.85\hsize}}X|%
  >{\setlength{\hsize}{1.15\hsize}}X|}
Superconsciousness is a long word &
Moge\-lijk\-heden en hoop          \\
Some text in column one          &
A somewhat longer text in column two\\
\end{tabularx}
```

Различия между окружениями tabularx и tabular*

Как окружение tabular* (стандартное средство LaTeX'a), так и окружение tabularx используют одни и те же аргументы, чтобы организовать таблицу заданной ширины. Ниже перечислены основные различия между этими двумя окружениями.

- Окружение tabularx модифицирует ширину *колонок*, тогда как tabular* изменяет величину *промежутков* между колонками.
- Окружения tabular и tabular* могут быть вложенными без ограничений. Однако, если одно окружение tabularx появляется внутри другого, внутреннее окружение *должно* быть заключено в скобки { }.
- Тело окружения tabularx является фактически аргументом для команды. Из этого обстоятельства вытекает ряд ограничений: команды \verb и \verb* можно использовать, однако они могут некорректно обращаться с пробелами; аргумент не может содержать символ %, а также непарные скобки { и }.
- Окружение tabular* использует базовую возможность TeX'a модифицировать межколонные промежутки при выравнивании. Окружение tabularx предпринимает несколько попыток при организации (верстке) таблицы, чтобы найти оптимальную ширину колонок. Вследствие этого tabularx много медленнее, чем tabular*. Тот факт, что тело таблицы расширяется несколько раз, может негативно повлиять на некоторые конструкции TeX'a.

Рисунок 5.1 дает сравнение окружения tabularx с окружением TabularC, описанным в разд. 5.3.2.

Трассировка выхода на терминале

\tracingtabularx

Если задана эта декларация, например, в преамбуле документа, то все следующие далее окружения tabularx будут выдавать информацию о ширинах колонок, когда они будут последовательно изменяться при поиске подходящих значе-

```
\begin{TabularC}{4}                                                  \hline
Material in column one        & column two              &
This is column three          & column four                      \\
Material in column one  again & and column two         &
This is column three          & column four                      \\
Once more column one          & column two              &
column three                  & \hfill column four               \\\hline
\end{TabularC}

\begin{tabularx}{\linewidth}{|X|X|X|X|}                             \hline
        . . . . . same contents . . . . .
                                                                     \hline
\end{tabularx}

\newcolumntype{Y}{>{\centering\arraybackslash}X}%
\begin{tabularx}{\linewidth}{|Y|Y|Y|Y|}                             \hline
        . . . . . same contents . . . . .
                                                                     \hline
\end{tabularx}
```

Material in column one	column two	This is column three	column four
Material in column one again	and column two	This is column three	column four
Once more column one	column two	column three	column four

Material in column one	column two	This is column three	column four
Material in column one again	and column two	This is column three	column four
Once more column one	column two	column three	column four

Material in column one	column two	This is column three	column four
Material in column one again	and column two	This is column three	column four
Once more column one	column two	column three	column four

Рис. 5.1. Сравнение окружений `TabularC` и `tabularx`

Как видно из примера, приведенного выше, оба окружения — `TabularC` и `tabularx` — могут решать задачу верстки требуемых таблиц. Дэвид Карлайл, автор пакета tabularx, заметил тем не менее: «Если пользователь ничего не имеет против работы с арифметическими возможностями TEX'а, то в ситуации, где ширину колонок можно вычислить заранее, метод, реализованный в `TabularC`, много лучше, поскольку он предотвращает многоступенчатое наращивание ширины таблицы (которое применяется в пакете tabularx). Окружение `tabularx` предпочтительнее главным образом тогда, когда часть колонок имеют тип `c`, так что ширина колонок типа `p` (p-колонок) не может быть вычислена заранее, поскольку зависит от вида элементов (записей), составляющих таблицу.»

ний. Например, в случае последнего из рассмотренных примеров будет получен следующий листинг:

```
Target width: \linewidth  = 204.0pt.
    Table Width     Column Width     X Columns
    433.19998pt     204.0pt            3
    203.99998pt     89.40001pt         2
Reached target.
```

5.3.6 delarray — определение вида ограничителей для окружения array

Полезное расширение пакета array, касающееся вида ограничителей, может быть найдено в пакете delarray (автор — Дэвид Карлайл). Он позволяет пользователю задать ограничители, которые должны стоять справа и слева от результата, порождаемого окружением array, посредством неявного введения пары команд \left...\right вокруг окружения в целом.

$$\left(\begin{array}{cc} A & B \\ C & D \end{array} \right)$$

```
\[ \begin{array}{cc}
      A & B  \\  C & D
   \end{array}
\]
```

В преамбуле может быть использован любой ограничитель. Пользователям систем Plain TeX и $\mathcal{AMS}$-TEX знакомы, например, следующие окружения (см. также разд. 8.4.1 и 8.4.2):

```
\newcolumntype{L}{>{$}l<{$}}
\newenvironment{Cases}{\begin{array}\{{1L}.}{\end{array}}
\newenvironment{Matrix}{\begin{array}|{*{20}{c}}|}{\end{array}}
\newenvironment{Pmatrix}{\begin{array}({*{20}{c}})}{\end{array}}
```

$$|x| = \begin{cases} x, & \text{if } x \ge 0; \\ -x, & \text{otherwise.} \end{cases}$$

```
\[  |x| = \begin{Cases}
            x,& if $x\ge0$; \\[2mm]
           -x,& otherwise.  \\
          \end{Cases}
\]
```

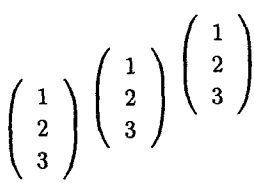
$$a = \begin{vmatrix} x-\lambda & 1 & 0 \\ 0 & x-\lambda & 1 \\ 0 & 0 & x-\lambda \end{vmatrix}^2.$$

```
\[  a = {\begin{Matrix}
            x-\lambda & 1 & 0 \\
            0 & x-\lambda & 1 \\
            0 & 0 & x-\lambda \\
         \end{Matrix}
        }^2.
\]
```

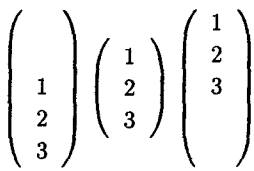
$$A = \begin{pmatrix} x-\lambda & 1 & 0 \\ 0 & x-\lambda & 1 \\ 0 & 0 & x-\lambda \end{pmatrix}.$$

```
\[  A = \begin{Pmatrix}
            x-\lambda & 1 & 0 \\
            0 & x-\lambda & 1 \\
            0 & 0 & x-\lambda \\
        \end{Pmatrix}.
\]
```

Это свойство особенно полезно, когда используются аргументы [t] или [b]. В этих случаях получаемый результат не эквивалентен простому заключению окружения внутрь пары команд \left...\right, как это можно видеть из следующего ниже примера:



```
\[  \begin{array}[t]{c}1\\2\\3\end{array}
    \begin{array}[c]{c}1\\2\\3\end{array}
    \begin{array}[b]{c}1\\2\\3\end{array}
\]
```



```
\[  \left(
      \begin{array}[t]{c}1\\2\\3\end{array}
    \right)
    \left(
      \begin{array}[c]{c}1\\2\\3\end{array}
    \right)
    \left(
      \begin{array}[b]{c}1\\2\\3\end{array}
    \right)                                  \]
```

5.4 Многостраничные таблицы

В первоначальном варианте, реализованном Лесли Лэмпортом, таблицы, формируемые при помощи окружения tabular, должны были размещаться целиком на одной странице. Если это условие не выполнялось, таблица выходила за пределы страницы и выдавалось сообщение Overfull \vbox (переполнение бокса при работе в вертикальном режиме).

Формирование таблиц, требующих для своего размещения более чем одну страницу, может быть осуществлено при помощи пакетов supertab и longtable. Возможности их примерно одинаковы, однако при верстке несложных таблиц команды пакета supertab, по-видимому, более понятны на интуитивном уровне. Если же необходимы более тонкие средства управления шириной таблицы, то целесообразнее использовать пакет longtable. В следующих разделах рассмотрены возможности каждого из этих двух пакетов, после чего с использованием более сложного примера будет продемонстрирована разница между подходами, реализуемыми данными пакетами.

5.4.1 supertab — верстка многостраничных таблиц

Пакет supertab (автор первоначального варианта — Тео Юринс, переработанного — Йоханнес Брамс) определяет окружение supertabular. Данный пакет основан на использовании окружения tabular в сочетании с процедурой проверки места, остающегося на формируемой странице, после обнаружения очередной команды \\. Если при такой проверке обнаруживается, что достигнуто предель-

ное значение параметра \textheight, то генерируется команда \end{tabular}, производится переход на новую страницу, на ней формируется головка таблицы, после чего продолжается обработка окружения tabular. Из этого следует, что ширина колонок и, следовательно, таблицы в целом может меняться от страницы к странице.

К числу новых команд, вводимых в окружении supertabular, относятся следующие:

\tablehead{...}	Задает содержимое головки («шапки») таблицы, за исключением случаев, когда перед этой командой записана команда \tablefirsthead. Аргумент этой команды может содержать как строки таблицы (заканчивающиеся командой \\), так и межстрочные элементы, например, задаваемые командой \hline.
\tablefirsthead{...}	Определяет содержимое головки таблицы при ее первом появлении. Этой командой, являющейся необязательной, следует пользоваться в тех случаях, когда начальная головка таблицы (т. е. головка той ее части, которая будет размещена на первой из страниц, занимаемых таблицей) отличается от той, которая определяется командой \tablehead.
\tabletail{...}	Определяет материал, который должен быть размещен в конце каждого из фрагментов таблицы (размещенного на соответствующей странице), за исключением завершающего фрагмента, если используется специальная команда \tablelasttail.
\tablelasttail{...}	Определяет материал, который должен быть помещен в конец таблицы, формируемой при помощи окружения supertabular. Эта команда относится к необязательным и необходима в тех случаях, когда вид завершающей части многостраничной таблицы отличается от того, который задается командой \tabletail.
\topcaption{...} \bottomcaption{...} \tablecaption{...}	Определяет заголовок (наименование) таблицы, который будет помещен окружением supertabular над таблицей (\topcaption) или под ней (\bottomcaption). Если используется команда \tablecaption, то заголовок будет размещен так, как это принято по умолчанию (над таблицей).

Внутри окружения supertabular переход к новой строке таблицы обозначается обычным способом, при помощи команды \\. Можно использовать все команды формирования колонок, включая @{...} и p{...}. Однако необяза-

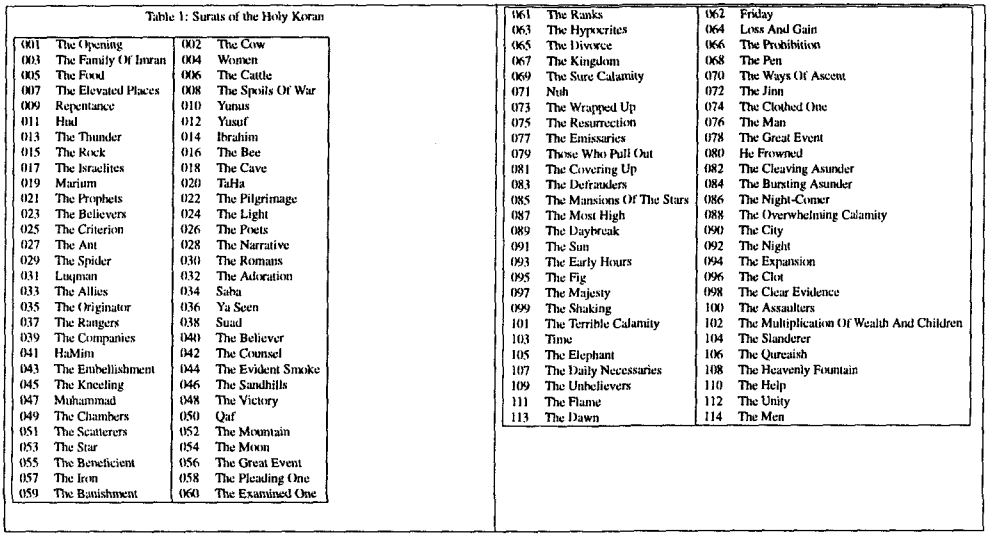


Table 1: Surats of the Holy Koran

001	The Opening	002	The Cow
003	The Family Of Imran	004	Women
005	The Food	006	The Cattle
007	The Elevated Places	008	The Spoils Of War
009	Repentance	010	Yunus
011	Hud	012	Yusuf
013	The Thunder	014	Ibrahim
015	The Rock	016	The Bee
017	The Israelites	018	The Cave
019	Marium	020	TaHa
021	The Prophets	022	The Pilgrimage
023	The Believers	024	The Light
025	The Criterion	026	The Poets
027	The Ant	028	The Narrative
029	The Spider	030	The Romans
031	Luqman	032	The Adoration
033	The Allies	034	Saba
035	The Originator	036	Ya Seen
037	The Rangers	038	Suad
039	The Companies	040	The Believer
041	HaMim	042	The Counsel
043	The Embellishment	044	The Evident Smoke
045	The Kneeling	046	The Sandhills
047	Muhammad	048	The Victory
049	The Chambers	050	Qaf
051	The Scatterers	052	The Mountain
053	The Star	054	The Moon
055	The Beneficient	056	The Great Event
057	The Iron	058	The Pleading One
059	The Banishment	060	The Examined One
061	The Ranks	062	Friday
063	The Hypocrites	064	Loss And Gain
065	The Divorce	066	The Prohibition
067	The Kingdom	068	The Pen
069	The Sure Calamity	070	The Ways Of Ascent
071	Nuh	072	The Jinn
073	The Wrapped Up	074	The Clothed One
075	The Resurrection	076	The Man
077	The Emissaries	078	The Great Event
079	Those Who Pull Out	080	He Frowned
081	The Covering Up	082	The Cleaving Asunder
083	The Defrauders	084	The Bursting Asunder
085	The Mansions Of The Stars	086	The Night-Comer
087	The Most High	088	The Overwhelming Calamity
089	The Daybreak	090	The City
091	The Sun	092	The Night
093	The Early Hours	094	The Expansion
095	The Fig	096	The Clot
097	The Majesty	098	The Clear Evidence
099	The Shaking	100	The Assaulters
101	The Terrible Calamity	102	The Multiplication Of Wealth And Children
103	Time	104	The Slanderer
105	The Elephant	106	The Qureaish
107	The Daily Necessaries	108	The Heavenly Fountain
109	The Unbelievers	110	The Help
111	The Flame	112	The Unity
113	The Dawn	114	The Men

Рис. 5.2. Суры Корана (пример применения окружения `supertabular`)

тельные параметры позиционирования колонок, например, `t` и `b`, задаваемые в `\begin{tabular}` и `\begin{tabular*}`, применять не разрешается.

Пример применения окружения `supertabular`

Первый пример (рис. 5.2) имеет простую структуру, и формирование многостраничной таблицы в этом случае не вызывает затруднений. Обратите внимание на изменение ширины таблицы при переходе от ее первого фрагмента ко второму (левая и правая половины рисунка соответственно). Этот пример был получен с использованием следующей совокупности команд:

```
\tablecaption{Surats of the Holy Koran}
\tablehead{\hline}
\tabletail{\hline}
\begin{supertabular}{|c|c|c|c|}
001 & The Opening             & 002 & The Cow              \\
003 & The Family Of Imran     & 004 & Women                \\
  ....
113 & The Dawn                & 114 & The Men              \\
\end{supertabular}
```

Пример применения окружения `supertabular*`

Ширина многостраничной таблицы может быть зафиксирована, например, ее можно положить равной ширине текста на странице, задаваемой при помощи

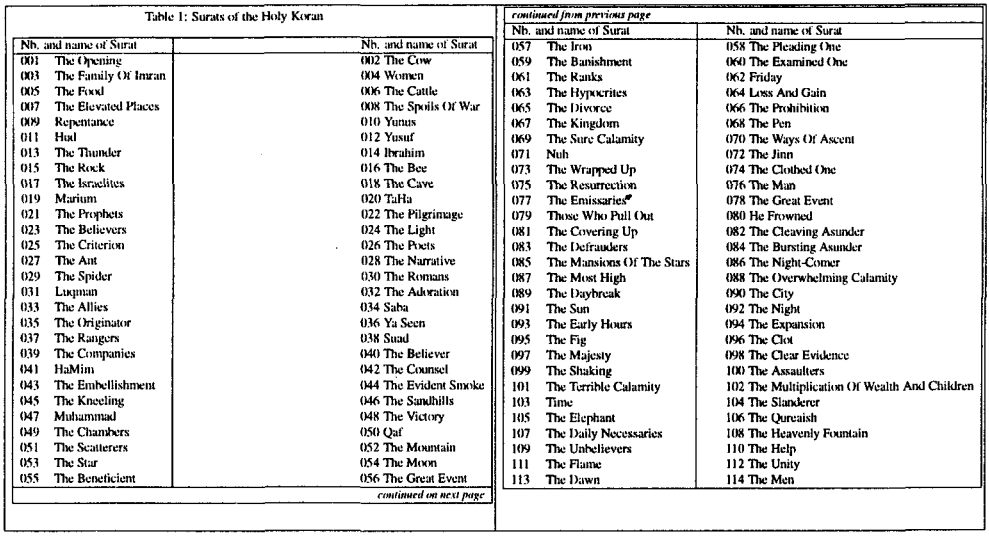


Table 1: Surats of the Holy Koran

Nb.	and name of Surat	Nb.	and name of Surat
001	The Opening	002	The Cow
003	The Family Of Imran	004	Women
005	The Food	006	The Cattle
007	The Elevated Places	008	The Spoils Of War
009	Repentance	010	Yunus
011	Hud	012	Yusuf
013	The Thunder	014	Ibrahim
015	The Rock	016	The Bee
017	The Israelites	018	The Cave
019	Marium	020	TaHa
021	The Prophets	022	The Pilgrimage
023	The Believers	024	The Light
025	The Criterion	026	The Poets
027	The Ant	028	The Narrative
029	The Spider	030	The Romans
031	Luqman	032	The Adoration
033	The Allies	034	Saba
035	The Originator	036	Ya Seen
037	The Rangers	038	Suad
039	The Companies	040	The Believer
041	HaMim	042	The Counsel
043	The Embellishment	044	The Evident Smoke
045	The Kneeling	046	The Sandhills
047	Muhammad	048	The Victory
049	The Chambers	050	Qaf
051	The Scatterers	052	The Mountain
053	The Star	054	The Moon
055	The Beneficient	056	The Great Event

continued on next page

continued from previous page

Nb.	and name of Surat	Nb.	and name of Surat
057	The Iron	058	The Pleading One
059	The Banishment	060	The Examined One
061	The Ranks	062	Friday
063	The Hypocrites	064	Loss And Gain
065	The Divorce	066	The Prohibition
067	The Kingdom	068	The Pen
069	The Sure Calamity	070	The Ways Of Ascent
071	Nuh	072	The Jinn
073	The Wrapped Up	074	The Clothed One
075	The Resurrection	076	The Man
077	The Emissaries	078	The Great Event
079	Those Who Pull Out	080	He Frowned
081	The Covering Up	082	The Cleaving Asunder
083	The Defrauders	084	The Bursting Asunder
085	The Mansions Of The Stars	086	The Night-Comer
087	The Most High	088	The Overwhelming Calamity
089	The Daybreak	090	The City
091	The Sun	092	The Night
093	The Early Hours	094	The Expansion
095	The Fig	096	The Clot
097	The Majesty	098	The Clear Evidence
099	The Shaking	100	The Assaulters
101	The Terrible Calamity	102	The Multiplication Of Wealth And Children
103	Time	104	The Slanderer
105	The Elephant	106	The Qureaish
107	The Daily Necessaries	108	The Heavenly Fountain
109	The Unbelievers	110	The Help
111	The Flame	112	The Unity
113	The Dawn	114	The Men

Рис. 5.3. Суры Корана (пример применения окружения `supertabular*`)

параметра `\textwidth` (см. рис. 5.3 в качестве примера). По сравнению с рис. 5.2, в рис. 5.3 добавились головки и завершения фрагментов таблицы на обеих страницах, а также пустая (растяжимая) вертикальная полоса между колонками, добавленная в первый фрагмент для того, чтобы уравнять его ширину с шириной второй колонки. Обращает на себя внимание различная величина промежутков между двумя первыми и двумя последними колонками. Рассматриваемая таблица была построена при помощи следующих команд:

```
\tablecaption{Surats of the Holy Koran}
\tablefirsthead{\hline \multicolumn{2}{|l|}{Nb. and name of Surat}
                & \multicolumn{2}{l|}{Nb. and name of Surat}\\\hline}
\tablehead{\hline \multicolumn{4}{|l|}%
                  {\small\slshape continued from previous page}\\
          \hline  \multicolumn{2}{|l|}{Nb. and name of Surat}
          &  \multicolumn{2}{l|}{Nb. and name of Surat} \\ \hline}
\tabletail{\hline\multicolumn{4}{|r|}%
                  {\small\slshape continued on next page}\\\hline}
\tablelasttail{\hline}
\begin{supertabular*}{\textwidth}%
    {|cl@{\hspace*{2mm}\extracolsep{\fill}}|c@{\extracolsep{1mm}}l|}
001 & The Opening            & 002 & The Cow                 \\
003 & The Family Of Imran    & 004 & Women                   \\
  ....
113 & The Dawn               & 114 & The Men                 \\
```

Проблемы, возникающие при использовании окружения `supertabular`

- Если на той же странице, где начинается таблица, формируемая при помощи окружения `supertabular`, имеется плавающий объект, результат непредсказуем. В этом случае многостраничная таблица может завершиться после печати первого же из ее фрагментов. Если список необработанных плавающих объектов непуст, он может стать пустым после формирования первого фрагмента многостраничной таблицы (поскольку будет исполнена команда `\clearpage`).
- Окружение `supertabular` нельзя использовать *внутри* плавающих окружений, таких как `table`, поскольку в этом случае TEX будет пытаться разместить многостраничную таблицу на *одной* странице.
- Иногда обработка окружения `supertabular` может завершаться аварийно с выдачей сообщения о переполнении бокса в вертикальном режиме `Overfull \vbox`. Случаи такого рода описаны в документированной версии пакета supertab.

5.4.2 longtable — усложненные многостраничные таблицы

Как уже отмечалось выше, в случае, когда необходимо более гибко управлять шириной таблицы при переходе со страницы на страницу, следует пользоваться пакетом longtable (автор — Дэвид Карлайл). Как и в случае окружения `supertabular`, окружение `longtable` унаследовало часть своих свойств от окружения `tabular`. В частности, оно использует тот же самый счетчик (`table`) и имеет идентичную команду `\caption`. Команда `\listoftables` формирует список таблиц, порождаемых как окружением `table`, так и окружением `longtable`.

Основное отличие между окружениями `supertabular` и `longtable` состоит в том, что второе сохраняет информацию о ширине фрагментов таблицы во вспомогательном файле с расширением `.aux` (`.aux`-файле) с тем, чтобы при следующем прогоне можно было найти наибольшее из значений ширины всех фрагментов многостраничной таблицы[2]. Таким образом, возникает возможность формирования таблицы требуемой ширины. Это свойство окружения `longtable` активизируется командой `\setlongtables`. Отметим, однако, что попытка добавить еще одно окружение `longtable`, не завершив упомянутую выше последовательность, приведет к тому, что все последующие фрагменты таблицы будут иметь неверную ширину, поскольку используется информация, относящаяся к другой таблице (нумерация будет сдвинута на единицу). Возникают также проблемы, если для одной из таблиц будет уменьшена ширина наиболее широкого фрагмента. Поскольку `longtable` сохраняет в `.aux`-файле значение этого параметра, необходимого для формирования таблицы в целом, и не может в дальнейшем воздействовать на это значение, таблица будет получаться слишком широкой. Единственная возможность исправить таблицу в этой ситуации — повторить обработку текста, содержащего `longtable`, без команды `\setlongtable`,

[2] Это значит, что команда `\nofiles`, подавляющая выдачу во все файлы, кроме `.dvi`- и `.log`-файлов, в пакете longtable использована быть не может.

Surats of the Holy Koran

001	The Opening	002	The Cow
003	The Family Of Imran	004	Women
005	The Food	006	The Cattle
007	The Elevated Places	008	The Spoils Of War
009	Repentance	010	Yunus
011	Hud	012	Yusuf
013	The Thunder	014	Ibrahim
015	The Rock	016	The Bee
017	The Israelites	018	The Cave
019	Mariam	020	TaHa
021	The Prophets	022	The Pilgrimage
023	The Believers	024	The Light
025	The Criterion	026	The Poets
027	The Ant	028	The Narrative
029	The Spider	030	The Romans
031	Luqman	032	The Adoration
033	The Allies	034	Saba
035	The Originator	036	Ya Seen
037	The Rangers	038	Suad
039	The Companies	040	The Believer
041	HaMim	042	The Counsel
043	The Embellishment	044	The Evident Smoke
045	The Kneeling	046	The Sandhills
047	Muhammad	048	The Victory
049	The Chambers	050	Qaf
051	The Scatterers	052	The Mountain
053	The Star	054	The Moon
055	The Beneficient	056	The Great Event
057	The Iron	058	The Pleading One

Surats of the Holy Koran

059	The Banishment	060	The Examined One
061	The Ranks	062	Friday
063	The Hypocrites	064	Loss And Gain
065	The Divorce	066	The Prohibition
067	The Kingdom	068	The Pen
069	The Sure Calamity	070	The Ways Of Ascent
071	Nuh	072	The Jinn
073	The Wrapped Up	074	The Clothed One
075	The Resurrection	076	The Man
077	The Emissaries	078	The Great Event
079	Those Who Pull Out	080	He Frowned
081	The Covering Up	082	The Cleaving Asunder
083	The Defrauders	084	The Bursting Asunder
085	The Mansions Of The Stars	086	The Night-Comer
087	The Most High	088	The Overwhelming Calamity
089	The Daybreak	090	The City
091	The Sun	092	The Night
093	The Early Hours	094	The Expansion
095	The Fig	096	The Clot
097	The Majesty	098	The Clear Evidence
099	The Shaking	100	The Assaulters
101	The Terrible Calamity	102	The Multiplication Of Wealth And Children
103	Time	104	The Slanderer
105	The Elephant	106	The Qureaish
107	The Daily Necessaries	108	The Heavenly Fountain
109	The Unbelievers	110	The Help
111	The Flame	112	The Unity
113	The Dawn	114	The Men

Рис. 5.4. Суры Корана (пример применения окружения `longtable`)

чтобы исключить неверные значения из `.aux`-файла, после чего (уже с восстановленной командой `\setlongtables`) запустить LaTeX дважды, чтобы получить корректное значение ширины таблицы. Поскольку в пакете longtable используется LaTeX'овский алгоритм завершения страницы, точная ширина колонок фрагментов таблицы не влияет на разрыв страницы. Следовательно, нет нужды тревожиться по поводу того, что фрагменты таблицы имеют разную ширину, а команду `\setlongtables` следует активизировать только тогда, когда формирование документа завершено и надо получить его окончательный вариант.

Чтобы это различие стало более отчетливым, первый из примеров, относящихся к окружению `supertabular` (рис. 5.2), воспроизведем на этот раз при помощи окружения `longtable` (рис. 5.4). Поскольку в рассматриваемом случае команда `\setlongtables` активизирована, видно, что ширина обоих фрагментов таблицы одинакова (левая и правая части рис. 5.4). Последовательность команд, использованных для формирования рис. 5.4, имеет вид:

```
\setlongtables
\begin{longtable}{|cl|cl|}
\caption*{Surats of the Holy Koran}              \\
\hline
\endhead
\hline
\endfoot
001 & The Opening            &  002 & The Cow    \\
  ....
113 & The Dawn               & 114 & The Men     \\
\end{longtable}
```

Команды, управляющие глобальным форматированием в окружении `longtable`, приведены в табл. 5.2.

Окружение `longtable` никогда не прерывает строку таблицы до ее завершения (это важно для элементов-абзацев со спецификаторами `p`, `m` и `b`). В окружении `longtable` разрешается использовать сноски, а также команду `\newpage`.

В табл. 5.2 дан обзор параметров и команд, управляющих окружением `longtable`. Например, команды `\endhead` и `\endfirsthead` задают текст, появляющийся в верхней части фрагментов таблицы на каждой из страниц и на первой странице соответственно (ср. эти команды с командами `\tablehead` и `\tablefirsthead` для окружения `supertabular`, описанными в разд. 5.4.1), тогда как команды `\endfoot` и `\endlastfoot` определяют то, что появится в нижней части фрагментов таблицы на каждой из страниц и на последней странице соответственно (ср. с командами `\tabletail` и `\tablelasttail` окружения `supertabular`).

Обсудим теперь другие параметры и команды окружения `longtable`.

`\setcounter{LTchunksize}{`*nrows*`}`

Для того чтобы TEX мог формировать многостраничные таблицы, необходимо разбить их на отдельные технологические фрагменты так, чтобы не было необходимости держать в оперативной памяти целиком всю таблицу. По умолчанию в `longtable` используется величина такого фрагмента, равная 20 строкам таблицы. Эта величина может быть изменена при помощи команды, такой как `\setcounter{LTchunksize}{10}`. Разбиение страницы на технологические фрагменты не влияет на то, каким образом будет производиться разрыв страниц. Если доступна оперативная память достаточно большого размера, значение параметра `\LTchunksize` можно задать также большим, что обеспечит более быструю работу TEX'a. Следует подчеркнуть, что величина `\LTchunksize` должна быть не меньше, чем число строк в головке или завершающей части таблицы.

`\setlongtables`

Если поместить команду `\setlongtables` до того, как начнется формирование таблицы, LATEX будет использовать ту информацию о ширине таблицы, которая была сохранена в его предыдущем запуске. Как уже было сказано выше, эту команду следует активизировать только тогда, когда формирование документа завершено и в нем осталось сделать только изменения «косметического» характера.

`\caption[`*short title*`]{`*full title*`}`

Команда `\caption` и ее вариант `\caption*` эквивалентны команде `\multicolumn` вида

`\multicolumn{`*n*`}{p{\LTcapwidth}}{...}`

где *n* — число колонок в таблице. Ширина заголовка таблицы может варьироваться путем переопределения параметра `\LTcapwidth`. Это значит, что в преам-

Параметры (в скобках показаны значения по умолчанию)	
`\LTleft` (`\fill`)	Клей слева от таблицы.
`\LTright` (`\fill`)	Клей справа от таблицы.
`\LTpre` (`\bigskipamount`)	Клей перед таблицей.
`\LTpost` (`\bigskipamount`)	Клей после таблицы.
`\LTchunksize` (20)	Число строк в технологическом фрагменте таблицы (LaTeX'овский счетчик).
`\LTcapwidth` (4in)	Ширина парбокса, содержащего заголовок таблицы.
Настройка окружения `longtable`	
`\setlongtables`	Использовать информацию о ширине колонок из предыдущего прогона.
Необязательные аргументы для `\begin{longtable}`	
none	Позиционирование согласно параметрам `\LTleft` и `\LTright` (обычно центрируется).
`[c]`	Центрирование таблицы.
`[l]`	Выравнивание таблицы по левому краю.
`[r]`	Выравнивание таблицы по правому краю.
Команды, доступные внутри окружения `longtable`	
`\endhead`	Текст, выдаваемый в верхней части фрагментов таблицы на каждой странице.
`\endfirsthead`	Текст, выдаваемый в верхней части фрагмента таблицы на первой странице.
`\endfoot`	Текст, выдаваемый в нижней части фрагментов таблицы на каждой странице.
`\endlastfoot`	Текст, выдаваемый в нижней части фрагмента таблицы на последней странице.
`\kill`	Строка таблицы не набирается (подавляется), но принимается во внимание при вычислении ширины колонок.
`\caption{`*foo*`}`	Заголовок таблицы «Table *xx*: foo» с включением «foo» в список таблиц.
`\caption[`*bar*`]{`*foo*`}`	Заголовок таблицы «Table *xx*: foo» с включением «bar» в список таблиц.
`\caption[]{`*foo*`}`	Заголовок таблицы «Table *xx*: foo» без записи чего-либо в список таблиц.
`\caption*{`*foo*`}`	Заголовок таблицы «foo» без записи чего-либо в список таблиц.
`\newpage`	Вызывает разрыв страницы.

Таблица 5.2. Перечень команд окружения `longtable`

буле документа можно записать команду \setlength{\LTcapwidth}{*width*}. Значение этого параметра по умолчанию равно 4 дюйма. Как и в случае с командой \caption в окружениях figure и table необязательный аргумент задает текст, предназначенный для помещения в список таблиц в том случае, если этот текст отличается от заголовка таблицы.

Если заголовок таблицы на первой странице должен отличаться от ее заголовков на последующих страницах, следует поместить команду \caption с полным текстом заголовка в первый заголовок, а также ввести вспомогательный текст (при помощи команды \caption[ ]) в главный заголовок, поскольку (в данном случае) записей в список таблиц не производится. В качестве альтернативного варианта, если номер таблицы не надо повторно воспроизводить в заголовках ее фрагментов, можно использовать команду \caption*. Как и в случае с окружением table, для обеспечения возможности ссылаться на таблицу можно воспользоваться командой \label.

По умолчанию окружение longtable центрирует таблицы (например, \LTleft и \LTright оба принимают значения \fill). Этим параметрам может быть задано любое значение (длина), но хотя бы один из них должен иметь значение \fill (т. е. быть растяжимым промежутком) с тем, чтобы была возможность довести ширину таблицы до ширины страницы, в противном случае растяжимый промежуток будет добавлен между колонками посредством команды \extracolsep. Например, таблица может быть выровнена по левому краю посредством определения

```
\setlength{\LTleft}{0pt}
\setlength{\LTright}{\fill}
```

или, что то же самое, посредством \begin{longtable}[l].

Можно также использовать параметры \LTleft и \LTright для верстки многостраничной таблицы, занимающей всю ширину страницы; например, для того чтобы получить результат, эквивалентный тому, что дает команда

```
\begin{supertabular*}{\textwidth}{l@{\extracolsep{\fill}}l}
```

в примере, обсуждавшемся на с. 141 для рис. 5.3, для окружения longtable следует записать

```
\setlength{\LTleft}{0pt}
\setlength{\LTright}{0pt}
\begin{longtable}{l@{\extracolsep{\fill}}l}
```

В общем случае, если параметры \LTleft и \LTright имеют фиксированное значение, таблица будет иметь следующую ширину:

```
\textwidth - \LTleft - LTright
```

5.4.3 Завершающее сравнение окружений supertabular и longtable

Рисунок 5.5 показывает результат, полученный при верстке многостраничной таблицы (на трех страницах; границы этих страниц схематически показаны тонкими линиями). Из этого примера наглядно видна разница в результате, получаемом при помощи окружений `supertabular` и `longtable`. Используя одни и те же входные данные, окружение `supertabular` дает на каждой из страниц фрагменты таблицы разной ширины, тогда как окружение `longtable` формирует все эти фрагменты равными по ширине, исходя из ширины наиболее широкого фрагмента. Еще одно отличие заключается в том, что страницы, сформированные окружением `supertabular`, короче «стандартных» страниц документа. Это является следствием того, что `supertabular` использует собственный алгоритм разрыва страниц. Окружение `longtable` использует стандартный LaTeX'овский алгоритм разрыва (завершения) страниц, поэтому уменьшения длины страницы не происходит. Пользователю необходимо решить, оправдано ли некоторое усложнение работы, связанное с использованием окружения `longtable`, применительно к отображаемым данным. Из табл. 5.3 видно, что определения самих таблиц для обоих случаев выглядят совершенно одинаково.

5.5 Дополнительные штрихи

В данном разделе описываются два пакета, прекрасно работающих совместно с окружениями, описанными выше. Первый из них облегчает выравнивание десятичных чисел в пределах колонок таблицы, тогда как второй предоставляет средства для разнообразного комбинирования отрезков вертикальных и горизонтальных линий внутри окружения `tabular` и подобных ему.

5.5.1 dcolumn — управление выравниванием в колонках таблицы

Пакет dcolumn (автор — Дэвид Карлайл) содержит набор средств определения колонок из элементов таблицы в окружениях `array` или `tabular`, которые могут быть выровнены по «десятичной точке». Элементы, не содержащие дробной части, целой части, а также пустые элементы также обрабатываются корректно.

Пакет dcolumn определяет спецификатор `D` (от Decimal — десятичный) тремя аргументами

D{*inputsep*}{*outputsep*}{*decimal places*}

inputsep Единичный символ, используемый в качестве разделителя в исходном `.tex`-файле (например, «`.`» или «`,`»).

Table 9: Codes of the languages of the world (supertabular)

Languages codes (ISO 639:1988)

code	language	code	language	code	language
aa	Afar	ab	Abkhazian	af	Afrikaans
am	Amharic	ar	Arabic	as	Assamese
ay	Aymara	az	Azerbaijani		
ba	Bashkir	be	Byelorussian	bg	Bulgarian
bh	Bihari	bi	Bislama	bn	Bengali; Bangla
bo	Tibetan	br	Breton		
ca	Catalan	co	Corsican	cs	Czech
cy	Welch				
da	Danish	de	German	dz	Bhutani
el	Greek	en	English	eo	Esperanto
es	Spanish	et	Estonian	eu	Basque
fa	Persian	fi	Finnish	fj	Fiji
fo	Faeroese	fr	French	fy	Frisian
ga	Irish	gd	Scots Gaelic	gl	Galician
gn	Guarani	gu	Gujarati		
ha	Hausa	hi	Hindi	hr	Croatian
hu	Hungarian	hy	Armenian		

continued on next page

continued from previous page

code	language	code	language	code	language
ia	Interlingua	ie	Interlingue	ik	Inupiak
in	Indonesian	is	Icelandic	it	Italian
iw	Hebrew				
ja	Japanese	ji	Yiddish	jw	Javanese
ka	Georgian	kk	Kazakh	kl	Greenlandic
km	Cambodian	kn	Kannada	ko	Korean
ks	Kashmiri	ku	Kurdish	ky	Kirghiz
la	Latin	ln	Lingala	lo	Laothian
lt	Lithuanian	lv	Latvian, Lettish		
mg	Malagasy	mi	Maori	mk	Macedonian
ml	Malayalam	mn	Mongolian	mo	Moldavian
mr	Marathi	ms	Malay	mt	Maltese
my	Burmese				
na	Nauru	ne	Nepali	nl	Dutch
no	Norwegian				
oc	Occitan	om	(Afan) Oromo	or	Oriya
pa	Punjabi	pl	Polish	ps	Pashto, Pushto
pt	Portuguese				
qu	Quechua				

continued on next page

continued from previous page

code	language	code	language	code	language
rm	Rhaeto-Romance	rn	Kirundi	ro	Romanian
ru	Russian	rw	Kinyarwanda		
sa	Sanskrit	sd	Sindhi	sg	Sangro
sh	Serbo-Croatian	si	Singhalese	sk	Slovak
sl	Slovenian	sm	Samoan	sn	Shona
so	Somali	sq	Albanian	sr	Serbian
ss	Siswati	st	Sesotho	su	Sudanese
sv	Swedish	sw	Swahili		
ta	Tamil	te	Tegulu	tg	Tajik
th	Thai	ti	Tigrinya	tk	Turkmen
tl	Tagalog	tn	Setswana	to	Tonga
tr	Turkish	ts	Tsonga	tt	Tatar
tw	Twi				
uk	Ukrainian	ur	Urdu	uz	Uzbek
vi	Vietnamese	vo	Volapuk		
wo	Wolof				
xh	Xhosa				
yo	Yoruba				
zh	Chinese	zu	Zulu		

Table 10: Codes of the languages of the world (longtable)

Language codes (ISO 639:1988)

code	language	code	language	code	language
aa	Afar	ab	Abkhazian	af	Afrikaans
am	Amharic	ar	Arabic	as	Assamese
ay	Aymara	az	Azerbaijani		
ba	Bashkir	be	Byelorussian	bg	Bulgarian
bh	Bihari	bi	Bislama	bn	Bengali; Bangla
bo	Tibetan	br	Breton		
ca	Catalan	co	Corsican	cs	Czech
cy	Welch				
da	Danish	de	German	dz	Bhutani
el	Greek	en	English	eo	Esperanto
es	Spanish	et	Estonian	eu	Basque
fa	Persian	fi	Finnish	fj	Fiji
fo	Faeroese	fr	French	fy	Frisian
ga	Irish	gd	Scots Gaelic	gl	Galician
gn	Guarani	gu	Gujarati		
ha	Hausa	hi	Hindi	hr	Croatian
hu	Hungarian	hy	Armenian		
ia	Interlingua	ie	Interlingue	ik	Inupiak
in	Indonesian	is	Icelandic	it	Italian

continued on next page

continued from previous page

code	language	code	language	code	language
iw	Hebrew				
ja	Japanese	ji	Yiddish	jw	Javanese
ka	Georgian	kk	Kazakh	kl	Greenlandic
km	Cambodian	kn	Kannada	ko	Korean
ks	Kashmiri	ku	Kurdish	ky	Kirghiz
la	Latin	ln	Lingala	lo	Laothian
lt	Lithuanian	lv	Latvian, Lettish		
mg	Malagasy	mi	Maori	mk	Macedonian
ml	Malayalam	mn	Mongolian	mo	Moldavian
mr	Marathi	ms	Malay	mt	Maltese
my	Burmese				
na	Nauru	ne	Nepali	nl	Dutch
no	Norwegian				
oc	Occitan	om	(Afan) Oromo	or	Oriya
pa	Punjabi	pl	Polish	ps	Pashto, Pushto
pt	Portuguese				
qu	Quechua				
rm	Rhaeto-Romance	rn	Kirundi	ro	Romanian
ru	Russian	rw	Kinyarwanda		
sa	Sanskrit	sd	Sindhi	sg	Sangro
sh	Serbo-Croatian	si	Singhalese	sk	Slovak

continued on next page

continued from previous page

code	language	code	language	code	language
sl	Slovenian	sm	Samoan	sn	Shona
so	Somali	sq	Albanian	sr	Serbian
ss	Siswati	st	Sesotho	su	Sudanese
sv	Swedish	sw	Swahili		
ta	Tamil	te	Tegulu	tg	Tajik
th	Thai	ti	Tigrinya	tk	Turkmen
tl	Tagalog	tn	Setswana	to	Tonga
tr	Turkish	ts	Tsonga	tt	Tatar
tw	Twi				
uk	Ukrainian	ur	Urdu	uz	Uzbek
vi	Vietnamese	vo	Volapuk		
wo	Wolof				
xh	Xhosa				
yo	Yoruba				
zh	Chinese	zu	Zulu		

Рис. 5.5. Сравнение результатов применения окружений `longtable` и `supertabular`

Определение в случае использования окружения supertabular

```
\tablefirsthead{\hline
            \multicolumn{6}{|c|}{Languages codes (ISO 639:1988)}  \\
            code & language & code & language & code & language   \\\hline}
\tablehead{\hline \multicolumn{6}{|l|}{\small\slshape
                                  continued from previous page}   \\\hline
            code & language & code & language & code & language   \\\hline}
\tabletail{\hline\multicolumn{6}{|r|}{\small\slshape continued on next page}
                                                                   \\\hline}
\tablelasttail{\hline}
\topcaption{Codes of the languages of the world (supertabular)}
\begin{center}
\begin{supertabular}{|*{3}{c|}|}
\ttfamily aa &  Afar     &  \ttfamily ab &  Abkhazian    &  .....
.....
\ttfamily zh &  Chinese  &  \ttfamily zu &  Zulu         &    &    \\
\end{supertabular}
\end{center}
```

Определение в случае использования окружения longtable

```
\setlongtables
\begin{longtable}[c]{|*{3}{c|}|}
\caption{Codes of the languages of the world (longtable)}
\\ \hline
\multicolumn{6}{|c|}{Language codes (ISO 639:1988)}                \\
         code & language & code & language & code & language      \\ \hline
\endfirsthead
\hline
\multicolumn{6}{|l|}{\small\slshape continued from previous page} \\ \hline
         code & language & code & language & code & language      \\ \hline
\endhead
\hline\multicolumn{6}{|r|}{\small\sl continued on next page}       \\ \hline
\endfoot
\hline
\endlastfoot
\ttfamily aa &  Afar     &  \ttfamily ab &  Abkhazian    &  .....
.....
\ttfamily zh &  Chinese  &  \ttfamily zu &  Zulu         &    &  \\
\end{longtable}
```

Таблица 5.3. Сравнение определений таблиц для окружений longtable и supertabular

outputsep Разделитель, который следует использовать в сформированном выходе. Он может совпадать со значением первого аргумента, но может быть и любым выражением, допускаемым в математическом режиме, например, `\cdot`.

decimal places Максимальное число десятичных разрядов в колонке. Если этот параметр имеет отрицательное значение, в колонке допускается любое число десятичных знаков и все элементы таблицы должны выравниваться по символу-разделителю. Это может привести к тому, что колонка получится слишком широкой (см. две первые колонки в примере, приводимом ниже).

Если вы не хотите использовать все эти три элемента в преамбуле, можно модифицировать спецификаторы преамбулы при помощи команды `\newcolumntype` так, как это показано ниже.

```
\newcolumntype{d}[1]{D{.}{\cdot}{#1}}
```

Заново определенный спецификатор `d` дает единственный аргумент, показывающий требуемое число десятичных разрядов. Десятичная точка во входном `.tex`-файле — обычная точка «.», тогда как в выходном — точка «·» из математического режима.

```
\newcolumntype{.}{D{.}{.}{-1}}
```

В этом случае спецификатор «.» не имеет аргументов; обычная точка используется как во входном, так и в выходном файле, а элементы таблицы при ее верстке будут центрироваться (выравниваться) по этой точке.

```
\newcolumntype{,}{D{,}{,}{2}}
```

Спецификатор «,», определенный этим выражением, использует в качестве десятичного разделителя запятую «,» как во входном, так и в выходном тексте, а верстаемые колонки должны иметь не более двух десятичных разрядов после запятой.

Эти определения использованы в примере ниже, из которого видно, что первая колонка с отрицательным значением параметра, определяющего число десятичных разрядов (это значение предопределяет расположение десятичной точки-разделителя в центре колонки), шире второй колонки, хотя они отображают одни и те же числа.

1·2	1·2	1.2	1,2
1·23	1·23	12.5	300,2
1121·2	1121·2	861.20	674,29
184	184	10	69
·4	·4		,4
		.4	

```
\begin{tabular}{|d{-1}|d{2}|.|,|}
   1.2    & 1.2    &1.2      &1,2     \\
   1.23   & 1.23   &12.5     &300,2   \\
   1121.2& 1121.2&861.20    &674,29  \\
   184    & 184    &10       &69      \\
   .4     & .4     &         &,4      \\
          &        &.4       &
\end{tabular}
```

Ниже приводится несколько измененный пример из книги LaTeX book [Л 182].

GG&A Hoofed Stock			
Year	Price low–high	Comments	Other
1971	97–245	Bad year for farmers in the West.	23,45
72	245–245	Light trading due to a heavy winter.	435,23
73	245–2001	No gnus was very good gnus this year.	387,56

```
\newcolumntype{+}{D{/}{\mbox{--}}{4}}
% Разделителями служат / и ,
\begin{tabular}{|r||+|p{2cm}|,|}   \hline
\multicolumn{4}{|c|}{GG\&A Hoofed Stock}
  \\ \hline\hline
& \multicolumn{1}{c|}{Price}& &
\\ \cline{2-2} \multicolumn{1}{|c||}{Year}
& \mbox{low}/\mbox{high}
& \multicolumn{1}{c|}{Comments}
& \multicolumn{1}{c|}{Other}    \\ \hline
1971 & 97/245  &Bad year for farmers in
        the West. & 23,45       \\ \hline
  72 &245/245  &Light trading due to a
        heavy winter.   & 435,23\\ \hline
  73 &245/2001 &No gnus was very good
        gnus this year. & 387,56\\ \hline
\end{tabular}
```

5.5.2 hhline — комбинирование горизонтальных и вертикальных отрезков

В пакете hhline (автор — Дэвид Карлайл) вводится команда \hhline, подобная \hline во всем, за исключением способа взаимодействия с вертикальными линиями.

\hhline{*decl*}

Декларация *decl* представляет собой список лексем (токенов) со следующими значениями:

= Двойная горизонтальная линия (\hline) в ширину колонки.

\- Одинарная горизонтальная линия (\hline) в ширину колонки.

~ Колонка без горизонтальной линии \hline, т. е. пробел, шириной в колонку.

| Вертикальная линия (\vline), «врезанная» в двойную (или одинарную) горизонтальную линию (\hline).

: Вертикальная линия (\vline), пересекаемая двойной горизонтальной линией (\hline).

Сегмент двойной горизонтальной линии (\hline) между двумя вертикальными линиями (\vline).

t Верхняя линейка в сегменте двойной горизонтальной линии (\hline).

b Нижняя линейка в сегменте двойной горизонтальной линии (\hline).

* *{3}{==#} раскрывается в ==#==#==#, как в варианте со звездочкой для преамбулы.

Если при помощи спецификации || или :: задана двойная вертикальная линия (\vline), горизонтальные линии (\hline), порождаемые командой \hhline,

будут нарушены. Чтобы получить эффект «врезания» горизонтальной линии `\hline` в сдвоенную `\vline`, следует пользоваться спецификатором `#`.

Лексемы-спецификаторы `t` и `b` можно использовать между двумя вертикальными линейками. Например, декларация `|tb|` дает тот же результат, что и `#`, но гораздо менее эффективно. Основное назначение этих спецификаторов состоит в том, чтобы строить конструкции типа `|t:` (верхний левый угол) и `:b|` (нижний правый угол).

Если команда `\hhline` используется для формирования только одинарных горизонтальных линий, то аргумент ее должен содержать только лексемы «`-`», «`~`» и «`|`» (а также `*`-выражения).

Вот пример, в котором использованы многие из описанных средств:

a	b	c	d
1	2	3	4
i	j	k	l
w	x	y	z

```
\setlength{\arrayrulewidth}{.8pt}
\begin{tabular}{||cc||c|c||}
                        \hhline{|t:==:t:==:t|}
a & b & c & d \\ \hhline{|:==:|~|~||}
1 & 2 & 3 & 4 \\ \hhline{#==#~|=#}
i & j & k & l \\ \hhline{||--||--||}
w & x & y & z \\ \hhline{|b:==:b:==:b|}
\end{tabular}
```

Линии, порождаемые командой `\hline`, состоят из единственной линейки `\hrule` (один из видов базисных средств-примитивов TEX'a). Линии же, которые формируются командой `\hhline`, строятся из некоторого числа небольших сегментов. TEX все эти сегменты точно отображает в `.dvi`-файле, однако `.dvi`-драйвер, используемый для отображения получаемого результата на экране монитора или на принтере, может оказаться не в состоянии строить эти сегменты с достаточной точностью. При возникновении такой ситуации можно попробовать увеличить значение параметра `\arrayrulewidth`, чтобы повысить качество получаемого результата.

5.6 Приложения

Ниже излагается ряд примеров усложненного вида, включая верстку вложенных таблиц. В них будет продемонстрировано также применение многих описанных в данной главе средств.

5.6.1 Переносы в узких колонках

TEX не делает переноса в первом слове абзаца, в связи с чем слишком узкие колонки в таблицах могут вызывать переполнение. Эту ситуацию можно исправить, если начать текст, записываемый в графы таблицы, с команды `\hspace{0pt}`.

Characteristics

Char-acteris-tics

```
\fbox{\parbox{11mm}{Characteristics}}\hfill
\fbox{\parbox{11mm}%
                {\hspace{0pt}Characteristics}}
```

5.6.2 Сноски в таблицах

Как уже было показано в разд. 3.4.1, стандартным LaTeX'ом сноски внутри таблицы не допускаются. Только в окружениях `tabularx` и `longtable`, рассматривавшихся выше, такого рода сноски обрабатываются корректно. Поскольку требуется, чтобы «табличные сноски» появлялись сразу же после соответствующей таблицы, в случае других окружений вам придется самостоятельно взяться за дело, управляя значками сносок и, например, используя команды `\multicolumn` в нижней части окружения `tabular`, чтобы разместить «табличные» сноски.

Можно также поместить окружение `tabular` или `array` внутри окружения `minipage`, поскольку в данном случае сноска будет набрана сразу же после окружения `minipage`. Отметим переопределение `\thefootnote`, которое позволяет использовать команду `\footnotemark` внутри окружения `minipage`. Без такого переопределения `\footnotemark` будет порождать сноску в стиле сносок для основной страницы, как это уже объяснялось в разд. 3.4.1.

PostScript type 1 fonts

Courier[a]	cour, courb, courbi, couri
Charter[b]	bchb, bchbi, bchr, bchri
Nimbus[c]	unmr, unmrs
URW Antiqua[c]	uaqrrc
URW Grotesk[c]	ugqp
Utopia[d]	putb, putbi, putr, putri

[a] Donated by IBM.
[b] Donated by Bitstream.
[c] Donated by URW GmbH.
[d] Donated by Adobe.

```
\begin{minipage}{\linewidth}
\renewcommand{\thefootnote}
                    {\thempfootnote}
\begin{tabular}{ll}
\multicolumn{2}{c}{\bfseries PostScript
                    type 1 fonts}    \\
Courier\footnote{Donated by IBM.}
      & cour, courb, courbi, couri  \\
Charter\footnote{Donated by Bitstream.}
      & bchb, bchbi, bchr, bchri    \\
Nimbus\footnote{Donated by URW GmbH.}
      & unmr, unmrs                 \\
URW Antiqua\footnotemark
                    [\value{mpfootnote}]
      & uaqrrc                      \\
URW Grotesk\footnotemark
                    [\value{mpfootnote}]
      & ugqp                        \\
Utopia\footnote{Donated by Adobe.}
      & putb, putbi, putr, putri
\end{tabular}
\end{minipage}
```

Разумеется, такой подход не обеспечивает автоматического совпадения ширины сносок с шириной таблицы, так что может потребоваться вручную обеспечить такое совпадение путем итерационной подстройки аргумента, задающего ширину `minipage`.

Другой подход к верстке сносок к таблицам предложен в пакете threeparttable (автор — Дональд Арсено). Сильная сторона данного пакета состоит в том, что при его использовании четко и недвусмысленно указывается то, что вы имеете дело со сносками внутри таблицы; более того, дается возможность исчерпывающим образом управлять появлением указаний на сноски в таблице, давать наименова-

ние табличному материалу. В этом смысле окружение threeparttable подобно неплавающему варианту окружения table, описываемому в разд. 6.3.1.

Таблица 5.4. PostScript type 1 fonts

Courier[a]	cour, courb, courbi, couri
Charter[b]	bchb, bchbi, bchr, bchri
Nimbus[c]	unmr, unmrs
URW Antiqua[c]	uaqrrc
URW Grotesk[c]	ugqp
Utopia[d]	putb, putbi, putr, putri

[a]Donated by IBM.

[b]Donated by Bitstream.

[c]Donated by URW GmbH.

[d]Donated by Adobe.

```
\begin{threeparttable}
\caption[Example of a
   \nxLenv{threeparttable}
   environment]{\textbf{PostScript
   type 1 fonts}}
\begin{tabular}{@{}ll@{}}
Courier\tnote{a}
  & cour, courb, courbi, couri    \\
Charter\tnote{b}
  & bchb, bchbi, bchr, bchri      \\
Nimbus\tnote{c}  & unmr, unmrs    \\
URW Antiqua\tnote{c} & uaqrrc     \\
URW Grotesk\tnote{c} & ugqp       \\
Utopia\tnote{d}
  & putb, putbi, putr, putri      \\
\end{tabular}
\begin{tablenotes}
\item[a] Donated by IBM.
\item[b] Donated by Bitstream.
\item[c] Donated by URW GmbH.
\item[d] Donated by Adobe.
\end{tablenotes}
\end{threeparttable}
```

5.6.3 Таблицы с широкими графами

Иногда промежутки между узкими колонками в широкой таблице требуется распределить равномерно. Например, в показанной ниже таблице имеется довольно широкая первая строка, под которой располагаются несколько узких колонок.

это-очень-широкая-строка		
C1	C2	C3
2.1	2.2	2.3
3.1	3.2	3.3

```
\begin{tabular}{ccc}
\multicolumn{3}{c}{это-очень-широкая-строка}\\
C1 &C2 &C3 \\ 2.1&2.2&2.3 \\ 3.1&3.2&3.3
\end{tabular}
```

В этом случае можно задать растяжимый промежуток перед каждой из колонок, используя команду \extracolsep. Фактическая величина этого промежутка не существенна, поскольку он может быть изменен так, чтобы заполнить требуемое пространство. В этом случае следует задать, разумеется, общую ширину таблицы.

это-очень-широкая-строка		
C1	C2	C3
2.1	2.2	2.3
3.1	3.2	3.3

```
\begin{tabular*}{\linewidth}%
        {!{\extracolsep{4in minus 4in}}ccc}
\multicolumn{3}{c}{это-очень-широкая-строка}\\
C1 &C2 &C3 \\ 2.1&2.2&2.3 \\ 3.1&3.2&3.3
\end{tabular*}
```

Предыдущий пример не слишком удачен, так как элементы таблицы оказались раздвинуты слишком широко. Можно, однако, заранее вычислить ширину «большого» элемента, определяющего общую ширину таблицы, и использовать ее в окружении `tabular*`.

это-очень-широкая-строка		
C1	C2	C3
2.1	2.2	2.3
3.1	3.2	3.3

```
\newlength{\Mylen}
\settowidth{\Mylen}{это-очень-широкая-строка}
\begin{tabular*}{\Mylen}%
        {!{\extracolsep{4in minus 4in}}ccc}
\multicolumn{3}{c}{это-очень-широкая-строка}\\
C1 &C2 &C3 \\ 2.1&2.2&2.3 \\ 3.1&3.2&3.3
\end{tabular*}
```

Полученный результат выглядит уже лучше, но чтобы более корректно выполнить выравнивание колонок таблицы, необходимо учесть ширину межколонных разделителей (`\tabcolsep`) по обе стороны каждого элемента таблицы. Можно еще улучшить данный вариант адекватным подбором выравнивания колонок и промежутков между колонками в таблице.

это-очень-широкая-строка		
C1	C2	C3
2.1	2.2	2.3
3.1	3.2	3.3

```
\settowidth{\Mylen}{это-очень-широкая-строка}
\addtolength{\Mylen}{2\tabcolsep}
\begin{tabular*}{\Mylen}%
        {!{\extracolsep{4in minus 4in}}ccc}
\multicolumn{3}{c}{это-очень-широкая-строка}\\
C1 &C2 &C3 \\ 2.1&2.2&2.3 \\ 3.1&3.2&3.3
\end{tabular*}
```

В альтернативном варианте можно подавить промежутки между колонками слева и справа от окружения `tabular*`, используя выражения `@{}` .

это-очень-широкая-строка		
C1	C2	C3
2.1	2.2	2.3
3.1	3.2	3.3

```
\settowidth{\Mylen}{это-очень-широкая-строка}
\begin{tabular*}{\Mylen}%
        {@{\extracolsep{4in minus 4in}}ccc@{}}
\multicolumn{3}{@{}c@{}}%
                {это-очень-широкая-строка}\\
C1 &C2 &C3 \\ 2.1&2.2&2.3 \\ 3.1&3.2&3.3
\end{tabular*}
```

5.6.4 Колонки, занимающие несколько строк таблицы

Имеется возможность формировать графы таблицы, занимающие по вертикали несколько ее строк, если поместить материал в бокс нулевой высоты и воспользоваться командой `\raisebox`.

100	qqq	
	A	B
20000000	10	10

```
\begin{tabular}{|c|c|c|}              \hline
        & \multicolumn{2}{c|}{qqq}\\\cline{2-3}
\raisebox{1.5ex}[0cm][0cm]{100}
        & A         & B          \\\hline
20000000 & 10       & 10         \\\hline
\end{tabular}
```

multirow — выравнивание в таблицах по вертикали

Пакет multirow (автор — Джерри Лейхтер) автоматизирует процедуру построения граф колонок, простирающихся на несколько строк таблицы, путем определения команды `\multirow`. Имеется несколько необязательных аргументов, позволяющих проводить точную подстройку параметров (внешнего вида) таблицы. Это может оказаться полезным, если некоторые из пересекаемых строк таблицы необычно велики, если распорки (`\strut`) используются асимметрично относительно средней линии пересекаемых строк или если не приняты во внимание выносные элементы букв. В этих случаях вертикальное центрирование не будет выполняться правильно, но можно воспользоваться подстроечным параметром *vmove*, чтобы ввести требуемые смещения по вертикали вручную.

`\multirow{`*nrow*`}[`*njot*`]{`*width*`}[`*vmove*`]{`*contents*`}`

Внутри окружения `array` эта команда несколько менее полезна, поскольку линии имеют лишнюю длину, определяемую величиной параметра `\jot` (значение его по умолчанию равно `3pt`), не учитываемую командой `\multirow`. Устранить это в общем случае практически невозможно, но можно данную ситуацию избежать. Надо установить параметр длины `\bigstrutjot` равным `\jot` и использовать затем второй аргумент *njot* команды `\multirow` со значением, равным половине числа пересекаемых строк таблицы.

Имеется ряд средств, позволяющих управлять форматированием внутри граф таблицы. Непосредственно перед выполнением операции размещения набираемого текста в графе таблицы выполняется макрокоманда `\multirowsetup`, чтобы установить некоторое специальное окружение. Первоначально команда `\multirowsetup` содержит только `\raggedright`, однако ее можно переопределить при помощи `\renewcommand`.

Область действия команды `\multirow` может распространяться на одну или несколько колонок, как это видно из примера.

Текст в колонке 1	C2a C2b C2c C2d	Текст в колонке 3	C4a C4b C4c C4d

```
\begin{tabular}{|l|l|l|l|}   \hline
\multirow{4}{14mm}{Текст в колонке 1}
    & C2a & \multirow{4}{14mm}%
            {Текст в колонке 3}
                & C4a       \\
    & C2b &     & C4b       \\
    & C2c &     & C4c       \\
    & C2d &     & C4d       \\\hline
\end{tabular}
```

Теперь можно повторить решение небольшого примера, рассматривавшегося в начале этого раздела, не прибегая к команде `\raisebox`. Во-первых, необходимо изменить выравнивание внутри абзаца `\multirow`, чтобы выполнить центрирова-

ние (\centering) и затем вычислить ширину текста в колонке, которая требуется команде \multirow. Если графа колонки, охватывающая несколько строк таблицы, имеет фиксированную ширину, как это было в других рассматривавшихся примерах, этот шаг перестает быть необходимым.

100	qqq	
	A	B
20000000	10	10

```
\renewcommand{\multirowsetup}{\centering}
\newlength{\LL} \settowidth{\LL}{100}
\begin{tabular}{|c|c|c|}
      \hline
\multirow{2}{\LL}{100}&
      \multicolumn{2}{c|}{qqq}  \\\cline{2-3}
                & A         & B  \\\hline
20000000        & 10        & 10 \\\hline
\end{tabular}
```

Влияние необязательного параметра вертикального позиционирования *vmove* иллюстрируется примером ниже. Обратите внимание на перемещение вверх на 3 мм нижней трети таблицы.

Общий текст в колонке 1	Колонка g2a
	Колонка g2b
	Колонка g2c
	Колонка g2d
Общий текст в колонке 1	Колонка g2a
	Колонка g2b
	Колонка g2c
	Колонка g2d
Общий текст в колонке 1	Колонка g2a
	Колонка g2b
	Колонка g2c
	Колонка g2d

```
\begin{tabular}{|l|l|}
\hline
\multirow{4}{25mm}{Общий текст в колонке 1}
& Колонка g2a \\\cline{2-2} & Колонка g2b\\
                                    \cline{2-2}
& Колонка g2c \\\cline{2-2} & Колонка g2d\\\hline
\multirow{4}{25mm}[-3mm]{Общий текст в колонке 1}
& Колонка g2a \\\cline{2-2} & Колонка g2b\\
                                    \cline{2-2}
& Колонка g2c \\\cline{2-2} & Колонка g2d\\\hline
\multirow{4}{25mm}[3mm]{Общий текст в колонке 1}
& Колонка g2a \\\cline{2-2} & Колонка g2b\\
                                    \cline{2-2}
& Колонка g2c \\\cline{2-2} & Колонка g2d\\\hline
\end{tabular}
```

5.6.5 Таблицы внутри таблиц

Ниже приведен пример, из которого видно, что достаточно небольших дополнительных усилий, чтобы сформировать в LaTeX'е таблицы сложной структуры.

Семейство окружений tabular обеспечивает возможность позиционирования по вертикали по отношению к базовой линии текста, в котором появилось данное окружение. По умолчанию данное окружение при размещении его в тексте центрируется, однако этот вариант может быть изменен на выравнивание по первой или последней строке окружения, если необязательному параметру позиционирования придать значение t или b соответственно. Однако это средство не будет работать, если первый или последний элемент в окружении представляет собой

команду \hline. В этом случае окружение будет выравниваться по формируемой ею горизонтальной линии.

Tables with no hline commands used versus tables

with some hline commands

used.

```
Tables
\begin{tabular}[t]{l}
  with no\\ hline \\
           commands \\ used
\end{tabular} versus tables
\begin{tabular}[t]{|l|}
 \hline
  with some \\ hline \\
                commands \\
 \hline
\end{tabular} used.
```

Чтобы добиться требуемого выравнивания в таком случае, можно использовать команды (\firsthline и \lasthline), определенные в последних версиях пакета array и являющиеся специальными вариантами команды \hline. Эти команды обеспечивают соответствующее выравнивание, если только первая или последняя строка таблицы не содержит слишком больших граф.

Tables with no line commands used versus tables

with some line commands

used.

```
Tables
\begin{tabular}[t]{l}
  with no\\ line \\ commands
                      \\ used
\end{tabular} versus tables
\begin{tabular}[t]{|l|}
 \firsthline
  with some \\ line  \\
                commands \\
 \lasthline
\end{tabular} used.
```

Реализация этих двух команд содержит параметр, именуемый \extratabsurround, позволяющий добавить некоторый дополнительный промежуток в верхней и нижней частях такого окружения. Данный параметр полезен для выравнивания соответствующим образом вложенных таблиц, как это видно из примера, приводимого на рис. 5.6.

Этот пример был получен с использованием следующего входного текста:

```
\setlength{\extratabsurround}{2pt}

\begin{tabular}{|cc|}                                          \hline
\textit{name} & \textit{telephone}                        \\\hline\hline
    John &
     \begin{tabular}[t]{|cc|}                                 \firsthline
     \textit{day} & \multicolumn{1}{c|}{\textit{telephone}}
                                                          \\\hline\hline
     Wed & 5554434                                            \\\hline
     Mon &
        \begin{tabular}[t]{|cc|}                              \firsthline
         \textit{time} & \textit{telephone}                 \\\hline\hline
```

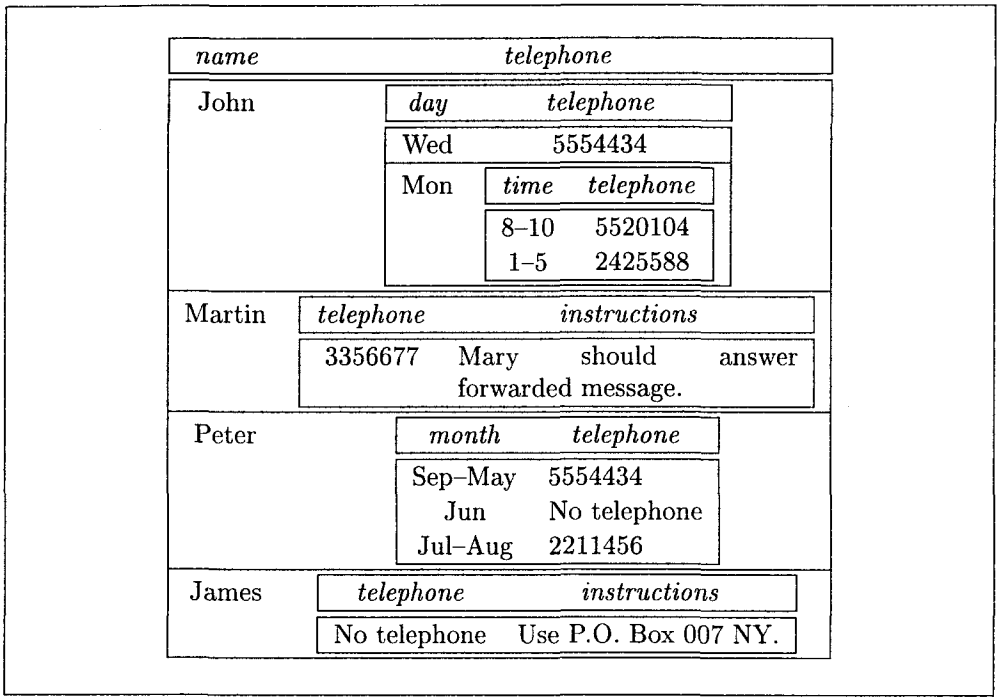


Рис. 5.6. Пример вложенных таблиц

```
        8--10 & 5520104 \\ 1--5 & 2425588                    \\\lasthline
       \end{tabular}                                         \\\lasthline
   \end{tabular}                                                \\\hline
Martin   &
   \begin{tabular}[t]{|cp{4.5cm}|}                            \firsthline
    \textit{telephone} & \multicolumn{1}{c|}{\textit{instructions}}
                                                             \\\hline\hline
    3356677 & Mary should answer forwarded message.          \\\lasthline
   \end{tabular}                                                \\\hline
 Peter   &
   \begin{tabular}[t]{|cl|}                                   \firsthline
    \textit{month} &\multicolumn{1}{c|}{\itshape telephone}
                                                             \\\hline\hline
    Sep--May & 5554434  \\  Jun &  No telephone  \\
    Jul--Aug & 2211456  \\                                     \lasthline
   \end{tabular}                                               \\\hline
 James   &
    \begin{tabular}[t]{|cl|}                                 \firsthline
    \textit{telephone} & \multicolumn{1}{c|}{\itshape instructions}
                                                             \\\hline\hline
    No telephone &  Use P.O. Box 007 NY.                     \\\lasthline
   \end{tabular}                                             \\\lasthline
\end{tabular}
```

5.6.6 Еще два примера

LaTeX'овский текст, приводимый ниже, показывает, как можно комбинировать различные методы и пакеты, описанные в данной главе. Будем использовать пакет tabularx для формирования таблицы из 12 колонок, причем колонки с 3 по 12 будут иметь одинаковую ширину. Для формирования структурированной торцевой колонки с именами строк таблицы используем пакет multirow. Эта колонка (prefix) охватывает две строки в колонке 1. Для того чтобы обеспечить соответствующее размещение такой колонки, необходимо вычислять заранее ее ширину.

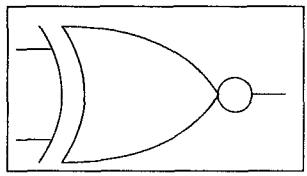
```
\setlength{\tabcolsep}{1mm}
\newlength{\Tl}\settowidth{\Tl}{Prefix}
\newcommand{\T}[1]{$10^{#1}$}
\begin{tabularx}{\linewidth}{|l|l|*{10}{>{\small}X|}}   \hline
\multicolumn{12}{|c|}{\textbf{Prefixes used in the SI system of units}}  \\
\hline
\multicolumn{2}{|c|}{Factor} &
\T{24} &\T{21} &\T{18} &\T{15} &\T{12} &\T{9} &\T{6} &\T{3} &\T{2} &\T{ }\\
\cline{1-2}
\multirow{2}{\Tl}{Prefix}&Name   &
yotta  &zetta  &exa    &peta   &tera   &giga  &mega  &kilo  &hecto &deca \\
                         &Symbol &
Y      &Z      &E      &P      &T      &G     &M     &k     &h     &da   \\
\hline
\multirow{2}{\Tl}{Prefix}&Symbol &
y      &z      &a      &f      &p      &n     &$\mu$ &m     &c     &d    \\
                         &Name   &
yocto  &zepto  &atto   &femto  &pico   &nano  &micro &milli &centi &deci \\
\cline{1-2}
\multicolumn{2}{|c|}{Factor} &
\T{-24}&\T{-21}&\T{-18}&\T{-15}&\T{-12}&\T{-9}&\T{-6}&\T{-3}&\T{-2}&\T{-1}\\
\hline
\end{tabularx}
```

Prefixes used in the SI system of units											
Factor		10^{24}	10^{21}	10^{18}	10^{15}	10^{12}	10^{9}	10^{6}	10^{3}	10^{2}	10
Prefix	Name	yotta	zetta	exa	peta	tera	giga	mega	kilo	hecto	deca
	Symbol	Y	Z	E	P	T	G	M	k	h	da
Prefix	Symbol	y	z	a	f	p	n	μ	m	c	d
	Name	yocto	zepto	atto	femto	pico	nano	micro	milli	centi	deci
Factor		10^{-24}	10^{-21}	10^{-18}	10^{-15}	10^{-12}	10^{-9}	10^{-6}	10^{-3}	10^{-2}	10^{-1}

В окружении tabular можно использовать различные другие окружения LaTeX'а, что иллюстрирует пример, приводимый ниже.

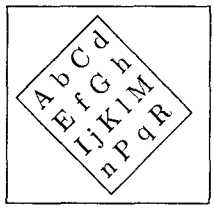
Окружение picture (см. с. 321)

```
\fbox{\begin{picture}(12,6.3)
      \bezier{200}(2.00,6.00)
       (7.00,6.00)(9.00,3.00)
    ....
\end{picture}}
```



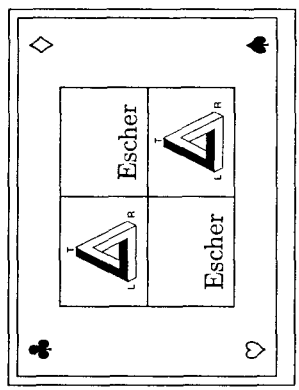
Вращение объекта (окружение turn, см. рис. 11.7)

```
\fbox{\begin{turn}{-45}
\fbox{\parbox[b]{13mm}{A b C d E f G h
                       I j K l M n P q R}}
      \end{turn}}
```



Вращение для tabular (см. разд. 11.4.1) с рисунками Encapsulated PostScript

```
\fbox{\begin{turn}{-90}
\begin{tabular}{|ccc|}          \hline
$\clubsuit$& &$\diamondsuit$ \\
&\begin{tabular}{|c|c|}         \hline
 \epsfig{file=Escher.eps,width=10mm}
                & Escher     \\\hline
 Escher &
 \epsfig{file=Escher.eps,width=10mm}
                               \\\hline
  \end{tabular} &              \\
&\heartsuit$ &  & $\spadesuit$\\\hline
\end{tabular}
\end{turn}}
```



ГЛАВА 6

Плавающие объекты

Восприятие документа упрощается, если его взаимосвязанные элементы размещаются в пределах одной и той же страницы. Однако зачастую это технически невозможно, и тогда TEX по умолчанию будет стремиться так перераспределить текстовый материал между смежными страницами, чтобы предотвратить получение частично заполненных страниц. В тех случаях, когда это нежелательно (как это имеет место применительно к рисункам и таблицам), соответствующий материал должен быть перемещен в некоторое удобное для восприятия место, такое, как нижняя или верхняя часть текущей или следующей страницы, что дает возможность избежать появления полупустых страниц.

По умолчанию LATEX обеспечивает средства формирования окружений (называемых «плавающими») [$\mathcal{L}$ **59,60,176–8**], [$\mathcal{A}$ **166–70**], обеспечивающих возможность перемещать материал способом, указанным выше. К таким окружениям относятся `figure` и `table`, стилевые параметры которых рассматриваются в разд. 6.1. Иногда могут потребоваться окружения, позволяющие сделать плавающими и другие виды объектов, например, компьютерные распечатки. Такого рода средства описываются в разд. 6.3. В разделе 6.4 обсуждаются несколько плавающих окружений и средства включения рисунков в другой рисунок, а в разд. 6.5 даются разъяснения по поводу модификации названий плавающих объектов.

6.1 Параметры плавающих объектов

Использование плавающих окружений в существующей версии LATEX'а часто порождает проблемы, поскольку данная система разрабатывалась в то время, когда объем используемых в документе графических материалов был существенно меньше, чем сейчас. Вследствие этого механизм размещения плавающих объектов (таблиц и рисунков) работает относительно хорошо до тех пор, пока простран-

ство, занимаемое этими объектами, невелико по сравнению с пространством, занимаемым текстом. Если, однако, имеется большое число плавающих объектов (т. е. много рисунков и таблиц), то часто все эти объекты оказываются размещенными в конце документа. Если такого рода эффект нежелателен, приходится пользоваться время от времени командой `\clearpage`, которая вызывает печать всех плавающих объектов, не обработанных к моменту ее активизации. Можно также попытаться выполнить подстройку стилевых параметров плавающих объектов для данного документа или воспользоваться каким-либо из пакетов, дающих возможность печатать таблицы и рисунки в том месте документа, где соответствующий объект появляется. В списке параметров, приводимом ниже, термин «плавающий объект» означает таблицу или рисунок, а термином «плавающая страница» обозначена страница документа, содержащая только плавающие объекты и не включающая текст. Изменения значений параметров сказываются только на следующей странице [$\mathcal{L}$ **176–8**], [$\mathcal{A}$**300–4**], не влияя на текущую.

`topnumber` Счетчик, задающий максимальное число плавающих объектов, которые могут быть размещены в верхней части страницы (значение по умолчанию равно 2). Его значение можно изменить при помощи команды `\setcounter`.

`bottomnumber` Счетчик, задающий максимальное число плавающих объектов, которые могут быть размещены в нижней части страницы (значение по умолчанию равно 1). Можно изменить при помощи команды `\setcounter`.

`totalnumber` Счетчик, задающий максимальное число плавающих объектов, размещаемых на одной странице (значение по умолчанию равно 3). Можно изменить при помощи команды `\setcounter`.

`\topfraction` Максимальная доля объема страницы, которая может быть занята плавающими объектами в верхней части страницы (например, 0.2 означает, что 20% объема страницы в ее верхней части может быть занято плавающими объектами; значение по умолчанию равно 0.7). Можно изменить при помощи команды `\renewcommand`.

`\bottomfraction` Максимальная доля объема страницы, которая может быть занята плавающими объектами в нижней части страницы (значение по умолчанию равно 0.3). Можно изменить при помощи команды `\renewcommand`.

`\textfraction` Минимальная доля нормальной страницы, которая должна быть занята текстом (значение по умолчанию равно 0.2). Можно изменить при помощи команды `\renewcommand`.

`\floatpagefraction` Минимальная доля плавающей страницы, которая должна быть занята плавающими объектами, т. е. ограничение объема незаполненного пространства, допускаемого на плавающей странице (значение по умолчанию равно 0.5). Можно изменить при помощи команды `\renewcommand`.

`dbltopnumber` Аналог счетчика `topnumber` для плавающих объектов, простирающихся на две колонки при двухколонном наборе (значение по умолчанию равно 2). Можно изменить при помощи команды `\renewcommand`.

\dbltopfraction Аналог параметра \topfraction для плавающих объектов, простирающихся на две колонки при двухколонном наборе (значение по умолчанию равно 0.7). Можно изменить при помощи команды \renewcommand.

\dblfloatpagefraction Аналог параметра \floatpagefraction для плавающих объектов, простирающихся на две колонки при двухколонном наборе (значение по умолчанию равно 0.5). Можно изменить при помощи команды \renewcommand.

\floatsep Величина «отбивки», задающая отступ по вертикали, добавляемый между плавающими объектами, появляющимися в верхней или нижней части страницы (значение по умолчанию равно 12pt plus 2pt minus 2pt для размеров шрифта основного текста 10 pt и 11 pt; 14pt plus 2pt minus 4pt для шрифта 12 pt). Значение этого параметра может быть изменено при помощи команды \setlength.

\textfloatsep Величина «отбивки», задающая отступ по вертикали, добавляемый между плавающими объектами, появляющимися в верхней или нижней части страницы, и текстом (значение по умолчанию равно 20pt plus 2pt minus 4pt). Значение этого параметра может быть изменено при помощи команды \setlength.

\intextsep Величина «отбивки», задающая отступ по вертикали, добавляемый сверху и снизу плавающего объекта, размещенного среди текста, если задан спецификатор h (значение по умолчанию совпадает с аналогичным значением для параметра \floatsep). Значение этого параметра может быть изменено при помощи команды \setlength.

\dblfloatsep Величина «отбивки», аналогичная параметру \floatsep для плавающих объектов, простирающихся на две колонки при двухколонном наборе (значение по умолчанию совпадает с аналогичным значением для параметра \floatsep). Значение этого параметра может быть изменено при помощи команды \setlength.

\dbltextfloatsep Величина «отбивки», аналогичная параметру \textfloatsep для плавающих объектов, простирающихся на две колонки при двухколонном наборе (значение по умолчанию для текстовой страницы аналогично значению параметра \textfloatsep, однако для страницы, содержащей только плавающие объекты, оно будет равно 8pt plus 2fil). Значение этого параметра может быть изменено при помощи команды \setlength.

\topfigrule Команда, позволяющая поместить линейку или какой-либо другой разделитель между плавающими объектами в верхней части страницы и текстом. Эта команда выполняется непосредственно перед формированием отступа \textfloatsep, отделяющего плавающие объекты от текста. Так же, как и в случае команды \footnoterule, линейка не должна, с точки зрения TEX'a, занимать места по вертикали (т. е. она располагается внутри отступа между плавающими объектами и текстом).

`\botfigrule` Подобна команде `\topfigrule`, но выполняется после отступа `\textfloatsep`, отделяющего текст от плавающих объектов в нижней части страницы.

`\dblfigrule` Подобна команде `\topfigrule`, но для случая плавающих объектов шириной в две колонки.

Изменяя значения перечисленных выше параметров, можно воздействовать на то, как будет работать LaTeX'овский алгоритм размещения плавающих объектов. Однако для улучшения получаемых результатов следует учитывать взаимосвязи, имеющиеся между этими параметрами.

Если воспользоваться значениями параметров управления размещением плавающих объектов, предлагаемыми по умолчанию, то можно обнаружить, что в случаях, когда число таких объектов достаточно велико, итоговый документ будет включать несколько плавающих страниц, т. е. страниц, содержащих только плавающие объекты. Зачастую на таких страницах будет много пустого пространства, например, возможно появление страниц, каждая из которых содержит единственный плавающий объект, занимающий лишь половину места, доступного на странице; в этом случае явно было бы лучше, если бы LaTeX заполнял пространство, остающееся от плавающего объекта, текстом. Источник такого поведения LaTeX'а состоит в том, что алгоритм размещения плавающих объектов пытается разместить возможно большее число таких объектов, остающихся после завершения страницы. Данный алгоритм поэтому создает столько плавающих страниц, сколько это возможно до тех пор, пока плавающих объектов набирается на полную (плавающую) страницу. Формирование плавающей страницы управляется параметром `\floatpagefraction`, задающим минимальную долю страницы, которая должна быть занята одним или несколькими плавающими объектами — по умолчанию это половина страницы. Поскольку стандартные настройки допускают размещение каждого плавающего объекта на отдельной странице (значение спецификации размещения по умолчанию равно `tbp`) [$\mathcal{L}$ **176**], [$\mathcal{A}$**166**], то для каждого такого объекта, даже незначительно превышающего по объему половину страницы, будет сформирована плавающая страница с одним этим объектом на ней. Отсюда следует, что, увеличивая значение параметра `\floatpagefraction`, можно избежать появления полупустых страниц.

Однако увеличение значения параметра `\floatpagefraction` затрудняет формирование плавающих страниц и в результате размещение некоторых плавающих объектов будет отложено, что, в свою очередь, помешает нормальному размещению других плавающих объектов. По этой причине зачастую лучше задать допустимые варианты компоновки явно (например, записав `\begin{figure}[tb]`) для тех плавающих объектов, размещение которых вызвало трудности.

Еще одно общее соображение, положенное в основу алгоритма размещения плавающих объектов, из-за которого все такие объекты могут оказаться сгруппированными в конце соответствующей главы (документа), состоит в характере использования спецификатора `[b]`, задающего размещение плавающих объектов в нижней части страницы. Такая спецификация значит, что допустимым местом

размещения является только нижняя часть страницы, и если размещаемый объект окажется больше, чем доля страницы в ее нижней части, предназначенная для размещения плавающих объектов (`\bottomfraction`), а по умолчанию значение этого параметра весьма невелико, то рассматриваемый объект не сможет быть там размещен. Это обстоятельство будет препятствовать размещению всех плавающих объектов такого же типа. Описанная ситуация может возникнуть, только если заданы спецификаторы `[h]` или `[t]`, а плавающий объект слишком велик для остающейся части страницы или же слишком велик, чтобы поместиться в разрешенной верхней части страницы, определяемой значением параметра `\topfraction`.

При вычислении доли страницы, которую можно использовать для размещения плавающих объектов, LaTeX принимает во внимание отступ (т. е. значение параметра `\textfloatsep`) между плавающими объектами и основным текстом. Увеличение значения этого параметра автоматически ведет к уменьшению максимального размера плавающих объектов, которые разрешается размещать в верхней или нижней части страницы.

В общем случае, всякий раз когда в сформированном документе значительное число плавающих объектов оказалось сосредоточенными в его конце, следует в первую очередь проанализировать спецификаторы и параметры компоновки этих объектов, чтобы исключить такие их комбинации, которые препятствуют требуемому размещению имеющихся плавающих объектов.

6.2 Улучшенное размещение плавающих объектов

Алгоритм размещения плавающих объектов отдает предпочтение варианту с расположением их в верхней части страницы, даже если при этом объект появляется до фактической ссылки на него. Такой вариант не всегда допустим, однако простых средств решения данной проблемы путем частичного изменения LaTeX'овского алгоритма не существует. Эта задача решается пакетом flafter (автор — Франк Миттельбах), гарантирующим, что плавающий объект не появится в документе до тех пор, пока не встретится ссылка на него.

В ряде случаев, однако, можно использовать и менее сильнодействующие средства. Например, если плавающий объект принадлежит разделу, начинающемуся в середине страницы, но размещается в ее верхней части, то это выглядит так, как будто бы данный объект относится к предыдущему разделу. Естественно, возникает желание воспрепятствовать такой ситуации, но оставить возможность для других плавающих объектов появляться в верхней части страницы. Для этой цели в LaTeX 2_ε есть команда

`\suppressfloats[`*placement*`]`

Необязательный аргумент *placement* здесь может принимать значения `t` или `b`. Если в каком-либо месте документа поместить команду `\suppressfloats`, то на

странице, содержащей эту команду, в области, задаваемой параметром *placement*, все имеющиеся плавающие объекты будут перемещены на следующую страницу. Если параметр *placement* не задан, отсроченными станут все плавающие объекты на данной странице. Таким образом, если требуется предотвратить появление плавающего объекта за пределами соответствующего раздела (до его начала), в этом разделе можно определить следующую команду:

```
\renewcommand{\section}{\suppressfloats[t]%
                        \@startsection{section}{..}{..}{..} ...
                       }
```

Возможные аргументы команды `\@startsection` обсуждаются в разд. 2.3.2.

Еще один путь воздействия на характер размещения плавающих объектов в LaTeX2ε заключается в задании спецификатора `!` в сочетании со спецификаторами `h`, `t`, `b`. Указанные способы размещения плавающих объектов не будут работать на плавающих страницах. Это значит, что, если данный плавающий объект размещается в одиночку, ограничения, устанавливаемые параметрами, обсуждавшимися выше (например, `\textfraction`), будут проигнорированы. Следовательно, такой объект может быть помещен в определенных областях, если только не нарушено одно из двух ограничений:

- плавающий объект укладывается в данную страницу, т. е. его высота плюс высота материала, уже размещенного на этой странице, не превышает значения параметра `\textheight`.
- нет отложенных плавающих объектов того же самого типа.

Все другие ограничения, учитываемые в обычном варианте работы алгоритма размещения плавающих объектов (такие, как число плавающих объектов, которое разрешается разместить на одной странице, и т. п.), в данной ситуации игнорируются. Например, если задать `[!b]`, то соответствующий плавающий объект будет размещен в нижней части страницы, даже если он окажется по размеру больше, чем это определено командой `\bottomfraction`. Кроме того, при обработке данного плавающего объекта игнорируются также любые команды `\suppressfloats`.

Следует помнить, что порядок перечисления спецификаторов неважен и что все спецификаторы должны быть использованы хотя бы один раз. Например, запись `[bt]` означает то же самое, что и `[tb]`, и *не является* инструкцией для LaTeX'а попытаться разместить плавающий объект в нижней части страницы и только после этого попробовать поместить его в верхнюю ее часть. LaTeX использует всегда такую последовательность проверок, которая должна обеспечить нахождение приемлемого варианта размещения плавающего объекта:

1. Если имеется спецификатор `!`, исключить из рассмотрения большинство ограничений, как это было описано выше, после чего продолжить.
2. Если задан спецификатор `h`, попытаться разместить плавающий объект в заданной позиции. Если такая попытка не удается и другие варианты размещения данного объекта отсутствуют, то изменить спецификатор на `t` (для возможного размещения объекта в верхней части следующей страницы).

3. Если задан спецификатор t, попробовать разместить плавающий объект в верхней части текущей страницы.
4. Если задан спецификатор b, попробовать разместить плавающий объект в нижней части текущей страницы.
5. Если задан спецификатор p, попытаться поместить плавающий объект на плавающую страницу (или в плавающую колонку — при двухколонном наборе), если текущая страница (или колонка) завершилась.
6. Этапы 3 и 4 повторяются, если это необходимо, в начале каждой последующей страницы, после чего следует этап 5 .

Пакет afterpage (автор — Дэвид Карлайл) реализует команду \afterpage, которая исполняется после того, как выведена текущая страница.

Этот пакет может быть использован в следующих ситуациях.

- Иногда механизм размещения плавающих объектов, имеющийся в LaTeX'e, оказывается перегруженным, вследствие чего все плавающие рисунки и таблицы перемещаются в конец документа. Можно принудительно вывести все плавающие объекты, оставшиеся необработанными, при помощи команды \clearpage, однако ее выполнение приведет к преждевременному завершению текущей страницы. Пакет afterpage дает возможность записать команду \afterpage{\clearpage}. Это приведет к заполнению страницы текстом (как в обычном варианте ее формирования), после чего и будет выдана команда \clearpage, которая обеспечит принудительный вывод всех имеющихся плавающих объектов до начала новой текстовой страницы.
- При использовании окружения longtable для формирования многостраничных таблиц (см. разд. 5.4.2) могут возникнуть трудности при верстке текста, окружающего создаваемую таблицу, так что может представить интерес попытка заставить «плавать» окружение longtable. Однако, поскольку таблица, порождаемая данным окружением, может занимать подряд несколько страниц, хранение ее в памяти целиком, что требуется при использовании окружения table, становится в ряде случаев невозможным. Тем не менее, если таблица размещена в отдельном файле, например, с именем ltfile.tex, можно записать следующие команды:

```
\afterpage{\clearpage\input{ltfile}}
\afterpage{\clearpage\input{ltfile}\clearpage}
```

  Первая из этих двух команд обеспечивает появление текста сразу после окружения longtable, а вторая гарантирует, что окружающий таблицу текст вновь появится на новой странице.
- Можно также использовать команду \afterpage совместно с пакетом float и спецификатором [H], как это объясняется в конце разд. 6.3.

Следует отметить, что команда \afterpage не работает в режиме twocolumn (набор страниц в две колонки).

6.3 float — создание новых видов плавающих объектов

Пакет float (автор — Ансельм Линьно) расширяет возможности LaTeX'а по формированию определений плавающих объектов, таких как рисунки и таблицы. Он добавляет описатель «float style» (вид плавающего объекта), управляющий появлением плавающих объектов. Новые виды плавающих объектов могут быть определены с использованием команды `\newfloat`.

`\newfloat{`*type*`}{`*placement*`}{`*ext*`}[`*within*`]`

Команда `\newfloat` имеет четыре аргумента — три обязательных и один необязательный:

type Тип нового класса плавающих объектов, например, `program` или `algorithm`. После того как задана соответствующая команда `\newfloat`, можно использовать команды `\begin{program}`, `\end{algorithm*}` и им подобные.

placement Значения по умолчанию параметров размещения для данного класса плавающих объектов. Этими параметрами, как и в стандартном LaTeX'е, будут `t`, `b`, `p` и `h` как сокращения для ключевых слов `top`, `bottom`, `page` и `here` соответственно. Кроме того, вводится еще новый спецификатор `H`, который на самом деле не отвечает природе плавающего объекта, поскольку означает: поместить данный объект «здесь» и нигде более. Спецификатор `H` играет особую роль и, вследствие специфики реализации, не может быть использован во втором аргументе команды `\newfloat`. Другими словами, нельзя определить новый плавающий объект, а затем сказать — размещать его по умолчанию «здесь».

ext Расширение для именования вспомогательного файла, который будет содержать список иллюстраций (или каких-либо других плавающих объектов). В этот файл LaTeX будет записывать названия соответствующих плавающих объектов.

within Этот (необязательный) аргумент определяет уровень структурной единицы документа, в пределах которого будет осуществляться непрерывная нумерация плавающих объектов данного вида. Например, если значение *within* равно `chapter`, соответствующие плавающие объекты будут иметь нумерацию, возобновляемую с началом каждой главы (в стандартном LaTeX'е так обстоит дело с рисунками и таблицами в классах документов report и book).

`\floatstyle{`*style*`}`

Команда `\floatstyle` устанавливает значение по умолчанию для стиля плавающего объекта. Она используется для всех видов плавающих объектов, определяемых затем при помощи команды `\newfloat`; введенное определение действует до тех пор, пока в тексте не встретится следующая команда `\floatstyle`. Для

переопределения стиля плавающих окружений `figure` и `table` служит команда `\restylefloat`, обсуждаемая ниже. Команда `\floatstyle` имеет один аргумент: наименование стиля, например, `\floatstyle{ruled}`. Если задать в качестве значения данного аргумента цепочку литер, не совпадающую с одним из правильных наименований стилей, будет выдано сообщение об ошибке. Аргумент *style* может принимать одно из следующих значений:

`plain` Это тот стиль, который LaTeX обычно реализует, работая с плавающими объектами, т. е. ничего особенного не добавляется.

`boxed` Тело плавающего объекта заключается в рамку. Наименование этого объекта печатается под рамкой.

`ruled` Этот стиль плавающих объектов заимствован из стиля представления таблиц в книге *Concrete Mathematics* [21]. Заголовок, заключенный между парой горизонтальных линеек, печатается сверху, еще одна линейка завершает объект.

`\floatname{`*float*`}{`*floatname*`}`

Команда `\floatname` определяет имя, которое LaTeX использует, формируя название соответствующего плавающего объекта, т. е. «Рис.» для рисунка и т. д. Пример такой команды: `\floatname{program}{Program}`. По умолчанию команда `\newfloat` устанавливает имя плавающего объекта совпадающим со значением его параметра *type*, если только впоследствии не было задано другое имя.

`\floatplacement{`*float*`}{`*placement*`}`

Команда `\floatplacement` определяет значение по умолчанию для спецификатора размещения плавающих объектов данного класса, например, `\floatplacement{figure}{tp}`.

`\restylefloat{`*float*`}`

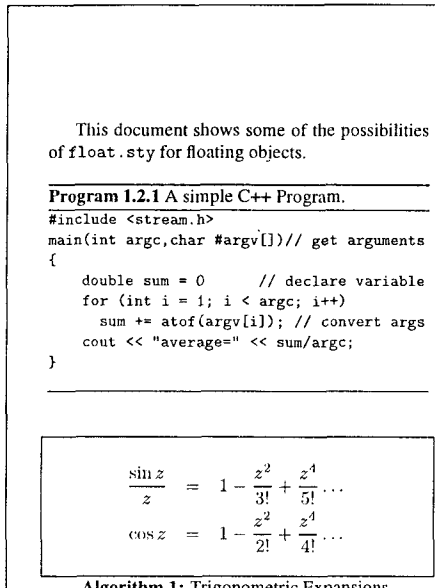
Команда `\restylefloat` необходима для изменения стиля стандартных типов плавающих объектов, т. е. `figure` и `table`. Поскольку они обычно не определяются командой `\newfloat`, соответственно нет и параметра-стиля, связанного с ними. В этом случае, чтобы сверстать таблицу в стиле `ruled`, надо записать:

```
\floatstyle{ruled}
\restylefloat{table}
```

Эта команда дает также возможность изменить стиль плавающих объектов, определенных при помощи команды `\newfloat`, хотя такой подход и нельзя признать слишком удачным.

`\listof{`*type*`}{`*title*`}`

Команда `\listof` формирует список всех плавающих объектов данного вида. Первый параметр (*type*) в ней задает вид плавающего объекта в той форме, как это было записано в команде `\newfloat`. Второй аргумент (*title*) определяет текст,



This document shows some of the possibilities of `float.sty` for floating objects.

Program 1.2.1 A simple C++ Program.

```
#include <stream.h>
main(int argc,char #argv[])// get arguments
{
    double sum = 0      // declare variable
    for (int i = 1; i < argc; i++)
      sum += atof(argv[i]); // convert args
    cout << "average=" << sum/argc;
}
```

$$\frac{\sin z}{z} = 1 - \frac{z^2}{3!} + \frac{z^4}{5!} \cdots$$
$$\cos z = 1 - \frac{z^2}{2!} + \frac{z^4}{4!} \cdots$$

Algorithm 1: Trigonometric Expansions.

```
\documentclass{book}
\usepackage{float,times}
\floatstyle{ruled}
\pagestyle{empty}
\newfloat{Program}{thp}{lop}[section]
\floatstyle{boxed}
\newfloat{algorithm}{thp}{loa}
\floatname{algorithm}{Algorithm}
\begin{document}
....
This document shows some of the
possibilities of \texttt{float.sty}
for floating objects.
\begin{Program}
\begin{verbatim}
#include <stream.h>
 ...
\end{verbatim}
\caption{A simple C++ Program.}
\end{Program}
\begin{algorithm}
\caption{Trigonometric Expansions.}
\begin{eqnarray*} ... \end{eqnarray*}
\end{algorithm}
\end{document}
```

Рис. 6.1. Пример определения двух «нестандартных» плавающих объектов — `Program` и `algorithm`

используемый в качестве заголовка списка сообщений об этих объектах, извлекаемых из команд `\caption`, для каждого из объектов, включенных в список. Команда `\listof` аналогична встроенным LaTeX'овским командам `\listoffigures` и `\listoftables`.

На рис. 6.1 приведен пример определения двух «нестандартных» плавающих объектов — `Program` и `algorithm`.

6.3.1 Разместить плавающий объект «здесь»!

Иногда может оказаться, что варианты размещения плавающего объекта, предлагаемые LaTeX'ом, не устраивают автора документа. Например, может потребоваться разместить плавающий объект точно в том месте, где он появился во входном файле, т. е. необходимо сделать, чтобы он вообще перестал быть плавающим. Распространенное заблуждение состоит в том, что спецификация `[h]` означает «здесь и только здесь». На самом же деле этот спецификатор только предписывает LaTeX'у *попытаться* разместить плавающий объект в текущей позиции. Если места для этого на странице оказалось недостаточно или если такое

«встроенное» размещение объекта запрещено соответствующей установкой стилевых параметров (см. разд. 6.1), то LaTeX проигнорирует данное предписание и попытается разместить плавающий объект в соответствии с другими заданными спецификаторами. Следовательно, если задана спецификация [ht], она означает, что объект, не поместившийся на текущей странице, будет размещен в верхней части одной из следующих страниц. Это часто случается при попытках разместить довольно большой плавающий объект среди текста, когда из-за отсутствия места возникает описанная выше ситуация. Пренебрегая спецификатором h и пытаясь использовать другое значение спецификатора размещения, LaTeX предотвращает появление существенно незаполненных страниц, порождаемых в подобного рода ситуациях. Однако в некоторых случаях предпочтительнее как раз допустить существование таких незаполненных фрагментов на странице. По этой причине в пакете float вводится спецификатор [H], обозначающий «поместить плавающий объект здесь».

Если на текущей странице места для размещения плавающего объекта остается недостаточно, он будет помещен в верхней части следующей страницы вместе с плавающими объектами, следующими за ним, если даже на этой (текущей) странице остается незаполненное пространство. При таком («ручном») управлении размещением плавающих объектов при помощи спецификатора H забота о том, чтобы на страницах документа не было слишком много пустого пространства в нижней части этих страниц, ложится целиком на автора подготавливаемого документа. Если в одном и том же документе используются плавающие объекты со стандартными значениями параметра размещения и со значением [H], то плавающий объект со спецификатором [t] (например), появляющийся во входном файле до одного из объектов со спецификатором [H], может быть обработан некорректно, так что, например, рис. 4 может оказаться напечатанным перед рис. 3.

Следует подчеркнуть, что использовать спецификатор H вместе с другими в одной спецификации нельзя. Вследствие этого любая комбинация наподобие [Hht] будет считаться ошибочной. Кроме того, [H] нельзя использовать в качестве значения параметра размещения по умолчанию для класса плавающих объектов в целом. Подытожить изложенный выше материал относительно спецификаторов размещения плавающих объектов можно в виде следующей табл. 6.3.1, которая была задана именно как объект со спецификацией [H] при помощи следующего описания:

t	Верхняя часть страницы	b	Нижняя часть страницы
p	Плавающая страница	h	Здесь, если можно
H	Здесь, в любом случае		

Таблица 6.1. Спецификаторы размещения плавающих объектов пакета float

```
\begin{table}[H]
\begin{tabular}{*2{>{\ttfamily}cl}}
t & Верхняя          & b & Нижняя          \\
  & часть страницы   & & часть страницы\\
p & Плавающая        & h & Здесь,          \\
  & страница         &   & если можно      \\
H & Здесь, \\
  &  в любом случае
\end{tabular}
\caption{Спецификаторы размещения
плавающих объектов пакета
\nxLpack{float}}
\end{table}
```

Еще одно важное замечание состоит в том, что пакет float не модифицирует стандартных окружений `figure` и `table`, в отличие от устаревшего уже пакета here (автор — Дэвид Карлайл). Поэтому, если возникает необходимость использовать спецификатор `[H]` с этими окружениями, необходимо выполнить вначале команду `\restylefloat{figure}` и/или `\restylefloat{table}`.

Пакет afterpage, описанный в разд. 6.2, совместно с рассматриваемыми здесь средствами позволяет добиться дополнительного улучшения управления размещением плавающих объектов. По сути дела, в некоторых случаях, когда задается значение параметра размещения, равное `[H]`, вовсе не имеется в виду «строго в данном месте», а на самом деле подразумевается «как можно ближе к данному месту». Такой интерпретации спецификатора `[H]` можно добиться, если следующим образом воспользоваться командой `\afterpage`:

```
\afterpage{\clearpage\begin{figure}[H]...\end{figure}}
```

Выполнение этой команды обеспечивает появление рисунка в верхней части следующей страницы. Она решает также проблему возможного нарушения последовательности плавающих объектов, упомянутую выше, поскольку команда `\clearpage` не дает возможности «перескочить» какому-либо другому рисунку назад, через рисунок со спецификатором `[H]`.

6.4 Другие виды плавающих окружений

6.4.1 floatfig — узкие плавающие рисунки «в оборку»

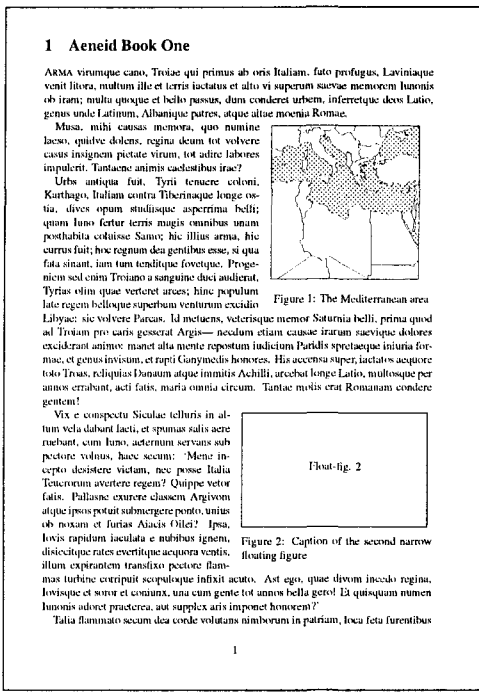
Пакет floatfig (автор — Томас Рейд [67]) содержит средства формирования рисунков, не заполняющих всю ширину страницы, так называемых рисунков «в оборку».

`\begin{floatingfigure}{`*width*`}`

Если ширина таких рисунков составляет только часть ширины страницы, остающееся свободным место сбоку от рисунка может быть заполнено текстом. Пакет floatfig полностью совместим со стандартным LaTeX'овским окружением `figure`:

1. Плавающие рисунки «в оборку» и обычные могут чередоваться в любом порядке.
2. Плавающим рисункам «в оборку» можно давать названия, подобно стандартным рисункам.
3. Плавающие рисунки «в оборку», имеющие название, помещаются в список иллюстраций, который может быть получен стандартной командой `\listoffigures`.

Окружение `floatingfigure` может использоваться, только когда TeX находится в вертикальном режиме, т. е. между абзацами. Плавающий рисунок «в оборку» должен быть помещен в документ возможно раньше, после того как встретилась ссылка на него. Это значит, что LaTeX проверит, достаточно ли места (по



1 Aeneid Book One

ARMA virumque cano, Troiae qui primus ab oris Italiam, fato profugus, Laviniaque venit litora, multum ille et terris iactatus et alto vi superum saevae memorem Iunonis ob iram; multa quoque et bello passus, dum conderet urbem, inferretque deos Latio, genus unde Latinum, Albanique patres, atque altae moenia Romae.

Musa, mihi causas memora, quo numine laeso, quidve dolens, regina deum tot volvere casus insignem pietate virum, tot adire labores impulerit. Tantaene animis caelestibus irae?

Urbs antiqua fuit, Tyrii tenuere coloni, Karthago, Italiam contra Tiberinaque longe ostia, dives opum studiisque asperrima belli; quam Iuno fertur terris magis omnibus unam posthabita coluisse Samo; hic illius arma, hic currus fuit; hoc regnum dea gentibus esse, si qua fata sinant, iam tum tenditque fovetque. Progeniem sed enim Troiano a sanguine duci audierat, Tyrias olim quae verteret arces; hinc populum late regem belloque superbum venturum excidio Libyae: sic volvere Parcas. Id metuens, veterisque memor Saturnia belli, prima quod ad Troiam pro caris gesserat Argis— necdum etiam causae irarum saevique dolores exciderant animo: manet alta mente repostum iudicium Paridis spretaeque iniuria formae, et genus invisum, et rapti Ganymedis honores. His accensa super, iactatos aequore toto Troas, reliquias Danaum atque immitis Achilli, arcebat longe Latio, multosque per annos errabant, acti fatis, maria omnia circum. Tantae molis erat Romanam condere gentem!

Vix e conspectu Siculae telluris in altum vela dabant laeti, et spumas salis aere ruebant, cum Iuno, aeternum servans sub pectore vulnus, haec secum: 'Mene incepto desistere victam, nec posse Italia Teucrorum avertere regem? Quippe vetor fatis. Pallasne exurere classem Argivom atque ipsos potuit submergere ponto, unius ob noxam et furias Aiacis Oilei? Ipsa, Iovis rapidum iaculata e nubibus ignem, disiecitque rates evertitque aequora ventis, illum expirantem transfixo pectore flammas turbine corripuit scopuloque infixit acuto. Ast ego, quae divom incedo regina, Iovisque et soror et coniunx, una cum gente tot annos bella gero! Et quisquam numen Iunonis adoret praeterea, aut supplex aris imponet honorem?'

Talia flammato secum dea corde volutans nimborum in patrium, loca feta furentibus

1

Figure 1: The Mediterranean area

Float-fig. 2

Figure 2: Caption of the second narrow floating figure

```
\documentclass{article}
\usepackage{floatfig,epsfig}
\begin{document}
\initfloatingfigs
\section{Aeneid Book One}
\textsc{Arma} virumque cano, ...
..., atque altae moenia Romae.
\par
\begin{floatingfigure}{6cm}
\mbox{\epsfig{file=Mediterranean.eps}}
\caption{The Mediterranean area}
\end{floatingfigure}
\quad Musa, mihi causas ...
...erat Romanam condere gentem!
\par
\begin{floatingfigure}{7cm}
\fbox{\parbox{66mm}{\rule[-2cm]{0mm}%
     {4cm}\hfil        Float-fig. 2}}
\caption{Caption of the second narrow
         floating figure}
\end{floatingfigure}
\quad Vix e conspectu Siculae telluris
in altum  ....
\end{document}
```

Рис. 6.2. Пример узких плавающих рисунков «в оборку»

высоте) на текущей странице. Если ответ на этот вопрос отрицателен, рисунок перемещается на следующую страницу. Плавающие рисунки «в оборку» размещаются на странице с чередованием их положения: справа — для нечетных страниц, слева — для четных. Рисунок 6.2 демонстрирует пример применения окружения `floatingfigure`.

Отметим, что пакет floatfig несовместим с режимом twocolumn, а также то обстоятельство, что плавающий рисунок «в оборку» никогда не появится в абзаце, начинающемся в самой верхней части страницы.

6.4.2 wrapfig — неплавающие рисунки «в оборку»

Пакет wrapfig (автор — Дональд Арсено) определяет окружение `wrapfigure`. Это окружение позволяет разместить рисунок вручную, прижатым к правому или левому полю страницы, и организовать «обтекание» этого рисунка текстом.

`\begin{wrapfigure}[`*nlines*`]{`*placement*`}{`*width*`}`

Окружение `wrapfig` имеет три аргумента, из которых два обязательных:

nlines Этот факультативный аргумент определяет число коротких строк, обтекающих рисунок. Если в обтекающем текстовом фрагменте имеются выключные формулы, то считается, что каждая из них занимает три строки.

placement Расположение рисунка по горизонтали (`l` — слева, `r` — справа).

width Ширина рисунка.

Абзац внизу, с размещенным в нем рис. 6.3, порожден следующим исходным текстом:

```
\begin{wrapfigure}{r}{3in}
\begin{boxit}
  \begin{center} Это рисунок "<в оборку">. \end{center}
  \caption{Пример окружения \nxLenv{wrapfigure}}
\end{boxit}
\end{wrapfigure}
%%
Окружение \Lenv{wrapfigure} "--- один из вариантов
\emph{неплавающих} окружений в \LaTeX 'е.
```

Это рисунок «в оборку».

Рис. 6.3. Пример окружения `wrapfigure`

Окружение `wrapfigure` — один из вариантов *неплавающих* окружений в LaTeX'е. Рисунок заданной ширины должен появиться в правой или левой части формируемой страницы. LaTeX будет стараться организовать обтекание данного рисунка текстом, оставляя промежуток величиной `\columnsep` между рисунком и текстом, формируя соответствующие короткие строки. Число таких коротких строк зависит от высоты рисунка, к которой прибавляется длина промежутка `\intextsep`. Можно воздействовать на число формируемых коротких строк, задавая значение необязательного аргумента *nlines*.

Окружение `wrapfigure` нельзя использовать внутри другого окружения (например, `list`), кроме того, оно не работает в режиме формирования страницы в две колонки (twocolumn). Поскольку данное окружение не является плавающим, его можно исключить из последовательности плавающих рисунков.

LaTeX не будет размещать окружение `wrapfigure` в оптимальном для этого окружения месте, эту операцию придется выполнить вручную. Подбор такого места для рисунка с обтеканием его текстом следует выполнять непосредственно перед выводом на печать *окончательного варианта* документа, поскольку любое изменение в документе может испортить результаты проведенной «подгонки». Можно сформулировать несколько правил, позволяющих улучшить размещение рисунка, обтекаемого текстом:

- Окружение надо поместить так, чтобы оно не вышло за пределы страницы.
- Для «обтекания» рисунка допускается только простой текст, в нем не может быть заголовков разделов. Математические выражения в обтекающем тексте использовать можно, если они укладываются в длину формируемых коротких строк.

- Удобно помещать команду `\begin{wrapfigure}` сразу после завершения абзаца. Если имеется необходимость разместить окружение `wrapfigure` внутри абзаца, его надо помещать между двумя словами в том месте абзаца, где естественным образом происходит завершение строки.

6.4.3 subfigure — рисунки внутри рисунков

Пакет subfigure (автор — Стефен Кокрен) позволяет помещать рисунки внутри рисунков, каждый из них со своей собственной подписью, и, кроме того, есть возможность поместить (необязательную) общую подрисуночную подпись под всем рисунком в целом.

По горизонтали рисунок центрируется, над внутренними рисунками формируется вертикальный отступ размером `\subfigtopskip` (значение по умолчанию равно 10pt), между внутренними рисунками и общей подрисуночной подписью оставляется еще один отступ по вертикали, равный `\subfigcapskip` (значение по *умолчанию равно* 10pt).

Снизу внутреннего рисунка добавляется еще один вертикальный отступ, равный `\subfigtopskip`.

Если задано название, заключенное в квадратные скобки (включая и пустое название `[]`), то внутренние рисунки будут печататься при помощи счетчика, обеспечиваемого макро `\thesubfigure` (по умолчанию метка подрисунка определена как `(\alph{subfigure}){space}`, что дает такие значения меток, как `(a)`, `(b)` и т. д.). При необходимости это макро может быть переопределено. Счетчик, используемый для формирования меток внутренних рисунков, именуется `subfigure`, и значение его меняется при переходе от одного внутреннего рисунка к другому независимо от того, имеется у того или иного название или нет.

Рисунок 6.4 на развороте дает пример того, как расположить по горизонтали три внутренних рисунка при помощи окружения `tabular`. Каждый из них имеет свою собственную подпись, кроме того, формируется общая подрисуночная подпись для рисунка в целом. На рисунок можно ссылаться в тексте по ключевому слову, определяемому командой `\label`.

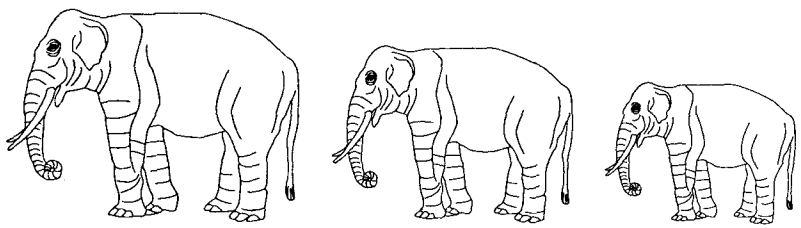
6.4.4 endfloat — размещение рисунков и таблиц в конце документа

В некоторых журналах требуется, чтобы рисунки и таблицы были отделены от текста и сгруппированы в конце документа. Может также потребоваться, чтобы рисункам и таблицам предшествовал их список.

Пакет endfloat (автор — Джеймс Даррелл Мак-Коули) помещает рисунки и таблицы изолированно от текста в конце статьи в раздел, озаглавленный, соответственно, `Figures` (Иллюстрации) или `Tables` (Таблицы).

Формирование списков иллюстраций и таблиц, помещенных в конце документа, может быть отключено при помощи команд `\nofiglist` и `\notablist` соответственно, размещаемых в преамбуле.

```
\begin{figure}
\centering
\mbox{\subfigure[Большой]{\epsfig{figure=elephant.eps,width=.30
      \textwidth}}\quad
      \subfigure[Средний]{\epsfig{figure=elephant.eps,width=.25
      \textwidth}}\quad
      \subfigure[Малый]{\epsfig{figure=elephant.eps,width=.20
      \textwidth}}}
\caption{Три внутренних рисунка в рисунке}
\label{fig:subfigures}
\end{figure}
```



(a) Большой (b) Средний (c) Малый

Рис. 6.4. Три подрисунка

В текст статьи вставляются примечания вида «[Figure 4 about here]», показывающие примерное место, где ориентировочно появился бы плавающий объект в нормальном режиме формирования документа. Эти примечания можно отключить, поместив в преамбулу документа команду `\nomarkersintext`. Текст данных примечаний, определяемый строками `\figureplace` и `\tableplace`, можно изменить при помощи `\renewcommand`. Определения этих команд, принятые по умолчанию, имеют вид

```
\newcommand{\figureplace}{% Для рисунков
 \begin{center}[\figurename~\thepostfig\ about here.]\end{center}}
\newcommand{\tableplace}{%  Для таблиц
 \begin{center}[\tablename~\theposttbl\  about here.]\end{center}}
```

Пакет endfloat создает два дополнительных файла с расширениями `.fff` и `.ttt`. При использовании данного пакета может понадобиться дополнительный запуск LaTeX'а, чтобы уладить дело с перекрестными ссылками, изменившимися из-за перемещения плавающих объектов.

6.5 Создание своих названий

Если требуется пояснить, что изображает то или иное плавающее окружение (`figure` или `table` в стандартном LaTeX'e), обычно используется команда `\caption`. Эта команда определена только внутри окружений. Вне формируемого текста она обеспечивает также помещение названий в список объектов соответствующего вида и увеличение значения счетчика, связанного с объектами данного вида. Эта команда имеет следующий синтаксис:

`\caption[`*short text*`]{`*long text*`}`

Необязательный аргумент *short text* содержит текст, помещаемый в список иллюстраций или таблиц. Если имеется только обязательный аргумент *long text*, то в упомянутых списках он и используется. Если название не умещается в одну строку, настоятельно рекомендуется использовать необязательный аргумент, чтобы сделать описание плавающего объекта в списке коротким и информативным. В противном случае список иллюстраций или таблиц станет неудобочитаемым, а также возникнут трудности с размещением в нем соответствующей информации.

В теле команды `\caption` производится выдача следующей внутренней команды:

`\@makecaption{`*numb*`}{`*text*`}`

Номер (*numb*) генерируется внутренними средствами команды `\caption` в зависимости от вида плавающего объекта. Аргумент *text* представляет собой текст для печати (текст названия). По умолчанию, определение для части команды, осуществляющей печать названия, имеет примерно следующий вид:

```
\newcommand{\@makecaption}[2]{% #1 is e.g. Figure 1, #2 is caption text
   \vspace{10pt}\sbox{\tempbox}{#1: #2}%
   \ifthenelse{\lengthtest{\wd\tempbox > \linewidth}}%
     { #1: #2\par}% More than one line
     {\begin{center}#1: #2\end{center}}%
}
```

После первоначального отступа (skip) по вертикали, равного 10 pt, материал названия верстается во временный бокс `\tempbox`, и его ширина сравнивается с шириной строки. Если материал названия укладывается в одну строку, то текст его центрируется; если же название не помещается в одной строке, то оно оформляется в виде абзаца с шириной, равной ширине строки текста.

Можно, конечно, определить и другие способы форматирования названий, можно даже задать разные команды для разных видов плавающих объектов. Например, для форматирования названий в окружении `figure` вместо команды `\@makecaption` можно использовать команду `\@makefigcaption`, определенную следующим образом:

```
\newcommand{\@makefigcaption}[2]{....}
\renewcommand{\figure}
    {\let\@makecaption\@makefigcaption\@float{figure}}
```

В качестве примера одного из возможных способов форматирования названий в пакете hangcaption (автор — Дэвид Джоунз) определена команда \isucaption (вариант команды \caption), которая порождает названия с отступом. Если название короче полной текстовой строки на данной странице, оно будет отцентрировано. Для управления шириной названия можно использовать переменную \captionwidth. Интересный фрагмент определения пакета, который можно сравнить с текстом, приведенным выше, имеет вид:

```
\newcommand{\@isucaption}[2]{% #1 is e.g. Figure 1, #2 is caption text
  \par\vspace{10pt}\sbox{\tempbox}{#1: #2}%
  \ifthenelse{\lengthtest{\wd\tempbox > \linewidth}}%     <> 1 line?
   {\sbox{\tempbox}{#1:\ }%                    Measure text of part 1
    \addtolength{\captionwidth}{-\wd\tempbox} % Subtract from width
    \mbox{#1:\ }\parbox[t]{\captionwidth}{#2}}}%     place two boxes
     {\begin{center}#1: #2\end{center}}%               One line only
}
```

ГЛАВА 7

Переключение шрифтов

7.1 Введение в NFSS

(LA)TEX как издательская система выполняет половину работы, связанной с подготовкой верстки: на основе исходного файла пользователя она вычисляет позиции литер (символов) на странице. Она основывается при этом лишь на простейшей информации о размещаемых символах, которые интерпретируются ею как черные прямоугольники (боксы), обладающие шириной, высотой и глубиной (долей прямоугольника, располагающейся ниже базовой линии). Эта информация содержится во внешних файлах, по одному на каждый шрифт, называемых файлами метрик шрифтов или, короче, `.tfm`-файлами.

Информация о начертании символов, согласующаяся с их параметрами в `.tfm`-файле, требуется уже после того, как (LA)TEX'ом сформирован выходной `.dvi`-файл. А именно, информация о размещении символов из `.dvi`-файла совместно с информацией о форме символов, содержащейся в `.pk`-файле или же в описании символа (например, средствами языка PostScript), объединяется программой-драйвером, которая и формирует образ документа, соответствующий устройству вывода определенного вида. Обычно для каждого класса устройств вывода (экран монитора, принтер и т. п.) требуется своя собственная программа-драйвер.

К моменту создания TEX'а в 1979 г. с ним можно было использовать всего около десятка шрифтов семейства Almost Computer Modern, также созданных Дональдом Кнутом. Поскольку число доступных шрифтов было невелико, для выбора того или иного из них использовался достаточно простой подход: было определено несколько управляющих последовательностей, позволявших переключаться с одного шрифта на другой.

Пять лет спустя, когда была выпущена первая версия LATEX'а, ситуация не слишком изменилась; сменилось только имя семейства шрифтов, поставлявшихся с LATEX'ом: с Almost Computer Modern на Computer Modern, причем сами

шрифты остались практически теми же. Естественно, что в такой ситуации схема переключения шрифтов была унаследована LaTeX'ом от plain TeX'а с добавлением к ней команд изменения размера шрифта, позволявших выбирать шрифт одного из десяти предопределенных размеров.

В результате LaTeX'овская схема переключения шрифтов оказалась далекой от общности. Например, если определить команду формирования заголовка, набираемого полужирным шрифтом (используя команду `\bf` в создаваемом определении), то другая команды переключения шрифта, к примеру `\sf` (для рубленого шрифта, т. е. шрифта без засечек) внутри заголовка отнюдь не приведет к тому, что заголовок будет набран полужирным рубленым шрифтом. Переключение в данном случае будет произведено на рубленый шрифт нормальной насыщенности, т. е. одна из команд переключения атрибута шрифта будет проигнорирована. Аналогично, если, например, использовать команду `\bf` внутри выделенного (при помощи команды `\emph`) текста, то результатом будет не полужирный курсив, как это требовалось, а обычный прямой полужирный шрифт.

Такого рода результаты обусловлены тем, что команды переключения шрифтов, подобные `\bf`, жестко связаны с определенными шрифтами. Поэтому выдача команды изменения атрибута шрифта приводит к замене текущего шрифта другим, связанным с выданной командой. Конечно, в LaTeX'е сделан определенный шаг вперед по сравнению с plain TeX'ом в части управления атрибутами шрифтов благодаря введению набора команд изменения размера шрифта. Однако концепция, положенная в основу механизма управления шрифтами в первоначальной версии LaTeX'а, страдала серьезным изъяном: таблицы соответствия имен атрибутов шрифтов и самих шрифтов были жестко встроены в LaTeX, так что переключение шрифтов становилось задачей очень трудной, если вообще выполнимой.

Через некоторое время после выпуска первоначальной версии LaTeX'а появились недорогие лазерные принтеры, а вместе с ними — большое число семейств шрифтов (PostScript-шрифтов и других). Число шрифтов в формате METAFONT'а (свободно доступных везде, где использовался (La)TeX) также росло очень быстро. Однако, к сожалению, не было простого и стандартного метода, позволяющего интегрировать эти новые шрифты в LaTeX — работать в LaTeX'е почти всегда означало работать только с семейством шрифтов Computer Modern. Отдельные шрифты можно было, разумеется, загрузить командой `\newfont` [$\mathcal{L}$ **116,200**], [$\mathcal{M}$**311–2**], однако назвать эту возможность интеграцией было никак нельзя: работа с таким шрифтом требовала больших усилий от пользователя, так как подключаемые при помощи `\newfont` дополнительные шрифты не подчинялись командам изменения размера шрифта и не давали возможности набирать документ целиком шрифтом, отличающимся от стандартного.

Было предпринято несколько попыток обеспечить интеграцию новых шрифтов в LaTeX, однако все они сводились к замене одной жесткой таблицы соответствия на другую, LaTeX в модифицированном варианте оставался таким же негибким, как и был, только заставлял работать с другим набором шрифтов.

Эта тяжелая ситуация была в конце концов изменена в результате создания Франком Миттельбахом и Райнером Шопфом в 1989 г. новой схемы выбора шрифтов, названной ими NFSS (New Font Selection Scheme) [52, 54]. Она стала широко известна после того, как была успешно применена в пакете $\mathcal{AMS}$-LaTeX (см. гл. 8). Эта система базируется на общей концепции индивидуального изменения атрибутов шрифта и простой интеграции новых семейств шрифтов в существующую систему LaTeX.

Упомянутая концепция основывается на пяти атрибутах, которые могут быть определены независимо для доступа к различным шрифтам, их характеристикам и семействам. Чтобы реализовать эту концепцию, пришлось переопределить некоторые команды LaTeX'а и добавить ряд новых команд.

Несколько позже Марком Петиллом был реализован опытный вариант набора масштабируемых шрифтов. Опираясь на эти результаты, Франк Миттельбах разработал новый вариант схемы переключения шрифтов — NFSS2; проводились также работы по интеграции с LaTeX'ом на этой базе PostScript-шрифтов (Себастьян Ратц и др.).

В последующих разделах дается описание версии 2 схемы NFSS, разработка которой была завершена к концу 1992 г. Эта версия в настоящее время является составной частью системы LaTeX2_ε, и, что касается интерфейса с пользователем, должна войти в систему LaTeX3.

Поскольку концепции, используемые в NFSS, полностью отличаются от подхода к управлению шрифтами в предыдущих версиях (La)TeX'а, целесообразно начать с общего обсуждения характеристик шрифтов, а также ввести главные атрибуты, используемые в NFSS для ортогонального переключения шрифтов. Затем будет описан высокоуровневый интерфейс с NFSS, т. е. те команды, которые используются в обычной практике. Сюда входят команды, применяемые в тексте (разд. 7.3), специальные средства для использования в математических формулах (разд. 7.4), а также обзор пакетов, доступных в рамках NFSS (разд. 7.5). В следующей части данной главы содержится описание низкоуровневого интерфейса с NFSS, полезного, когда приходится определять новые сложные команды, и, что еще важнее, когда приходится подключать к LaTeX'у новые шрифты. В завершение главы (в разд. 7.8) дается описание всех сообщений об ошибках и предупреждений, связанных с механизмом управления шрифтами.

7.2 Характеристики шрифтов

Имеется целый ряд характеристик, по которым шрифты можно разделить на отдельные пересекающиеся классы. Знание этих характеристик часто оказывается полезным, когда приходится решать проблему выбора семейства шрифтов, отвечающего тем или иным предъявляемым условиям.

7.2.1 Моноширинные и пропорциональные шрифты

Шрифты могут быть либо моноширинными, либо пропорциональными. В моноширинном шрифте каждая отдельная литера, независимо от ее начертания, занимает одно и то же место по горизонтали. Литерам пропорциональных шрифтов для их размещения требуются, напротив, горизонтальные промежутки разных размеров, зависящие от начертания той или иной литеры. Из рис. 7.1 видно, в частности, что буква `i` в моноширинном шрифте занимает по горизонтали ровно столько же места, что и `m`, тогда как в пропорциональном шрифте место, занимаемое `i`, существенно меньше места, занимаемого `m`.

В результате пропорциональные шрифты (называемые также полиграфическими шрифтами) позволяют разместить на странице больший объем текста, причем текст этот более удобочитаемый, чем в случае моноширинного шрифта. Лишние промежутки по горизонтали, окружающие отдельные символы в моноширинных шрифтах, затрудняют их зрительное восприятие из-за менее четко выраженных границ слов, что и обусловливает меньшую удобочитаемость текстов, набранных моноширинными шрифтами. Однако у моноширинных шрифтов имеется своя сфера применения. В ряде случаев они повышают качество печатного документа. Например, в таблицах или компьютерных распечатках, где важным является соответствующее выравнивание представляемых данных, моноширинные шрифты будут вполне уместны. В книгах по информатике, например, принято набирать компьютерные программы именно моноширинным шрифтом, что позволяет выделить визуально эти программы на фоне основного текста, содержащего, в частности, пояснения к приводимой программе.

Однако использование моноширинных шрифтов выходит за рамки потребностей визуального выделения фрагментов документа. Рассматривался даже вариант, когда моноширинный шрифт применяется для набора целиком всего документа, точнее, в качестве основного (базового) шрифта документа. Такой шрифт придает документу вид выполненного на механической или электрической пишущей машинке и даже производит впечатление написанного от руки, когда оставляется рваный правый край абзацев, так что в некоторых случаях такой документ будет, возможно, лучше отвечать ситуации, в которой он создается, чем профессионально подготовленный текст типографского качества. Один из вариантов,

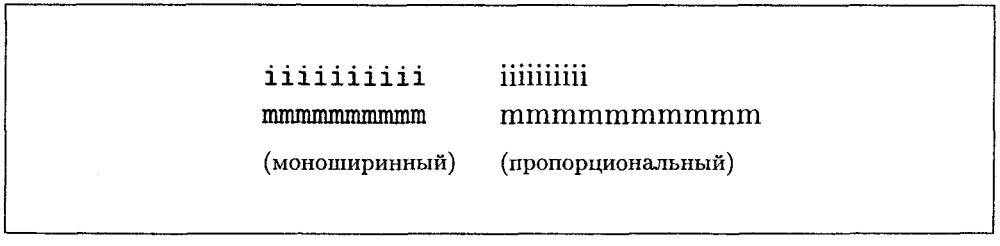


Рис. 7.1. Основные характеристики шрифта



Рис. 7.2. Сравнение букв с засечками и без засечек

когда оправдано применение моноширинных шрифтов — это письма частного характера, поскольку они выглядят в таком случае более личностно окрашенными. Следует, однако, иметь в виду, что моноширинные шрифты выглядят очень невыразительно, если абзацы выровнены по правому краю (см. разд. 3.1.4, где сказано, как отключить это выравнивание).

7.2.2 Шрифты с засечками и без засечек

Другая полезная классификация шрифтов основывается на наличии или отсутствии засечек — крошечных горизонтальных черточек в крайних точках линии, образующей ту или иную литеру (см. рис. 7.2). Первоначально засечки выполнялись резцом при гравировке на камне букв латинского (древнеримского) капитального письма. По этой причине шрифты с засечками часто называют «романскими» шрифтами.

Шрифты с засечками традиционно используются при наборе длинных текстов, поскольку считается, что при этом повышается уровень удобочитаемости текста. Долгое время считалось также, что буквы с засечками лучше и быстрее распознаются глазом. Это действительно так, если видна только часть символа, но, как показали последние исследования, для текста, доступного для обозрения полностью, скорость чтения мало зависит от наличия или отсутствия засечек [71].

7.2.3 Семейства шрифтов и их атрибуты

Кроме приведенной выше довольно грубой классификации (шрифты с засечками против шрифтов без засечек (рубленых); моноширинные шрифты против пропорциональных шрифтов) используется также разделение шрифтов на семейства. Шрифты, входящие в семейство, характеризуются общим для них графическим решением и различаются между собой по размеру, жирности, ширине очка литеры и начертанию.

Начертания шрифтов

Важный элемент классификации шрифтов, входящих в некоторое семейство — это их начертание. Объединить несколько шрифтов с различными начертаниями в одно семейство или в несколько — это, конечно, дело вкуса того или иного автора. Например, Дональд Кнут назвал созданную им коллекцию из 31 шриф-

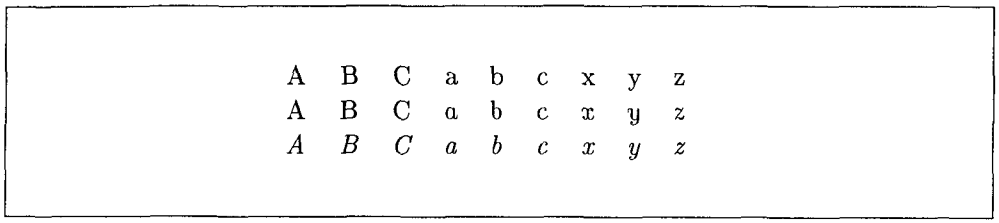


Рис. 7.3. Сравнение прямого и курсивного начертаний

Первая строка дает образец букв шрифта семейства Computer Modern Serif (шрифт с засечками) в прямом начертании, а третья показывает эти же буквы в курсивном начертании. Для лучшего сравнения этих двух видов начертаний во вторую строку помещены курсивные буквы без их обычного наклона, т. е. буквы, курсивные по начертанию, но искусственно лишенные наклона.

та семейством Computer Modern [21, 36], хотя в традиционном понимании это, скорее, метасемейство, т. е. набор семейств шрифтов[1].

Не существует общепринятых соглашений по именованию начертаний шрифтов, но это не имеет особого значения, если ограничиться только частным случаем схемы переключения шрифтов, реализованной в рамках NFSS.

Практически в каждом семействе шрифтов есть шрифт с начертанием, именуемым «прямым» (upright)[2]. Например, в семействе шрифтов, использованном для набора данной книги (Computer Modern), шрифт текста, который вы в данный момент читаете, имеет прямое начертание.

Другое важное начертание шрифта, присутствующее в большинстве семейств шрифтов, — курсивное; текст, набранный шрифтом с таким начертанием, в семействе Computer Modern выглядит *подобно этому фрагменту строки.* Курсивные буквы наклонены вправо, причем их рисунок обычно отличаются от рисунка аналогичных прямых букв, как это видно из рис. 7.3.

В семействах шрифтов без засечек соответствующее курсивное начертание обычно отсутствует, вместо этого имеется наклонное начертание, в котором буквы наклонены вправо, но во всем остальном они идентичны своим аналогам в прямом начертании. В англоязычной литературе для обозначения шрифтов с наклонным

[1] METAFONT как средство разработки шрифтов обеспечивает возможность получения совершенно различных шрифтов из одного и того же исходного описания, так что нет ничего удивительного в том, что в 1989 г. на базе исходных описаний шрифтов Computer Modern было сформировано другое семейство шрифтов [36]. Это семейство было получено варьированием ряда METAFONT'овских параметров в исходном тексте шрифтов. Результат вышел настолько *сильно отличающимся от существовавшего (Computer Modern), что Кнут решил дать вновь* полученному семейству шрифтов новое имя — Concrete Roman.

[2] Иногда такое начертание называют «латинским» или «романским» (roman). Причина здесь заключается в том, что до недавнего времени для набора в подавляющем большинстве случаев использовались шрифты с засечками. Таким образом, с точки зрения большинства людей, термину «романский» противопоставляется термин «курсивный». Так что надо смотреть, как следует понимать термин «романский» в каждом конкретном случае — как обозначение шрифтов с прямым начертанием или как обозначение семейства шрифтов с засечками.

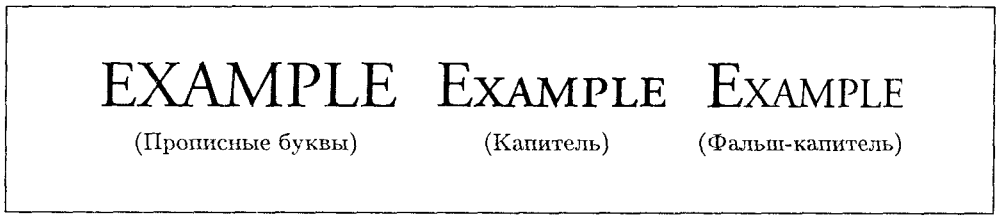


Рис. 7.4. Сравнение прописных букв и капители

начертанием кроме термина «slanted» используются также термины «sloped» и «oblique».

Еще один общеупотребительный вариант начертания представляет собой «капитель» (small caps). В этом начертании малые (строчные) буквы являются уменьшенными копиями больших (прописных) букв, как это показано на рис. 7.4. Если такое начертание в требуемом семействе отсутствует, в издательской практике иногда используют прямые прописные буквы из шрифта, меньшего размера[3], однако такой подход дает результат более низкого качества, чем специально разработанная капитель. Буквы капители отличаются по ширине очка и жирности от прописных букв, полученных из шрифта меньшего кегля (это отчетливо видно из рис. 7.4, в частности то, что линии, образующие букву, полученную из шрифта меньшего кегля, заметно тоньше, чем у буквы, специально разработанной в начертании «капитель»).

Есть еще несколько начертаний шрифтов, менее важных с практической точки зрения. Существуют семейства шрифтов, в которых внутренняя часть графических элементов, образующих букву, имеет специальный вид. Наиболее важным здесь будет, по-видимому, вариант, в котором упомянутая внутренняя часть остается незаполненной, пустой. Это так называемое «контурное» (outline) начертание. В числе шрифтов, ориентированных в первую очередь на вывод информации на экран монитора, имеются также шрифты с «оттененным» (shaded) начертанием, имитирующим эффект трехмерности. Примеры букв контурного и оттененного начертаний приведены на рис. 7.5.

С помощью системы METAFONT, варьируя соответствующие параметры в исходных текстах шрифтов, можно получать специальные варианты шрифтов, относящихся к метасемейству Computer Modern. Например, можно построить «прямой курсив», буквы которого по форме являются курсивными, но без обычного для курсива наклона вправо (см. в качестве примера вторую строку букв на рис. 7.3). Это начертание используется обычно в иллюстративных целях, чтобы продемонстрировать возможности METAFONT'а как средства метапроектирова-

[3] Хорошее эмпирическое правило в данной ситуации — использовать прописные буквы из шрифта, который примерно на полпункта больше, чем x-высота основного шрифта. Способ определения x-высоты для любого шрифта, используемого в TEX'е, рассматривается в разд. 7.7.2 на с. 234.



Рис. 7.5. Контурное и оттененное начертания

ния шрифтов, однако в каких-то экзотических случаях так можно набрать отдельные фрагменты документа.

Жирность и ширина

Шрифты определенного начертания в некотором семействе могут различаться между собой по «жирности». Эта характеристика относится к толщине линий, используемых для изображения символов из данного шрифта. Как и в случае с начертанием, общеупотребительная терминология, характеризующая жирность, отсутствует, но, тем не менее, подходящую градацию уровней жирности нетрудно построить. Некоторые разработчики шрифтов называют жирность букв, применяемых при наборе обычного текста, «книжной», другие — «нормальной» или «средней». Для шрифтов, символы которых образованы при помощи элементов, более тонких, чем в шрифтах нормальной жирности, общеупотребительно название «светлые» (light), а более толстых — название «полужирные» (bold). В больших семействах шрифтов может потребоваться и более тонкая градация, так что в них жирность шрифтов может изменяться от различных уровней очень светлого начертания (extra light, ultra light), через светлое (light), полусветлое (semi light) и т. д., вплоть до жирного (extra bold) и сверхжирного (ultra bold) начертаний. В других случаях в семейство могут входить шрифты лишь довольно ограниченного набора жирности. Например, семейство шрифтов Computer Modern Roman имеет только две градации жирности — (normal) и **(bold)**. Другой не менее важной характеристикой шрифта является его «ширина», т. е. большая или меньшая ширина очка литер в нем по сравнению с очком нормальной (средней) ширины из состава нормального (стандартного) шрифта данного семейства. Семейство Computer Modern Roman имеет в своем составе полужирные шрифты двух видов: **(medium width)** и **(extended width)**. Есть еще особые узкие (condensed) шрифты, которые можно было бы, например, использовать для набора элементов документа, расположенных на его полях. Некоторые издательские системы автоматически переходят на узкий шрифт в случае, когда текст не укладывается в отведенное ему место, например, при размещении заголовка. Эта возможность напрямую в (LA)TEX'е не доступна, да и ее эстетическая ценность зачастую довольно спорна.

Размеры шрифтов

Размеры шрифтов (или кегли) традиционно измеряются в типографских пунктах (pt). В одном дюйме содержится 72.27 пункта. Размер шрифта — это не какая-то абсолютная мера для него, а скорее некоторая описательная характеристика, даваемая разработчиком шрифта для ориентировки пользователя. Например, в шрифте 10-го кегля (размером 10pt) буквы по высоте обычно меньше 10pt и только символы наподобие скобок занимают почти всю эту высоту.

Два шрифта одного и того же размера могут плохо сочетаться визуально друг с другом из-за различий в таких характеристиках, как высота малых (строчных) букв в них (так называемая x-высота), величина нижних выносных элементов в буквах (т. е. той их части, которая находится ниже базовой линии), как, например, в букве q и т. д.

В (LA)TEX'овском мире обычно доступны шрифты, размер которых меняется как степень числа 1.2 (т. е. в геометрической прогрессии) [30, с. 21–22]. Выбор именно такого набора размеров шрифтов объясняется тем, что так проще получить оригинал-макет с подходящим увеличением, чтобы затем его уменьшить фото-способом и получить бóльшее разрешение. Например, если готовится выпуск брошюры с форматом страниц A5, оригинал-макет следует делать с увеличением $1.44 \approx \sqrt{2}$, т. е. на бумаге формата A4. Фотографическое уменьшение страницы, отпечатанной с разрешением 300 dpi (dots per inch — точек на дюйм), типичной для обычного лазерного принтера, позволит получить страницу итогового документа в формате A5 с разрешением уже 432 dpi, т. е. существенно повысить качество печати по сравнению с тем, что дал бы непосредственный вывод в формате A5.

Однако такая схема пересчета размеров шрифтов, используемая в (LA)TEX'е и поддерживаемая системой METAFONT, не является общепринятой в профессиональном издательском мире, где последовательность используемых шрифтов выглядит обычно как 7, 8, 9, 10, 11, 12, 14, 16, 18, 20, 24, 30 и 36 pt. Однако не все шрифты доступны во всех этих размерах, а иногда предлагаются и некоторые дополнительные их размеры, например, выделительные шрифты для больших заголовков, а также шрифты особо малых размеров для верхних и нижних индексов в формулах.

Следует заметить, что использование шрифтов, полученных увеличением или уменьшением из шрифтов некоторых других размеров, дает обычно менее удовлетворительный результат, чем использование шрифтов, специально разработанных

Ten point type is different from magnified five point type

Рис. 7.6. Шрифт 10-го кегля и шрифт, полученный масштабированием из 5-го кегля

для требуемого размера, поскольку операция масштабирования, выполняемая человеческим глазом, имеет нелинейный характер. В частности, символы шрифтов специальной разработки обычно бывают более узкими, чем символы той же высоты, полученные увеличением из некоторого базового шрифта. Поэтому масштабирование шрифтов допустимо только в небольшом диапазоне размеров, и всегда, когда это только возможно, следует применять шрифты, разработанные специально для требуемого размера. Разница между шрифтами, масштабированными до нужного размера и разработанными специально под этот размер, иллюстрируется рис. 7.6.

7.2.4 Схемы кодирования

Для того чтобы получить доступ к тому или иному символу из требуемого шрифта, необходим способ задания этого символа в исходном тексте (файле). Эта проблема может показаться тривиальной, поскольку можно просто нажать клавишу 'A' на клавиатуре, чтобы получить данную букву в формируемом выходном документе. Как быть, однако, если надо получить, например, перевернутый вопросительный знак (¿), открывающий испанские вопросительные предложения? Этот знак вы не найдете на обычной клавиатуре (если только она не адаптирована специально к испанскому языку). В (LA)TEX'е, как и во многих других компьютерных программах, эта проблема решается путем соотнесения символов шрифта с целыми числами (номерами) из диапазона 0–255. Такого рода отображение в целом называется кодовой страницей или схемой кодирования. Чтобы проиллюстрировать, что получается, если взять шрифт с неверной схемой кодирования, приведем здесь первое предложение данного раздела (на английском языке), переписанное для кодовой страницы, не отвечающей данному шрифту[4]:

> То аццесс а чарацтер фром а фонт ин а цомпутер тыпесеттинг програм, ыоу неед а щаы то специфы тхат чарацтер ин тхе инпут соурце.

Полученный результат выглядит забавной головоломкой и ничего общего не имеет с текстом нормального документа.

В (LA)TEX'е имеется несколько способов доступа к символам из текущего шрифта. Наиболее общий из них заключается в том, что в текст документа помещается некоторый ASCII-символ, принадлежащий к разряду отображаемых (видимых), который затем преобразуется в значок соответствующего символа, полученного из текущего шрифта согласно схеме кодирования данного шрифта и ASCII-коду рассматриваемого символа. Для шрифтов, используемых а (LA)TEX'е, в их схемах кодирования позиции букв латинского алфавита совпадают с позициями этих же букв в ASCII-таблице. Вследствие этого, набирая в исходном тексте букву 'a', точно такую же букву получим и в итоговом документе, чего не будет, например, в случае символа '⊣.' Однако для символов, не являющихся буквами,

[4] Исходная фраза выглядела так: «To access a character from a font in a computer typesetting program, you need a way to specify that character in the input source».— *Прим. перев.*

такого рода соответствие совсем не обязательно будет иметь место. Например, для стандартной (LA)TEX'овской схемы кодирования, вводя с клавиатуры '<|>', на выходе получим '¡—¿'. Ряд подробностей, а также внутренние (LA)TEX'овские схемы кодирования приводятся в разд. 7.3.4 и 7.5.5.

7.3 Переключение шрифтов в тексте

При подготовке LATEX'овского документа соответствующие шрифты выбираются автоматически на основе логических ключей, определяющих структуру документа. Например, атрибуты шрифта для заголовка раздела, такие как размер шрифта `large` и жирность `bold`, определяются классом документа и включаются каждый раз, когда используется команда `\section`, так что пользователю самостоятельно менять атрибуты шрифтов приходится относительно редко.

Временами, однако, возникает необходимость непосредственно воздействовать на атрибуты шрифтов. Один из возможных случаев такого рода связан с требованием изменить атрибуты шрифта глобально, т. е. в масштабах документа в целом сменить основной шрифт, взяв для этого шрифт из другого семейства. Зачастую для этого достаточно воспользоваться соответствующим пакетом. Ряд пакетов, предназначенных для решения этой задачи, описывается в разд. 7.5 и 11.9.

Другой пример, когда требуется явно изменить атрибуты шрифта, — визуальное выделение некоторых фрагментов текста документа, например, обозначение специальным образом примеров, аббревиатур, наименований фирм и т. д. Так, в данной книге имена пакетов набираются шрифтом без засечек (sans serif), что достигается командой `\textsf{...}`, содержащей в качестве аргумента имя соответствующего пакета. Более эффективное решение — определить для этой цели новую команду (например, `\Lpack`), поскольку таким способом можно добавлять к документу некоторую добавочную информацию. Такой подход к определению логически различных элементов в виде индивидуальных команд, даже если они и задаются первоначально совершенно одинаково, ценен тем, что впоследствии очень легко при необходимости менять, причем независимо от других, форму представления данного элемента на протяжении всего документа в целом.

Наконец, что немаловажно, может возникнуть потребность перекрыть решение, задаваемое определением соответствующего класса (стиля): скажем, надо набрать таблицу более мелким, чем основной текст, шрифтом, чтобы разместить ее на одной странице. Это желание вполне закономерно, поскольку способность LATEX'а обеспечивать автоматическое форматирование документа заданного класса имеет определенные границы. Часто требуется ручное форматирование, например, указание LATEX'у места, где надо произвести разрыв страницы, особенно на стадии формирования окончательного варианта документа. Однако явное ручное форматирование документа осложняет дальнейшее его использование и является потенциальным источником ошибок. Поэтому надо стремиться как можно меньше вмешиваться в текст документа напрямую, в том числе и с командами шрифтового выделения каких-либо его фрагментов.

7.3.1 Стандартные шрифтовые команды NFSS

Шрифт, используемый в основном тексте документа, именуется «основным шрифтом». Он выбирается автоматически в начале документа, а также в некоторых других ситуациях (сноски, рисунки и т. д.). Для отдельных структурных элементов документа (например, для заголовков) производится автоматическое переключение на шрифт другого вида или размера, в зависимости от класса документа. Иногда, однако, может потребоваться выделить какие-то фрагменты текста, соответственно изменив используемый в них вид шрифта. Эту операцию можно выполнить при помощи описываемых ниже команд.

Большинство команд переключения шрифтов существует в двух разновидностях, а именно, как команды с одним аргументом, такие как `\textbf{...}`, и в декларативной форме, например, `\bfseries`. Декларации не имеют аргументов, они предписывают LaTeX'у выполнить некоторое действие, причем область действия данной команды распространяется до конца текущей группы скобок или окружения. Таким образом, не следует писать чего-то вроде `\bfseries`, поскольку в результате до конца следующего окружения все символы станут полужирными.

Чтобы изменить символ в отдельном слове или в короткой фразе, следует пользоваться командами с одним аргументом, тогда как декларативная форма больше подходит для определения новых окружений или команд. Для более длинных фрагментов документа можно также применять декларативный способ переключения шрифта, записываемый в виде окружения (в начальной и конечной строках этого окружения используется имя соответствующей декларации без начального символа '\'), как это показано в приведенном ниже примере:

Некоторые слова в этом предложении **набраны полужирными** буквами.

```
Некоторые слова в этом предложении
\begin{bfseries}набраны полужирными
\end{bfseries} буквами.
```

Подробное сопоставление командной и декларативной форм переключения шрифтов, анализ их преимуществ и недостатков в различных ситуациях содержатся в разд. 7.3.3.

Основной шрифт документа

Чтобы сменить текущий шрифт документа на тот, который является для него основным, можно воспользоваться командой `\textnormal` или декларацией `\normalfont`. Они используются обычно только в определениях команд или окружений, когда важно обеспечить неизменность результата независимо от того, в каком контексте выполняется данная команда или окружение. Например, команда набора имен команд в данной книге выглядит следующим образом:

```
\newcommand{\Lcs}[1]{{\normalfont\ttfamily\bslash#1}%
                     \index{#1@{\normalfont\ttfamily\bslash#1}}}
```

Это препятствует возникновению ситуаций, когда в некоторых местах текста команда выглядела бы подобно `\this`.

Стандартные семейства шрифтов

По умолчанию LaTeX позволяет работать с тремя семействами шрифтов, которые можно выбирать, используя короткие последовательности команд. В число этих семейств входят: текстовые шрифты с засечками, выбираемые при помощи команды `\textrm`, шрифты без засечек (команда `\textsf`) и шрифт, имитирующий пишущую машинку, обычно это моноширинный шрифт (команда `\texttt`). Декларативная форма этих команд имеет вид `\rmfamily`, `\sffamily` и `\ttfamily` соответственно.

Имена внешних семейств шрифтов, доступных через эти команды, зависят от класса документа, но могут быть изменены при помощи пакетов или в преамбуле документа (см. разд. 7.3.5). По умолчанию при установке LaTeX'а в качестве шрифта с засечками принимается Computer Modern Roman, шрифта без засечек — Computer Modern Sans, а шрифта пишущей машинки — Computer Modern Typewriter. Если вы предполагаете изменить эти установки, то надо позаботиться, чтобы при произвольном смешивании букв из разных шрифтов не возникало визуального «конфликта» между ними. Кроме того, следует позаботиться, чтобы подключаемые внешние шрифты существовали в форме с разрешением, подходящим для используемого устройства вывода.

В данной книге используются семейства шрифтов, перечисленные выше[5].

В большинстве классов документов шрифт с засечками, выбираемый при помощи команды `\textrm`, принимается в качестве основного шрифта документа, так что команда `\textrm` используется не слишком часто. Однако, если разработчик документа выбрал в качестве основного шрифт без засечек, то команда `\textrm` уже будет вызывать смену основного шрифта на альтернативный.

Стандартные насыщенности шрифтов

Еще один атрибут шрифта — это его *насыщенность*. В NFSS насыщенность есть комбинация из двух атрибутов: ширины очка литеры и жирности.

В NFSS имеется две команды для управления насыщенностью шрифта: `\textmd` и `\textbf`, или, в декларативной форме, `\mdseries` и `\bfseries` соответственно. Первая команда позволяет выбрать шрифт нормальной насыщенности со средними (умеренными) значениями ширины и жирности, а вторая включает шрифт полужирного начертания. Фактические значения используемых имен

[5] В оригинале книги в качестве семейства шрифтов с засечками использовалось семейство Lucida Bright, без засечек — Lucida Sans, пишущей машинки — семейство Computer Modern Typewriter. Эти шрифтовые семейства подключались при помощи пакета lucidbrb. Чтобы вместе с этим пакетом использовать семейство Computer Modern Typewriter в качестве шрифта пишущей машинки, дополнительно пришлось переопределить команду `\ttdefault`; см. разд. 7.3.5, из которого можно узнать, как изменить значение имени семейства шрифтов, принимаемого по умолчанию. — *Прим. перев.*

шрифтов будут зависеть от класса документа и значений его настроечных параметров или же от вида используемого пакета переключения шрифтов. По умолчанию, применительно к семейству шрифтов Computer Modern команда `\textbf` используется для переключения на полужирный вариант текущего шрифта, а команда `\textmd` обеспечивает возврат к текущему шрифту, устанавливая для него нормальные ширину и жирность.

Если требуется более тонкое управление насыщенностью шрифта, то лучше определить соответствующие команды высокого уровня. Это можно проделать при помощи низкоуровневой декларации `\fontseries`, описанной в разд. 7.6.1. В ряде пакетов, обеспечивающих доступность в LaTeX'е больших семейств шрифтов, такие команды иногда уже реализованы.

Стандартные начертания шрифтов

Третий атрибут шрифта, который можно менять независимо от остальных — это *начертание* текущего шрифта. По умолчанию для большинства документов принимается прямое начертание шрифта, включаемое командой `\textup` или декларацией `\upshape`.

По-видимому, наиболее важными командами изменения начертания являются `\textit` и `\textsc`, включающие, соответственно, *курсивное начертание* и начертание типа ПРОПИСНЫЕ И КАПИТЕЛЬ. Парные им декларации имеют вид `\itshape` и `\scshape`.

Альтернативной по отношению к команде `\textit` является команда `\textsl` (ее декларативная форма — `\slshape`), включающая *наклонное начертание* шрифта. Чаще всего семейство шрифтов содержит либо курсивный, либо наклонный шрифт, однако в состав семейства Computer Modern Roman входят оба эти варианта.

В том месте, где происходит переключение с наклонного начертания на прямое, символы обычно оказываются слишком близко друг к другу, особенно если последняя наклонная буква имеет верхний выносной элемент. Дополнительный пробел, который следует добавить между этими двумя буквами (последней наклонной и первой прямой), носит название «поправка на курсив». Значения этих поправок зависят от начертания индивидуальных символов в шрифте и по этой причине должны запоминаться в соответствующем `.tfm`-файле. В случае, когда используется командное переключение начертания шрифта, поправка на курсив добавляется автоматически. Если же переключение осуществляется при помощи декларации, данную поправку следует ввести вручную, используя команду `\/` [*Л* **17**], [*Л* **95**]. Для шрифта с прямым начертанием поправка на курсив для символов обычно будет нулевой или очень небольшой. Из этого правила имеются, однако, исключения. (В семействе Computer Modern, чтобы набрать полужирную 'f' в кавычках, надо написать `'{\bfseries f\/}'` или `'\textbf{f}'`, чтобы не получить данную букву с «прилипшей» к ней правой кавычкой, т. е. 'f'.) В наклонных и курсивных шрифтах поправка на курсив обычно положительна, а ее фактическое значение зависит от начертания конкретного символа. Таким обра-

зом, примером правильного использования деклараций управления начертанием шрифта можно считать следующий:

Переключаясь с *курсивного* или *наклонного начертания шрифта* на прямое, следует добавлять *поправку на курсив*, за исключением случая, когда в месте переключения стоит небольшой знак препинания (запятая, например).

```
\raggedright
Переключаясь с {\itshape курсивного\/}
или {\slshape наклонного\/} начертания шрифта
на прямое, следует добавлять
{\itshape поправку на курсив}, за исключением
случая, когда в месте переключения стоит
небольшой знак препинания (запятая, например).
```

При использовании командной формы с одним аргументом вместо декларативной формы переключения начертания шрифтов добавление поправки на курсив осуществляется автоматически. Дальнейшее обсуждение этого вопроса содержится в разд. 7.3.3.

Капитель иногда используется в заголовках или для форматирования имен. В последнем из этих двух случаев можно, например, определить команду `\names`

```
\newcommand{\names}[1]{\textsc{#1}}
```

или, используя декларативное переключение,

```
\newcommand{\names}[1]{{\normalfont\scshape #1}}
```

Первое из этих двух определений обеспечивает простую смену одного начертания другим, тогда как второе вначале осуществляет переустановку всех четырех атрибутов шрифта, исходя из значений этих атрибутов, принятых по умолчанию. Какой из этих двух вариантов применить, зависит от доступного набора шрифтов, а также от вида документа. Если используется семейство шрифтов Computer Modern, то капитель фигурирует только в наборе шрифтов с засечками. В этой ситуации в ряде приложений более предпочтительным оказывается второй вариант, поскольку он будет использовать капитель, даже если текущий контекст был определен командой `\sffamily`, т. е. в качестве текущего используется шрифт без засечек. Первая команда определяет запрос на переключение начертания шрифта в значение «капитель». В ситуации, описанной выше (текущий шрифт относится к семейству Computer Modern Sans), попытка найти вариант шрифта с заданным начертанием завершится, естественно, неудачей. В подобном случае согласно схеме NFSS производится поиск значения начертания по умолчанию, которое и подставляется, приводя в итоге совсем не к тому результату, который требуется. (О таких подстановках более подробно рассказывается в разд. 7.6.3.)

Еще один интересный пример применения декларации `\scshape` — это пример *построения команды для получения аббревиатуры*:

```
\newcommand{\acro}[1]{{\scshape\lowercase{#1}}}
```

Это определение использует TEX'овскую команду-примитив `\lowercase`, которая переводит все буквы текста, составляющего его аргумент (точнее, все символы, которые не принадлежат именам команд), в строчную форму. В результате ис-

пользования команды \acro ее аргумент будет представлен следующим образом: если в нем были прописные буквы, они будут заменены строчными, а затем все они будут превращены в капитель.

В LaTeX'е имеется еще одна команда управления начертанием символов, называемая \emph [$\mathcal{L}$ 16,38], [$\mathcal{A}$ 100]. Данная команда предназначена для шрифтового выделения текстовых фрагментов; ее декларативная форма имеет вид \em. Традиционно выделение слов и других фрагментов в тексте принято выполнять путем использования курсива, если же шрифт основного текста имеет курсивное начертание, то при выделении фрагментов в таком тексте осуществляется переход к прямому шрифту. Команда \emph следует этим соглашениям, активизируя команду \itshape, если шрифт основного текста прямой, и команду \upshape, если шрифт основного текста — наклонный (т.е. располагается в зоне действия команды \itshape или \slshape). Таким образом, пользователю нет необходимости беспокоиться о том, каков основной шрифт текста в данный конкретный момент; чтобы правильно выполнить выделение текстового фрагмента, ему достаточно применить команду \emph или декларацию \em.

Следует быть весьма *осмотрительным, применяя курсивные поправки до и после выделенного текста.* Поэтому для этой цели лучше воспользоваться командой \emph, которая *целиком* берет на себя заботу о поправках на курсив с обеих сторон.

```
{\em Следует быть {\em весьма\/}
осмотрительным, применяя курсивные
поправки до и после выделенного текста}.
Поэтому для этой цели лучше
воспользоваться командой \verb=\emph=,
которая \emph{целиком} берет на себя заботу
о поправках на курсив с обеих сторон.
```

Надо заметить, что использование подчеркивания для выделения текстовых фрагментов считается в издательской практике признаком дурного тона. Подчеркивание применяется сейчас только тогда, когда нет другого способа выделить слово или фрагмент текста вследствие особенностей печатающего устройства, например, пишущей машинки. Тем не менее в разд. 3.1.2 описывается LaTeX'овский пакет, позволяющий модифицировать декларацию \em так, чтобы выделение текстовых фрагментов осуществлялось путем их подчеркивания.

Стандартные размеры шрифтов

В LaTeX'е [$\mathcal{L}$ 115,200], [$\mathcal{A}$ 97] имеется десять деклараций для изменения размера шрифта (см. табл. 7.1). Поскольку обычно изменение размера шрифта производится только в определениях команд, то парные данным декларациям команды с одним аргументом отсутствуют. В NFSS имена этих деклараций остались теми же самыми, но характер выполняемых ими действий несколько изменился. В NFSS команда изменения размера шрифта воздействует только на размер текущего шрифта, остальные же его атрибуты остаются неизменными, тогда как в старой схеме замены шрифтов, используемой в LaTeX 2.09, команда изменения размера шрифта автоматически вызывала переход к основному шрифту документа.

Как в LaTeX2_ε, так и в старом LaTeX'е размер, задаваемый этими командами, зависит от конкретных значений настроечных параметров, использован-

\tiny	Size	\normalsize	Size	\huge	Size
\scriptsize	Size	\large	Size	\Huge	Size
\footnotesize	Size	\Large	Size		
\small	Size	\LARGE	Size		

Таблица 7.1. Стандартные команды изменения размеров шрифтов

ных в классе документа, включая такие необязательные параметры, как размер основного шрифта текста (например, 11pt). В общем случае команда `\normalsize` соответствует размеру основного шрифта документа, а другие команды изменения размеров шрифта образуют упорядоченную последовательность, начиная от `\tiny` для самого маленького шрифта, кончая `\Huge` — для самого большого. Иногда одному и тому же реальному размеру шрифта отвечает более чем одна команда, например, если `\normalsize` задан достаточно большим, команды `\Huge` и `\huge` могут давать один и тот же результат. Однако упомянутый порядок всегда остается неизменным.

К сожалению, в настоящее время в LaTeX'е нет команд для изменения относительных размеров шрифтов, например, нет команд, позволяющих задать размер шрифта на 2pt больший, чем размер текущего шрифта.

7.3.2 Комбинирование стандартных команд управления шрифтами

Как уже отмечалось, стандартные команды и декларации переключения шрифтов можно комбинировать. Результатом такого комбинирования будет выбор шрифта, соответствующего всем заданным атрибутам. Например:

Можно набирать текст **крупным полужирным рубленым шрифтом.**

```
Можно набирать текст
{\sffamily\bfseries\large
крупным полужирным рубленым шрифтом.}
```

В данном случае было выполнено переключение на семейство шрифтов без засечек (`\sffamily`), используемое по умолчанию, затем из этого семейства выбран шрифт полужирного начертания (`\bfseries`), после чего изменен размер шрифта (`\large`); все остальные атрибуты шрифта остались неизменными. Файлы метрик шрифтов (т. е. `.tfm`-файлы) загружаются для всех промежуточных комбинаций значений атрибутов, даже если они и не будут никогда использованы. В приведенном выше примере к таким комбинациям относятся: «шрифт без засечек, нормальной насыщенности, размера 10pt» после выдачи команды `\sffamily`, затем «шрифт без засечек, полужирный, широкий, размера 10 pt», после выдачи команды `\bfseries`, после чего «шрифт без засечек, полужирный, широкий, размера 14pt», который и является нужным шрифтом, т. е. отвечающим заданной комбинации атрибутов. Таким образом, комбинации высокоуровневых команд переключения атрибутов шрифтов вынуждают NFSS загружать кроме требуемого

еще и шрифты, не нужные в данной конкретной ситуации. Обычно это не имеет особого значения и вызывает лишь небольшое снижение скорости обработки исходного текста, когда та или иная комбинация появляется в первый раз. Если же, однако, число подобных комбинаций велико, следует дать их определения в терминах низкоуровневых деклараций переключения шрифтов (см. разд. 7.6).

7.3.3 Сравнение командного и декларативного способов переключения шрифтов

Выше уже был представлен ряд примеров шрифтовых команд с аргументом, при помощи которых производилось переключение шрифтов. Использование таких команд вместо соответствующих им декларативных форм имеет то преимущество, что они отвечают по общему стилю другим LaTeX'овским структурам. Эти команды предназначены для набора небольших текстовых фрагментов, в которых надо изменить семейство, насыщенность или начертание шрифта. В табл. 7.2 показан характер воздействия этих команд на соответствующие текстовые фрагменты. Еще одно преимущество, связанное с применением такого (командного) подхода, заключается в том, что поправка на курсив в требуемых случаях вводится в текст автоматически (по обе стороны от текстового фрагмента, являющегося аргументом команды).

Таким образом, используя командное переключение атрибутов шрифтов, нет необходимости беспокоиться о том, чтобы не забыть добавить поправку на курсив при переключении шрифта. Имеется лишь небольшое число ситуаций, когда промежуток, добавляемый при автоматическом введении поправки на курсив, будет неадекватным. Например, в издательской практике рекомендуется обычно не вводить поправки, если сразу за текстовым фрагментом с измененным атрибутом следует небольшой по размерам знак препинания, в частности, запятая. Поскольку требуемая величина коррекции является в определенной степени делом вкуса, пользователю в данной ситуации предоставляется возможность отменить введение в текст поправки на курсив. Это осуществляется путем перечисления в команде `\nocorrlist`[6] списка символов, перед которыми запрещается применение поправки на курсив. Определение этой команды по умолчанию имеет вид

```
\newcommand{\nocorrlist}{,.}
```

Те символы, которые используются чаще других, лучше поместить в начале списка, поскольку в таком случае обработка текста несколько ускоряется.

Кроме модификации способа введения поправок на курсив в рамках документа в целом, можно осуществлять и локализованное их подавление. Для этой цели существует команда `\nocorr`. Эту команду следует разместить слева, спра-

[6] Любой пакет, меняющий категорию (`\catcode`) символа, задаваемого командой `\nocorrlist`, должен переопределять этот список. В противном случае измененный символ перестает распознаваться алгоритмом подавления поправок на курсив.

	Декларация	*Действие*
`\textrm{...}`	`{\rmfamily...}`	Набирает текст шрифтом семейства roman.
`\textsf{...}`	`{\sffamily...}`	Набирает текст шрифтом семейства sans serif.
`\texttt{...}`	`{\ttfamily...}`	Набирает текст шрифтом семейства `typewriter`.
`\textmd{...}`	`{\mdseries...}`	Набирает текст шрифтом нормальной насыщенности.
`\textbf{...}`	`{\bfseries...}`	Набирает текст шрифтом **полужирной** насыщенности.
`\textup{...}`	`{\upshape...}`	Набирает текст шрифтом прямого начертания.
`\textit{...}`	`{\itshape...}`	Набирает текст шрифтом *курсивного* начертания.
`\textsl{...}`	`{\slshape...}`	Набирает текст шрифтом *наклонного* начертания.
`\textsc{...}`	`{\scshape...}`	Набирает текст шрифтом начертания «КАПИТЕЛЬ».
`\emph{...}`	`{\em...}`	Набирает *выделенный* текст.
`\textnormal{..}`	`{\normalfont..}`	Набирает текст основным шрифтом документа.

Таблица 7.2. Команды и декларации переключения шрифтов

Все команды переключения шрифтов с аргументом начинаются с `\text...` (за исключением команды `\emph`), чтобы подчеркнуть их ориентированность на использование в обычном тексте, а также для облегчения их запоминания. При необходимости они обеспечивают автоматическое добавление поправки на курсив как перед началом текстового фрагмента, являющегося аргументом соответствующей команды, так и после его окончания.

ва или внутри аргумента команды `\text...` в зависимости от того, где именно требуется подавить поправку на курсив.

При использовании высокоуровневых команд NFSS требуемое *управление поправками на курсив осуществляется автоматически.* Только *в редких случаях* приходится помогать LaTeX'у, добавляя в текст команду `\nocorr`.

```
\emph{При использовании высокоуровневых
команд \NFSS{} \emph{требуемое} управление
поправками на курсив осуществляется
автоматически}. Только \emph{в редких
случаях} приходится помогать \LaTeX'у,
добавляя в текст команду \verb=\nocorr=.
```

Декларативная форма команд переключения атрибутов шрифтов оказывается более подходящей в случаях, когда пользователь формирует собственные команды или окружения.

- **Это окружение порождает элементы перечня, набранные полужирным шрифтом.**
- **Оно определено при помощи таких средств, как LaTeX'овское окружение itemize и декларации NFSS.**

```
\exampleoffset{-24pt}
\newenvironment{bfitemize}
  {\begin{itemize}\normalfont\bfseries}
  {\end{itemize}}
\begin{bfitemize}
\item Это окружение порождает элементы
      перечня, набранные полужирным шрифтом.
\item Оно определено при помощи таких
      средств, как \LaTeX'овское окружение
      \texttt{itemize} и декларации NFSS.
\end{bfitemize}
```

7.3.4 Доступ ко всем литерам шрифта

Для ряда литер шрифта возможность прямого ввода с клавиатуры отсутствует. Поэтому часть подобных символов доступна лишь посредством имен-команд [$\mathcal{L}$ **40**], [$\mathcal{A}$**93**], подобных `\ss` или `\AE`, дающих на печати соответственно 'ß' и 'Æ'. Другие символы подобного рода генерируются неявно, когда в тексте встречаются соответствующие последовательности букв (это встроенное свойство шрифтов), например, `\ffi` , порождающая 'ffi' и `\---` , дающая «длинное» тире '—' в стандартных TEX'овских шрифтах.

Кроме того, имеется команда `\symbol` [$\mathcal{L}$ **200**]-, обеспечивающая доступ к любой литере шрифта на основе информации о ее номере в текущей схеме кодирования (этот номер используется в качестве аргумента команды `\symbol`), причем номер этот может быть десятичным, восьмеричным (тогда ему предшествует одинарная кавычка (')) или шестнадцатеричным (с предшествующей двойной кавычкой (")).

In the Cork font encoding (T1), characters like Þ, §, and ␣ are included and can be accessed with the `\symbol` command.

```
In the Cork font encoding (\texttt{T1}),
characters like \symbol{"DE}, \symbol{'237},
and \symbol{32} are included and can be
accessed with the \verb=\symbol= command.
```

Номера символов для соответствующего шрифта могут быть получены при помощи программы `nfssfont.tex`, описанной в разд. 7.5.5.

Для обеспечения более простого доступа к цифрам, стилизованным «под старину», например, 1982, в NFSS есть команда `\oldstylenums`, которую можно использовать как в текстовой части документа, так и при наборе формул. Ее аргументом является последовательность цифр, которые необходимо набирать в невыровненном виде. При использовании данной команды в текстовом режиме пробелы в аргументе остаются на своих местах, но другие литеры, отличные от цифр, вводить в состав аргумента нельзя, поскольку результат непредсказуем.

7.3.5 Изменение значений по умолчанию для атрибутов текстовых шрифтов

Чтобы упростить изменение внешнего вида документа в целом, в NFSS предусмотрен набор встроенных параметров-ключей, модифицирующих поведение высокоуровневых команд переключения шрифтов, обсуждавшихся в предыдущих разделах. Эти ключи показаны в табл. 7.3.

Значения этих ключей можно устанавливать либо в пакетах, подключаемых к документу, либо в преамбуле документа, используя команду `\renewcommand`. Подходящие значения для этих параметров можно определить из таблиц атрибутов шрифтов, приводимых в данной главе.

Параметр	*Значение по умолчанию*	*Описание*
`\encodingdefault`	`OT1`	Схема кодирования для основного шрифта.
`\familydefault`	`\rmdefault`	Семейство, соответствующее основному шрифту.
`\seriesdefault`	`m`	Насыщенность основного шрифта.
`\shapedefault`	`n`	Начертание основного шрифта.
`\rmdefault`	`cmr`	Семейство, выбираемое `\rmfamily` и `\textrm`.
`\sfdefault`	`cmss`	Семейство, выбираемое `\sffamily` и `\textsf`.
`\ttdefault`	`cmtt`	Семейство, выбираемое `\ttfamily` и `\texttt`.
`\bfdefault`	`bx`	Насыщенность, выбираемая `\bfseries` и `\textbf`.
`\mddefault`	`m`	Насыщенность, выбираемая `\mdseries` и `\textmd`.
`\itdefault`	`it`	Начертание, выбираемое `\itshape` и `\textit`.
`\sldefault`	`sl`	Начертание, выбираемое `\slshape` и `\textsl`.
`\scdefault`	`sc`	Начертание, выбираемое `\scshape` и `\textsc`.
`\updefault`	`n`	Начертание, выбираемое `\upshape` и `\textup`.

Таблица 7.3. Параметры управления атрибутами текстовых шрифтов

Например, если записать в преамбуле

```
\renewcommand{\familydefault}{cmss}
```

то документ в целом будет набираться шрифтом без засечек, т. е. из семейства Computer Modern Sans, поскольку данная команда переопределяет значение атрибута «семейство» для шрифта, используемого NFSS в качестве основного шрифта документа. Точнее, вид основного шрифта документа задается значениями ключей `\encodingdefault`, `\familydefault`, `\seriesdefault` и `\shapedefault`. Формируя значения этих ключей, надо убедиться, что комбинации их имеют соответствия среди шрифтов, реально подключенных через NFSS.

Значение по умолчанию для ключа `\encodingdefault` (схема кодирования) равно `OT1`. Это означает, что NFSS исходит из предположения, что большинство используемых шрифтов базируются на первоначальной TEX'овской кодировке. Ожидается, что будет введен в действие новый стандарт, получивший наименование «корковская кодировка» (Cork encoding), рассматриваемый в разд. 7.5.1, который заменит первоначальную TEX'овскую кодировку. Тогда значение по умолчанию для ключа `\encodingdefault` изменится на `T1`. Более подробное рассмотрение вопросов, связанных со схемами кодирования, содержится в разд. 7.5.5

Другой пример, относящийся на этот раз к команде изменения насыщенности шрифта, связан с изменением значения ключа `\bfdefault` с `bx` на `b`, чтобы по команде `\bfseries` происходило переключение шрифта не на **bold extended**, как это имеет место по умолчанию в семействе Computer Modern, а на нормальный (по ширине) **bold**. В подобном переопределении, однако, есть известный риск, поскольку в Computer Modern требуемый шрифт (нормальный полужирный) имеется только в семействе Computer Modern Roman, а в семействах Computer Modern Typewriter и Computer Modern Sans присутствует только широкий полужирный шрифт. Таким образом, если не прибегать к дополнительной настройке, в данной ситуации требование переключиться на полужирный шрифт без засечек (т. е. `\sffamily\bfseries`) заставит NFSS прибегнуть к подстановке и в конце концов будет выбран шрифт нормальной насыщенности вместо затребованной полужирной. (Эти действия можно исключить, как объясняется в разд. 7.7.1, если указать заранее, что полужирные широкие варианты шрифтов из семейства с засечками следует использовать в качестве подстановок для полужирных нормальных шрифтов.)

Пример, в котором показано изменение значений по умолчанию для некоторых атрибутов шрифтов, дается в главе, посвященной использованию средств языка PostScript (разд. 11.9.6).

Начальная установка параметра `\familydefault` означает, что если изменить только значение ключа `\rmdefault`, новое значение получит и `\familydefault`, хотя явно такая команда и не выдавалась. Однако изменение только `\rmdefault` на значение ключа `\familydefault` влияния не окажет.

7.3.6 Шрифтовые команды LaTeX 2.09

Двухбуквенные команды переключения шрифтов, используемые в LaTeX 2.09, подобные команде `\bf`, в LaTeX2_ε теперь непосредственно не определены: их определения содержатся в файлах классов LaTeX2_ε. По соображениям совместимости стандартные классы включают определения двухбуквенных команд, эмулирующих их действие в LaTeX 2.09 . Вполне допустимо, однако, переопределить их в каком-либо пакете или же в преамбуле документа, исходя из собственных вкусов и предпочтений, чего нельзя сделать с базовыми командами NFSS вроде `\bfseries`.

7.4 Переключение шрифтов в формулах

В отличие от переключения шрифтов в тексте при наборе математических формул автоматическое изменение шрифтов в общем случае нежелательно. Для математика начертание индивидуального символа несет, как правило, специфическую информацию. Например, буквы прямого полужирного начертания могут обозначать векторы. Если начертание символов в формуле изменилось вследствие изменения контекста, в котором находится данная формула, результат может получиться некорректным. По этим причинам управление шрифтами в математических формулах организовано способом, отличающимся от того, какой применяется в «чистом» тексте.

Литеры, используемые в формулах, можно разделить на два класса: математические символы и алфавитно-цифровые символы (буквы и цифры). В соответствии со своей внутренней организацией (LA)TeX различает восемь видов математических символов (это необходимо для правильной расстановки пробелов и определения их размеров), но с точки зрения пользователя вполне достаточно упомянутого выше разделения всех символов в формуле на два класса.

Некоторые символы (наподобие знака равенства =) можно ввести непосредственно с клавиатуры, однако подавляющее их большинство вводится при помощи управляющих последовательностей — например, `\leq` для $\leq$. Символы из второй группы (алфавитно-цифровые) вводятся, как обычно, с использованием клавиатуры.

В стандартный набор средств (LA)TeX'а входят свыше 200 предопределенных символов, что дает возможность пользователю построить почти любую формулу. Эти символы рассеяны по различным шрифтам, но доступ к ним организован таким образом, что пользователь не обязан знать о такого рода деталях внутренней организации (LA)TeX'а. Если возникнет необходимость, аналогичным образом можно организовать доступ к символам и из других шрифтов (см. разд. 7.7.6).

Наиболее существенное различие между символами математическими и алфавитно-цифровыми состоит в том, что математические символы всегда имеют одно и то же графическое представление в пределах одной формулы, тогда как внешний вид алфавитно-цифровых символов в формуле пользователь может менять. Будем называть команды, задающие тот или иной вид алфавитно-

цифровым символам в формуле, «идентификаторами математических алфавитов», а шрифты, которые они идентифицируют, — «математическими алфавитами». Эти идентификаторы математических шрифтов не зависят от шрифтовых команд окружающего текста, так что формула не изменится, если ее поместить (например) в окружение типа «теорема», текст которой по умолчанию набирается курсивом. Такое поведение формулы очень важно, поскольку начертание символов в математической формуле несет определенную смысловую нагрузку, которая не должна меняться при перемещении формулы из одного места документа в другое.

Пользователи, применяющие старый метод переключения шрифтов, с удивлением обнаружат, что команды, подобные `\bfseries`, употреблять в формулах теперь нельзя. Это та цена, которую приходится платить за повышение гибкости механизма выбора атрибутов шрифтов, поскольку подобная гибкость недопустима в формуле. Вследствие этого для изменения вида некоторых алфавитно-цифровых символов в сложных формулах приходится пользоваться другим механизмом управления — идентификаторами математических алфавитов.

7.4.1 Специальные идентификаторы математических алфавитов

Одного алфавита даже в совокупности с огромным числом математических символов математикам для выражения своих мыслей недостаточно. Они пытаются использовать для этих целей любые доступные средства набора. Кроме обычной практики применения иностранных алфавитов, в частности, греческого (символы которого получают специальными командами — `\alpha`, `\beta` и т. д.), часто используются и буквы латинского алфавита в различных начертаниях, например, рубленые буквы — для обозначения матриц, полужирные из шрифта с засечками — для векторов, готические — для обозначения групп, идеалов и полей. Буквы «рукописного» начертания применяются часто для обозначения множеств. Такого рода соглашений очень много, и что еще более существенно, сами эти соглашения меняются от одной дисциплины к другой. По этой причине в NFSS вместо жесткой фиксации заранее предопределенного набора идентификаторов математических алфавитов допускается объявление новых идентификаторов и связывание их с некоторой группой начертаний шрифтов. Эти идентификаторы представляют собой специальные команды, предназначенные для использования в формулах. Каждая такая команда воздействует на алфавитно-цифровые символы, содержащиеся в аргументе; начертание математических символов изменять таким способом нельзя. Эти команды могут быть так организованы, что в разных формулах они будут давать различный результат (начертание алфавитно-цифрового символа); данный вопрос рассматривается в разд. 7.4.3. Однако в пределах одной формулы результат действия этих команд будет одним и тем же вне зависимости от контекста, в котором находится формула.

Команда	*Пример*	
\mathcal	`$\mathcal{A}=a$`	$\mathcal{A} = a$
\mathrm	`$\mathrm{max}_i$`	max_i
\mathbf	`$\sum x = \mathbf{v}$`	$\sum x = \mathbf{v}$
\mathsf	`$\mathsf{G}_1^2$`	G_1^2
\mathtt	`$\mathtt{W}(a)$`	$\mathtt{W}(a)$
\mathnormal	`$\mathnormal{abc}=abc$`	$\mathnormal{abc} = abc$
\mathit	`$differ\neq\mathit{differ}$`	$differ \neq \mathit{differ}$

Таблица 7.4. Набор идентификаторов математических алфавитов, встроенных в NFSS

Как видно из последних двух строк таблицы, буквы, используемые в формулах по умолчанию, берутся из математического алфавита \mathnormal. Пример в последней строке показывает, что в последовательностях букв, образованных \mathnormal и \mathit, по-разному формируются пробелы; команда \mathit может быть использована для задания имен переменных, представляющих собой слова естественного языка, что характерно для некоторых дисциплин.

Предопределенные идентификаторы математических алфавитов

Пользователь может, конечно, сам вводить новые идентификаторы математических алфавитов в соответствии со своими потребностями, но в NFSS уже имеется встроенный набор заранее определенных идентификаторов (см. табл. 7.4).

В NFSS идентификаторы математических алфавитов являются командами с одним аргументом, представляющим собой обычно единичную букву или изолированное слово. Над этим аргументом производится действие, заключающееся в придании ему определенного начертания, соответствующего заданному шрифту, например,

Следовательно, G можно вычислить как

$$\mathsf{G} = \mathcal{A} + \sum_{i=1}^{n} \mathcal{B}_i \qquad (7.1)$$

```
Следовательно, $\mathsf{G}$ можно
вычислить как
\begin{equation}
  \mathsf{G} = \mathcal{A} +
     \sum_{i=1}^{n} \mathcal{B}_{i}
\end{equation}
```

Эта процедура отличается от подхода к использованию шрифтовых команд в LaTeX 2.09, где команды типа \rm ведут к изменению шрифта (..{\rm A}..). Для наиболее важных двухбуквеных команд изменения атрибутов шрифтов \rm, \sf, \bf, \it и \tt в LaTeX 2_ε пока еще осуществляется поддержка старого синтаксиса. Это выполнено путем включения соответствующих средств в описания стандартных классов документов. LaTeX 2_ε можно заставить использовать и другие команды этого вида, если подключить пакет oldlfont (см. разд. 7.5.5). Однако от использования этих команд во вновь создаваемых документах лучше воздерживаться.

Как уже отмечалось, еще одно различие между старой схемой переключения шрифтов и NFSS состоит в том, что декларации для шрифтов, используемых в основном тексте документа, нельзя более использовать в формулах, поскольку эти декларации выполняют только изменение отдельных атрибутов текущего шрифта вместо требуемого переключения на некоторый специальный шрифт. Таким образом, если в формуле написать `\bfseries..` вместо `\mathbf...`, то NFSS выдаст сообщение об ошибке.

При формировании имен для идентификаторов математических алфавитов предпочтение отдается наглядности (а не простоте их ввода с клавиатуры), вследствие чего все такие идентификаторы начинаются с `\math`. Поэтому, если подобные команды должны использоваться в документе многократно, имеет смысл в преамбуле дать определения их кратких вариантов, например,

```
\newcommand{\mrm}{\mathrm}
```

Может вызвать удивление способ, которым в некоторых случаях назначается математический алфавит по умолчанию (т. е. из какого алфавита будут взяты символы в случае, когда имя математического алфавита не указано явно), например, в формуле `$x = 123$`. Дело здесь заключается в том, что единый математический алфавит, из которого брались бы символы по умолчанию, отсутствует. (LA)TEX настроен так, что буквы и цифры выбираются по умолчанию из различных алфавитов, если только пользователь не указал явным образом обратное; данный подход иллюстрируется следующим примером:

$x = 12345$ (7.2)

x = 12345 (7.3)

$x = 12345$ (7.4)

```
\begin{align}
  x              &= 12345                 \\
 \mathrm{x}      &= \mathrm{12345}        \\
 \mathnormal{x}  &= \mathnormal{12345}
\end{align}
```

Как видно из этого примера, команда `\mathrm` не оказывает никакого воздействия на цифры, а команда `\mathnormal` не влияет на начертание букв. Значение начертания по умолчанию для цифр при стандартной настройке LaTeX'а определяется математическим алфавитом, связанным с командой `\mathrm`, а для букв — с командой `\mathnormal`. Этими настройками можно управлять при помощи команды `\DeclareMathSymbol`, описываемой в разд. 7.7.6.

Определение новых идентификаторов математических алфавитов

Новые идентификаторы математических алфавитов задаются при помощи декларации `\DeclareMathAlphabet`. Предположим, что в качестве математического алфавита будет взят наклонный шрифт без засечек. Первое, что надо задать — это имя, под которым данный шрифт будет подключен как математический алфавит, например `\mathsfsl`. Затем, основываясь на материале таблиц значений шрифтовых атрибутов (они расположены начиная со с. 209), надо подобрать подходящую группу начертаний шрифтов. Можно выбрать, например, семейство шрифтов Computer Modern Sans нормальной насыщенности в прямом и наклонном на-

чертании. Если решено использовать шрифт наклонного начертания, об этом информируется NFSS путем включения в преамбулу документа следующей команды

```
\DeclareMathAlphabet{\mathsfsl}{OT1}{cmss}{m}{sl}
```

Более формально, команда `\DeclareMathAlphabet` кроме определяемого идентификатора математического алфавита имеет еще четыре аргумента: схему кодирования, семейство, насыщенность и начертание используемого шрифта. Теперь определенный таким образом идентификатор доступен для использования в формуле: команда с этим именем будет всегда переключать (для своего аргумента) шрифт на Computer Modern, без засечек, наклонный, нормальной насыщенности.

Покажем это на примере формулы

$$\sum A_i = a \tan \beta \qquad (7.5)$$

```
Покажем это на примере формулы
\begin{equation}
  \sum \mathsfsl{A}_{i} = a \tan \beta
\end{equation}
```

Можно также в подключаемом пакете или в преамбуле переопределить существующий идентификатор математического алфавита. Например, декларация

```
\DeclareMathAlphabet{\mathsf}{OT1}{pss}{m}{n}
```

перекроет значения по умолчанию для идентификатора математического алфавита `\mathsf`. После такого переопределения команда `\mathsf` будет в формулах переключаться на шрифт Pandora Sans.

7.4.2 Текстовые шрифтовые команды при наборе математических формул

Как уже отмечалось выше, команды декларативного типа для переключения шрифтов, наподобие `\rmfamily`, использовать при наборе математических формул нельзя. Это не относится, однако, к шрифтовым командам с одним аргументом, например, `\textrm`, которые можно применять при наборе как основного текста, так и формул. Такие команды можно применять для временного выхода из математического контекста, чтобы внутри формулы набрать некоторый текстовый фрагмент, который логически связан с текстом, окружающим формулу. Следует помнить, что атрибуты шрифта, используемого в данном фрагменте, зависят от того окружения, в котором находится рассматриваемая формула, а именно, значения атрибутов (кодировка, семейство, насыщенность, начертание) для шрифта текста, окружающего формулу, и для текстового фрагмента в формуле полностью совпадают, как это можно видеть из следующего примера:

Результат имеет вид

$$x = 10 \textbf{ and thus } y = 12$$

```
\sffamily Результат имеет вид
\[  x = 10
   \textbf{ and thus }
    y = 12 \]
```

Как видно из этого примера, шрифт основного текста и текстового фрагмента в формуле принадлежат к одному семейству (Sans), а насыщенность для шрифта

во фрагменте изменена с нормальной на полужирную путем использования команды \textbf. Более полезной может оказаться команда \text, определенная в пакете amstext, которая берет значения атрибутов текущего шрифта документа (см. разд. 8.3.12) и их не изменяет.

7.4.3 Версии математических формул

NFSS позволяет не только вносить локальные изменения в формулу, пользуясь идентификаторами математических алфавитов, но и влиять на внешний вид формулы в целом. Формулы набираются в некоторой «математической версии», переключение между различными версиями осуществляется вне математического режима при помощи команды \mathversion, которая и меняет в целом облик следующих за ней формул.

NFSS известны две версии математических формул, называемых «нормальная» и «полужирная». В некоторых специальных пакетах содержатся определения и других версий. Например, пакет concrete (см. разд. 7.5.1) подключает математическую версию, получившую наименование «эйлерова», позволяющую набирать формулы в том стиле, который был применен в книге *Конкретная математика* [21].

Нормальная математическая версия, подключаемая командой \mathversion{normal}, устанавливается по умолчанию. Во втором варианте (полужирная математическая версия) для набора формул используются алфавитно-цифровые и специальные символы с большей жирностью, чем в нормальной версии. В примере, следующем ниже, одна и та же формула показана вначале в нормальной версии, а затем — в полужирной[7]:

$$\sum_{j=1}^{z} j = \frac{z(z+1)}{2} \qquad (7.6)$$

```
\begin{equation}
  \sum_{j=1}^{z} j = \frac{z(z+1)}{2}
\end{equation}
\mathversion{bold}
\begin{equation}
  \sum_{j=1}^{z} j = \frac{z(z+1)}{2}
\end{equation}
```

$$\sum_{j=1}^{z} j = \frac{z(z+1)}{2} \qquad (7.7)$$

Использование команды \mathversion может быть оправданным в ряде ситуаций, таких, как печать заголовка, однако следует помнить, что меняется вид формулы в целом, а вместе с видом, возможно, и смысл формулы, если в данном контексте различие между символами нормальной и полужирной насыщенности играет какую-то роль. Если есть необходимость перейти к полужирному начертанию лишь для части формулы, пользоваться командой \mathversion не надо. Вместо этого следует применить команду \mathbf для требуемых символов, и/или использовать команду \boldsymbol из пакета amsbsy (см. разд. 8.2.1 и табл. 8.1).

[7] Для сохранения преемственности в NFSS имеются две дополнительные команды переключения между стандартными математическими версиями: \boldmath и \unboldmath.

Если командой \mathversion произведена смена математической версии, NFSS просматривает свои внутренние таблицы, чтобы найти, где размещаются все те символы, которые необходимы для новой математической версии. Это позволяет изменить все или некоторые из идентификаторов математических алфавитов и связать их для данной версии с другими шрифтами.

Но что произойдет с теми идентификаторами математических алфавитов, которые были определены дополнительно, например, \mathsfsl в одном из примеров, приведенных выше? Как только такой идентификатор задан командой \DeclareMathAlphabet, он остается неизменным в любой математической версии.

Если же такой идентификатор должен давать разные результаты в различающихся математических версиях, надо информировать об этом NFSS при помощи команды \SetMathAlphabet. Например, команда \mathsf по умолчанию определена в следующем виде:

```
\DeclareMathAlphabet{\mathsf}{OT1}{cmss}{m}{n}
\SetMathAlphabet{\mathsf}{bold}{OT1}{cmss}{bx}{n}
```

Из первой строки следует, что по умолчанию командой \mathsf во всех математических версиях используется семейство шрифтов Computer Modern Sans нормальной жирности (насыщенности). Вторая строка показывает, что вместо этого шрифта в полужирной математической версии будет использован шрифт семейства Computer Modern Sans широкого полужирного начертания.

Из предшествующего примера видно, что команда \SetMathAlphabet имеет шесть аргументов: первый — это имя идентификатора математического алфавита, второй — имя математической версии, в которой надо осуществить корректировку атрибутов, а оставшиеся четыре — это кодировка, семейство, насыщенность и начертание, которые требуется придать скорректированной версии.

Как отмечалось выше, можно переопределить существующий идентификатор математического алфавита, используя для этой цели команду \DeclareMathAlphabet. Если выполнить эту операцию, все предыдущие декларации \SetMathAlphabet для данного идентификатора будут удалены из внутренних таблиц NFSS, что приведет к изменению характера ее работы во всех математических версиях до тех пор, пока не будут добавлены новые декларации \SetMathAlphabet для этой команды.

7.5 Стандартные пакеты

Совместно с NFSS распространяется ряд пакетов, которые будут рассмотрены в данном разделе. Много других можно получить из различных электронных архивов или от организаций, осуществляющих поддержку и сопровождение TEX'а; информацию о них можно найти в приложении B. Все семейства шрифтов, упоминаемых в данном разделе, доступны свободно, в виде исходных METAFONT'овских текстов.

Семейство	Насыщенность	Начертание	Пример
Computer Modern Roman (T1, OT1)			
cmr	m	n, it, sl, sc, u	COMPUTER ROMAN SMALL CAPS
cmr	bx	n, it, sl	***Comp. Mod. Roman bold extended italic***
cmr	b	n	**Computer Modern Roman bold upright**
Computer Modern Sans (T1, OT1)			
cmss	m	n, sl	*Computer Modern Sans slanted*
cmss	bx	n	**Computer Modern Sans bold extended**
cmss	sbc	n	Computer Modern Sans semibold condensed
Computer Modern Typewriter (T1, OT1)			
cmtt	m	n, it, sl, sc	*Computer Modern Typewriter italic*
Computer Modern Fibonacci (T1, OT1)			
cmfib	m	n	Computer Modern Fibonacci
Computer Modern Funny Roman (T1, OT1)			
cmfr	m	n, it	Computer Modern Funny Roman
Computer Modern Dunhill (T1, OT1)			
cmdh	m	n	Computer Modern Dunhill

Таблица 7.5. Классификация шрифтов Computer Modern, принятая в NFSS

7.5.1 Добавление новых текстовых шрифтов

Одно из важных достоинств NFSS — простота интеграции новых шрифтов, которые можно использовать для набора основного текста документа. Кроме шрифтовых семейств Computer Modern, подключаемых по умолчанию, весьма просто обеспечить использование и других семейств шрифтов, загрузив соответствующие пакеты командами, помещенными за командой `\documentclass`. Разумеется, чтобы нормально обработать и затем распечатать документ с подключенными новыми шрифтами, надо располагать соответствующими шрифтовыми файлами (например, `.tfm`-файлами и `.pk`-файлами).

DC-шрифты

На конференции пользователей TEX'а (TEX Users conference), состоявшейся в 1990 г. в Корке, была выработана стандартная схема кодирования для текстовых

Семейство	Насыщенность	Начертание	Пример
Concrete Roman (T1, OT1)			
ccr	m	n, it, sl, sc	Concrete Roman medium
ccr	c	sl	*Concrete Roman condensed slanted*
Concrete Math (OML)			
ccm	m, c	it	*Concrete Math.* $\alpha\ \Omega$

Таблица 7.6. Семейство шрифтов Concrete

шрифтов (T1), включающая большое число диакритических знаков (см. табл. 9.1) и обеспечивающая набор документов на более чем 30 языках. В университете г. Бочам (University of Bochum) под руководством Норберта Шварца была осуществлена переработка семейств шрифтов Computer Modern с добавлением ряда новых символов и приведением шрифтов, входящих в эти семейства, в соответствие с новой кодировкой. В настоящее время эти шрифты распространяются под наименованием DC-шрифтов, а их окончательный вариант называется EC-шрифтами. Настоятельно рекомендуется использовать шрифты из этих семейств и мы надеемся, что и большая часть других шрифтовых семейств будет в скором времени доступна в новой (корковской) кодировке[8].

Если после команды `\documentclass` поставить команду загрузки пакета `t1enc`, корковская схема кодирования (T1) станет кодировкой, используемой по умолчанию. Соответственно, в данном случае вместо набора семейств шрифтов Computer Modern в кодировке (OT1) LaTeX будет использовать DC-шрифты. С корковской схемой кодирования могут применяться и PostScript-шрифты. Этот вопрос рассматривается в разд. 11.10.

Шрифты семейства Concrete

Для книги *Конкретная математика* [21] Дональд Кнут разработал новый набор текстовых шрифтов Concrete Roman, который вместе с соответствующими эйлеровыми математическими шрифтами, созданными Германом Цапфом, образуют семейство Concrete Roman, полученное на основе исходных METAFONT'овских текстов путем варьирования ряда параметров в них. Основываясь на результатах для DC-шрифтов, довольно просто получить шрифты Concrete Roman в корковской кодировке; соответствующий набор METAFONT'овских драйверов является составной частью дистрибутива пакета NFSS. Шрифты, входящие в семейство Concrete Roman, показаны в табл. 7.6.

[8] С 1997 г. DC-шрифты заменены на EC-шрифты, релиз 1.0, поддерживающие кодировку T2. Для кириллических шрифтов на этой же основе существует кодировка T2, включенная в релиз 1999 г. как бета-версия.— *Прим. ред.*

Семейство	Насыщенность	Начертание	Пример
Pandora Roman (OT1)			
panr	m	n, sl	Pandora Roman medium
panr	b	n	**Pandora Roman bold**
Pandora Sans (OT1)			
pss	m	n, sl	*Pandora Sans medium slanted*
pss	b	n	**Pandora Sans bold**

Таблица 7.7. Семейство шрифтов Pandora

Чтобы иметь возможность пользоваться шрифтами семейства Concrete Roman в основном тексте документа, лучше всего воспользоваться пакетом beton (автор — Франк Йенсен), который, кроме всего прочего, реализует ряд небольших, но важных модификаций общего стиля документа, таких, например, как некоторое увеличение `\baselineskip`.

Имеется также пакет concrete, который помимо определения семейства Concrete Roman как основного шрифта документа содержит также новый вариант математического шрифта, именуемый euler. Этот файл вместе с документацией для него можно использовать в качестве образца, если возникнет необходимость формирования собственной комбинации из текстового и математического шрифтов.

Шрифты семейства Pandora

Назин Билавала разработал два семейства шрифтов: Pandora Roman и Pandora Sans [7, 8]. В табл. 7.7 представлен полный перечень шрифтов, входящих в эти семейства. К сожалению, оба этих семейства шрифтов имеются пока только в традиционной TEX'овской кодировке (OT1). Чтобы использовать шрифты Pandora в качестве основных для текста документа, можно воспользоваться пакетом pandora, загрузив его после команды `\documentclass`.

Готические шрифты

Существует набор прекрасных готических шрифтов, созданных Яннисом Хараламбусом [80], включающий шрифты Gothisch, also called Textur (Gothisch, называемых также Textur), Schwabacher (Schwabacher) и Fraktur (Fraktur). В этот набор входит также шрифт для буквиц; отдельные буквицы показаны на рис. 7.7.

Перечисленными шрифтами можно воспользоваться, если после команды `\documentclass` поставить команду загрузки пакета oldgerm. Для переключения на соответствующие семейства шрифтов пакет oldgerm определяет команды `\gothfamily`, `\frakfamily` и `\swabfamily`. Поскольку каждое из этих семейств со-



Рис. 7.7. Шрифт `yinit` Янниса Хараламбуса для буквиц

У этого шрифта следующие значения атрибутов: семейство — `yinit`, насыщенность — `m`, начертание — `n`, схема кодирования — `U`.

Семейство	Насыщенность	Начертание	Пример
Gothic (U)			
ygoth	m	n	Yannif: Gothic
Fraktur (U)			
yfrak	m	n	Yanniſ: Fraktur
Schwabacher (U)			
yswab	m	n	Yanniſ: Schwabacher

Таблица 7.8. Семейства готических текстовых шрифтов

Поскольку эти шрифты содержат много лигатур, необходимых при наборе основного текста документа, они не следуют стандартным схемам кодирования и в NFSS включаются как имеющие кодировку `U`.

держит шрифт в единственном начертании для одного значения насыщенности, команды вроде `\bfseries` или `\itshape` в данной ситуации не работают. Команды изменения размеров шрифтов можно применять обычным порядком. Дополнительно, пакет oldgerm определяет шрифтовые команды с одним аргументом, т. е. `\textgoth`, `\textfrak` и `\textswab`.

7.5.2 Подключение новых математических шрифтов

Эйлеровы шрифты

Как отмечалось выше, Герман Цапф разработал семейство отличных математических шрифтов (символы прямого начертания с «рукописным оттенком»), названных им эйлеровыми шрифтами в честь знаменитого математика Леонарда Эйлера. Этими шрифтами можно воспользоваться при помощи пакета euler, который не вводит какого-то дополнительного набора математических символов, а

Семейство	Насыщенность	Начертание	Пример
Euler Roman (U)			
eur	m	n	Euler Roman medium
eur	b	n	**Euler Roman bold**
Euler Script (U)			
eus	m	n	*EULER SCRIPT*
Euler Fraktur (U)			
euf	m	n	Euler Fraktur

Таблица 7.9. Эйлеровы семейства математических шрифтов

К сожалению, эйлеровы семейства математических шрифтов в их существующем варианте имеют кодировку, отличающуюся от всех других схем кодирования для математических шрифтов. Поэтому при использовании эйлеровых шрифтов необходимо переопределение ряда математических символов; эта операция выполняется пакетом euler. По этой причине считается, что эйлеровы шрифты имеют кодировку U, если только она не будет переопределена.

переопределяет текущие математические шрифты в нормальном и полужирном начертании таким образом, чтобы в их качестве использовались соответствующие эйлеровы шрифты.

Если из всех эйлеровых шрифтов необходим только шрифт *SCRIPT*, можно обеспечить его подключение при помощи пакета euscript, который подключает данный шрифт под именем `\EuScript`.

Использование шрифта Euler Fraktur в формулах обеспечивается пакетом eufrak, который подключает математический алфавит `\EuFrak`. Полностью набор эйлеровых шрифтов приведен в табл. 7.9.

latexsym — подключение LaTeX'овских шрифтов

В базовом варианте NFSS исключены определения следующих одиннадцати математических символов, которые были доступны в LaTeX 2.09:

`\mho` ℧ `\Join` ⋈ `\Box` □ `\Diamond` ◇ `\leadsto` ⇝ `\sqsubset` ⊏
`\sqsupset` ⊐ `\lhd` ◁ `\unlhd` ⊴ `\rhd` ▷ `\unrhd` ⊵

Если есть необходимость в использовании этих символов, подключение их можно организовать при помощи пакета latexsym. Кроме того, перечисленные символы становятся доступными после загрузки пакетов amsfonts и amssymb (см. также разд. 8.6.6).

Дополнительные математические шрифты

Обычно для набора крупных математических символов доступен шрифт только одного размера. Как правило, этого вполне достаточно, поскольку каждый из требуемых символов включен в данный шрифт в нескольких вариантах по величине, а (LA)TEX обладает средствами автоматического выбора того из вариантов соответствующего символа, который наилучшим образом подходит в данной конкретной ситуации. Однако, если в документе содержится большое число математических выражений, в состав которых входит много крупных математических символов, например, в заголовках, наличные символы могут оказаться в таком случае слишком мелкими. В этой ситуации можно воспользоваться пакетом exscale, предоставляющим возможность масштабирования математических шрифтов.

При использовании пакета exscale потребуются варианты шрифта cmex10, увеличенные до размера 10.95pt, 12pt, 14.4pt, 17.28pt, 20.74pt и 24.88pt. Кроме того, необходимы будут также его варианты для размеров 7pt и 9pt. Эти шрифты являются составной частью шрифтового пакета AMS, найти их можно на многих серверах.

7.5.3 slides — получение демонстрационных слайдов

Когда существовал только LaTeX, основанный на старой схеме переключения шрифтов, для получения демонстрационных слайдов была необходима специальная версия LaTeX'а, именуемая SliTeX [$\mathcal{L}$ **131–38**], поскольку для слайдов требуется набор шрифтов, совершенно отличающийся от стандартного. В настоящее время, однако, того же самого результата можно добиться простым использованием класса документов slides, обладающего практически теми же самыми функциональными возможностями, что и программа SliTeX.

Этот класс, существующий в рамках схемы NFSS, обладает целым рядом достоинств: нет необходимости в специальном, заранее скомпилированном форматном файле; любой требуемый шрифт можно подключить простой загрузкой соответствующего пакета. Например, если имеется семейство шрифтов PostScript Times, можно формировать слайды с его использованием, если написать

```
\documentclass{slides}
\usepackage{times}
```

7.5.4 Обработка ранее созданных документов

oldlfont — обеспечения совместимости с LaTeX 2.09

Как было показано, NFSS и, следовательно, LaTeX2ε, существенно отличаются от LaTeX 2.09 в части реализуемых команд переключения шрифтов. Эта разница особенно заметна в математических формулах, где команды наподобие \bfseries

теперь применять запрещено. Тем не менее в NFSS набирть старые (т. е. подготовленные в LaTeX 2.09) документы очень просто.

Если требуется просто воспроизвести такого рода документ, это обеспечивается встроенными средствами LaTeX2_ε, осуществляющими после обнаружения команды `\documentstyle` переключение в режим совместимости с LaTeX 2.09, в котором выполняется эмуляция старой схемы переключения шрифтов, описанной в первом издании LaTeX book. Другой вариант состоит в том, чтобы после команды `\documentclass` загрузить пакет oldlfont. В таком случае будут определены все старые команды переключения шрифтов, все же другие команды переключения шрифтов будут блокироваться. Переопределенные команды можно использовать в математических формулах.

Некоторые авторы пользовались такими внутренними шрифтовыми командами, как `\twlrm` и `\nintt`, а также другими, подобными этим. Теперь включение этих команд в текст документа вызывает появление сообщения об ошибке, поскольку они больше не поддерживаются даже в режиме совместимости. Одно из оснований для такого подхода состоит в том, что определения данных команд не были общепринятыми. Чтобы иметь возможность обработать документ, включающий в явном виде указанные команды, необходимо определить их в преамбуле, используя команды, описанные в разд. 7.6. Например, для команд `\twlrm` и `\nintt` требуемое определение могло бы выглядеть следующим образом:

```
\newcommand{\twlrm}{\fontsize{12}{14pt}\normalfont\rmfamily}
\newcommand{\nintt} {\fontsize{9}{11pt}\normalfont\ttfamily}
```

Повторное использование частей старых документов также не вызывает обычно затруднений. Попробуйте вначале просто включить требуемый фрагмент в новый документ и посмотрите, что получится. Как правило, NFSS вполне корректно справляется с этим фрагментом, если же этого не произойдет, будут в явном виде указаны те места введенного фрагмента, в которые требуется внести изменения.

newlfont — обеспечение совместимости с NFSS1

В первой редакции NFSS двухбуквенные команды переключения шрифтов были переопределены таким образом, чтобы воздействовать на отдельные атрибуты шрифтов, например, команды `\sf` или `\it` действовали точно так же, как и команды `\sffamily` и `\itshape` в NFSS2. Если предпочтительнее использовать краткие варианты таких команд, то после команды `\documentclass` следует загрузить пакет newlfont.

7.5.5 Специальные пакеты для NFSS

syntonly — проверка синтаксиса TeX'овского документа

Пакет syntonly реализует LaTeX'овскую декларацию `\suntaxonly`. Эту команду затем можно использовать в преамбуле документа, что позволяет проверить его

синтаксическую корректность (без формирования выходного образа документа), что выполняется примерно вчетверо быстрее, чем обработка этого же документа в обычном режиме.

tracefnt — трассировка NFSS

Пакет tracefnt может оказать помощь в выявлении источников ошибок, возникших при использовании NFSS. Для управления характером и объемом выдачи информации на экран терминала и в файл протокола в пакете tracefnt есть несколько необязательных параметров.

errorshow Если задать этот параметр, то все предупреждения и информационные сообщения будут выдаваться только в файл протокола, выдача их на терминал будет подавляться. На экран терминала в данном режиме будут выводиться только сообщения об ошибках. Поскольку предупреждения о некорректных подстановках шрифтов могут означать, что получаемый окончательный вид не соответствует задуманному макету, следует тщательно проанализировать протокольный файл перед завершающей распечаткой документа.

warningshow На экран терминала выдаются предупреждения и сообщения об ошибках; объем выдаваемой информации будет таким же, как если бы LaTeX 2_ε работал без использования пакета tracefnt.

infoshow Используется пакетом tracefnt по умолчанию. На терминал выдается дополнительная информация, которая обычно только записывается в протокольный файл.

debugshow Формируются и выводятся дополнительные сведения, относящиеся к переключениям текстовых шрифтов, включая восстановление вида шрифта при выходе из группы. Пользоваться этой возможностью надо с определенной осторожностью, поскольку она ведет к существенному увеличению размера протокольного файла.

В дополнение к этим параметрам «стандартной трассировки»[9] в пакет tracefnt включены еще два таких параметра:

pausing В этом режиме все предупреждающие сообщения переводятся в категорию сообщений об ошибках, чтобы помочь автору в обнаружении возможных источников некорректностей при формировании важных документов.

loading В данном режиме выдается информация о загрузке внешних шрифтов. Однако если какие-либо шрифты загружаются средствами форматного файла и класса документов, то для этих шрифтов информация об их загрузке выдаваться не будет.

[9] Имеется в виду, что разработчики сервисных пакетов, обеспечивающих средства трассировки для пользовательских пакетов, используют по мере возможности именно эти ключевые слова.

Программа nfssfont

В комплект поставки NFSS входит LaTeX'овский файл, называемый `nfssfont.tex`, который можно использовать для тестирования новых шрифтов, выдачи шрифтовых таблиц, показывающих все символы данного шрифта и т. д. Этот файл представляет собой доработанную программу `testfont.tex`, написанную первоначально Дональдом Кнутом. При запуске LaTeX'а для обработки файла `nfssfont.tex` будет задан вопрос относительно имени проверяемого шрифта. Ответ должен содержать внешнее имя шрифта без расширения, например, `cmr10` (Computer Modern Roman 10pt) или `yinit` (буквицы Янниса Хараламбуса). После этого появляется запрос на ввод команды. По-видимому, наиболее важной из этих команд, реализуемых пакетом tracefnt, является команда `\table`, которая выдает таблицу раскладки шрифта в виде, показанном на с. 241. Чтобы перейти к работе с другим шрифтом, надо ввести команду `\init`; для прекращения работы с программой следует воспользоваться командой `\bye` или `\stop`. Справку по другим возможностям программы `nfssfont.tex` (в настоящее время они относятся все еще к шрифтам с кодировкой `OT1`) можно получить, если ввести команду `\help`.

7.6 Низкоуровневый интерфейс

Шрифтовые команды высокого уровня, рассматривавшиеся выше, предназначены для непосредственного их использования при подготовке документа. Наряду с ними NFSS реализует также набор низкоуровневых команд, необходимых главным образом для формирования новых высокоуровневых команд в пакетах, либо в преамбуле документа (см. также разд. 7.6.4). Чтобы эффективнее использовать эти средства, полезно иметь представление о внутренней организации шрифтов, принятой в NFSS.

Одна из задач, решаемых NFSS — дать в распоряжение пользователя средства для удобного и рационального выбора требуемых шрифтов. Соответствующие алгоритмы строятся на основе общих принципов разметки текста при его верстке. С этой целью желательно иметь возможность независимо изменять как можно большее число шрифтовых атрибутов. Однако в действительности семейства шрифтов обычно отвечают не полному набору всех возможных комбинаций атрибутов, а лишь некоторому подмножеству этого набора. Следовательно, если допустить независимое варьирование слишком большого числа атрибутов, получим в итоге очень много таких их комбинаций, которым не соответствует никакой внешний (реальный) шрифт; в такой ситуации NFSS прибегает к подстановке некоторого шрифта, определенного по умолчанию, вместо недостающего.

NFSS в ходе своей работы отслеживает значения пяти независимых шрифтовых атрибутов, в число которых входят: текущая схема кодирования, текущее семейство, текущая насыщенность, текущее начертание и текущий размер. Атрибут «схема кодирования» (кодировка) был введен в состав второй редакции NFSS после того, как стало ясно, что эффективная поддержка работы в многоязыко-

вой среде возможна лишь в случае, когда имеется возможность менять кодировку шрифта независимо от всех остальных его атрибутов.

Совокупность значений перечисленных выше атрибутов определяет тот шрифт, который в данный конкретный момент будет текущим. В состав NFSS входит большое число внутренних таблиц, назначение которых — связать комбинации значений шрифтовых атрибутов с реальными внешними шрифтами (т. е. с `.tfm`-файлами, содержащими информацию о соответствующем шрифте, необходимую LaTeX'у при обработке документа). Выбор шрифта в NFSS производится в два этапа:

- при помощи низкоуровневых команд `\fontencoding`, `\fontfamily`, `\fontseries` и `\fontsize` осуществляется изменение всех шрифтовых атрибутов или же какой-либо их части;
- при помощи команды `\selectfont` выполняется подбор шрифта, соответствующего заданной комбинации атрибутов.

Второй этап включает в себя несколько действий. Вначале NFSS проверяет, подключен ли шрифт, отвечающий требуемой комбинации атрибутов (т. е. имеется ли уже в памяти соответствующий `.tfm`-файл); если да, то производится переключение на него. Если требуемый шрифт не подключен, то NFSS при помощи своих внутренних таблиц пробует найти имя внешнего шрифта с необходимыми значениями атрибутов. Если такое имя шрифта удается найти, производится загрузка в память `.tfm`-файла для него, после чего выполняется переключение на данный шрифт. В случае же, когда такой поиск завершается неудачей, NFSS пытается найти шрифт для замены недостающего так, как это объясняется в разд. 7.6.3.

7.6.1 Установка индивидуальных шрифтовых атрибутов

Для каждого из шрифтовых атрибутов имеется одна команда, позволяющая изменить его текущее значение. Все эти команды в качестве аргумента имеют текстовую строку более или менее произвольного вида. Однако очень немногие из допустимых строк имеют смысл. Такие «осмысленные» значения аргументов шрифтовых команд не являются жестко встроенными в NFSS, они представляют собой лишь соглашения, отраженные во внутренних таблицах NFSS. В последующих разделах дается описание этих соглашений применительно к стандартному варианту настройки NFSS; используемые настройки можно изменениить, добавив новые шрифтовые декларации во внутренние таблицы NFSS. При подключении новых шрифтов надо стараться в максимальной степени следовать соглашениям по именованию, принятым в NFSS. Такой подход облегчает осуществление непротиворечивого именования шрифтов, что в свою очередь гарантирует выбор правильного шрифта при верстке документа.

Если упомянутые интерфейсные средства используются для выбора конкретного шрифта (например, Computer Modern Dunhill, полужирный, узкий, курсивный, размером 14pt), то обнаруживается, что знания только соглашений по именованию шрифтовых атрибутов недостаточно, поскольку внешние шрифты

существуют не для каждой из возможных комбинаций атрибутов. Вы можете, конечно, попытать счастья, записав что-то вроде следующего набора команд:

```
\fontencoding{OT1}\fontfamily{cmdh}\fontseries{bc}\fontshape{it}%
\fontsize{14}{16pt}\selectfont
```

который вполне корректен, как это будет видно из материала следующих разделов, с точки зрения соглашений по именованию атрибутов. Но поскольку задаваемой этими командами комбинации атрибутов не отвечает ни один из существующих реальных шрифтов, NFSS прибегнет к подстановке шрифта, определенного по умолчанию, вместо недостающего. Алгоритм такой подстановки может выбрать шрифт, совершенно непохожий на тот, который хотелось бы получить, так что прежде чем задавать ту или иную комбинацию шрифтовых атрибутов, надо свериться с таблицами раскладки шрифтов и определить, есть ли шрифт, удовлетворяющий заданным требованиям (более подробно процесс подстановки описан в разд. 7.6.3). В каждом инсталляционном комплекте LaTeX'а должна быть информация о том, какие именно шрифты доступны в имеющейся конкретной конфигурации системы.

Выбор семейства шрифтов

Семейство шрифтов выбирается при помощи команды `\fontfamily`. Аргумент этой команды представляет собой символьную (текстовую) строку, обозначающую имя семейства так, как оно определено во внутренних таблицах NFSS. Указанная строка задается при формировании этих таблиц и обычно представляет собой короткую последовательность букв, например, `cmr` для семейства Computer Modern Roman. Имена шрифтовых семейств не должны превышать по длине пяти символов, поскольку при формировании имени соответствующего файла может быть добавлено еще до трех символов, а в среде некоторых операционных систем имя файла должно включать не более 8 символов.

Выбор насыщенности шрифта

Атрибут «насыщенность» для шрифта изменяется командой `\fontseries`. Насыщенность является комбинированным атрибутом, учитывающим жирность шрифта и ширину очка литеры. Вследствие этого нельзя изменить ширину литер шрифта независимо от его жирности. Такой подход обусловлен тем, что едва ли нужно отдельно менять ширину очка литеры и жирность шрифта. Напротив, изменение жирности шрифта (например, на полужирный) сопровождается обычно изменением значения ширины очка литеры (например, на широкое начертание). В этом нет ничего удивительного, поскольку варьирование жирности изменяет внешний вид символа (соотношение ширины его штрихов по горизонтали) и для того, чтобы этот вид оставался визуально сбалансированным, приходится корректировать и его ширину.

В соглашениях по именованию для аргумента команды `\fontseries` названия соответствующих градаций жирности и ширины сокращены таким образом,

Градация жирности

Ultra Light	ul
Extra Light	el
Light	l
Semi Light	sl
Medium (normal)	m
Semi Bold	sb
Bold	b
Extra Bold	eb
Ultra Bold	ub

Градация ширины

Ultra Condensed	50%	uc
Extra Condensed	62.5%	ec
Condensed	75%	c
Semi Condensed	87.5%	sc
Medium	100%	m
Semi Expanded	112.5%	sx
Expanded	125%	x
Extra Expanded	150%	ex
Ultra Expanded	200%	ux

Таблица 7.10. Классификация ширины очка литеры и жирности шрифтов

Указанные процентные соотношения взяты из работы [95]. При объединении этих обозначений для получения аргумента команды `\fontseries` первым берется обозначение для жирности, причем любое вхождение буквы `m` в получаемую строку опускается, кроме случая, когда значение «нормальный» имеют оба атрибута, т. е. жирность и ширина. В таком случае оставляется одна буква `m`.

чтобы каждой из допустимых комбинаций этих атрибутов отвечало уникальное обозначение. Эти соглашения представлены в табл. 7.10. Данные классификации объединяются в аргументе команды `\fontseries`, при этом, однако, значение `m` (оно соответствует «нормальному» значению жирности или ширины) всякий раз опускается, за исключением случая, когда это значение имеют оба рассматриваемых атрибута. Такой случай обозначается единственной буквой `m`. Например, полужирный широкий шрифт будет обозначаться как `bx`, тогда как нормальный (по жирности) широкий шрифт имеет обозначение `x`, а полужирный нормальный (по ширине) — обозначение `b`.

Выбор начертания шрифта

Для изменения значения атрибута «начертание» в NFSS существует команда `\fontshape`. Как и в случае насыщенности шрифта, вводятся одно- и двухбуквенные сокращения обозначений начертания шрифта; эти сокращения перечислены в табл. 7.11.

Выбор размера шрифта

Размер шрифта можно изменить командой `\fontsize{`⟨*size*⟩`}{`⟨*skip*⟩`}`. Это единственная команда смены атрибута шрифта, у которой имеется два аргумента: параметр ⟨*size*⟩, задающий требуемый размер шрифта, и параметр ⟨*skip*⟩, устанавливающий расстояние между базисными линиями для двух смежных строк текста документа. По умолчанию считается, что что размер шрифта (⟨*size*⟩) задается в типографских пунктах, обозначение pt при этом опускается. То же самое

Наименование	*Описание*
`n`	Прямой (upright)
`it`	*Курсив (italic)*
`sl`	*Наклонный (slanted)*
`sc`	Капитель (small caps)
`ui`	Прямой курсив (upright italic)
`ol`	КОНТУРНЫЙ (OUTLINE)

Таблица 7.11. Классификация начертаний шрифтов

Стандартные сокращенные наименования начертаний шрифтов даны здесь для семейства Computer Modern Roman в качестве примера. Начертание outline shape построено с использованием METAFONT'овского исходного текста, приведенного в [81].

верно и для второго аргумента (⟨*skip*⟩). Однако, если межстрочный промежуток (baseline skip) должен быть растяжимым (т. е. если он содержит `plus` или `minus`), единицы измерения следует указывать в явном виде. В таком случае правильная команда изменения размера шрифта могла бы, например, иметь такой вид

```
\fontsize{14.4}{17}\selectfont
```

Даже если такой запрос и правилен в принципе, внешнего шрифта соответствующего размера может не оказаться. В этом случае NFSS попытается найти шрифт, возможно более близкий по размеру к требуемому, если внутренние таблицы разрешают проводить корректировку размера шрифта. Если же такая корректировка недопустима, выдается сообщение об ошибке.

Если вы используете шрифты, допускающие произвольное масштабирование (например, PostScript-шрифты), можно, конечно, указывать в качестве требуемого любой размер шрифта; например, команда

```
\fontsize{1in}{1.2in}\selectfont Happy Birthday
```

приведет к получению строки с размером букв в один дюйм для поздравительного плаката.

Есть, правда, одна проблема, связанная с применением произвольно масштабируемых шрифтов: поскольку NFSS не знает точно, будут или нет набираться формулы, она произведет подстройку всех шрифтов, используемых в формулах, под новый размер. Это значит, что требуется пересчитать и размеры символов, входящих в верхние и нижние индексы в формулах и т. д. А это, в свою очередь, означает, что будет загружено дополнительно большое число новых шрифтов (информацию об этом можно получить из протокольного файла). Если загружаемых таким образом шрифтов будет слишком много, то в итоге может оказаться исчерпанной память, отводимая для размещения шрифтов. Если такое произойдет, можно попытаться (при помощи декларации `\DeclareMathSizes`) явным образом сообщить NFSS, какие шрифты необходимо загружать для использования их в

Кодировка	*Описание*	*Определено в ...*
T1	Корковская схема кодирования для TEX'овских шрифтов.	NFSS
OT1	Кодировка для TEX'овских шрифтов, введенная Дональдом Кнутом.	NFSS
OML	Кодировка для «математического курсива», введенная Дональдом Кнутом.	NFSS
OMS	Кодировка для математических символов, введенная Дональдом Кнутом.	NFSS
OMX	Кодировка для расширенного набора математических символов, введенная Дональдом Кнутом.	NFSS
U	Неизвестная кодировка (для различных нестандартных ситуаций).	NFSS
L..	Местная (локальная) кодировка (для схем кодирования частного характера).	—

Таблица 7.12. Стандартные схемы кодирования, используемые в NFSS

Мы надеемся, что большинство из этих схем кодирования будет вытеснено стандартами, определяемыми различными (национальными) группами пользователей TEX'а, с тем, чтобы со временем смена кодировки могла потребоваться только при переключении с одного языка на другой. По этой причине наименования большинства кодировок начинаются с буквы O, чтобы подчеркнуть, что они являются устаревшими (от английских слов **O**ld или **O**bsolete).

формулах, чтобы блокировать ее попытки решить эту задачу при помощи алгоритма, имеющегося в NFSS. В разд. 7.7.6 приведена дополнительная информация по этому вопросу.

Выбор схемы кодирования

Изменение кодировки осуществляется командой `\fontencoding`, аргумент которой — внутреннее имя требуемой схемы кодирования. Это имя должно быть известно NFSS. Для этого оно должно либо принадлежать набору предопределенных имен кодировок (см. табл. 7.12)[10], либо его следует определить при помощи команды `\DeclareFontEncoding` (см. разд. 7.7.4).

NFSS исходит из предположения, что большая часть шрифтов (а еще лучше — все) доступна в одной и той же кодировке, пока они используются для набора текста на одном и том же языке. Другими словами, смена схемы кодирования может потребоваться только при переключении с одного языка на другой. Например,

[10] В настоящее время для языков, использующих кириллицу, зафиксирована кодовая таблица T2; см. с. 9.— *Прим. ред.*

окружение для набора фрагмента русского текста (при основном тексте, набираемом латинскими буквами) могло бы иметь вид

Some text по-русски embedded

```
\newenvironment{Cyr}
    {\fontencoding{OT2}\fontfamily{wnr}\selectfont}{}
Some text \begin{Cyr}po-russki\end{Cyr} embedded
```

где OT2 задает Вашингтонскую (кириллическую) кодировку. В таком случае требуемая схема кодирования должна быть объявлена в преамбуле документа или же в пакете.

7.6.2 Установка значений для нескольких шрифтовых атрибутов

При разработке стилей макета полосы (см. разд. 4.3) или команд, меняющих внешний вид документа, часто возникает необходимость выбора некоторого конкретного шрифта, для чего требуется задать значения всех шрифтовых атрибутов. Для решения этой задачи NFSS располагает командой `\usefont`, имеющей четыре аргумента: кодировку, семейство, насыщенность и начертание. Данная команда модифицирует значения перечисленных атрибутов, после чего вызывает команду `\selectfont`. Если в дополнение к этим операциям надо изменить также размер шрифта и интерлиньяж (межстрочный интервал), следует перед командой `\usefont` поместить соответствующую команду `\fontsize`, например,

```
\fontsize{14}{16pt}\usefont{OT1}{cmdh}{bc}{it}
```

что приводит к выполнению тех же действий, что и в примере, приведенном на с. 219.

Помимо `\usefont` в NFSS определена декларация `\DeclareFixedFont`, которую можно использовать при формировании новых команд, осуществляющих переключение на новый основной шрифт документа. Такие команды будут выполняться очень быстро, поскольку в них совсем не применяется просмотр каких-либо внутренних таблиц NFSS. Поэтому они будут очень полезными в определениях команд, требующих реализации постоянных переключений между фиксированными (т. е. заранее заданными) шрифтами. Например, в пакете doc (см. гл. 14) номера страниц исходного текста (программного кода) набираются с использованием следующих определений:

```
\DeclareFixedFont{\CodelineFont}{\encodingdefault}{\familydefault}
                 {\seriesdefault}{\shapedefault}{7pt}
\newcommand{\theCodelineNo}{\CodelineFont\arabic{CodelineNo}}
```

Как видно из этого примера, декларация `\DeclareFixedFont` имеет шесть аргументов — имя определяемой команды, за которым следуют пять шрифтовых атрибутов, используемых NFSS. Вместо фиксированных значений в данном определении для аргументов используются встроенные ссылки, набор которых опи-

сывает основной шрифт документа (см. разд. 7.3.5). В этом примере все еще зависит от общего вида документа только команда `\CodelineFont` (через значения параметров `\encodingdefault` и т. п.). Однако, после того как определение сформировано, их значения будут зафиксированы (заморожены).

7.6.3 Автоматические подстановки шрифтов

Всякий раз как запрос на переключение шрифта не может быть выполнен из-за того, что предъявленная комбинация атрибутов неизвестна NFSS, система предпринимает попытку разрешить возникшую проблему путем использования шрифта с близкими значениями атрибутов. При этом происходит следующее. Если комбинация из кодировки, семейства, насыщенности и начертания не определена (см. разд. 7.7.2), NFSS меняет значение атрибута «начертание» на то, которое присвоено ему по умолчанию. Если получившаяся комбинация параметров по-прежнему неизвестна NFSS, значение по умолчанию присваивается атрибуту «насыщенность». Как последнее средство используется изменение значения атрибута «семейство» (также на значение, которое этот атрибут имеет по умолчанию). Завершающая операция состоит в просмотре внутренней таблицы для поиска шрифта требуемого размера. Например, если задать последовательность команд `\ttfamily\bfseries\itshape`, т. е. запросить переключение на шрифт пишущей машинки, курсивного начертания, полужирный (такого шрифта обычно нет), то в действительности в документе будет использован шрифт пишущей машинки, но нормальной насыщенности и прямого начертания, поскольку NFSS изменит вначале значение атрибута «начертание» и лишь потом — атрибута «насыщенность» на их значения по умолчанию. Если в подобной ситуации предпочтительнее все-таки шрифт пишущей машинки в курсивном начертании, надо довести это до сведения NFSS при помощи функции `sub`, описываемой на с. 231.

Схема кодирования процессом подстановки не затрагивается никогда, поскольку такого рода изменения могут привести к появлению неверных символов в тексте документа. Напомним, что схема кодирования определяет, каким образом интерпретировать входные символы, тогда как остальные атрибуты задают, как должен выглядеть формируемый документ. Было бы крайне нежелательно, а то и катастрофично, если бы, например, знак £ в документе оказался заменен знаком $ только из-за претензий программы быть «умной».

Таким образом, каждая схема кодирования должна иметь значения по умолчанию для семейства, насыщенности и начертания и хотя бы одну комбинацию из кодировки и этих атрибутов, определенную в NFSS так, как это объясняется в разд. 7.7.4.

7.6.4 Использование низкоуровневых команд в документе

Низкоуровневые команды, описанные в предыдущих разделах, предназначены для использования их в определениях высокоуровневых команд, классов и пакетов, а также в преамбуле документа.

Всегда, когда есть такая возможность, следует избегать непосредственного применения низкоуровневых команд в документе и пользоваться вместо них соответствующими высокоуровневыми командами вроде `\textsf`. Причина здесь в том, что низкоуровневые команды являются инструкциями, позволяющими с высокой точностью управлять переключениями конкретных шрифтов, тогда как высокоуровневые команды дают возможность модифицировать их при помощи пакетов или деклараций в преамбуле документа. Предположим, например, что в формируемом документе при помощи команды `\fontfamily{cmss}\selectfont` выбран шрифт Computer Modern Sans. Если впоследствии вы решите перепечатать весь документ, используя PostScript-шрифты, например, семейства Times, то, к сожалению, загрузка соответствующего пакета приведет к изменению только тех частей документа, которые не содержат команд `\fontfamily`.

7.7 Подключение новых шрифтов

7.7.1 Общая схема

Подключение новых шрифтов для их использования с NFSS в первую очередь состоит в заполнении внутренних таблиц NFSS информацией, необходимой в дальнейшем для связывания шрифтового запроса, обнаруженного в документе, с соответствующим внешним `.tfm`-файлом, содержащим информацию о символах, которая нужна LaTeX'у в процессе обработки документа. Например, внутренние таблицы NFSS обеспечивают связь запроса вида

```
\fontencoding{OT1}\fontfamily{cmdh}\fontseries{m}\fontshape{n}%
\fontsize{10}{12pt}\selectfont
```

с внешним `.tfm`-файлом с именем `cmdunh10.tfm`. Чтобы добавить новый шрифт, надо обратить этот процесс. Для каждого нового внешнего шрифта надо ответить на следующие пять вопросов:

1. Какова схема кодирования подключаемого шрифта, т. е. какие символы и в каких позициях размещаются?
2. Каково имя семейства, к которому принадлежит данный шрифт?
3. Какова насыщенность подключаемого шрифта (жирность и ширина очка литеры)?
4. Каково начертание этого шрифта?
5. Каков его размер (кегль)?

Ответы на эти вопросы дают необходимую информацию, позволяющую классифицировать предъявленный внешний шрифт в соответствии с соглашениями, принятыми в NFSS, как это было описано в разд. 7.6. В следующих нескольких разделах обсуждается, каким образом включить сведения о новых шрифтах в таблицы NFSS так, чтобы эти шрифты можно было использовать в основном тексте документа. Обычно такая информация необходима, когда предпринимается попытка подключения новых шрифтов, например, путем написания соответствующего пакета, обеспечивающего доступность нового семейства шрифтов. В последующих разделах рассматриваются более сложные вопросы, например, подключение специальных шрифтов в качестве математических вместо стандартных, используемых в LaTeX'e.

7.7.2 Объявление новых семейств шрифтов и групп начертаний шрифтов

Каждую комбинацию из семейства шрифтов и схемы кодирования надо сделать известной NFSS, используя для этой цели команду `\DeclareFontFamily`. Эта команда имеет три аргумента. Первые два — наименования схемы кодирования и семейства шрифтов. Третий аргумент обычно представляет собой пустую строку, однако он может содержать специальный параметр управления загрузкой шрифта (он объясняется на с. 232). Таким образом, если надо подключить новое семейство шрифтов, например Computer Modern Dunhill со старой TeX'овской кодировкой, это можно сделать следующей командой:

```
\DeclareFontFamily{OT1}{cmdh}{}
```

В норме семейство шрифтов состоит из нескольких индивидуальных (отдельных) шрифтов. Вместо того чтобы вводить в NFSS каждый из элементов данного семейства индивидуально, можно объединить шрифты, различающиеся только размером, и объявить их как шрифтовую группу.

Такая группа вводится во внутренние таблицы NFSS командой `\DeclareFontShape`, которая имеет шесть аргументов. Первые четыре из них — это схема кодирования, имя семейства, наименование насыщенности и наименование начертания, которые должны использоваться впоследствии в командах вызова данных шрифтов. Пятый аргумент представляет собой список размеров и внешних имен шрифтов, записанных в специальном формате, который будет описан ниже. Шестой аргумент обычно представляет собой пустую строку, использование его будет разъяснено на с. 232.

Приведем вначале несколько примеров и введем необходимую терминологию, затем обсудим эти вопросы подробнее.

В качестве примера можно привести команду, обеспечивающую ввод соответствующей информации в таблицы NFSS для шрифта Computer Modern Dunhill нормальной насыщенности, прямого начертания, со схемой кодирования «TeXtext»:

```
\DeclareFontShape{OT1}{cmdh}{m}{n}{ <10> cmdunh10 }{}
```

В этом примере считается, что доступен единственный внешний шрифт размера 10pt. Если же этот шрифт есть еще и для размера 12pt, декларация, показанная выше, перепишется в следующем виде:

```
\DeclareFontShape{OT1}{cmdh}{m}{n}{ <10> <12>cmdunh10 }{}
```

Если внешний шрифт доступен для всех возможных размеров (как, например, в случае шрифтов PostScriptType 1 или же при наличии программы, вызывающей METAFONT и заставляющей его сформировать шрифты недостающих размеров), упомянутая декларация принимает очень простой вид. Например, шрифт Times Roman, полужирной насыщенности, прямого начертания, в текстовой корковской кодировке мог бы быть подключен следующей командой:

```
\DeclareFontShape{T1}{times}{b}{n}{ <-> pstimb }{}
```

В этом примере используется объявление диапазона размеров шрифтов с открытыми границами, т. е. конкретная величина наименьшего и наибольшего размеров здесь не задана. В результате для PostScript-шрифтов всех размеров будет использоваться один и тот же внешний `.tfm`-файл (назовем его `pstimb`), данные из которого будут пересчитываться (масштабироваться) для каждого потребовавшегося конкретного размера. Если для каждого PostScript-шрифта имеется более одного `.tfm`-файла, например, `pstim` для печати документа на принтере и `pstdim` — для вывода документа на экран терминала, соответствующая декларация примет вид

```
\DeclareFontShape{T1}{times}{m}{n}{ <-12> pstim <12-> psdtim }{}
```

В таком случае `.tfm`-файл с именем `pstim` будет использоваться для размеров менее 12pt, а `pstdim` — для всех размеров, которые больше или равны 12pt.

Из предшествующих примеров видно, что пятый аргумент команды `\DeclareFontShape` состоит из спецификаций размера шрифта, заключенных в угловые скобки (т. е. `<...>`) и перемежающихся информацией, необходимой для загрузки отдельных шрифтов (т. е. именами этих шрифтов). Часть аргумента, содержащаяся внутри угловых скобок, представляет собой информацию о размерах шрифтов, а часть его, следующая за закрывающейся угловой скобкой — информацию об именах шрифтов, которая оформлена в виде так называемой «размерной функции» (часто пустой) и ее аргументов; этот случай будет обсужден ниже. Чтобы сделать аргументы команды `\DeclareFontShape` более удобочитаемыми, в них можно использовать пробелы, которые при обработке команды игнорируются. Если вдруг все-таки потребуется ввести здесь пробел как отдельный знак, можно воспользоваться командой `\space`.

Простые размеры и диапазоны размеров

Информация о размерах шрифтов, т. е. та часть пятого аргумента команды `\DeclareFontShape`, которая содержится в угловых скобках, подразделяется на два вида: «простые размеры» и «диапазон размеров». Простой размер задается

единственным (десятичным) числом, например, <10> или <14.4>, и в принципе может иметь в качестве значения любое положительное число. Однако, поскольку это число представляет собой размер шрифта в типографских пунктах, вряд ли вам понадобятся его значения меньше 4 и больше 120. Диапазон размеров задается двумя десятичными числами, разделенными знаком переноса (-), и обозначает совокупность размеров шрифтов, основывающихся на одной и той же шрифтовой информации. Нижняя граница диапазона (т. е. размер, стоящий слева от знака переноса) в него включается, тогда как верхняя исключается. Например, запись вида <5-10> обозначает совокупность размеров шрифтов, больших или равных 5pt и меньших, чем 10pt. Как левую, так и правую границу диапазона можно опустить, интерпретация получаемых выражений очевидна: <-> обозначает все возможные размеры, <-10> — все размеры меньше 10pt, а <12-> — все размеры, которые больше или равны 12pt.

Часто с несколькими простыми размерами связана одна и та же информация об именах шрифтов. Для этого случая существует удобная сокращенная запись, *получаемая удалением всех таких ссылок, кроме последней, например*

```
\DeclareFontShape{OT1}{panr}{m}{n}{ <5> <6> <7> <8> <9> <10>
        <10.95> <12> <14.4> <17.28> <20.74> <24.88> pan10 }{}
```

В этом примере содержится описание шрифта Pandora прямого начертания, нормальной насыщенности, доступного в нескольких размерах, которые получаются масштабированием из одного и того же исходного шрифта.

Размерные функции

Как отмечалось выше, шрифтовая информация (строка после закрывающейся угловой скобки) структурируется далее на размерную функцию и ее аргумент. Если в строке, представляющей собой шрифтовую информацию, имеется звездочка *, то часть этой строки, находящаяся левее звездочки, будет именем размерной функции, а ее часть, расположенная справа от звездочки — аргументом данной функции. Если звездочка в шрифтовой информации отсутствует, как это было во всех примерах выше, это означает, что размерная функция именуется `empty`, а вся строка интерпретируется как аргумент этой функции.

Основываясь на размере шрифта, затребованном в пользовательском запросе, и на информации, содержащейся в команде `\DeclareFontShape`, размерная функция формирует спецификацию, необходимую NFSS для того, чтобы отыскать соответствующий внешний шрифт и загрузить его в правильном размере. Эти функции отвечают также за информирование пользователя обо всех аномалиях, которые могут возникнуть в процессе поиска и загрузки шрифта. Например, некоторые функции отличаются друг от друга только тем, что одни из них выдают сообщения-предупреждения, а другие — нет. Это свойство дает возможность использовать NFSS наилучшим для каждой конкретной ситуации образом.

Имя размерной функции состоит из последовательности букв, в частном случае эта последовательность может быть пустой (т. е. именем функции может быть

пустая строка). Некоторые из размерных функций имеют два аргумента, один из которых обязательный, а второй — необязательный. Необязательный аргумент заключается в квадратные скобки. Например, спецификация

```
<-> s * [0.9] cmfib8
```

задает для всех возможных размеров (здесь — в диапазоне от 0 до ∞) размерную функцию `s` с необязательным аргументом `0.9` и обязательным аргументом `cmfib8`.

Спецификации размеров в команде `\DeclareFontShape` просматриваются в том порядке, в котором они записаны. Когда обнаруживается информация о размере шрифта, отвечающая запросу пользователя, исполняется соответствующая размерная функция. Если ее исполнение завершается успешно и найден требуемый шрифт, дальше список размеров не просматривается, в противном случае поиск продолжается на основе следующего элемента этого списка.

Ниже представлены стандартные размерные функции. В документации на NFSS описывается, как определить дополнительные функции, если это вдруг понадобится.

Пустая функция Поскольку пустая функция (empty) используется чаще всех, для нее было выбрано максимально короткое имя. (Каждая запись во внутренней таблице занимает определенное место в оперативной памяти, так что при выборе синтаксиса команд приходится балансировать между понятностью команды для пользователя и ее компактностью.) Пустая функция загружает шрифтовую информацию только в случае, когда запрашиваемый размер в точности соответствует одному из перечисленных простых размеров. Если определен диапазон размеров и требуемый размер в него укладывается, то пустая функция загружает шрифт именно того размера, который задан пользователем.

Например, если пользователь запросил шрифт размера `14.4`, спецификация вида

```
<-> panr10
```

загрузит `.tfm`-файл `panr10.tfm` для размера 14.4pt. Поскольку этот шрифт проектировался для размера 10pt (Pandora Roman, кегль 10pt), значения всех параметров в `.tfm`-файле будут умножены на коэффициент 1.44.

Иногда желательно загрузить шрифт, несколько больший или меньший по размеру, чем затребованный пользователем. Такая подстройка бывает нужна, в частности, если символы одного семейства выглядят слишком большими рядом с символами другого семейства шрифтов, используемого в одном и том же документе. С целью такой подстройки в пустую функцию введен второй (необязательный) аргумент, задающий значение коэффициента масштабирования, который исполь-

зуется при вычислении фактического размера загружаемого шрифта. Тогда по спецификации

```
<-> [0.95] helvetica
```

будет загружаться шрифт Helvetica, размер которого равен 95% размера, содержащегося в пользовательском запросе. Если этот необязательный аргумент используется, пустая размерная функция выдает соответствующее предупреждение, напоминающее пользователю, что шрифт будет загружен с изменением размера.

Функция `s` По своему поведению функция `s` совпадает с пустой функцией, отличие же между ними состоит в том, что функция `s` не выдает предупреждающих сообщений (от слова silence — молчание). Если записать

```
\DeclareFontShape{T1}{times}{b}{n}{ <-> s * [0.9] pstimb }{}
```

то все сообщения на терминал, которые выдавались бы в аналогичной ситуации пустой функцией (empty), будут подавляться. В протокольный файл эта информация будет записываться по-прежнему, так что в случае каких-либо аномалий имеется возможность проследить и проанализировать процесс загрузки шрифтов.

Функция `gen` Зачастую внешние имена шрифтов формируются путем присоединения числа, обозначающего размер шрифта, к цепочке литер, именующей семейство шрифтов, например, `cmtt8`, `cmtt9` и `cmtt10` — это внешние имена для шрифтов Computer Modern Typewriter размера `8`, `9` и `10pt`. Если имена шрифтов организованы по такой схеме, для сокращения числа размеров можно воспользоваться функцией `gen`. Эта функция объединяет (комбинирует) шрифтовую информацию с информацией о размерах для генерации (отсюда ее имя `gen` — generation) внешних имен шрифтов. Используя функцию `gen`, можно записать

```
<8> <9> <10> gen * cmtt
```

как сокращенный вариант спецификации вида

```
<8> cmtt8 <9> cmtt9 <10> cmtt10
```

что дает возможность сэкономить восемь символов во внутренних таблицах NFSS. Функция `gen` объединяет обе части внешнего имени шрифта буквально, так что в случае десятичных размеров шрифтов (например, `14.4`) ее использовать нельзя. Кроме того, надо быть уверенным, что число во внешнем имени шрифта действительно отвечает его проектному размеру (например, шрифт с внешним именем `cmr17` на самом деле представляет собой Computer Modern Roman размера 17.28pt).

Во всех остальных отношениях функция `gen` ведет себя аналогично пустой функции, т. е. если задан необязательный аргумент, то он представляет собой мас-

штабный коэффициент, при использовании которого выдается соответствующее информационное сообщение.

Функция sgen Функция sgen представляет собой вариант функции gen с отключенной выдачей сообщений на экран терминала. Запись этих сообщений в протокольный файл будет производиться в полном объеме.

Функция sub Еще одна важная размерная функция — это функция sub, используемая для организации подстановки одной группы начертаний шрифтов вместо другой, когда не удается найти внешний шрифт для текущей шрифтовой группы. В таком случае аргумент функции sub — это не имя некоторого внешнего шрифта, а имя, составленное из семейства, насыщенности и начертания шрифта, разделенных символом "/" (на схему кодирования процесс подстановки не распространяется по причинам, объяснявшимся выше). Например, в семействе Computer Modern Sans отсутствует шрифт с курсивным начертанием, имеется только шрифт наклонного начертания. Поэтому имеет смысл определить наклонное начертание в качестве значения этого атрибута, которое будет использоваться для выполнения подстановки, если встретится запрос на шрифт этого семейства в курсивном начертании. Это можно выполнить при помощи следующей команды:

```
\DeclareFontShape{OT1}{cmss}{m}{it}{ <-> sub * cmss/m/sl }{}
```

Без такой декларации алгоритм подстановки, реализуемый NFSS (см. разд. 7.6.3), приведет к замене отсутствующего курсивного шрифта семейства Computer Modern Sans на шрифт прямого начертания из этого же семейства.

Кроме подстановки на уровне шрифтовой группы в целом, у команды sub есть еще одно полезное применение:

```
\DeclareFontShape{OT1}{cmss}{m}{sl}{ <-8> sub * cmss/m/n
   <8> cmssi8  <9> cmssi9  <10><10.95> cmssi10  <12><14.4> cmssi12
   <17.28><20.74><24.88> cmssi17 }{}
```

Из этой декларации следует, что для шрифтов с размерами, меньшими 8 pt, в NFSS должно использоваться описание шрифта OT1/cmss/m/n. Такие подстановки можно выстраивать в цепочку. Те, кто знаком с составом шрифтового пакета в его стандартном варианте, знают, что шрифтов Computer Modern Sans с размерами меньше 8pt не существует, так что группа, используемая при подстановке шрифтов, должна, по-видимому, содержать еще какой-то вариант подстановки. Преимущество такого подхода состоит в том, что при добавлении нового шрифта придется изменить декларацию только для одной шрифтовой группы, остальные будут подстроены автоматически.

Размерная функция sub выдает на терминал информационные сообщения о произведенных подстановках. Если эти сообщения надо подавить, вместо функции sub следует воспользоваться функцией ssub.

Функция ssub Функция ssub обладает теми же самыми возможностями, что и функция sub, однако не выдает на экран терминала сообщений, предупреждающих о выполненных подстановках.

Функция subf Эта функция является в определенной степени гибридом пустой функции и функции sub. Она загружает шрифты тем же самым способом, что и пустая функция, но выдает сообщение, если пришлось прибегнуть к операциям подстановки из-за отсутствия требуемого шрифта. Поэтому функцию subf можно использовать для выполнения подстановок одних внешних шрифтов вместо других, причем определять отдельную шрифтовую группу для них, как это имеет место в случае функции sub, нет необходимости.

Функция ssubf Это вариант функции subf с отключенной выдачей сообщений на терминал, все порождаемые сообщения записываются только в протокольный файл.

Функция fixed Эта функция позволяет, не обращая внимания на размер шрифта, содержащийся в пользовательском запросе, загрузить шрифт с размером, указанным в аргументе. Если задан необязательный аргумент, то он указывает требуемый размер (в типографских пунктах) загружаемого шрифта. Эта функция позволяет задать диапазон размеров, из которого будет загружен шрифт с некоторым фиксированным размером.

Функция sfixed Данная функция представляет собой вариант функции fixed с отключенной выдачей информационных сообщений. Эта функция используется, например, для загрузки шрифта, содержащего крупные математические символы, которые доступны обычно лишь в одном размере.

Параметры управления загрузкой шрифтов

Как уже отмечалось, каждое семейство шрифтов должно быть объявлено при помощи команды `\DeclareFontFamily`. Аргумент этой команды, так же, как и шестой аргумент команды `\DeclareFontShape`, можно использовать для определения некоторых специальных операций, выполняемых при загрузке шрифта. Таким способом можно изменять параметры, связанные со шрифтом в целом.

Для любого внешнего шрифта кроме информации о каждом символе, входящем в него, (LA)TEX использует также набор глобальных размеров и других значений, связанных со шрифтом. Например, каждый шрифт включает свой собственный «знак переноса», т. е. символ, вставляемый (LA)TEX'ом автоматически, когда производится перенос слова. Другой пример связан с нормальной шириной и растяжимостью межсловного интервала. Здесь опять с каждым шрифтом связано определенное значение данного параметра, изменяемое при переключении с одного шрифта на другой. Корректируя эти значения при загрузке шрифтов, можно добиваться различных специальных эффектов.

Обычно вносимые изменения относятся к семейству шрифтов в целом. Пусть, например, требуется запретить переносы во всех словах, набираемых шрифтом пишущей машинки (т. е. шрифтом из семейства Computer Modern Typewriter). Эта задача решается третьим аргументом команды `\DeclareFontFamily`. Если некоторые изменения должны распространяться только на некоторую шрифтовую группу, следует использовать шестой аргумент команды `\DeclareFontShape`. Другими словами, когда загружается некоторый шрифт, NFSS вначале использует данные аргумента команды `\DeclareFontFamily`, а затем — шестого аргумента команды `\DeclareFontShape`, что в итоге приводит к перекрытию значений параметров загрузки шрифта для семейства шрифтов в целом.

Ниже анализируется информация, которая может формироваться подобным образом (к сожалению, не всегда она будет изменяемой), после чего обсуждается ряд полезных примеров. Эта часть интерфейса основывается на TEX'овских командах низшего уровня. Поскольку данные команды являются узкоспециализированными, попыток придать обсуждаемому интерфейсу LATEX'овский вид даже и не предпринималось. В связи с этим методы присваивания значений (целые числа, размеры) соответствующим параметрам могут выглядеть несколько необычно.

Команда (LA)TEX'а `\hyphenchar\font=`⟨*number*⟩ вводит обозначение для символа, который будет использоваться в качестве знака переноса при выполнении операции переноса слова. Параметр ⟨*number*⟩ представляет собой позицию слова в схеме кодирования. Значение по умолчанию здесь — это значение параметра `\defaulthyphenchar`, равное 45, что соответствует позиции символа '-' в большинстве схем кодирования. Если значение этого параметра положить равным −1, то перенос в словах будет подавляться. Таким образом, если записать декларацию вида

```
\DeclareFontFamily{OT1}{cmtt}{\hyphenchar\font=-1}
```

это приведет к отключению переносов во всех шрифтах семейства `cmtt` со схемой кодирования `OT1`. Шрифты с корковской кодировкой имеют альтернативный знак переноса в позиции 127, так что можно задать, например,

```
\DeclareFontFamily{T1}{dcmr}{\hyphenchar\font=127}
```

Это дает возможность использовать различные символы в качестве знака переноса и в качестве дефиса в составных словах, вроде «so-called». (LA)TEX не делает переносов в словах, содержащих в явном виде знаки переноса (как исключение — только сразу после знака переноса), что может породить серьезные проблемы для языков, в которых средняя длина слова существенно больше, чем в английском языке. Эта проблема может быть решена приведенными выше настройками.

С каждым (LA)TEX'овским шрифтом связан набор параметров-размеров, которые можно изменить командой присваивания вида `\fontdimen`⟨*number*⟩`\font=`⟨*dimen*⟩, где ⟨*number*⟩ — ссылочный номер для соответствующего размера, а ⟨*dimen*⟩ — присваиваемое значение. Значения этих размеров по умолчанию извлекаются из `.tfm`-файла при загрузке требуемо-

го шрифта. Каждый шрифт характеризуется по крайней мере семью такими размерами:

`\fontdimen1` Задает наклон шрифта на единицу его высоты 1pt. Если значение этого параметра равно нулю, шрифт имеет прямое начертание.

`\fontdimen2` Задает нормальную ширину межсловного промежутка (пробела между двумя смежными словами).

`\fontdimen3` Задает добавочную растяжимость для межсловного интервала, т. е. величину дополнительного пробела между словами, который (LA)TEX'у позволяется применить при выравнивании правого края абзаца формируемого документа. В крайнем случае допускается увеличение межсловного интервала и сверх этого значения, но тогда будет выдано предупреждение «underfull box».

`\fontdimen4` Задает добавочную сжимаемость межсловного интервала, т. е. величину пробела, которую (LA)TEX имеет право вычесть из нормального значения этого интервала (задаваемого параметром `\fontdimen2`) при выравнивании правого края абзаца. Ниже этого минимума (LA)TEX не будет уменьшать величину межсловного интервала никогда.

`\fontdimen5` Представляет собой так называемую x-высоту шрифта и определяет размер `1 ex`, зависящий от шрифта.

`\fontdimen6` Определяет так называемый «квадрат» (quad), равный `1 em` и зависящий от шрифта.

`\fontdimen7` Задает размер дополнительного промежутка, добавляемого после точки в конце предложения, если включен режим `\nonfrenchspacing`.

Изменяя межсловный интервал, связанный с некоторым шрифтом, нельзя задавать его значение в абсолютных величинах, поскольку оно должно быть применимо для всех размеров шрифтов, включенных в данную шрифтовую группу.

Следовательно, вы должны определить это значение, используя какие-то другие параметры, зависящие от шрифта. Можно было бы записать, например,

```
\DeclareFontShape{OT1}{cmr}{m}{n}{...}
  {\fontdimen2\font=.7\fontdimen2\font}
```

Эта декларация уменьшает нормальный межсловный интервал до 70% его первоначального значения. Аналогичным образом можно изменять значения растяжимости и сжимаемости.

Для некоторых шрифтов, используемых в формулах, семи параметров-размеров недостаточно. Это относится к шрифтам «`symbols`» и «`largesymbols`», рассматриваемых в разд. 7.7.6. TEX не может напечатать формулу, если для этих шрифтов не определены величины 22 и 13 параметров `\fontdimen` соответственно. Значения этих параметров используются для определения положения символов в математических формулах. Разъяснение смысла этих параметров выходит за рамки задач данной книги, требуемые подробности можно найти в приложении G книги TEXbook [30].

TEX имеет одну особенность — .tfm-файл в целях оптимизации расходования памяти загружается один раз для каждого размера. В связи с этим нельзя определить одну шрифтовую группу NFSS (командой \DeclareFontShape) для загрузки некоторого внешнего шрифта, например, cmtt10, и использовать другую команду \DeclareFontShape для загрузки точно такого же внешнего шрифта, но с измененными к этому моменту времени значениями параметров \fontdimen или любых других параметров, связанных с этим шрифтом. Попробуйте выполнить эти изменения значений для обеих шрифтовых групп.

Предположим, например, что вы пытаетесь определить начертание шрифта с уменьшенными значениями межсловных интервалов, просто уменьшая значение данного параметра, т. е., например,

```
\DeclareFontShape{T1}{times}{m}{n}{ <-> pstim }{}
\DeclareFontShape{T1}{times}{c}{n}{ <-> pstim }
                  {\fontdimen2\font=.7\fontdimen2\font}
```

Эта декларация работать не будет. При загрузке шрифта со сжатыми межсловными интервалами уменьшится и расстояние между словами для шрифта нормального начертания, а это не тот результат, который требовалось получить. Наиболее приемлемый способ решения данной проблемы состоит в том, чтобы определить виртуальный шрифт, который содержал бы те же самые символы, что и исходный шрифт, и отличался бы от него значениями требуемых размерных параметров (см. также разд. 9.1.1).

7.7.3 Параметры управления загрузкой шрифтов

Если вам необходимо объявить нестандартную шрифтовую группу для конкретного документа, просто поместите соответствующую декларацию в некоторый пакет или же в преамбулу документа. В таком варианте новая декларация перекроет все уже существующие декларации для соответствующих комбинаций атрибутов шрифтов. Здесь следует заметить, что использование команды \DeclareFontFamily предотвращает последующую загрузку соответствующего .fd-файла (см. разд. 7.7.5). На ранее загруженные шрифты вновь введенная декларация никакого влияния не окажет.

7.7.4 Ввод определений новых схем кодирования

Выполнение изменений в шрифтах, затрагивающих схему их кодирования, требует соблюдения некоторых предосторожностей. Например, в корковской кодировке для большинства букв с акцентами имеются отдельные соответствующие им значки (глифы), тогда как в традиционной TEX'овской текстовой кодировке акцентированные буквы получаются комбинированием буквы без акцента с соответствующим значком акцента с использованием команды-примитива \accent. (Кроме всего прочего, наличие отдельных глифов для акцентированных букв по

сравнению с подходом, основанным на использовании команды \accent, облегчает решение проблемы правильной расстановки переносов в словах). Если приходится смешивать оба этих подхода, например, из-за того, что не все требуемые в документе шрифты доступны в одной и той же кодировке, следует скорректировать определения таких команд, как \", поскольку их поведение будет изменяться со сменой шрифтов.

По этой причине каждая из схем кодирования должна быть формально определена в NFSS командой \DeclareFontEncoding. Эта команда имеет три аргумента. Первый представляет собой имя схемы кодирования, по которому она будет вызываться командой \fontencoding. Список стандартных LaTeX'овских кодировок и их внутренние имена в NFSS приводились в табл. 7.12. Второй аргумент содержит некоторый текст (такой, как определения), который должен исполняться всякий раз, когда NFSS производит переключение с одной кодировки на другую, используя команду \fontencoding. Последний, третий, аргумент включает код, используемый в тех случаях, когда производится переключение на один из математических алфавитов. Перечисленные аргументы могут быть использованы для переопределения команд, зависящих от позиции символа в рассматриваемой схеме кодирования. Чтобы исключить появление на печати лишних промежутков, обусловленных избыточными пробелами в таких аргументах, такие пробелы игнорируются. В маловероятном случае, когда все же потребуется поместить в аргумент пробелы, это можно сделать командой \space.

Начальные буквы T, O и M в обозначениях кодировок зарезервированы для стандартных схем кодирования, так что при вводе определения каких-либо собственных кодировок их использовать нельзя. В такой ситуации следует формировать имя схемы кодирования так, чтобы оно начиналось с буквы L (от Local — местный, локальный). Такой подход гарантирует мобильность файлов, использующих официально принятые кодировки. При появлении новых стандартных кодировок они будут добавлены в NFSS теми, кто отвечает за сопровождение данной подсистемы LaTeX'а.

Как это было видно из материала разд. 7.6.3, посвященного подстановкам шрифтов, в значениях по умолчанию для семейства, насыщенности и начертания шрифта может потребоваться учесть их зависимость от конкретной кодировки. По этой причине в NFSS есть команда \DeclareFontSubstitution, в которой первым аргументом будет имя схемы кодирования. Остальные три аргумента данной команды представляют собой значения по умолчанию семейства, насыщенности и начертания для данной схемы кодирования, которые будут использованы в процессе автоматической подстановки шрифтов, описанном в разд. 7.6.3.

7.7.5 Внутренняя организация файла

Семейства шрифтов могут быть объявлены при формировании LaTeX'овского форматного файла или же в преамбуле документа. Кроме того, эти семейства могут загружаться по мере надобности, когда команда переключения шрифта запрашивает шрифт с такой комбинацией значений атрибутов, которая до это-

го момента не использовалась. Если использовать первый из трех перечисленных выше вариантов, то внутренняя память системы будет расходоваться на размещение шрифтов в любом случае, в том числе и тогда, когда эти шрифты не нужны. Второй же и третий варианты ведут к некоторому увеличению времени обработки документа, поскольку чтение определений шрифтов теперь производится в ходе этой обработки. Несмотря на это обстоятельство, второй и третий варианты предпочтительнее первого, так как при одном и том же LaTeX'овском форматном файле появляется возможность обрабатывать широкий круг документов.

При генерации форматного файла LaTeX читает файл с именем `fontdef.ltx`, в котором должны содержаться определения семейств шрифтов, используемых обычно в документах. Все другие определения семейств шрифтов должны объявляться во внешних файлах, загружаемых при возникновении потребности в них, по запросу. Этими файлами могут быть либо файлы пакетов, либо файлы определений шрифтов (`.fd`-файлы; см. ниже).

Если определения семейств шрифтов размещаются в пакете, то команду загрузки такого пакета надо ставить после команды `\documentclass`. Однако существует еще и третья возможность. Пусть NFSS получает запрос на семейство шрифтов с именем `foo` и со схемой кодирования `BAR`, причем данная комбинация атрибутов NFSS неизвестна. В этой ситуации NFSS пытается загрузить файл `BARfoo..fd`. Если этот файл существует, то считается, что он содержит определения шрифтовых групп для семейства шрифтов `foo` со схемой кодирования `BAR`, т. е. декларации вида

```
\DeclareFontFamily{BAR}{foo}{..}
\DeclareFontShape{BAR}{foo}{..}{..}{..}{..}
  ...
\endinput
```

Таким способом можно определить в NFSS огромное количество шрифтов, не рискуя переполнить внутреннюю память системы информацией, которая скорее всего никогда не потребуется. К сожалению, этот подход не в полном объеме реализован в системах (LA)TeX, использующих различные способы поиска путей к файлам при выполнении команд `\input` и `\openin`. В таких системах указанное `.fd`-свойство можно активизировать при помощи NFSS на этапе начальной установки системы, если указать полностью путь, ведущий к каталогу, где размещаются `.fd`-файлы. В итоге локальные `.fd`-файлы, т. е. те, которые расположены в текущем каталоге, использоваться системой не будут. Это типичный случай для некоторых систем, работающих в среде операционной системы VMS, однако есть надежда, что такое ограничение в ближайшее время будет устранено.

Итак, каждый `.fd`-файл должен содержать все определения шрифтов для одного семейства и одной схемы кодирования. Он должен включать одну или более декларацию `\DeclareFontShape` и точно одну декларацию `\DeclareFontFamily`. Другие определения включать в такой файл не следует, за исключением, может быть, команд `\typeout`, информирующих пользователя о загрузке шрифтов. Вместо команды `\typeout` можно использовать команду `\wlog` из состава plain TeX'а,

которая записывает значение своего аргумента только в протокольный файл. Все вызываемые .fd-файлы должны помещать в протокольный файл информацию о загружаемых файлах и номерах их версий, поскольку такая информация может оказаться полезной для локализации ошибок в документе. Используя команду \typeout или \wlog, важно помнить, что пробелы и пустые строки в .fd-файле игнорируются. Поэтому, чтобы вывести в необходимое место экрана терминала или протокольного файла требуемое число пробелов, надо в соответствующих местах аргументов команд \typeout и \wlog использовать команду \space.

Нельзя вводить новые схемы кодирования , используя механизм .fd-файлов. NFSS отвергнет любой запрос на переключение кодировки, если требуемая схема кодирования не была объявлена в файле fontdef.ltx или в каком-либо пакете.

7.7.6 Объявление новых шрифтов для математических формул

Определение размеров шрифтов

Каждому размеру основного текстового шрифта документа NFSS ставит в соответствие три размера математических шрифтов, используемых при наборе формул (см. также разд. 8.9.1). Это размер, который устанавливается для большинства из символов в формуле (он выбирается командой \textstyle или \displaystyle); размер верхних и нижних индексов первого уровня (\scriptstyle); размер верхних и нижних индексов второго и последующих уровней (\scriptscriptstyle). Если размер основного шрифта текста изменяется на какой-либо другой, для которого соответствующие размеры математических шрифтов еще не известны, NFSS попытается вычислить их, основываясь на заданном размере основного шрифта. Чтобы не вынуждать NFSS проводить эти вычисления, можно задать требуемые размеры самостоятельно, используя команду \DeclareMathSizes. Эта декларация имеет четыре аргумента: базовый размер текстового шрифта и три размера для математических шрифтов, упомянутых выше. Например, файл класса документов compan, использованный для верстки оригинала данной книги (*The LaTeX Companion*), содержит декларации вида

```
\DeclareMathSizes{14}{14}{10}{7}    \DeclareMathSizes{36}{}{}{}
```

Первая из этих двух деклараций определяет размеры математических шрифтов при размере основного шрифта 14pt (они равны 14pt, 10pt и 7pt соответственно). Вторая декларация (размер шрифта для заголовков глав) информирует NFSS о том, что с текстовым шрифтом размером 36pt никаких математических шрифтов связывать не надо. Эта декларация предотвращает загрузку более чем 30 дополнительных шрифтов, которые никогда не понадобятся и привели бы к тому, что обработка книги *The LaTeX Companion* как единого документа стала бы невозможной. С отключением математических шрифтов ненужных размеров надо быть осторожным, поскольку если такой (отключенный) размер все-таки встре-

тится, то формула, в которой потребовались символы этого размера, будет печататься шрифтом (взамен недостающего), который остался активным от предыдущего режима работы, определяемого размером основного шрифта документа.

Добавление новых символьных шрифтов

Мы уже видели, как использовать команды переключения математических алфавитов, чтобы набирать в формулах символы требуемого начертания. Обсудим теперь, каким образом подключать шрифты, содержащие специальные символы, и как сделать эти символы доступными при наборе формул. Такие шрифты называются «символьными».

Процесс добавления новых символьных шрифтов подобен объявлению команд переключения математических алфавитов: команда `\DeclareSymbolFont` определяет значения по умолчанию для всех математических версий, а команда `\SetSymbolFont` перекрывает значения по умолчанию для конкретной версии.

Доступ к математическому символьному шрифту организуется посредством символьного имени,представляющего собой цепочку букв. Если, например, требуется подставить шрифты AMS с именем `msbm10`, показанные на рис. 7.8, его вначале надо объявить для NFSS, используя декларации, описанные в предыдущих разделах. Эти инструкции, включенные в состав некоторого пакета, могли бы выглядеть следующим образом:

```
\DeclareFontFamily{U}{msb}{}
\DeclareFontShape{U}{msb}{m}{n}{ <5> <6> <7> <8> <9> gen * msbm
      <10> <10.95> <12> <14.4> <17.28> <20.74> <24.88> msbm10}{}
```

Фактически лишь команда `\DeclareFontEncoding` должна быть определена в пакете, а остальные две можно поместить в `.fd`-файл. В таком случае вы должны объявить этот символьный шрифт для всех математических версий, используя для этой цели следующую команду:

```
\DeclareSymbolFont{AMSb}{U}{msb}{m}{n}
```

которая обеспечивает доступность шрифтовой группы `U/msb/m/n` в качестве символьного шрифта под именем `AMSb`. Если бы в данное семейство шрифтов входил шрифт полужирного начертания (к сожалению, на самом деле это не так), то можно было бы изменить настройку для полужирной математической версии следующей командой:

```
\SetSymbolFont{AMSb}{bold}{U}{msb}{b}{n}
```

После того как декларация, определяющая шрифт, задана, этот символьный шрифт можно использовать в математическом режиме работы LaTeX'а. Однако каким образом довести до сведения NFSS, например, что запись `$a\lessdot b$` должна обеспечить выдачу на печать выражения $a \lessdot b$? Чтобы это стало возможным, необходимо ввести в NFSS требуемые символьные имена-команды. Данная

Тип	*Значение*	*Пример*
\mathord	Обычный математический знак	/
\mathbin	Бинарный оператор	+
\mathopen	Открывающая скобка	(
\mathpunct	Знак препинания	,
\mathop	«Большой» оператор	\sum
\mathrel	Знак отношения	=
\mathclose	Закрывающая скобка	)
\mathalpha	Алфавитно-цифровой символ	A

Таблица 7.13. Типы математических символов

операция выполняется при помощи команды \DeclareMathSymbol. Например, команда \lessdot могла бы быть определена следующим образом:

```
\DeclareMathSymbol{\lessdot}{\mathbin}{AMSb}{"6C}
```

Первый аргумент в \DeclareMathSymbol представляет собой имя определяемой команды. Возможен другой вариант этого аргумента, когда вместо имени команды записывается единственный символ. Например, пакет euler содержит несколько деклараций вида

```
\DeclareMathSymbol{0}{\mathalpha}{letters}{"30}
```

которые задают, откуда следует выбирать цифры.

Второй аргумент — это одна из команд, перечисленных в табл. 7.13 и описывающих вид символа, т. е. является ли он бинарным оператором, знаком отношения и т. п. Эта информация необходима (LA)TEX'у, чтобы правильно формировать промежутки перед и после соответствующего символа, когда он входит в некоторую формулу. Все эти команды, кроме \mathalpha, можно непосредственно использовать в математических формулах как функции с одним аргументом. В таком случае аргумент той или иной функции, который может представлять собой сложное выражение, будет корректно отделен пробелами от соседних с ним частей математического выражения.

В третьем аргументе задается имя символьного шрифта, из которого будут извлекаться соответствующие символы, т. е. символьное имя, введенное командой \DeclareSymbolFont. Четвертый аргумент дает позицию рассматриваемого символа в схеме кодирования соответствующего шрифта, причем эта позиция может записываться в десятичном, восьмеричном или шестнадцатеричном виде. Восьмеричные (по основанию 8) и шестнадцатеричные (по основанию 16) числа обозначаются одинарной (') или двойной (") кавычкой соответственно, предшествующей данному числу. Позиция того или иного символа-значка в схеме кодирования определяется весьма просто, если представить ее в виде таблицы, пример

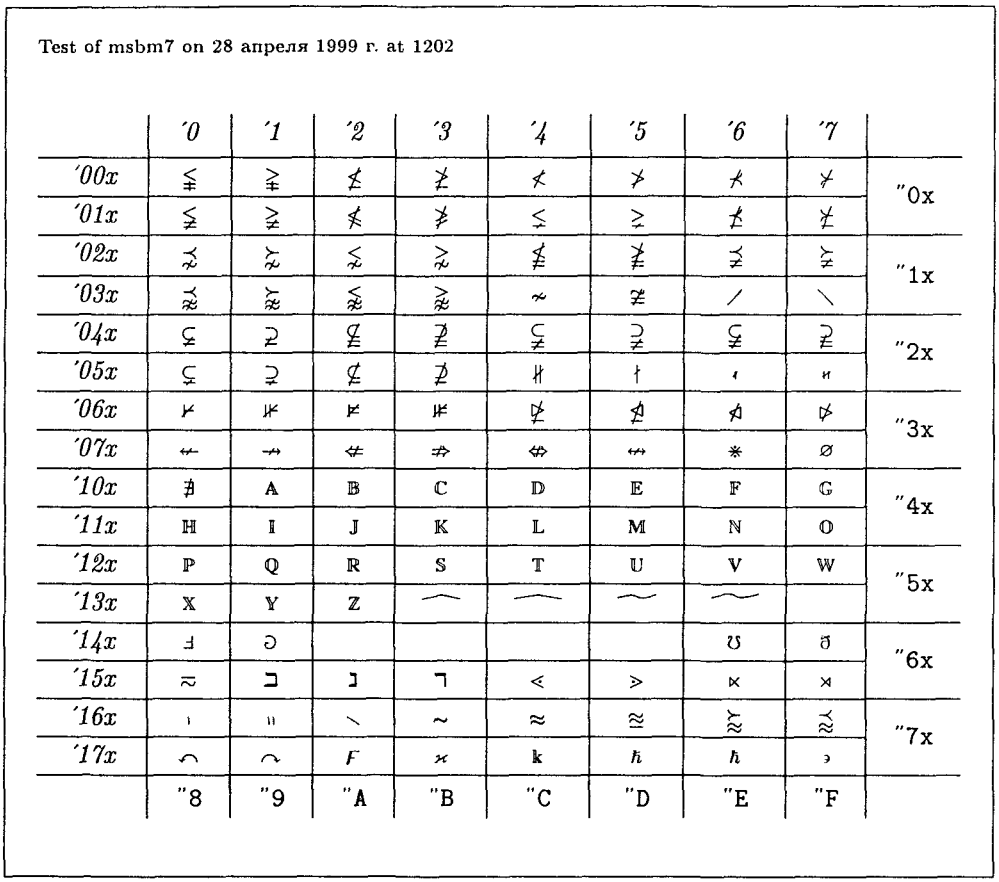


Рис. 7.8. Таблица кодировки для шрифта `msbm7`, полученная программой `nfssfont.tex`

которой показан на рис. 7.8. Такого рода таблицы можно строить при помощи LaTeX'овской программы `\nfssfont`, которая входит в комплект поставки пакета NFSS (см. также разд. 7.5.5).

Поскольку команда `\DeclareMathSymbol` используется для задания позиции знака в символьном шрифте, важно, чтобы все внешние шрифты, связанные с этим символьным шрифтом через команды `\DeclareSymbolFont` и `\SetSymbolFont`, содержали бы одни и те же символы в одних и тех же позициях. Простейший способ добиться этого — использовать только шрифты с одной и той же кодировкой (если только это не кодировка `U`, поскольку так помечаются «неизвестные» кодировки, которые могут меняться от одного шрифта к другому).

Из рис. 7.8 видно, что шрифт `msbm7` содержит так называемые «ажурные буквы» (blackboard letters), например, $\mathbb{ABC}$. Если имеется потребность применять

эти буквы в качестве символов математического алфавита, надо написать соответствующее определение, используя команду \DeclareMathAlphabet. Поскольку сам шрифт, содержащий ажурные буквы, уже известен системе, надо определить лишь команду вызова требуемых букв из него:

```
%%%\DeclareSymbolFontAlphabet{\mathbb}{AMSb} %Так в новом варианте
\DeclareSymbolFontAlphabet{\mathbb}{AMSb}    %Это старый вариант
```

Здесь \Bbb — команда вызова ажурной буквы, AMSb — ранее введенное имя символьного шрифта, содержащего такие буквы.

Важная причина, заставляющая избегать дублирования символьных шрифтов, заключается в том, что в любой момент времени в (LA)TEX'е активными могут быть не более 16 математических шрифтов. При подсчете этой величины учитывается каждый символьный шрифт, но математические алфавиты принимаются во внимание, только если они фактически используются в документе и, кроме того, по каждой математической версии они учитываются отдельно.

Итак, чтобы сделать доступными новые символьные шрифты, требуется только ввести соответствующее число деклараций \DeclareFont... и определенное (обычно значительно большее) число команд \DeclareMathSymbol; такого рода подключение лучше всего оформлять как пакет.

Введение новых математических версий

Как уже отмечалось, в стандартном варианте предопределены две математические версии — нормальная и полужирная. Если требуется ввести какие-то дополнительные версии, это можно сделать при помощи декларации \DeclareMathVersion. Эта декларация имеет один аргумент — имя определяемой версии. Все символьные шрифты и все математические алфавиты, введенные ранее, доступны и во вновь определенной версии, значения по умолчанию для них задаются декларациями \DeclareMathAlphabet или \DeclareSymbolFont.

Впоследствии настройки новой математической версии можно менять соответствующими командами \Set... так, как это было показано в предыдущих разделах для полужирной математической версии. Как и в случае со шрифтами, вводить новые математические версии лучше всего, оформляя их в виде пакетов.

Изменение настроек символьного шрифта

Кроме подключения новых символьных шрифтов, обеспечивающих возможность использования дополнительных символов, рассмотренные выше команды позволяют также изменять настройки шрифтов, объявленных ранее. Это свойство команд представляет значительный интерес в тех случаях, когда имеется необходимость сделать так, чтобы какие-то из специальных шрифтов были доступны в нескольких или даже во всех математических версиях.

Настройки по умолчанию имеют в NFSS следующий вид:

```
\DeclareMathVersion{normal}  \DeclareMathVersion{bold}
\DeclareSymbolFont{operators}      {OT1}{cmr}{m} {n}
\SetSymbolFont     {operators}{bold}{OT1}{cmr}{bx}{n}
\DeclareSymbolFont{letters}        {OML}{cmm}{m}{it}
\SetSymbolFont     {letters}  {bold}{OML}{cmm}{b}{it}
\DeclareSymbolFont{symbols}{OMS}{cmsy}{m}{n}
\DeclareSymbolFont{largesymbols}{OMX}{cmex}{m}{n}
```

В стандартном варианте настройки цифры и буквы, используемые для написания имен функций и операторов (так называемые «логарифмоподобные операторы», такие как `\log` и `\max`), берутся из символьного шрифта с именем `operators`.

Пусть, например, шрифт основного текста не Computer Modern Roman, а Computer Modern Sans. Тогда, чтобы добиться согласованности в начертании указанных выше элементов с основным текстом, надо воспользоваться следующими командами:

```
\SetSymbolFont{operators}{normal}{OT1}{cmss}{m} {n}
\SetSymbolFont{operators}{bold}  {OT1}{cmss}{bx}{n}
```

Символьные шрифты с именами `symbols` и `largesymbols` играют в TEX'е особую роль и по этой причине требуется, чтобы с ними были связаны параметры `\fontdimen` со специальными значениями. Из этого следует, что в качестве этих двух символьных шрифтов можно использовать только специально подготовленные шрифты. В принципе есть возможность добавить такие параметры к любому шрифту на этапе его загрузки, если воспользоваться третьим параметром в декларации `\DeclareFontFamily` или же шестым параметром в декларации `\DeclareFontShape`. Информация о специальных параметрах для этих символьных шрифтов содержится в приложении G книги [30].

7.7.7 Порядок записи деклараций

NFSS требует, чтобы все декларации, описывающие шрифты и действия с ними, следовали в строго определенном порядке, а также проверяет полноту задания информации, нужной для подключения шрифта. Если предписанный порядок будет нарушен, NFSS выдаст соответствующие сообщения об ошибках. Контролируются следующие взаимосвязи между декларациями, относящимися к математическим шрифтам:

- `\DeclareFontFamily` проверяет, была ли уже задана схема кодирования (при помощи декларации `\DeclareFontEncoding`);
- `\DeclareFontShape` проверяет, доступно ли указанное семейство шрифтов в требуемой кодировке (при помощи `\DeclareFontFamily`);

- `\DeclareSymbolFont` проверяет, правильна ли используемая схема кодирования;
- `\SetSymbolFont` дополнительно уточняет, определена ли уже запрашиваемая математическая версия (`\DeclareMathVersion`), а также определен ли уже требуемый символьный шрифт (`\DeclareSymbolFont`);
- `\DeclareSymbolFontAlphabet` проверяет, можно ли использовать имя соответствующей команды для идентификатора алфавита, а также определен ли уже требуемый символьный шрифт;
- `\DeclareMathAlphabet` контролирует возможность использования выбранного имени команды, а также проверяет, определена ли уже применяемая схема кодирования;
- `\SetMathAlphabet` проверяет, был ли уже ранее определен (при помощи `\DeclareMathAlphabet` или `\DeclareSymbolFontAlphabet`) идентификатор алфавита, а также смотрит, известны ли уже соответствующие математическая версия и схема кодирования;
- `\DeclareMathSymbol` уточняет, можно ли уже использовать соответствующее имя команды (т. е. смотрит, определен ли уже соответствующий математический символ), а также был ли уже ранее определен соответствующий символьный шрифт;
- и, наконец, когда в тексте документа встречена команда `\begin{document}`, NFSS предпринимает ряд дополнительных проверок, например, контроль того, соответствуют ли значения по умолчанию для шрифтовых подстановок для каждой схемы кодирования каким-либо ранее определенным шрифтовым группам.

7.8 Предупреждения и сообщения об ошибках

По-видимому, наиболее поразительное отличие старой схемы переключения шрифтов и NFSS состоит в появлении сообщений о переполнении боксов. Например, ранние черновые варианты этой книги[11] порождали сообщения наподобие следующего:

```
Overfull \hbox (14.13165pt too wide) in paragraph at lines 775--775
[]\OT1/cmtt/m/n/8 oldlfont.sty      {\errmessage{The package
'oldlfont' does not make sense if you[]
```

Видно, что это сообщение показывает фрагмент текста, вызвавший переполнение, с указанием использовавшегося внутреннего представления шрифта, в частности, `\OT1/cmtt/m/n/8`, т. е. NFSS отображает текущую ситуацию достаточно деталь-

[11] В оригинале. — *Прим. перев.*

но, показывая четко схему кодирования, семейство, насыщенность и начертание текстового шрифта, из-за которого возникло переполнение. Старый LaTeX в аналогичной ситуации выдал бы что-нибудь вроде `\\ptt\@viiipt` (в самом деле с двумя символами «\»), так что выдача NFSS в этой ситуации гораздо более информативна. Если, однако, бокс, вызывающий переполнение, содержит формульный материал, результат, к сожалению, становится совершенно неудобочитаемым, поскольку в формуле почти любой знак берется из специального шрифта. Например,

```
Overfull \hbox (10.01093pt too wide) detected at line 23
$\OML/cmm/m/it/10 A \OMS/cmsy/m/n/10 ^^T []  \OML/cmm/m/it/10 x$
```

есть символьное представление того, что (La)TeX отобразил бы в виде простой формулы

$A \leq \sum_{i=1}^{n} x$ `$A\leq\sum_{i=1}^n x$`

В таком случае лучше обратиться для анализа ситуации непосредственно к соответствующей строке исходного текста документа, используя ее номер (в данном случае 23), имеющийся в диагностическом сообщении.

Ниже приводится алфавитный список других предупреждений и сообщений об ошибках, выдаваемых NFSS, и объясняются наиболее вероятные причины появления этих сообщений. Предупреждающие сообщения обычно содержат в качестве префикса строку `Warning:` или `Info:`. Сообщения об ошибках преподносятся либо как LaTeX'овские, либо (если они являются следствием неверной генерации системы) как TeX'овские.

`Calculating math sizes for size` ⟨*text size*⟩
Размеры математических шрифтов вычислены для текстового шрифта размера ⟨*text size*⟩

NFSS пришлось сделать предположение относительно правильных размеров шрифтов для верхних и нижних индексов, поскольку информация о текущем размере текстового шрифта (⟨*text size*⟩) во внутренних таблицах не обнаружена. Это сообщение обычно сопровождается несколькими предупреждениями о корректировках размеров шрифтов, поскольку первоначальное предположение, использованное NFSS, редко бывает удачным. Такая ситуация возникает при выборе (командой `\fontsize`) необычного размера шрифта (см. также разд. 7.7.6).

`Checking defaults for` ⟨*cdp*⟩`/`⟨*font shape*⟩
Проверка значений по умолчанию для ⟨*cdp*⟩/⟨*font shape*⟩

Это сообщение записывается в протокольный файл, когда NFSS проверяет, имеют ли смысл подстановки по умолчанию для схемы кодирования ⟨*cdp*⟩. За ним следует либо `...okay`, либо сообщение об ошибке. Последний из этих двух случаев означает, что шрифтовая группа ⟨*font shape*⟩, задаваемая декларацией `\DeclareFontEncoding`, неизвестна NFSS.

`Command` ⟨*name*⟩ `already defined`
Команда ⟨*name*⟩ уже определена
NFSS выдает это сообщение, если вы попытаетесь объявить команду, которая была определена ранее. Вновь вводимая декларация для команды с именем ⟨*name*⟩ будет в этой ситуации отвергнута и для объявляемой команды надо будет выбрать другое имя, *отличное от* ⟨*name*⟩.

`Command` ⟨*name*⟩ `invalid in math mode`
Команда с именем ⟨*name*⟩ недопустима в математическом режиме
Это сообщение (оно может быть как предупреждением, так и сообщением об ошибке) означает, что имеет место попытка использования в математическом режиме команды, которая предназначена для применения ее только при наборе основного текста документа. В случае если появляется сообщение об ошибке, для получения дополнительной справочной информации надо нажать `h` .

`Command` ⟨*name*⟩ `not defined as a math alphabet`
Команда ⟨*name*⟩ не определена в качестве математического алфавита
Это сообщение об ошибке выдается в том случае, если вы попытаетесь использовать команду `\SetMathAlphabet` для команды с именем ⟨*name*⟩, не определенным ранее декларацией `\DeclareMathAlphabet` или `\DeclareSymbolFontAlphabet` в качестве идентификатора математического алфавита.

`Command \tracingfonts not provided`
Команда `\tracingfonts` не реализована
В преамбуле документа задано `\tracingfonts=...` , но пакет tracefnt из документа удален. Это сообщение представляет собой только предупреждение, которым можно пренебречь.

`Corrupted NFSS tables`
Ошибки в таблицах NFSS
Это сообщение выдается, когда NFSS при попытке выполнить шрифтовую подстановку обнаруживает цикл. Если команда `\DeclareErrorFont` была подключена правильно, можно продолжить работу, однако при появлении такой ошибки следует обратить на нее внимание специалиста, отвечающего за эксплуатацию данной системы[12].

`Encoding` ⟨*name*⟩ `has changed to` ⟨*new name*⟩ `for ...`
Кодировка с именем ⟨*name*⟩ изменена на кодировку с именем ⟨*new name*⟩
Это сообщение выдается в случае, если при объявлении символьного шрифта в различных математических версиях были использованы различные схемы кодирования. Это может означать, что команды `\DeclareMathSymbol` для дан-

[12] Декларация `\DeclareErrorFont` используется в процессе развертывания (генерации) системы и задает шрифт, который следует использовать в случае, если ни один другой не подходит. Более подробное описание этой ситуации содержится в документации по NFSS.

ного символьного шрифта не будут правильными для всех математических версий.

`Encoding scheme` ⟨*name*⟩ `unknown`
Схема кодирования с именем ⟨*name*⟩ неизвестна
Схема кодирования с именем ⟨*name*⟩, заданная в декларации или команде `\fontencoding`, не известна системе. Вы либо забыли объявить ее при помощи `\DeclareFontEncoding`, либо допустили ошибку при вводе ее имени.

`External font` ⟨*name*⟩ `loaded for size` ⟨*size*⟩
Внешний шрифт ⟨*name*⟩ загружен для размера ⟨*size*⟩
NFSS отвергла запрос на загрузку некоторого варианта шрифта с размером ⟨*size*⟩ и вместо него загрузила внешний шрифт с именем ⟨*name*⟩ (это сообщение выдается размерной функцией `fixed`).

`FontDef file:` ⟨*name*⟩ ...
Файл определений шрифтов: ⟨*name*⟩ ...
Такое сообщение может быть обнаружено в протокольном файле после запуска (LA)TEX'а. Здесь ⟨*name*⟩ — имя загруженного файла определений шрифтов. Подобные файлы содержат описания шрифтовых групп и рассматривались в разд. 7.7.5.

`Font family` ⟨*cdp*⟩+⟨*family*⟩ `unknown`
Семейство шрифтов ⟨*cdp*⟩+⟨*family*⟩ неизвестно
Такое сообщение появится, если имела место попытка объявить шрифтовую группу (декларацией `\DeclareFontShape`), не определив перед этим семейство шрифтов ⟨*family*⟩, доступное в кодировке ⟨*cdp*⟩, командой `\DeclareFontFamily`.

`Font` ⟨*name*⟩ `not found`
Шрифт ⟨*name*⟩ не найден
Из-за ошибки во внутренних таблицах NFSS не удалось найти внешний шрифт с именем ⟨*name*⟩.

`Font shape` ⟨*font shape*⟩ `in size` ⟨*size*⟩ `not available`
Шрифт начертания ⟨*font shape*⟩ размера ⟨*size*⟩ не доступен
Это сообщение выдается NFSS после неудачной попытки переключиться на требуемый шрифт, когда обнаружилось, что такой шрифт с заданной комбинацией значений его атрибутов недоступен. В зависимости от содержания внутренних таблиц NFSS за этим сообщением будут выданы дополнительно следующие:

`external font` ⟨*name*⟩ `used`
использован внешний шрифт с именем ⟨*name*⟩
NFSS в данной конкретной ситуации был выбран внешний шрифт с именем ⟨*name*⟩, однако неизвестно, к какой шрифтовой группе он относится (это сообщение выдается размерной функцией `subf`).

`size` ⟨*size*⟩ `substituted`
размер ⟨*size*⟩ заменен
NFSS выбран шрифт с правильным начертанием, но поскольку в требуемом размере он недоступен, была произведена замена его на шрифт с близким размером ⟨*size*⟩. Это действие выполняется автоматически, если ни один из имеющихся простых размеров или диапазонов размеров в шрифтовой группе ⟨*font shape*⟩ не удовлетворяет запросу.

`shape` ⟨*font shape*⟩ `tried`
испробовано начертание с именем ⟨*font shape*⟩
NFSS пришлось выбрать шрифтовую группу ⟨*font shape*⟩, отличающуюся от затребованной, поскольку был недоступен шрифт с необходимым размером ⟨*size*⟩ (это сообщение порождается размерной функцией `sub`).

`Font shape` ⟨*font shape*⟩ `will be scaled to size` ⟨*size*⟩
Шрифт с начертанием ⟨*font shape*⟩ будет промасштабирован до размера ⟨*size*⟩
NFSS информирует о том, что произведена загрузка затребованного шрифта с масштабированием его до необходимого размера. Результат такого действия удовлетворяет (LA)TEX, однако для печати документа, содержащего масштабированные шрифты, требуется, чтобы они либо были доступны программе печати в правильных размерах, либо эта программа должна располагать возможностью автоматического масштабирования шрифтов.

`Font shape` ⟨*font shape*⟩ `undefined. Using '`⟨*other shape*⟩`' instead`
Шрифт с начертанием ⟨*font shape*⟩ не определен.
Взамен использован шрифт с начертанием ⟨*other shape*⟩
Эта информация выдается NFSS, когда принимается решение о замене шрифта с одним начертанием на шрифт с другим начертанием.

`Font shape` ⟨*font shape*⟩ `not found`
Шрифт с начертанием ⟨*font shape*⟩ не найден
Это сообщение об ошибке выдается, если в декларации `\DeclareFontShape` содержатся какие-то серьезные ошибки, например, в ней вообще не заданы спецификации размеров шрифтов. В такой ситуации следует проверить информацию, по которой формируются шрифтовые группы.

`Math alphabet identifier` ⟨*id*⟩ `is undefined in math version` ⟨*name*⟩
Идентификатор математического алфавита ⟨*id*⟩ не определен
в математической версии с именем ⟨*name*⟩
Была предпринята попытка использовать в математической версии с именем ⟨*name*⟩ идентификатор математического алфавита ⟨*id*⟩, который в данной математической версии не определялся. При появлении такого сообщения необходимо добавить в преамбулу документа декларацию `\SetMathAlphabet`, связывающую шрифтовую группу с этим идентификатором.

`Math version` ⟨*name*⟩ `is not defined`
Математическая версия с именем ⟨*name*⟩ не определена

Обнаружена попытка использования математического алфавита или символьного шрифта в математической версии, которая неизвестна NFSS. Вы либо ошиблись в написании имени этой версии, либо забыли ее определить (может быть при помощи вызова соответствующего пакета). Не исключено, что математическая версия, переключение на которую задавалось командой `\mathversion`, вообще не описана в системе.

*** NFSS `release 1 command` ⟨*name*⟩ `found`
Обнаружена команда ⟨*name*⟩ из первой версии NFSS

Это сообщение может быть как предупреждением, так и сообщением об ошибке. Оно вызвано тем, что NFSS встретилась команда с именем ⟨*name*⟩, которая не поддерживается более во второй версии NFSS. В большинстве случаев NFSS способна преодолеть данную ситуацию, воспользовавшись подстановкой вместо команды NFSS1 соответствующей команды из NFSS2. Однако в любом случае следует внести исправления в исходный файл, содержащий обрабатываемый документ, чтобы исключить появление такого сообщения в дальнейшем.

`No declaration for shape` ⟨*font shape*⟩
Отсутствует декларация для начертания ⟨*font shape*⟩

Размерная функция `sub` или `ssub`, используемая в команде `\DeclareFontShape`, указывает на подстановку, выполненную из-за того, что задействован шрифт, не известный NFSS.

`Overwriting` ⟨*something*⟩ `in version` ⟨*name*⟩ ...
Изменения ⟨*something*⟩ в версии ⟨*name*⟩ ...

Декларации, подобные `\SetSymbolFont` или `\DeclareMathAlphabet` изменили значения начертаний шрифтов на ⟨*something*⟩ (имя символьного шрифта или математического алфавита) в математической версии с именем ⟨*name*⟩.

`Redeclaring math alphabet` ⟨*name*⟩
Переопределение математического алфавита ⟨*name*⟩

Для определения математического алфавита была выдана команда `\DeclareMathAlphabet` или `\DeclareSymbolFontAlphabet`, причем имя ⟨*name*⟩ уже было ранее определено в качестве идентификатора математического алфавита. Новая декларация перекрывает для ⟨*name*⟩ все ранее существовавшие установки.

`Redeclaring math symbol` ⟨*name*⟩
Переопределение математического символа ⟨*name*⟩

Команда с именем ⟨*name*⟩ уже была определена ранее как математический символ, поэтому вновь введенное определение для нее перекрывает существовавшее.

`Redeclaring math version` ⟨*name*⟩
Переопределение математической версии ⟨*name*⟩
Введена команда `\DeclareMathVersion` для определения математической версии с именем ⟨*name*⟩, которое уже использовано для тех же целей. Новая декларация заменит старую вместе со значениями ее параметров по умолчанию.

`Redeclaring symbol font` ⟨*name*⟩
Переопределение символьного шрифта ⟨*name*⟩
Обнаружена команда `\DeclareSymbolFont`, определяющая символьный шрифт с именем ⟨*name*⟩, которое уже известно системе. Новое определение перекрывает старое во всех математических версиях.

`Size substitution with differences to` ⟨*size*⟩ `have occured`
Произведены подстановки шрифтов с разницей размеров вплоть до ⟨*size*⟩
Это сообщение появляется в конце процесса обработки документа в случае, если NFSS будет обнаружена хотя бы одна подстановка шрифтов с существенной разницей в размерах замещающего и замещаемого шрифтов. Величина ⟨*size*⟩ обозначает максимальную разницу в размерах этих шрифтов для всего множества выполненных подстановок.

`Some font shapes were not available, defaults substituted`
Шрифты некоторых начертаний оказались недоступными,
были использованы подстановки значений по умолчанию
Это сообщение появляется в конце процесса обработки документа, если NFSS производилась автоматическая подстановка шрифтов.

`Symbol font` ⟨*name*⟩ `is not defined`
Символьный шрифт ⟨*name*⟩ не определен
Обнаружена попытка использования символьного шрифта с именем ⟨*name*⟩, например в команде `\DeclareMathSymbol`, без предварительного объявления этого шрифта декларацией `\DeclareSymbolFont`.

`The` ⟨*name*⟩ `package can only be used ...`
Пакет ⟨*name*⟩ может быть использован только ...
Это сообщение выдается всеми пакетами NFSS, если вы попытаетесь использовать их при неактивной системе NFSS. Здесь ⟨*name*⟩ — имя пакета, вызвавшего ошибку[13].

`This NFSS system isn't set up properly`
В системе NFSS обнаружена ошибка
Это сообщение выдается, когда NFSS обнаруживает ошибку при попытке проверки таблиц подстановки шрифтов. Такая ситуация возникает в случае, если нарушена декларация `\DeclareFontSubstitution` или `\DeclareErrorFont`. Для получения дополнительной информации введите в качестве ответа букву `h` и поставьте в известность специалиста, отвечающего за эксплуатацию

[13] Этой ошибки не может быть в LaTeX 2_ε, однако вполне вероятно ее появление в ранних редакциях NFSS, работающих совместно с LaTeX 2.09.

данной системы. Если таким специалистом являетесь вы сами, то прочитайте разд. 7.7.4.

`Try loading font information for` ⟨*cdp*⟩+⟨*family*⟩
Попытка загрузки информации о шрифте семейства ⟨*family*⟩ с кодировкой ⟨*cdp*⟩
Такое сообщение записывается в протокольный файл во всех случаях, когда NFSS пытается загрузить `.fd`-файл для заданной комбинации «кодировка / семейство» (⟨*cdp*⟩/⟨*family*⟩).

`Undefined font size function` ⟨*name*⟩
Шрифтовая размерная функция ⟨*name*⟩ не определена
Ошибка в написании имени размерной функции в декларации `\DeclareFontShape`. Проверьте правильность описания шрифтовых групп или проконсультируйтесь со своим специалистом, отвечающим за эксплуатацию системы.

`***` `Use` ⟨*command*⟩ `for` ⟨*old command*⟩ `***`
Используйте ⟨*command*⟩ вместо ⟨*old command*⟩
Исходный текст документа содержит команду ⟨*old command*⟩, которая является устаревшей в NFSS2, и вместо нее предлагается использовать команду ⟨*command*⟩.

ГЛАВА 8

Высшая математика

LaTeX в своей базовой комплектации предоставляет хорошие возможности для набора математических формул. Однако, если неоднократно требуется вводить сложные уравнения или другие сложные математические объекты, для облегчения набора каждый может для себя определять новые команды и окружения. Учитывая это обстоятельство, Американское математическое общество (The American Mathematical Society — AMS) субсидировало разработку расширений TeX'а, известных как $\mathcal{AMS}$-TeX. Его использование сокращает время на подготовку математических рукописей на компьютере и делает распечатки более совместимыми с издательскими требованиями.

Недавно такие же расширения были присоединены к LaTeX'у. Важно, однако, различать первоначальное не LaTeX'овское воплощение $\mathcal{AMS}$-TeX'а и его модифицированный вариант, который составляет LaTeX'овский пакет amsmath, широко известный как $\mathcal{AMS}$-LaTeX [19]. Рубленый шрифт будет использоваться для обозначения LaTeX'овского пакета amsmath, а стандартная эмблема $\mathcal{AMS}$-TeX будет применяться для исходной не LaTeX'овской версии.

8.1 Создание $\mathcal{AMS}$-LaTeX'а

Для повсеместного использования $\mathcal{AMS}$-TeX был выпущен в 1982 г. Основная его мощь состоит в том, что он упрощает набор математических формул, давая на выходе результат, который удовлетворяет высоким стандартам, принятым в математических издательствах. В нем заранее определены некоторые естественные команды, такие как `\matrix` и `\text`, что позволяет достаточно удобно набирать сложные математические выражения. При создании этих команд Американское математическое общество преследовало цель не обременять пользователя своими стандартами и заботами о тонкостях полиграфического оформления нештатных ситуаций, таких, например, как набор матриц внутри матриц или текста в качестве нижнего индекса.

В $\mathcal{AMS}$-TEX'е нет некоторых весьма удобных для авторов возможностей, которые есть в LATEX'е. К ним относятся автоматическая нумерация различных элементов документа, позволяющая не заботиться о результате при добавлении или удалении материала в исходном тексте, а также сокращающие затраты труда особенности LATEX'а, приспособленные для составления указателей, списков литературы, таблиц или простых диаграмм. Благодаря отмеченным удобствам, к середине 80-х годов LATEX приобрел широкую популярность (достаточно оформившаяся версия LATEX'а была доступна уже в конце 1983 г.), и в Американское математическое общество стали поступать предложения от авторов принимать электронные варианты их статей в LATEX'е.

Так, в 1987 г. возник проект создания $\mathcal{AMS}$-LATEX'а и три года спустя была выпущена его версия 1.0. Внедрение математических усовершенствований $\mathcal{AMS}$-TEX'а в LATEX и объединение с NFSS было проделано Франком Миттельбахом и Райнером Шопфом, работающими консультантами Американского математического общества, при содействии Майкла Даунза из Группы поддержки технического и программного обеспечения Американского математического общества.

Для того чтобы использовать возможности $\mathcal{AMS}$-LATEX'а, вам следует загрузить пакет amsmath при помощи команды \usepackage.

8.2 Шрифты и символы в формулах

8.2.1 Команды для математических шрифтов

Список команд для математических шрифтов, подключаемых пакетом amsmath, приведен в табл. 8.1, где также показаны примеры их использования. Кроме того, можно применять идентификаторы математических алфавитов из табл. 7.4.

Команду \boldsymbol из пакета amsmath следует применять для получения отдельных полужирных математических символов и полужирных греческих букв, т. е. любых математических символов, кроме букв (для которых нужно использовать команду \mathbf). Например, чтобы получить полужирные ∞, $+$, π или $\mathbf{0}$, следует соответственно ввести \boldsymbol{\infty}, \boldsymbol{+}, \boldsymbol{\pi} или \boldsymbol{0}.

Поскольку команда \boldsymbol довольно длинная для набора, при частом использовании полужирных символов вы можете ввести для них новые команды (макро):

$$B_\infty + \pi B_1 \sim \mathbf{B}_\infty + \pi\mathbf{B}_1$$

```
\newcommand{\bpi}{\boldsymbol{\pi}}
\newcommand{\binfty}{\boldsymbol{\infty}}
\[ B_\infty + \pi B_1 \sim
   \mathbf{B}_{\binfty} \boldsymbol{+}
   \bpi \mathbf{B}_{\boldsymbol{1}}
\]
```

Для тех математических символов, для которых команда \boldsymbol не действует из-за отсутствия полужирной версии символа в существующих шрифтах, имеется команда «полужирный шрифт для бедных» \pmb (от «Poor man's bold». — *Перев.*), которая воспроизводит полужирный символ, печатая несколько его ко-

пий с небольшими сдвигами. Эта команда должна применяться для символов больших операторов и символов из расширенного шрифта cmex, а также для символов из дополнительных математических шрифтов msam и msbm семейства $\mathcal{AMS}$.

$$\frac{\partial w}{\partial u}\Bigg|\frac{\partial u}{\partial v}$$

```
\[ \frac{\partial w}{\partial u}
   \pmb{\Bigg\vert}
   \frac{\partial u}{\partial v} \]
```

В настоящее время команда \pmb в применении к символам больших операторов (например, $\sum$ и $\prod$) и символам из расширенного шрифта cmex работает не очень хорошо, поскольку не до конца правильно расставляются пробелы и обрабатываются пределы в операторах. Поэтому следует использовать TEX'овскую команду \mathop (см. табл. 7.13).

$$\sum_{j<P}\prod_{\lambda}\lambda R(r_i) \qquad \sum_{x_j}\prod_{\lambda}\lambda R(x_j)$$

```
\[ \sum_{j<P}
   \prod_\lambda \lambda R(r_i) \qquad
   \mathop{\pmb{\sum}}_{x_j}
   \mathop{\pmb{\prod}}_\lambda
   \lambda R(x_j)
\]
```

Для того чтобы сделать полужирной всю математическую формулу (или возможно бóльшую ее часть, в зависимости от имеющихся шрифтов), перед этой формулой нужно поставить команду \boldmath.

Последовательность команд \mathbf{\hat{A}} дает полужирный акцент «крышку» над полужирной буквой **A**. Но такие комбинации, как \mathcal{\hat{A}}, не будут работать в обычном LaTeX'е, поскольку шрифт \mathcal не имеет собственных акцентов. В пакете amsmath команды изменения шрифтов определены таким образом, что символы акцентов будут браться из шрифта \mathrm, если текущий шрифт этих акцентов не имеет (наряду со шрифтом \mathcal шрифты \mathbb и \mathfrak также не имеют акцентов).

8.2.2 Математические символы

В табл. 8.3–8.12 ниже приведены математические символы, имеющиеся в стандартном LaTeX'е [$\mathcal{L}$ **42–47**], [$\mathcal{A}$ **43–51**]. Отрицание какого-либо LaTeX'овского символа вы можете получить, ставя перед ним команду \not [$\mathcal{L}$ **44**], [$\mathcal{A}$ **62**], например,

$u \not< v$ или $a \notin \mathbf{A}$ `$u \not< v$ или $a \not\in \mathbf{A}$`

В табл. 8.13–8.20 ниже приведены дополнительные математические символы из шрифтов семейства $\mathcal{AMS}$, которые автоматически становятся доступными, если вы загрузите пакет amssymb[1]. Однако если вы хотите определить лишь некоторые символы (например, в случае, когда TEX'у не хватает памяти для всех этих символов), то вы можете использовать пакет amsfonts и команду \DeclareMathSymbol, которая объяснена в разд. 7.7.6.

[1] Отметим, что в этой книге используются математические шрифты Lucida, которые содержат стандартные LaTeX'овские и $\mathcal{AMS}$-символы, но их начертание имеет другой вид по сравнению с математическими шрифтами семейства Computer Modern.

\mathbb	Ажурный алфавит; например, `$\mathbb{NQRZ}$` дает: $\mathbb{NQRZ}$ (определена в пакете amsfonts).
\boldsymbol	Используется для получения полужирных чисел и других небуквенных символов, а также для полужирных греческих букв (определена в пакете amsbsy).
\mathfrak	Готический алфавит; например, `$\mathfrak{E}=\mathfrak{mc}^2$` дает: $\mathfrak{E} = \mathfrak{mc}^2$ (определена в пакете amsfonts).
\pmb	«Полужирный шрифт для бедных» используется для математических символов, не имеющих полужирных вариантов; например, `$\pmb{\oint}$` дает: $\pmb{\oint}$, а `$\pmb{\triangle}$` дает: $\pmb{\triangle}$ (определена в пакете amsbsy).
\text	Дает обычный текст с правильной расстановкой пробелов в текущем шрифте вне математического режима; например, `$E=mc^2\quad\text{(Эйнштейн)}$` дает: $E = mc^2 \quad \text{(Эйнштейн)}$ (определена в пакете amstext).

Таблица 8.1. Идентификаторы математических алфавитов, подключаемые пакетом amsmath

$\hat{a}$	\hat–a″	$\acute{a}$	\acute–a″	$\bar{a}$	\bar–a″	$\dot{a}$	\dot–a″	$\breve{a}$	\breve–a″
$\check{a}$	\check–a″	$\grave{a}$	\grave–a″	$\vec{a}$	\vec–a″	$\ddot{a}$	\ddot–a″	$\tilde{a}$	\tilde–a″

Таблица 8.2. Акценты в математическом режиме

α	\alpha	β	\beta	γ	\gamma	δ	\delta	ϵ	\epsilon
ε	\varepsilon	ζ	\zeta	η	\eta	θ	\theta	ϑ	\vartheta
ι	\iota	κ	\kappa	λ	\lambda	μ	\mu	ν	\nu
ξ	\xi	o	o	π	\pi	ϖ	\varpi	ρ	\rho
ϱ	\varrho	σ	\sigma	ς	\varsigma	τ	\tau	υ	\upsilon
ϕ	\phi	φ	\varphi	χ	\chi	ψ	\psi	ω	\omega
Γ	\Gamma	Δ	\Delta	Θ	\Theta	Λ	\Lambda	Ξ	\Xi
Π	\Pi	Σ	\Sigma	Υ	\Upsilon	Φ	\Phi	Ψ	\Psi
Ω	\Omega								

Таблица 8.3. Греческие буквы

$\pm$	\pm	$\cap$	\cap	$\diamond$	\diamond	$\oplus$	\oplus
$\mp$	\mp	$\cup$	\cup	$\triangle$	\bigtriangleup	$\ominus$	\ominus
$\times$	\times	$\uplus$	\uplus	$\triangledown$	\bigtriangledown	$\otimes$	\otimes
$\div$	\div	$\sqcap$	\sqcap	$\triangleleft$	\triangleleft	$\oslash$	\oslash
$*$	\ast	$\sqcup$	\sqcup	$\triangleright$	\triangleright	$\odot$	\odot
$\star$	\star	$\vee$	\vee	$\lhd$[a]	\lhd	$\bigcirc$	\bigcirc
$\circ$	\circ	$\wedge$	\wedge	$\rhd$[a]	\rhd	$\dagger$	\dagger
$\bullet$	\bullet	$\setminus$	\setminus	$\unlhd$[a]	\unlhd	$\ddagger$	\ddagger
$\cdot$	\cdot	$\wr$	\wr	$\unrhd$[a]	\unrhd	$\amalg$	\amalg

[a] Не предопределены в NFSS. Используйте пакеты latexsym или amssymb.

Таблица 8.4. Символы бинарных операций

$\leq$	\leq\le	$\geq$	\geq	$\equiv$	\equiv	$\models$	\models	$\prec$	\prec
$\succ$	\succ	$\sim$	\sim	$\perp$	\perp	$\preceq$	\preceq	$\succeq$	\succeq
$\simeq$	\simeq	$\mid$	\mid	$\ll$	\ll	$\gg$	\gg	$\asymp$	\asymp
$\parallel$	\parallel	$\subset$	\subset	$\supset$	\supset	$\approx$	\approx	$\bowtie$	\bowtie
$\subseteq$	\subseteq	$\supseteq$	\supseteq	$\cong$	\cong	$\bowtie$	\Join	$\sqsubset$	\sqsubset
$\sqsupset$	\sqsupset	$\neq$	\neq	$\smile$	\smile	$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq
$\doteq$	\doteq	$\frown$	\frown	$\in$	\in	$\ni$	\ni	$\propto$	\propto
$=$	=	$\vdash$	\vdash	$\dashv$	\dashv	$<$	¡	$>$	¿

Таблица 8.5. Символы отношений

$\leftarrow$	\leftarrow	$\longleftarrow$	\longleftarrow	$\uparrow$	\uparrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow	$\Uparrow$	\Uparrow
$\rightarrow$	\rightarrow	$\longrightarrow$	\longrightarrow	$\downarrow$	\downarrow
$\Rightarrow$	\Rightarrow	$\Longrightarrow$	\Longrightarrow	$\Downarrow$	\Downarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow	$\updownarrow$	\updownarrow
$\Leftrightarrow$	\Leftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow	$\Updownarrow$	\Updownarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto	$\nearrow$	\nearrow
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow	$\searrow$	\searrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup	$\swarrow$	\swarrow
$\leftharpoondown$	\leftharpoondown	$\rightharpoondown$	\rightharpoondown	$\nwarrow$	\nwarrow

Таблица 8.6. Стрелки

$\ldots$	\ldots	$\cdots$	\cdots	$\vdots$	\vdots	$\ddots$	\ddots	$\aleph$	\aleph
$\prime$	\prime	$\forall$	\forall	∞	\infty	$\hbar$	\hbar	$\emptyset$	\emptyset
$\exists$	\exists	∇	\nabla	$\surd$	\surd	$\Box$	\Box[a]	$\triangle$	\triangle
$\Diamond$	\Diamond[a]	$\imath$	\imath	$\jmath$	\jmath	ℓ	\ell	$\neg$	\neg
$\top$	\top	$\flat$	\flat	$\natural$	\natural	$\sharp$	\sharp	$\wp$	\wp
$\bot$	\bot	$\clubsuit$	\clubsuit	$\diamondsuit$	\diamondsuit	$\heartsuit$	\heartsuit	$\spadesuit$	\spadesuit
$\mho$	\mho[a]	$\Re$	\Re	$\Im$	\Im	$\angle$	\angle	∂	\partial

[a] Не предопределены в NFSS. Используйте пакеты latexsym или amssymb.

Таблица 8.7. Разнородные символы

$\sum$	\sum	$\prod$	\prod	$\coprod$	\coprod	$\int$	\int	$\oint$	\oint
$\bigcap$	\bigcap	$\bigcup$	\bigcup	$\bigsqcup$	\bigsqcup	$\bigvee$	\bigvee	$\bigwedge$	\bigwedge
$\bigodot$	\bigodot	$\bigotimes$	\bigotimes	$\bigoplus$	\bigoplus	$\biguplus$	\biguplus		

Таблица 8.8. Символы больших операторов

```
\arccos  \cos   \csc  \exp  \ker     \limsup  \min  \sinh
\arcsin  \cosh  \deg  \gcd  \lg      \ln       \Pr   \sup
\arctan  \cot   \det  \hom  \lim     \log      \sec  \tan
\arg     \coth  \dim  \inf  \liminf  \max      \sin  \tanh
```

Таблица 8.9. Символы математических функций

$\uparrow$	\uparrow	$\Uparrow$	\Uparrow	$\downarrow$	\downarrow	$\Downarrow$	\Downarrow
$\{$	\{	$\}$	\}	$\updownarrow$	\updownarrow	$\Updownarrow$	\Updownarrow
$\lfloor$	\lfloor	$\rfloor$	\rfloor	$\lceil$	\lceil	$\rceil$	\rceil
$\langle$	\langle	$\rangle$	\rangle	/	/	$\backslash$	\backslash
$\vert$	\vert	$\Vert$	\Vert				

Таблица 8.10. Ограничители

$\rmoustache$	\rmoustache	$\lmoustache$	\lmoustache	$\rgroup$	\rgroup	$\lgroup$	\lgroup
$\arrowvert$	\arrowvert	$\Arrowvert$	\Arrowvert	$\bracevert$	\bracevert		

Таблица 8.11. Большие ограничители

$\widetilde{abc}$	\widetilde{abc}"	$\widehat{abc}$	\widehat{abc}"
$\overleftarrow{abc}$	\overleftarrow{abc}"	$\overrightarrow{abc}$	\overrightarrow{abc}"
$\overline{abc}$	\overline{abc}"	$\underline{abc}$	\underline{abc}"
$\overbrace{abc}$	\overbrace{abc}"	$\underbrace{abc}$	\underbrace{abc}"
$\sqrt{abc}$	\sqrt{abc}"	$\sqrt[n]{abc}$	\sqrt[n]{abc}
f'	f'	$\frac{abc}{xyz}$	\frac{abc}{xyz}

Таблица 8.12. Некоторые математические конструкции LaTeX'а

$\digamma$	\digamma	$\varkappa$	\varkappa	$\beth$	\beth	$\daleth$	\daleth	$\gimel$	\gimel

Таблица 8.13. Буквы греческого и еврейского алфавитов (имеются в amssymb)

$\ulcorner$	\ulcorner	$\urcorner$	\urcorner	$\llcorner$	\llcorner	$\lrcorner$	\lrcorner

Таблица 8.14. Символы типа ограничителей (имеются в amssymb)

⇢	\dashrightarrow	⇠	\dashleftarrow	⇇	\leftleftarrows
⇆	\leftrightarrows	⇚	\Lleftarrow	↞	\twoheadleftarrow
↢	\leftarrowtail	↫	\looparrowleft	⇋	\leftrightharpoons
↶	\curvearrowleft	↺	\circlearrowleft	↰	\Lsh
⇈	\upuparrows	↿	\upharpoonleft	⇃	\downharpoonleft
⊸	\multimap	↭	\leftrightsquigarrow	⇉	\rightrightarrows
⇄	\rightleftarrows	⇉	\rightrightarrows	⇄	\rightleftarrows
↠	\twoheadrightarrow	↣	\rightarrowtail	↬	\looparrowright
⇌	\rightleftharpoons	↷	\curvearrowright	↻	\circlearrowright
↱	\Rsh	⇊	\downdownarrows	↾	\upharpoonright
⇂	\downharpoonright	⇝	\rightsquigarrow		

Таблица 8.15. Стрелки семейства AMS (имеются в amssymb)

↚	\nleftarrow	↛	\nrightarrow	⇍	\nLeftarrow
⇏	\nRightarrow	↮	\nleftrightarrow	⇎	\nLeftrightarrow

Таблица 8.16. Отрицания стрелок семейства AMS (имеются в amssymb)

≦	\leqq	⩽	\leqslant	⪕	\eqslantless
≲	\lesssim	⪅	\lessapprox	≊	\approxeq
⋖	\lessdot	⋘	\lll	≶	\lessgtr
⋚	\lesseqgtr	⪋	\lesseqqgtr	≒	\doteqdot
≓	\risingdotseq	≒	\fallingdotseq	∽	\backsim
⋍	\backsimeq	⫅	\subseteqq	⋐	\Subset
⊏	\sqsubset	≼	\preccurlyeq	⋞	\curlyeqprec
≾	\precsim	⪷	\precapprox	◁	\vartriangleleft
⊴	\trianglelefteq	⊩	\vDash	⊪	\Vvdash
⌣	\smallsmile	⌢	\smallfrown	≏	\bumpeq
≎	\Bumpeq	≧	\geqq	⩾	\geqslant
⪖	\eqslantgtr	≳	\gtrsim	⪆	\gtrapprox
⋗	\gtrdot	⋙	\ggg	≷	\gtrless
⋛	\gtreqless	⪌	\gtreqqless	≗	\eqcirc
≗	\circeq	≜	\triangleq	∼	\thicksim
≈	\thickapprox	⫆	\supseteqq	⋑	\Supset
⊔	\sqsupset	≽	\succcurlyeq	⋟	\curlyeqsucc
≿	\succsim	⪸	\succapprox	▷	\vartriangleright
⊵	\trianglerighteq	⊨	\Vdash	∣	\shortmid
∥	\shortparallel	≬	\between	⋔	\pitchfork
∝	\varpropto	◀	\blacktriangleleft	∴	\therefore
϶	\backepsilon	▶	\blacktriangleright	∵	\because

Таблица 8.17. Бинарные отношения семейства AMS (имеются в amssymb)

$\not<$ \nless	$\not\leq$ \nleq	$\not\leqslant$ \nleqslant
$\not\leqq$ \nleqq	$\lneq$ \lneq	$\lneqq$ \lneqq
$\lvertneqq$ \lvertneqq	$\lnsim$ \lnsim	$\lnapprox$ \lnapprox
$\not\prec$ \nprec	$\not\preceq$ \npreceq	$\precnsim$ \precnsim
$\not\precapprox$ \precnapprox	$\not\sim$ \nsim	$\nshortmid$ \nshortmid
$\nmid$ \nmid	$\nvdash$ \nvdash	$\nvDash$ \nvDash
$\not\triangleleft$ \ntriangleleft	$\not\trianglelefteq$ \ntrianglelefteq	$\not\subseteq$ \nsubseteq
$\subsetneq$ \subsetneq	$\varsubsetneq$ \varsubsetneq	$\subsetneqq$ \subsetneqq
$\varsubsetneqq$ \varsubsetneqq	$\not>$ \ngtr	$\not\geq$ \ngeq
$\not\geqslant$ \ngeqslant	$\not\geqq$ \ngeqq	$\gneq$ \gneq
$\gneqq$ \gneqq	$\gvertneqq$ \gvertneqq	$\gnsim$ \gnsim
$\gnapprox$ \gnapprox	$\not\succ$ \nsucc	$\not\succeq$ \nsucceq
$\succnsim$ \succnsim	$\not\succapprox$ \succnapprox	$\not\cong$ \ncong
$\nshortparallel$ \nshortparallel	$\not\parallel$ \nparallel	$\not\nvDash$ \nvDash
$\not\nVDash$ \nVDash	$\not\triangleright$ \ntriangleright	$\not\trianglerighteq$ \ntrianglerighteq
$\not\supseteq$ \nsupseteq	$\not\supseteqq$ \nsupseteqq	$\supsetneq$ \supsetneq
$\supsetneqq$ \varsupsetneq	$\not\supseteqq$ \supsetneqq	$\varsupsetneqq$ \varsupsetneqq

Таблица 8.18. Отрицания бинарных отношений семейства AMS (имеются в amssymb)

$\dotplus$ \dotplus	$\smallsetminus$ \smallsetminus	$\Cap$ \Cap
$\Cup$ \Cup	$\barwedge$ \barwedge	$\veebar$ \veebar
$\doublebarwedge$ \doublebarwedge	$\boxminus$ \boxminus	$\boxtimes$ \boxtimes
$\boxdot$ \boxdot	$\boxplus$ \boxplus	$\divideontimes$ \divideontimes
$\ltimes$ \ltimes	$\rtimes$ \rtimes	$\leftthreetimes$ \leftthreetimes
$\rightthreetimes$ \rightthreetimes	$\curlywedge$ \curlywedge	$\curlyvee$ \curlyvee
$\circleddash$ \circleddash	$\circledast$ \circledast	$\circledcirc$ \circledcirc
$\centerdot$ \centerdot	$\intercal$ \intercal	

Таблица 8.19. Бинарные операторы семейства AMS (имеются в amssymb)

$\hbar$ \hbar	$\hslash$ \hslash	$\vartriangle$ \vartriangle
$\triangledown$ \triangledown	$\square$ \square	$\lozenge$ \lozenge
$\circledS$ \circledS	$\angle$ \angle	$\measuredangle$ \measuredangle
$\nexists$ \nexists	$\mho$ \mho	$\Finv$[a] \Finv
$\Game$[a] \Game	$\Bbbk$[a] \Bbbk	$\backprime$ \backprime
$\varnothing$ \varnothing	$\blacktriangle$ \blacktriangle	$\blacktriangledown$ \blacktriangledown
$\blacksquare$ \blacksquare	$\blacklozenge$ \blacklozenge	$\bigstar$ \bigstar
$\sphericalangle$ \sphericalangle	$\complement$ \complement	$\eth$ \eth
$\diagup$[a] \diagup	$\diagdown$[a] \diagdown	

[a] Не определены в старых версиях пакета amssymb; определяются при помощи команды `\DeclareMathSymbol`.

Таблица 8.20. Разнородные символы семейства AMS (имеются в amssymb)

8.3 Составные символы, ограничители и операторы

В этом разделе[2] представлены команды для математического набора из пакета amsmath, дополняющие LaTeX составными символами, большими ограничителями и т. п. В следующих ниже примерах используются окружения выравнивания из пакета amsmath. В принципе, на данном этапе детальное понимание того, как работают приводимые конструкции, не обязательно, а заинтересованный читатель может обратиться к разд. 8.5 за дополнительной информацией.

8.3.1 Кратные интегралы

Команды `\iint`, `\iiint` и `\iiiint` дают знаки кратных интегралов с правильными пробелами между отдельными знаками интеграла как в формулах в сплошном тексте, так и в выключных формулах. Команда `\idotsint` проставляет два знака интеграла с тремя центрированными точками между ними.

$$\iint\limits_V \mu(u,v)\,du\,dv \qquad (8.1)$$

$$\iiint\limits_V \mu(u,v,w)\,du\,dv\,dw \qquad (8.2)$$

$$\iiiint\limits_V \mu(t,u,v,w)\,dt\,du\,dv\,dw \qquad (8.3)$$

$$\int\limits_V\cdots\int \mu(u_1,\dots,u_k) \qquad (8.4)$$

```
\begin{gather}
\iint\limits_V \mu(u,v)\,du\,dv          \\
\iiint\limits_V \mu(u,v,w)\,du\,dv\,dw   \\
\iiiint\limits_V \mu(t,u,v,w)\,
                        dt\,du\,dv\,dw   \\
\idotsint\limits_V \mu(u_1,\dots,u_k)
\end{gather}
```

8.3.2 Стрелки сверху и снизу

Имеются дополнительные операции, проставляющие стрелки сверху и снизу формул. (В стандартном LaTeX'е применяются команды `\overrightarrow` и `\overleftarrow`.)

$$\overrightarrow{\psi_\delta(t)E_t h} = \underrightarrow{\psi_\delta(t)E_t h}$$

$$\overleftarrow{\psi_\delta(t)E_t h} = \underleftarrow{\psi_\delta(t)E_t h}$$

$$\overleftrightarrow{\psi_\delta(t)E_t h} = \underleftrightarrow{\psi_\delta(t)E_t h}$$

```
\begin{align*}
 \overrightarrow{\psi_\delta(t) E_t h}     &
=\underrightarrow{\psi_\delta(t) E_t h}    \\
 \overleftarrow{\psi_\delta(t) E_t h}      &
=\underleftarrow{\psi_\delta(t) E_t h}     \\
 \overleftrightarrow{\psi_\delta(t) E_t h} &
=\underleftrightarrow{\psi_\delta(t) E_t h}
\end{align*}
```

В индексах длина всех стрелок меняется пропорционально:

$\int_{\overrightarrow{uv}} vt\,dt$

```
$ \int_{\overrightarrow{uv}} vt\,dt $
```

[2] С разрешения Американского математического общества часть материала этого и следующего разделов заимствована из электронного документа `testmath.tex` (распространяемого вместе с $\mathcal{AMS}$-LaTeX'ом).

8.3.3 Многоточия

Многоточие должно почти всегда набираться как \dots. Положение точек (на базовой линии или по центру) автоматически выбирается в зависимости от того, что следует за командой \dots. Если следующий символ — знак плюс, то точки будут центрированы, а если запятая, то они будут помещены на базовую линию. При желании такое размещение многоточия, принятое по умолчанию в пакете amsmath, может быть изменено.

Если многоточие попадает в конец математической формулы, так что следующий символ есть что-то вроде \end, \) или $, то нет никакой информации о том, как разместить многоточие. В этом случае следует использовать команду \dotsc для «многоточия после запятой», \dotsb для «многоточия после бинарной операции или отношения», \dotsm для «многоточия вместо умножения» или \dotsi для «многоточия вместо интеграла». В приводимом примере первая из перечисленных выше команд дает многоточие на базовой линии, а остальные команды дают центрированные многоточия с правильными пробелами вокруг этих многоточий.

Последовательность $H_1, H_2, \dots$, бесконечная сумма $H_1 + H_2 + \cdots$, ортогональное произведение $H_1 H_2 \cdots$ и бесконечный интеграл

$$\int_{H_1}\int_{H_2}\cdots.$$

```
Последовательность $H_1,H_2,\dotsc$,
бесконечная сумма $H_1+H_2+\dotsb$,
ортогональное произведение $H_1H_2\dotsm$ и
бесконечный интеграл
\[ \int_{H_1}\int_{H_2}\dotsi. \]
```

8.3.4 Двойные акценты

Следующие команды правильным образом и автоматически проставляют двойной акцент:

$$\acute{\acute{A}} \quad \bar{\bar{B}} \quad \check{\check{C}} \quad \breve{\breve{D}}$$
$$\ddot{\ddot{E}} \quad \dot{\dot{F}} \quad \grave{\grave{G}} \quad \hat{\hat{H}}$$
$$\tilde{\tilde{I}} \quad \vec{\vec{J}}$$

```
\begin{gather*}
\Acute{\Acute{A}} \qquad\Bar{\Bar{B}} \qquad
\Breve{\Breve{C}} \qquad\Check{\Check{D}}  \\
\Ddot{\Ddot{E}}   \qquad\Dot{\Dot{F}} \qquad
\Grave{\Grave{G}} \qquad\Hat{\Hat{H}}      \\
\Tilde{\Tilde{I}} \qquad\Vec{\Vec{J}}
\end{gather*}
```

Операция двойного акцента является трудоемкой, что приводит к замедлению обработки LaTeX'овского файла. Если ваш документ содержит много символов с двойными акцентами, можете загрузить пакет amsxtra. В нем определяется команда \accentedsymbol, которую следует указывать в преамбуле вашего документа. Это способствует более быстрой обработке файла. Результат команды, проставляющей двойной акцент, сохраняется в некотором регистре, что дает быстрый доступ к символу с двойным акцентом. Используется команда \accentedsymbol так же, как и \newcommand:

$\hat{\hat{A}}$ — это двойная крышка над A, а это — $\ddot{\delta}$ — дельта с черточкой и точкой.

```
\accentedsymbol{\Ahathat}{\Hat{\Hat A}}
\accentedsymbol{\dbardot}{\Dot{\Bar\delta}}
\(\Ahathat\) "---это двойная крышка над \(A\), а это
"--- \(\dbardot\) "--- дельта с черточкой и точкой.
```

8.3.5 Акценты как верхние индексы

У ряда акцентов есть широкие варианты: так, `$\widehat{xy},\widetilde{xy}$` дает $\widehat{xy}$, $\widetilde{xy}$. Так как «широкие акценты» имеют некоторый максимальный размер, для слишком длинных выражений в пакете amsxtra предусмотрены другие обозначения: например, $(AmBD)\hat{}$ вместо $\widehat{AmBD}$. Чтобы было легче проставлять акценты, в пакете amsxtra имеются следующие командные последовательности:

$$(AmBD)\hat{} \qquad (AmBD)\check{} \qquad (8.5)$$

$$(AmBD)\tilde{} \qquad (AmBD)\dot{} \qquad (8.6)$$

$$(AmBD)\ddot{} \qquad (AmBD)\dddot{} \qquad (8.7)$$

$$(AmBD)\breve{} \qquad (8.8)$$

```
\begin{gather}
(AmBD)\sphat   \qquad (AmBD)\spcheck \\
(AmBD)\sptilde \qquad (AmBD)\spdot   \\
(AmBD)\spddot  \qquad(AmBD)\spdddot  \\
(AmBD)\spbreve
\end{gather}
```

8.3.6 Акценты в виде точек

В дополнение к уже имеющимся в LaTeX'е командам `\dot` и `\ddot` существуют команды `\dddot` и `\ddddot`, которые помещают соответственно три и четыре точки над символом:

$\dddot{Q} \qquad \ddddot{R}$

```
$  \dddot{Q}  \qquad  \ddddot{R}  $
```

8.3.7 Корни

В обычном LaTeX'е показатели (индексы) корня иногда располагаются не очень удачно. Корректировку положения показателя корня позволяют осуществить команды `\leftroot` и `\uproot` из пакета amsmath. Если аргумент у этих команд положителен, то показатель будет сдвинут влево и вверх соответственно, а если аргумент отрицателен — то вправо и вниз. Единицы измерения этих сдвигов достаточно малы и специально приспособлены для таких корректировок. В следующем примере показатель β сдвинут на 2 единицы влево и на 4 единицы вверх:

$$\sqrt[\beta]{k} \qquad \sqrt[\leftroot{2}\uproot{4}\beta]{k}$$

```
\[ \sqrt[\beta]{k} \qquad
   \sqrt[\leftroot{2}\uproot{4}\beta]{k} \]
```

8.3.8 Формулы в рамке

Команда `\boxed` рисует рамку вокруг своего аргумента подобно команде `\fbox`, но с содержимым в математическом режиме:

$$\boxed{W_t - F \subseteq V(P_i) \subseteq W_t}$$

```
\[ \boxed{W_t - F \subseteq
          V(P_i)  \subseteq W_t} \]
```

8.3.9 Растяжимые стрелки

Команды `\xleftarrow` и `\xrightarrow` воспроизводят стрелки, которые автоматически растягиваются, чтобы охватить необычно широкие нижние или верхние индексы. Текст нижнего или верхнего индекса набирается соответственно как необязательный и обязательный аргумент:

$$0 \xleftarrow[\zeta]{\alpha} F \times \triangle[n-1] \xrightarrow{\partial_0\alpha(b)} E^{\partial_0 b}$$

```
\[0 \xleftarrow[\zeta]{\alpha}
    F\times\triangle[n-1]
    \xrightarrow{\partial_0\alpha(b)}
    E^{\partial_0b}
\]
```

8.3.10 Команды \overset, \underset и \sideset

Для того чтобы поместить символ над бинарным отношением, в LaTeX'е предусмотрена команда `\stackrel`. В пакете amsmath определяются несколько более общие команды `\overset` и `\underset`. Их можно использовать для того, чтобы поставить один символ над или под другим независимо от того, отношение это или что-то еще. Команда `\overset{*}{X}` расположит символ $*$ такого же размера, как индекс, над буквой X; команда `\underset` осуществляет операцию, которую можно ожидать из ее названия (overset — «поместить над», underset — «поместить под». — *Перев.*)

$$\overset{*}{X} \qquad \underset{*}{X} \qquad \overset{a}{\underset{b}{X}}$$

```
\[ \overset{*}{X}           \qquad
   \underset{*}{X}          \qquad
   \overset{a}{\underset{b}{X}}   \]
```

Имеется также команда `\sideset` специального назначения: она помещает символ в позицию нижнего или верхнего индекса у больших операторов, таких как $\sum$ и $\prod$. Основной пример — это когда вы хотите поставить штрих у знака суммы. Если сумма не содержит пределов суммирования, можно просто указать `\nolimits`:

$$\sum\nolimits' E_n. \tag{8.9}$$

```
\begin{equation}
   \sum\nolimits' E_n.
\end{equation}
```

Но если вы хотите не только поставить штрих у знака суммы, но и указать пределы суммирования, то все не так просто. Предположим, что вы хотите добавить штрих к знаку суммы в следующем выражении:

$$\sum_{n<k,\ n \text{ нечетное}} nE_n; \tag{8.10}$$

```
\begin{equation}
   \sum_{n<k,\;n\ \mathrm{нечетное}}nE_n;
\end{equation}
```

тогда используйте \sideset, например, так: \sideset{}{'}\sum_{...}nE_n. Дополнительная пара фигурных скобок, ограничивающая пустую группу, объясняется тем, что \sideset имеет возможность ставить символ или символы в индексы со всех сторон большого оператора.

$$\sideset{_1^2}{_3^4}\prod_k \qquad \sideset{}{'}\sum_{0\le i\le m} E_i\beta x$$

```
\[\sideset{_1^2}{_3^4}\prod_k          \qquad
  \sideset{}{'}\sum_{0\le i\le m} E_i\beta x
\]
```

8.3.11 Команда \smash

Команда \smash plain TEX'а сохраняет содержимое бокса, в котором находится ее аргумент, но пренебрегает высотой этого бокса и его глубиной. В пакете amsmath для команды \smash предусмотрен необязательный аргумент, который может принимать значения t и b. Команда \smash[t]{...} пренебрегает только верхней (выше базовой линии) частью бокса, сохраняя нижнюю его часть, тогда как \smash[b]{...} пренебрегает нижней (ниже базовой линии) частью и сохраняет верхнюю часть содержимого бокса.

$$X_j = (1/\sqrt{\lambda_j})X_j' \qquad X_j = (1/\sqrt{\lambda_j})X_j$$

```
\[ X_j=(1/\sqrt{\smash[b]{\lambda_j}})X_j'
  \qquad  X_j=(1/\sqrt{\lambda_j})X_j    \]
```

В этом примере команда \smash используется с целью ограничить выступ радикала вниз, который в противном случае простирается ниже настолько, чтобы охватить также и индекс j (в равенстве справа).

8.3.12 Команда \text

Команда \text, определенная также в отдельном пакете amstext, используется главным образом для того, чтобы вставить одно или несколько слов в выделенную формулу. По своему действию она похожа на LATEX'овскую команду \mbox, но имеет некоторые преимущества. Если вы хотите поместить некоторый текст в индекс, то можете ввести

```
..._{\text{некоторый текст}}
```

и правильный размер содержимого индекса будет автоматически выбран. Команда \text не только более соответствует своему названию, но ею легче и пользоваться, чем эквивалентной командой \mbox:

```
..._{\mbox{\scriptsize некоторый текст}}
```

Еще один пример:

$$\mathbf{y}=\mathbf{y}' \quad \text{тогда и только тогда, когда} \quad y'_k=\delta_k y_{\tau(k)}$$

```
\[ \mathbf{y}=\mathbf{y}' \quad
   \text{тогда и только
            тогда, когда} \quad
   y'_k=\delta_k y_{\tau(k)} \]
```

8.3.13 Названия новых операций[3]

Названия математических функций и операций (называемых здесь также операторами) типа log, sin и lim традиционно набираются прямым шрифтом во избежание путаницы с отдельными переменными, которые набираются в формулах курсивом. Для наиболее общепринятых функций определены соответствующие им команды: `\log`, `\sin`, `\lim` и т. д. (см. табл. 8.9). Но в математической литературе постоянно возникает необходимость в новых названиях. В пакете amsmath предусмотрены команды `\DeclareMathOperator` и `\DeclareMathOperator*` для создания имен новых операторов, которые будут иметь требуемое начертание. Например, `\DeclareMathOperator{\xyz}{xyz}` дает xyz в правильно выбранном шрифте и при необходимости расставляет правильные пробелы вокруг названия, так что получается $A\operatorname{xyz}B$, а не $A\operatorname{xyz}B$. Ниже приводятся примеры определений новых операторов (команда `\,` в определении `\esssup` вставляет некоторое дополнительное пространство; см. табл. 8.21).

Исходный текст

```
\DeclareMathOperator*{\esssup}{ess\,sup}
\DeclareMathOperator{\meas}{meas}
\newcommand{\abs}[1]{\lvert#1\rvert}
\newcommand{\norm}[1]{\lVert#1\rVert}
\begin{align*}
\norm{f}_\infty                                          &=
\esssup_{x\in R^n}\abs{f(x)}                               \\
\meas_1\{u\in R_+^1\colon f^*(u)>\alpha\}                &=
\meas_n\{x\in R^n\colon\abs{f(x)}\geq\alpha\}\qquad\forall\alpha>0.
\end{align*}
```

$$\|f\|_\infty = \operatorname*{ess\,sup}_{x\in R^n}|f(x)|$$
$$\operatorname{meas}_1\{u\in R_+^1\colon f^*(u)>\alpha\} = \operatorname{meas}_n\{x\in R^n\colon |f(x)|\geq\alpha\} \qquad \forall\alpha>0.$$

Текст на выводе

Вариант этой команды со звездочкой `\DeclareMathOperator*` имеет единственное отличие, как видно из приведенного примера: размещение нижних и верхних индексов. Обратите также внимание на использование команд `\lvert`, `\rvert`, `\lVert` и `\rVert` (в сравнении с командами `\langle` и `\rangle`), которые делают применение вертикальных черточек более гибким.

[3] В оригинале Operator Names. — *Прим. перев.*

В пакете amsmath уже определены операции \varlimsup, \varliminf, \varinjlim и \varprojlim. Вот как они работают:

$$\overline{\lim}_{n\to\infty} Q(u_n, u_n - u^{\#}) \le 0 \qquad (8.11)$$

$$\underline{\lim}_{n\to\infty} |a_{n+1}| / |a_n| = 0 \qquad (8.12)$$

$$\varinjlim (m_i^{\lambda}\cdot)^* \le 0 \qquad (8.13)$$

$$\varprojlim_{p\in S(A)} A_p \le 0 \qquad (8.14)$$

```
\begin{gather}
\varlimsup_{n\rightarrow\infty}
 \mathcal{Q}(u_n,u_n-u^{\#}) \le 0      \\
\varliminf_{n\rightarrow\infty}
 \left|a_{n+1}\right|/\left|a_n\right|=0\\
\varinjlim (m_i^\lambda\cdot)^* \le 0   \\
\varprojlim_{p\in S(A)}A_p \le 0
\end{gather}
```

8.3.14 Команда \mod и ее аналоги

Команды \mod, \bmod, \pmod и \pod предусмотрены для расстановки специфических пробелов, традиционно принятых в обозначениях сравнения по модулю. Команды \bmod и \pmod имеются и в LaTeX'e, но при использовании пакета amsmath команда \pmod ставит меньшие пробелы в невыключной формуле. Некоторые пользователи предпочитают варианты \pmod — команды \mod и \pod: \mod опускает круглые скобки в обозначении, а \pod опускает само слово «mod», но оставляет скобки.

$$u \equiv v+1 \pmod{n^2} \qquad (8.15)$$

$$u \equiv v+1 \mod n^2 \qquad (8.16)$$

$$u \equiv v+1 \pod{n^2} \qquad (8.17)$$

```
\begin{align}
  u &\equiv v + 1 \pmod{n^2}  \\
  u &\equiv v + 1 \mod{n^2}   \\
  u &\equiv v + 1 \pod{n^2}
\end{align}
```

8.3.15 Дроби и родственные конструкции

В дополнение к команде \frac (имеющейся в LaTeX'e) пакет amsmath предоставляет команды \dfrac и \tfrac как удобные сокращения для {\displaystyle\frac ... } и {\textstyle\frac ... }.

$$\frac{1}{k}\log_2 c(f) \quad \tfrac{1}{k}\log_2 c(f)$$

и

$$\sqrt{\tfrac{1}{k}\log_2 c(f)} \quad \sqrt{\frac{1}{k}\log_2 c(f)}.$$

```
\[ \frac{1}{k}\log_2 c(f)\quad
   \tfrac{1}{k}\log_2 c(f)      \]
и
$  \sqrt{\frac{1}{k}\log_2 c(f)}\quad
   \sqrt{\dfrac{1}{k}\log_2 c(f)}    $.
```

Для биномиальных коэффициентов вида $\binom{n}{k}$ в пакете amsmath определены команды \binom, \dbinom и \tbinom.

$$\binom{k}{1}2^{k-1} + \tbinom{k}{2}2^{k-2} \qquad (8.18)$$

и

$$\tbinom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2}.$$

```
\begin{equation}
\binom{k}{1}2^{k-1}+\tbinom{k}{2}2^{k-2}
\end{equation}
и
$\binom{k}{1}2^{k-1}+\dbinom{k}{2}2^{k-2}$.
```

Команда \binom и ее варианты \dbinom и \tbinom, также как и команда \frac и ее варианты \dfrac и \tfrac, основаны на общей команде \genfrac, порождающей дроби и имеющей шесть параметров.

\genfrac{*ldelim*}{*rdelim*}{*thick*}{*style*}{*num*}{*denom*}

Первые два параметра *ldelim* и *rdelim* — это соответственно левый и правый ограничители. Третий параметр *thick* позволяет игнорировать толщину черты, разделяющей числитель и знаменатель (например, команда \binom использует этот параметр для того, чтобы установить толщину черты равной нулю, т. е. сделать невидимой). Если этот аргумент пуст, то толщина черты по умолчанию является «естественной». Четвертый параметр контролирует стиль для набора математики. Он может принимать целое значение между 0 и 3 и означает соответственно стили \displaystyle, \textstyle, \scriptstyle и \scriptscriptstyle. Наконец, пятый аргумент *num* является числителем дроби, а шестой аргумент *denom* — ее знаменателем.

В качестве иллюстрации покажем, каким образом могут быть определены команды \frac, \tfrac и \binom:

```
\newcommand{\frac}[2]{\genfrac{}{}{}{}{#1}{#2}}
\newcommand{\tfrac}[2]{\genfrac{}{}{}{1}{#1}{#2}}
\newcommand{\binom}[2]{\genfrac{(}{)}{0pt}{}{#1}{#2}}
```

Другие примеры — это переопределение TEX'овских примитивов для дробей.

$$\frac{n+1}{n} \qquad \left\langle \frac{n+1}{n} \right\rangle$$

$$\binom{n+2}{n} \qquad \left(\frac{n+2}{n} \right)$$

$$\frac{n-3}{n} \qquad \left[\frac{n-3}{n} \right]$$

```
\renewcommand{\over}[2]{%
  \genfrac{}{}{}{}{#1}{#2}}
\renewcommand{\overwithdelims}[2]{%
  \genfrac{\langle}{\rangle}{}{}{#1}{#2}}
\[ \over{n+1}{n}\qquad\overwithdelims{n+1}{n}   \]
\renewcommand{\atop}[2]{%
  \genfrac{}{}{0pt}{}{#1}{#2}}
\renewcommand{\atopwithdelims}[2]{%
  \genfrac{(}{)}{0pt}{}{#1}{#2}}
\[ \atop{n+2}{n}\qquad\atopwithdelims{n+2}{n}   \]
\renewcommand{\above}[2]{%
  \genfrac{}{}{1pt}{}{#1}{#2}}
\renewcommand{\abovewithdelims}[2]{%
  \genfrac{[}{]}{1pt}{}{#1}{#2}}
\[ \above{n-3}{n}\qquad\abovewithdelims{n-3}{n} \]
```

Разумеется, если вы хотите неоднократно использовать в документе какое-либо выражение с командой \genfrac, то поможете себе (и вашему издателю), если определите команду с осмысленным названием при помощи \newcommand как сокращение для этого выражения (см. приведенные примеры).

8.3.16 Непрерывные дроби

Непрерывную дробь можно получить следующим образом:

$$\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{3}+\cfrac{1}{\sqrt{4}+\cfrac{1}{\sqrt{5}+\cfrac{1}{\sqrt{6}+\cdots}}}}} \qquad (8.19)$$

```
\begin{equation}
\cfrac{1}{\sqrt{2}+
 \cfrac{1}{\sqrt{3}+
  \cfrac{1}{\sqrt{4}+
   \cfrac[l]{1}{\sqrt{5}+
    \cfrac[r]{1}{\sqrt{6}+\dotsb
}}}}}
\end{equation}
```

Сдвига любого числителя влево или вправо можно достигнуть, если в команде `\cfrac` воспользоваться необязательным аргументом `[l]` или `[r]` соответственно.

8.3.17 Ог-г-г-громные ограничители

Чтобы правильно назначать размеры ограничителей в математических формулах, стандартная комплектация TEX'а предоставляет четыре команды `\big`, `\Big`, `\bigg` и `\Bigg`, которые дают ограничители, указанные в аргументе этих команд, в порядке возрастания их размера. Эти команды применяются к любым ограничителям, к которым можно применять команды `\left` или `\right` (см. табл. 8.10, 8.11 и 8.14). Более того, для каждой из указанных четырех команд существуют три ее варианта для использования в качестве открывающего символа (например, `\bigl`), бинарного отношения (например, `\Bigm`) и закрывающего символа (например, `\Biggr`)[4]. Если в стандартном TEX'е размеры этих ограничителей фиксированы, то при использовании пакета amsmath их размеры адаптируются под окружающий материал.

$$\biggl(\mathbf{E}_y \int_0^{t_\varepsilon} L_{x,y^x(s)}\varphi(x)\,ds\biggr)$$

$$\left(\mathbf{E}_y \int_0^{t_\varepsilon} L_{x,y^x(s)}\varphi(x)\,ds\right)$$

```
\[
  \biggl(\mathbf{E}_{y}
    \int_0^{t_\varepsilon}
    L_{x,y^x(s)}\varphi(x)\,ds \biggr)
\]
{\Large
\[
  \biggl(\mathbf{E}_{y}
     \int_0^{t_\varepsilon}
     L_{x,y^x(s)}\varphi(x)\,ds \biggr)
\]}
```

[4] *Различные типы математических символов представлены в табл. 7.13.*

8.4 Окружения типа матрицы и коммутативные диаграммы

8.4.1 Окружение cases

Перечисления случаев в математических конструкциях могут быть получены при помощи окружения `cases`:

$$P_{r-j} = \begin{cases} 0, & \text{если } r-j \text{ нечетное,} \\ r!\,(-1)^{(r-j)/2}, & \text{если } r-j \text{ четное.} \end{cases} \tag{8.20}$$

```
\begin{equation}
P_{r-j}=
 \begin{cases}
   0, & \text{если $r-j$ нечетное},\\
   r!\,(-1)^{(r-j)/2}, &
     \text{если $r-j$ четное}.
  \end{cases}
\end{equation}
```

Обратите внимание на применение команды `\text` и использование математического выражения в ее аргументе.

8.4.2 Окружения типа matrix

Окружения типа `matrix` аналогичны LaTeX'овскому окружению `array` за исключением того, что у них не предусмотрен аргумент, определяющий формат столбцов. Вместо этого предоставляется формат по умолчанию: не более 10 центрированных столбцов. Следующие примеры показывают, как использовать окружения `matrix`, `pmatrix`, `bmatrix`, `vmatrix` и `Vmatrix`:

$$\begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix} \quad \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} \quad \begin{Vmatrix} 1 & 0 \\ 0 & 1 \end{Vmatrix}$$

```
\begin{gather*}
\begin{matrix} 0 & 1 \\ 1 & 0\end{matrix} \quad
\begin{pmatrix}0 &-i \\ i & 0\end{pmatrix}\quad
\begin{bmatrix}1 & 0 \\ 0 &-1\end{bmatrix}     \\
\begin{vmatrix}a & b \\ c & d\end{vmatrix}\quad
\begin{Vmatrix}1 & 0 \\ 0 & 1\end{Vmatrix}
\end{gather*}
```

Максимальное число столбцов в матрице определяется счетчиком `MaxMatrixCols`, который можно изменять при помощи обычных LaTeX'овских операций со счетчиками. Предположим, например, что имеется большая матрица из 19 или 20 столбцов. Тогда вы можете проделать следующее:

```
\begin{equation}
\setcounter{MaxMatrixCols}{20}
A=\begin{pmatrix}
...&...&...&...&...&...&...&...&...&...&...&  ... \\
  ...  \\
  ...
\end{pmatrix}
\end{equation}
\setcounter{MaxMatrixCols}{10}
```

В LaTeX'е счетчики определены глобально, поэтому после того как набор широкой матрицы завершен, вы, возможно, захотите установить значение счетчика

MaxMatrixCols в значение по умолчанию 10: дело в том, что при большом значении этого счетчика LaTeX'у гораздо труднее набирать матрицы.

Для того чтобы получить небольшую матрицу, пригодную для использования в тексте, следует привлечь окружение smallmatrix:

Чтобы показать, как матрица влияет на окружающие ее строки внутри абзаца, мы расположили ее здесь $\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \right)$ и продолжили писать текст до тех пор, пока под матрицей не оказалась по крайней мере одна полная строка.

```
Чтобы показать, как матрица влияет на окружающие
ее строки внутри абзаца, мы расположили ее здесь
\begin{math}
  \left( \begin{smallmatrix}
      a&b\\ c&d
    \end{smallmatrix} \right)
\end{math}
и продолжили писать текст до тех пор, пока под
матрицей не оказалась по крайней мере одна полная
строка.
```

Отточие в матрице, простирающееся на несколько столбцов, можно получить при помощи команды

`\hdotsfor[`*spacing-factor*`]{`*number*`}`

Расстояние между точками можно менять, используя необязательный аргумент *spacing-factor*, например, \hdotsfor[1.5]{3}. Число в квадратных скобках умножается на расстояние между точками; обычное значение этого расстояния равно единице[5].

Исходный текст

```
\[ W(\Phi)= \begin{Vmatrix}
  \dfrac\varphi{(\varphi_1,\varepsilon_1)}    & 0 & \hdotsfor{2} & 0 \\
  \dfrac{\varphi k_{n2}}{(\varphi_2,\varepsilon_1)}  &
  \dfrac\varphi{(\varphi_2,\varepsilon_2)}           & 0 & \dots & 0 \\
  \hdotsfor{5}                                                    \\
  \dfrac{\varphi k_{n1}}{(\varphi_n,\varepsilon_1)}  &
   \dfrac{\varphi k_{n2}}{(\varphi_n,\varepsilon_2)} & \dots &
   \dfrac{\varphi k_{n\,n-1}}{(\varphi_n,\varepsilon_{n-1})} &
   \dfrac{\varphi}{(\varphi_n,\varepsilon_n)}
\end{Vmatrix}\]
```

Текст на выводе

$$W(\Phi) = \begin{Vmatrix} \dfrac{\varphi}{(\varphi_1,\varepsilon_1)} & 0 & \dots\dots\dots\dots & 0 \\ \dfrac{\varphi k_{n2}}{(\varphi_2,\varepsilon_1)} & \dfrac{\varphi}{(\varphi_2,\varepsilon_2)} & 0 \quad \dots & 0 \\ \dots\dots\dots\

8.4.3 Команда \substack

Команда \substack применяется для набора нижних и верхних индексов, состоящих из нескольких строк; для разделения строк нужно набрать \\. Команда \substack применима всюду, где можно использовать обычные нижние и верхние индексы.

$$\sum_{\substack{0\le i\le m\\ 0<j<n}} P(i,j) \qquad (8.21)$$

```
\begin{equation}
  \sum _{\substack{0\le i\le m\\ 0<j<n}} P(i,j)
\end{equation}
```

Вместо центрирования все строки можно выровнять по левому краю, если воспользоваться окружением subarray.

$$\sum_{\substack{i\in\Lambda\\ 0<j<n}} P(i,j) \qquad (8.22)$$

```
\begin{equation}
  \sum_{\begin{subarray}{l}
          i\in \Lambda\\ 0<j<n
        \end{subarray}}
  P(i,j)
\end{equation}
```

8.4.4 Коммутативные диаграммы

Команды $\mathcal{AMS}$-TEX'а для создания коммутативных диаграмм не включены в пакет amsmath, а составляют отдельный пакет amscd. Это позволяет сэкономить память для тех пользователей, которым не нужны коммутативные диаграммы. Для сложных коммутативных диаграмм можно использовать окружение picture, а простые коммутативные диаграммы без диагональных стрелок удобнее создавать при помощи команд из пакета amscd[6].

$$\begin{array}{ccc} S^{\mathcal{W}_\Lambda}\otimes T & \xrightarrow{j} & T \\ \downarrow & & \downarrow{\scriptstyle \mathrm{End}\,P} \\ (S\otimes T)/I & = & (Z\otimes T)/J \end{array}$$

```
\DeclareMathOperator{\End}{End}
\[\begin{CD}
  S^{\mathcal{W}_\Lambda}\otimes T @>j>>  T        \\
     @VVV                        @VV{\End P}V    \\
   (S\otimes T)/I                @= (Z\otimes T)/J
   \end{CD}                                      \]
```

Аналогичный результат можно получить и в обычном LaTeX'е, но он не выглядит так же хорошо:

$$\begin{array}{ccc} S^{\mathcal{W}_\Lambda}\otimes T & \xrightarrow{j} & T \\ \downarrow & & \downarrow{\scriptstyle \mathrm{End}\,P} \\ (S\otimes T)/I & = & (Z\otimes T)/J \end{array}$$

```
\[\begin{array}{ccc}
  S^{\mathcal{W}_\Lambda}\otimes T &
  \stackrel{j}{\longrightarrow}    & T    \\
  \Big\downarrow    &   &
  \Big\downarrow\vcenter{%
   \rlap{$\scriptstyle{\mathrm{End}}\,P$}}\\
  (S\otimes T)/I    & = &  (Z\otimes T)/J
  \end{array}                                \]
```

[6] Имеются более обширные пакеты для коммутативных диаграмм — это система XY-pic Кристофера Роуза [70], пакет коммутативных диаграмм Пола Тейлора [75] и Diagram 3 Фрэнсиса Борсо [10].

При использовании пакета amscd горизонтальные стрелки получаются более длинными, а пространство между отдельными элементами диаграммы несколько бóльшим.

В окружении CD команды @>>>, @<<<, @VVV и @AAA дают стрелки, направленные соответственно вправо, влево, вниз и вверх. Для тех, у кого на клавиатуре отсутствуют клавиши > и <, команды @))) и @(((предусмотрены как альтернативы первым двум командам.

В горизонтальных стрелках содержимое между первым и вторым знаком > или первым и вторым знаком < набирается как верхний индекс, а содержимое между вторым и третьим знаками — как нижний индекс. Аналогично, содержимое между первой и второй (соответственно второй и третьей) буквами A или V в вертикальных стрелках будет набрано как левый (соответственно правый) «боковой индекс». Это обстоятельство было использовано в первом из приведенных выше примеров для того, чтобы поставить операцию «End P» справа от вертикальной стрелки вниз.

Последний пример снова демонстрирует применение команды \DeclareMathOperator:

$$\begin{array}{ccccc} \operatorname{cov}(\mathcal{L}) & \longrightarrow & \operatorname{non}(\mathcal{K}) & \longrightarrow & \operatorname{cf}(\mathcal{K}) \\ \downarrow & & \uparrow & & \uparrow \\ \operatorname{add}(\mathcal{L}) & \longrightarrow & \operatorname{add}(\mathcal{K}) & \longrightarrow & \operatorname{cov}(\mathcal{K}) \end{array}$$

```
\begin{equation*}
\DeclareMathOperator{\add}{add}
\DeclareMathOperator{\cf}{cf}
\DeclareMathOperator{\cov}{cov}
\DeclareMathOperator{\non}{non}
\begin{CD}
 \cov(\mathcal{L})@>>>\non(\mathcal{K})
                    @>>>\cf(\mathcal{K})\\
 @VVV @AAA @AAA                  \\
 \add(\mathcal{L})@>>>\add(\mathcal{K})
                    @>>>\cov(\mathcal{K})\\
\end{CD}
\end{equation*}
```

8.5 Выравнивание многострочных формул

В пакете amsmath определено несколько окружений для оформления многострочных выключных формул. Они аналогичны LaTeX'овским окружениям equation и eqnarray. Далее обсуждаются следующие окружения:

align	align*	выравнивание по одной позиции
flalign	flalign*	«расширенные» варианты align
alignat	alignat*	выравнивание по нескольким позициям
equation	equation*	формула, занимающая одну строку
gather	gather*	несколько формул без выравнивания
multline	multline*	многострочная формула (нумеруемая целиком)
split		разбивка на части длинных формул

Некоторые из этих окружений позволяют выравнивать лишь отдельные части формулы. В отличие от LaTeX'овских окружений eqnarray и eqnarray* в окружениях выравнивания из пакета amsmath применяется иное правило для указания позиции выравнивания: поскольку окружение eqnarray аналогично окружению array с преамбулой {rcl}, то по краям выравниваемой части формулы требуется ставить два знака & (по одному на каждый край); в структурах же выравнивания из пакета amsmath следует указывать позицию выравнивания (или несколько позиций, если используется, например, окружение alignat) при помощи только одного знака &, помещая его слева от символа, по которому должно произойти выравнивание с предыдущими или последующими строками.

Структуры выравнивания из пакета amsmath дают правильные пробелы вокруг позиций выравнивания, в то время как окружение eqnarray создает лишнее пространство около этих позиций, которое зависит от определяющих array параметров[7]. Это различие явно видно в следующем примере, где одни и те же формулы набираются в окружениях equation, align и eqnarray; в идеале все три окружения должны давать один и тот же результат, однако в окружении eqnarray формулы получаются слишком растянутыми.

$$x^2+y^2=z^2 \qquad (8.23)$$

$$x^2+y^2=z^2 \qquad (8.24)$$

$$x^3+y^3<z^3 \qquad (8.25)$$

$$x^2+y^2 = z^2 \qquad (8.26)$$

$$x^3+y^3 < z^3 \qquad (8.27)$$

```
\begin{equation}
  x^2+y^2  =  z^2
\end{equation}
\begin{align}
  x^2+y^2 &= z^2 \\   x^3+y^3 &< z^3
\end{align}
\begin{eqnarray}
  x^2+y^2 &=& z^2 \\ x^3+y^3 &<& z^3
\end{eqnarray}
```

8.5.1 Несколько формул без выравнивания

Окружение gather предназначено для набора двух или более формул в том случае, когда для них не требуется выравнивание. Каждая формула по отдельности будет центрировано относительно левого и правого полей.

$$(a+b)^2=a^2+2ab+b^2 \qquad (8.28)$$

$$(a+b)\cdot(a-b)=a^2-b^2 \qquad (8.29)$$

```
\begin{gather}
(a + b)^2 = a^2 + 2ab + b^2\\
(a + b) \cdot (a - b) = a^2 - b^2
\end{gather}
```

Другие примеры приведены в разд. 8.7.3.

[7] Таких как \arraycolsep, \arrayrulewidth, \doublerulesep и \arraystretch.— *Прим. перев.*

8.5.2 Несколько формул с выравниванием

Окружение `align` используется для вертикального выравнивания двух или более формул (обычно выравниваются знаки бинарных отношений, таких как знаки равенства). Здесь термин «формула» трактуется довольно широко и обозначает любое математическое соотношение, которое может рассматриваться как самостоятельный элемент большего выключного выражения и которая не всегда, но как правило, включает в себя некоторое бинарное отношение.

$$x^2 + y^2 = 1 \qquad x^3 + y^3 = 1 \tag{8.30}$$
$$x = \sqrt{1 - y^2} \qquad x = \sqrt[3]{1 - y^3} \tag{8.31}$$

```
\begin{align}
x^2  + y^2 & = 1 &
x^3  + y^3 & = 1 \\
x          & = \sqrt{1-y^2}    &
x          & = \sqrt[3]{1-y^3}
\end{align}
```

Другие примеры приведены в разд. 8.7.4.

В окружении `align` содержимое располагается равномерно вдоль строк[8]. Если вы хотите управлять пространством между выравниваемыми колонками, то можете воспользоваться окружением `alignat`. Оно имеет один обязательный аргумент, обозначающий количество выравниваемых формул на каждой строке. Если значение этого аргумента равно n, то на каждой строке потребуется поставить $2n - 1$ знаков `&` (по одному для каждой выравниваемой формулы на строке и $n - 1$ знаков `&`, отделяющих одну формулу от другой).

Специальное окружение `flalign` является вариантом окружения `align` с дополнительными (горизонтальными) пробелами между отдельными компонентами выравниваемых формул.

$$L_1 = R_1 \qquad L_2 = R_2 \tag{8.32}$$
$$L_3 = R_3 \qquad L_4 = R_4 \tag{8.33}$$

```
\begin{align}
L_1 & = R_1 & \qquad L_2 & = R_2 \\
L_3 & = R_3 & \qquad L_4 & = R_4
\end{align}
```

$$L_1 = R_1 \qquad L_2 = R_2 \tag{8.34}$$
$$L_3 = R_3 \qquad L_4 = R_4 \tag{8.35}$$

```
\begin{alignat}{2}
L_1 & = R_1 & \qquad L_2 & = R_2 \\
L_3 & = R_3 & \qquad L_4 & = R_4
\end{alignat}
```

$$L_1 = R_1 \qquad L_2 = R_2 \tag{8.36}$$
$$L_3 = R_3 \qquad L_4 = R_4 \tag{8.37}$$

```
\begin{flalign}
L_1 & = R_1 & \qquad L_2 & = R_2 \\
L_3 & = R_3 & \qquad L_4 & = R_4
\end{flalign}
```

$$L_1 = R_1 \qquad L_2 = R_2$$
$$L_3 = R_3 \qquad L_4 = R_4$$

```
\begin{flalign*}
L_1 & = R_1 & \qquad L_2 & = R_2 \\
L_3 & = R_3 & \qquad L_4 & = R_4
\end{flalign*}
```

Другие примеры приведены в разд. 8.7.6.

[8] Использует всю ширину выключной формулы.— *Прим. перев.*

8.5.3 Разбитые на части формулы без выравнивания

Окружение `multline` является разновидностью окружения `equation` и предназначено для формул, которые не умещаются в одну строку. Первая строка из этого окружения будет прижата к левому полю, а последняя — к правому, если не считать отступ от обоих полей, который равен параметру `\multlinegap`. Значение `\multlinegap` можно изменять при помощи LaTeX'овских команд `\setlength` и `\addtolength`. Если `multline` содержит более двух строк, то все строки, кроме первой и последней, будут по отдельности центрированы по ширине (если только не задействована опция `fleqn`). Есть возможность прижать формулу к левому полю или правому, если воспользоваться соответственно командами `\shoveleft` и `\shoveright`.

Первая строка формулы
Средняя строка по центру
Средняя строка вправо
Другая строка по центру
Средняя строка влево
Последняя строка формулы (8.38)

```
\begin{multline}
\text{Первая строка формулы}       \\
\text{Средняя строка по центру}    \\
\shoveright{\text{Средняя
                  строка вправо}}  \\
\text{Другая строка по центру}     \\
\shoveleft{\text{Средняя строка
                  влево}}          \\
\text{Последняя строка формулы}
\end{multline}
```

Другие примеры показаны в разд. 8.7.2.

8.5.4 Разбитые на части формулы с выравниванием

Как и `multline`, окружение `split` предназначено для отдельных длинных формул, не умещающихся в одну строку, а потому требующих разбивки на несколько строк. Однако в отличие от `multline`, в окружении `split` предусмотрено выравнивание расщепленных строк при помощи, как обычно, знака `&`, указывающего позицию, с которой начинается выравнивание. В дополнение (и в отличие от остальных окружений выравнивания из пакета `amsmath`) окружение `split` не генерирует номера формулы, так как оно приспособлено для использования только внутри других окружений выравнивания, таких, как `equation`, `align` или `gather`. Это внешнее окружение и генерирует нужный номер формулы.

$$\begin{aligned}(a+b)^4 &= (a+b)^2(a+b)^2 \\ &= (a^2+2ab+b^2)(a^2+2ab+b^2) \\ &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\end{aligned} \tag{8.39}$$

```
\begin{equation}
\begin{split}
(a+b)^4 &= (a+b)^2 (a+b)^2          \\
        &= (a^2+2ab+b^2)(a^2+2ab+b^2) \\
        &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\\
\end{split}
\end{equation}
```

Если указана опция `tbtags`, то номер формулы, порождаемый окружением `split`, будет помещен на последнюю (соответственно первую) строку, если номер формулы стоит справа (соответственно слева). По умолчанию задействована

опция centertags, которая помещает номер формулы вертикально по центру относительно высоты формулы в окружении split, если только для него хватает места.

$$\begin{aligned}(a+b)^3 &= (a+b)(a+b)^2 \\ &= (a+b)(a^2+2ab+b^2) \\ &= a^3+3a^2b+3ab^2+b^3\end{aligned} \tag{8.40}$$

```
\begin{equation}
\begin{split}
(a+b)^3
   &= (a+b) (a+b)^2       \\
   &= (a+b)(a^2+2ab+b^2)  \\
   &= a^3+3a^2b+3ab^2+b^3 \\
\end{split}
\end{equation}
```

Другие примеры приведены в разд. 8.7.1.

8.5.5 Окружения выравнивания для набора отдельных частей выключных формул

Наряду с окружением split, имеются и другие окружения выравнивания, которые используются для набора не целой формулы, а лишь некоторых ее частей. Они представляют собой самостоятельные образования, которые применяются внутри других формул, или могут располагаться рядом друг с другом. Вот их названия: aligned, gathered и alignedat. Эти окружения имеют необязательный аргумент, определяющий их вертикальное положение по отношению к содержимому справа и слева от них. Принятое по умолчанию (вертикальное) выравнивание — это центрирование ([c]), а его результат показан в следующем примере:

$$\begin{aligned}x^2+y^2 &= 1 \\ x &= \sqrt{1-y^2}\end{aligned} \qquad \begin{aligned}(a+b)^2 &= a^2+2ab+b^2 \\ (a+b)\cdot(a-b) &= a^2-b^2\end{aligned}$$

```
\begin{equation*}
\begin{aligned}
 x^2 + y^2 &= 1            \\
         x &= \sqrt{1-y^2}
\end{aligned}
\qquad
\begin{gathered}
 (a+b)^2 = a^2 + 2ab + b^2 \\
 (a+b) \cdot (a-b) = a^2 - b^2
\end{gathered}
\end{equation*}
```

Теперь те же самые формулы можно набрать, применяя иное вертикальное выравнивание для указанных окружений.

$$\begin{aligned}x^2+y^2 &= 1 \\ x &= \sqrt{1-y^2}\end{aligned} \qquad \begin{aligned}(a+b)^2 &= a^2+2ab+b^2 \\ (a+b)\cdot(a-b) &= a^2-b^2\end{aligned}$$

```
\begin{equation*}
\begin{aligned}[b]
 x^2 + y^2 &= 1            \\
         x &= \sqrt{1-y^2}
\end{aligned}
\qquad
\begin{gathered}[t]
 (a+b)^2 = a^2 + 2ab + b^2 \\
 (a+b) \cdot (a-b) = a^2 - b^2
\end{gathered}
\end{equation*}
```

8.5.6 Вертикальные пробелы и разрывы страниц при наборе формул

Чтобы получить дополнительный вертикальный пробел между строками, во всех окружениях пакета amsmath для выключных формул вы можете использовать команду `\\[`*dimension*`]`, как принято в LaTeX'е. В отличие от eqnarray, окружения из пакета amsmath не позволяют разрывать страницу между строками, если только не указаны команды `\displaybreak` или `\allowdisplaybreaks`. Причина здесь в том, что разрыв страниц в таких случаях должен контролироваться автором особо. Команда `\displaybreak` должна предшествовать команде `\\` в том месте, где предполагается разрыв страницы. Как и LaTeX'овская команда `\pagebreak`, `\displaybreak` имеет необязательный аргумент — целое число между 0 и 4, означающий степень желательности разрыва страницы. Команда `\displaybreak[0]` означает, что «разрыв здесь допустим», хотя он не «поощряется», а команда `\displaybreak` без аргумента означает то же, что и `\displaybreak[4]`, и вынуждает сделать разрыв.

У команды `\allowdisplaybreaks` также предусмотрен необязательный аргумент. Эта команда подчиняется обычным LaTeX'овским правилам группирования. Стандартный способ ограничения области ее действия — это указание `{\allowdisplaybreaks` в начале и закрывающая `}` в конце нужной области. В пределах действия команды `\allowdisplaybreaks` для запрещения разрыва страницы, как обычно, можно воспользоваться командой `\\*`.

8.5.7 Команда \intertext

Команда `\intertext` служит для вставки короткого, из одной или двух строк, текста между выравниваемыми выключными формулами. Отличительная ее черта состоит в том, что она позволяет сохранить выравнивание, что было бы невозможно, если просто завершить одну выключную формулу, а затем начать новую. Команда `\intertext` должна следовать сразу же после команд `\\` или `\\*`.

$$A_1 = N_0(\lambda;\Omega') - \phi(\lambda;\Omega'), \qquad (8.41)$$

$$A_2 = \phi(\lambda;\Omega')\phi(\lambda;\Omega), \qquad (8.42)$$

и, наконец,

$$A_3 = \mathcal{N}(\lambda;\omega). \qquad (8.43)$$

```
\begin{align}
A_1&=N_0(\lambda;\Omega') -
        \phi(\lambda;\Omega'), \\
A_2&=\phi(\lambda;\Omega')
        \phi(\lambda;\Omega), \\
\intertext{и, наконец,}
A_3&=\mathcal{N}(\lambda;\omega).
\end{align}
```

Здесь слова «и, наконец,» выпадают из выравниваемых формул и прижаты к левому полю.

8.6 Разное

В этом разделе обсуждаются команды пакета amsmath, которые еще не рассматривались, и приводится список файлов, отвечающих за классы документов, которые распространяются вместе с $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX'ом.

8.6.1 Нумерация формул

Каждое окружение, кроме `split`, имеет два варианта, со звездочкой и без, причем вариант без звездочки автоматически генерирует номер, используя LaTeX'овский счетчик `equation`. Номер любой строки в формуле можно опустить, поставив перед \\ команду `\notag`. Этот номер можно заменить на свой собственный при помощи команд:

`\tag{`*label*`}` `\tag*{`*label*`}`

где *label* — это произвольный текст, предназначенный для идентификации строки (уравнения).

Вариант со звездочкой `\tag*` печатает «номер» *label* без каких-либо выделений типа круглых скобок, которые иначе могут быть добавлены соответствующим классом документа. Команды `\tag` и `\tag*` можно применять также и в версиях со звездочкой для всех окружений выравнивания из пакета amsmath.

$$x^2+y^2=z^2 \qquad (8.44)$$
$$x^3+y^3=z^3$$
$$x^4+y^4=r^4 \qquad (*)$$
$$x^5+y^5=r^5 \qquad *$$
$$x^6+y^6=r^6 \qquad (8.44')$$

```
\begin{gather}
  x^2+y^2 = z^2 \label{eq:r2}      \\
  x^3+y^3 = z^3 \notag             \\
  x^4+y^4 = r^4 \tag{$*$}          \\
  x^5+y^5 = r^5 \tag*{$*$}         \\
  x^6+y^6 = r^6 \tag{\ref{eq:r2}$'$}
\end{gather}
```

Обратите внимание на использование команд `\label` и `\ref` в предыдущем примере, позволяющих создавать подчиненную нумерацию формул.

Если для пакета amsmath указана опция leqno, то номер формулы будет напечатан слева от этой формулы (с пакетом amsmath по умолчанию он ставится справа).

$$\sin^2\eta+\cos^2\eta=1 \qquad (8.45)$$

```
\begin{equation}
  \sin^2\eta + \cos^2\eta = 1
\end{equation}
```

8.6.2 Установка счетчика формул

Если в рамках LaTeX'а вы хотите, чтобы формулы нумеровались в зависимости от разделов, т. е. в разделе 1 формулы имели номера (1.1), (1.2), ... , в разделе 2 — номера (2.1), (2.2), ... и т. д., то вполне возможно вы переопределите команду `\theequation`:

```
\renewcommand{\theequation}{\thesection.\arabic{equation}}
```

Однако теперь вам придется устанавливать вид номера формулы вручную в начале каждого нового раздела или новой главы. Чтобы несколько облегчить задачу, в пакете amsmath предусмотрена команда `\numberwithin`. Для того чтобы нумерация формул была привязана к нумерации разделов с автоматической переустановкой счетчика формул, используется команда

```
\numberwithin{equation}{section}
```

В соответствии с ее названием (number within означает «нумеровать в пределах, в рамках, внутри». — *Перев.*) команду `\numberwithin` можно применять, кроме счетчика формул, и к другим счетчикам, но результат может не быть удовлетворительным во всех случаях. Там, где это уместно, например, с командой `\newtheorem`[9], следует применять обычные LaTeX'овские средства.

Чтобы облегчить ссылку на номера формул, предусмотрена команда `\eqref`. Она автоматически ставит круглые скобки вокруг номера формулы и, если необходимо, добавляет поправку на курсив перед закрывающей скобкой. Для того чтобы сослаться на формулу с меткой `e:baset`, следует набрать `\eqref{e:baset}`.

8.6.3 Подчиненная нумерация формул

Для того чтобы было легче нумеровать формулы в некоторой выделенной группе формул, в пакете amsmath предусмотрено окружение `subequations`, которое использует подчиненную нумерацию. Например, в части документа, выделенной как

```
\begin{subequations}
...
\end{subequations}
```

номера формул будут иметь вид (4.9a), (4.9b), (4.9c) и т. д., если предыдущая формула имела номер (4.8). Команда `\label`, стоящая сразу после `\begin{subequations}`, порождает при помощи команды `\ref` ссылку на «родительский» номер 4.9, а не на 4.9a. Окружение `subequations` использует счетчики `parentequation` и `equation`, которые можно установить при помощи LaTeX'овских команд `\addtocounter`, `\setcounter`, `\value` и т. д. Более того, стиль подчиненной нумерации контролируется обычными LaTeX'овскими сред-

[9] См. также обсуждение команды `\@addtoreset` на с. 40.

Положительный пробел			Отрицательный пробел		
Сокращенная форма	Пример	Полное название	Сокращенная форма	Пример	Полное название
`\,`	$x\,x$	`\thinspace`	`\!`	$x\!x$	`\negthinspace`
`\:`	$x\:x$	`\medspace`		$x\!\!x$	`\negmedspace`
`\;`	$x\;x$	`\thickspace`		$x\!\!\!x$	`\negthickspace`
	$x\quad x$	`\quad`			
	$x\qquad x$	`\qquad`			

Таблица 8.21. Команды, отвечающие за расстановку пробелов в формулах

ствами (см. разд. А.1.3). Например, переопределив команду `\theequation`, как показано ниже, мы получим подчиненную нумерацию римскими цифрами.

```
\begin{subequations}
\renewcommand{\theequation}{\theparentequation \roman{equation}}
...
```

8.6.4 Тонкая настройка в математическом режиме

Хотя TEX, как правило, хорошо расставляет пробелы между отдельными элементами в математической формуле, иногда требуется подправить положение того или иного элемента. Для этого предназначаются команды из табл. 8.21. И полная, и сокращенная формы этих команд не являются хрупкими и могут быть использованы также вне математического режима.

Чтобы еще более тонко управлять пробелами в математических выражениях, имеется команда `\mspace`. Ее единственный аргумент является LATEX'овской длиной, выражаемой в 'математических единицах' (по-английски 'math unit'). Одна математическая единица, обозначаемая `mu`, равняется 1/18 em (см. также табл. А.1). Таким образом, для того чтобы получить `\quad` отрицательной длины, следует написать `\mspace{-18.0mu}`.

8.6.5 На что еще обратить внимание

Многие команды, добавляемые пакетом amsmath, являются хрупкими [$\mathcal{L}$ 151–152], [$\mathcal{A}$ 165], и их требуется защищать с помощью `\protect` при использовании в командах с «перемещаемыми аргументами».

При наличии такого большого количества окружений выравнивания в пакете amsmath отпадает надобность в окружении `eqnarray`. К тому же, оно не предотвращает наложений длинных формул на их номера, а большинство окружений выравнивания с этим справляется. Таким образом, использование окружений пакета amsmath кажется предпочтительным. В пакете amsmath LATEX'овское окружение `equation` переопределяется как однострочное окружение `gather` и для симметрии добавляется его ненумеруемый вариант `equation*`. Заметьте, однако, что команда `\verb` может не работать в окружениях выравнивания.

Команда \nonumber эквивалентна команде \notag; последняя слегка предпочтительнее, так как более согласуется с названием \tag.

8.6.6 Опции к пакету amsmath и отдельные его составляющие

С пакетом amsmath может использоваться несколько опций и классов документов из $\mathcal{AMS}$-LaTeX'а[10]. Главным образом они влияют на расположение пределов (индексов) в математических операциях или номеров, генерируемых командой \tag.

centertags (Используется по умолчанию.) Номера формул в окружении split центрируются по вертикали относительно всей высоты этих формул.

tbtags «Номера сверху или снизу». Номер формулы из окружения split помещается на последнюю (соответственно первую) строку, занимаемую формулой, если номера ставятся справа (соответственно слева).

intlimits Как опция sumlimits, но используется для знаков интеграла.

nointlimits (Используется по умолчанию.) Противоположная опции intlimits.

namelimits (Используется по умолчанию.) Как опция sumlimits, но используется для некоторых «операторов», таких как det, inf, lim, max, min, у которых традиционно ставится нижний индекс; в выключной формуле этот индекс располагается под именем оператора.

nonamelimits Противоположная опции namelimits.

sumlimits (Используется по умолчанию.) В выключных формулах верхние и нижние индексы суммирования располагаются над и под знаком суммы. Эта опция также влияет на другие символы подобного типа: $\prod$, $\coprod$, $\bigotimes$, $\bigoplus$ и т. п., исключая интегралы (см. опцию intlimits).

nosumlimits Нижние и верхние индексы у знаков типа суммы располагаются сбоку от знака даже в выключной формуле.

Следующие три опции обычно являются глобальными в документе, а потому указываются в команде \documentclass. Однако они также распознаются, когда пакет amsmath загружается при помощи команды \usepackage.

leqno Номера формул располагаются слева.

reqno Номера формул располагаются справа.

fleqn Формулы располагаются с фиксированным отступом от левого поля, а не центрируются между полями.

Дистрибутив $\mathcal{AMS}$-LaTeX'а состоит из нескольких компонентов, которые можно загружать независимо друг от друга при помощи команды \usepackage. Единственным заслуживающим внимания пакетом, вероятно, является пакет amsmath,

[10] Это касается только версии $\mathcal{AMS}$-LaTeX'а из комплектации LaTeX 2_ε. В более ранних версиях $\mathcal{AMS}$-LaTeX'а эти опции представляют собой комплектующие (подпакеты) этого пакета.

а остальные пакеты можно использовать независимо. Отметим, что вместе с пакетом amsmath автоматически включаются и пакеты amsbsy, amsopn и amstext.

amsmath Определяет дополнительные окружения для многострочных выключных формул и имеет некоторые другие усовершенствования для набора математических формул.

amsbsy Определяет команды `\boldsymbol` и `\pmb` (жирный шрифт для бедных).

amsopn Предоставляет команду `\DeclareMathOperator` для определения «новых операторов», таких как `\sin` и `\lim`.

amstext Предоставляет команду `\text` для набора текста внутри выключной математической формулы.

Все другие пакеты, имеющие дополнительные возможности, следует загружать явно. В этой главе описывается только часть этих пакетов. Мы упоминаем их здесь для полноты изложения.

amscd Определяет некоторые команды для облегчения набора коммутативных диаграмм, вводя окружение `CD` (см. разд. 8.4.4). Диагональные стрелки не поддерживаются.

amsintx Предоставляет более описательный синтаксис команд для интегралов и сумм (еще не выпущен).

amsthm Предоставляет окружение `proof` (для оформления доказательств) и расширения команды `\newtheorem`.

amsxtra Определяет некоторые дополнительные команды типа `\accentedsymbol` (см. разд. 8.3.4).

upref Вне зависимости от контекста команда `\ref` всегда печатает номера ссылок светлым прямым шрифтом.

Наконец, имеется несколько пакетов, которые поставляются вместе с дистрибутивом шрифтов `AMSFonts`.

amsfonts определяет команды `\mathfrak` и `\mathbb` и делает доступными в математическом режиме шрифты `msam` (дополнительные математические символы подсемейства A), `msbm` (дополнительные математические символы подсемейства B и ажурные буквы), `eufm` (эйлерова готика), а также дополнительные размеры шрифтов `cmmib` (полужирный математический курсив и полужирные строчные греческие буквы) и `cmbsy` (полужирные математические символы и полужирные «рукописные» буквы).

amssymb определяет имена всех математических символов, имеющихся в семействах шрифтов $\mathcal{AMS}$. Этим пакетом загружается пакет amsfonts.

eufrak Делает доступными готические буквы.

eucal Команда `\mathcal` будет использовать рукописные буквы семейства Euler вместо обычных рукописных букв семейства Computer Modern.

Все эти пакеты распознают опцию psamsfonts, которая будет использовать вариант семейства шрифтов $\mathcal{AMS}$ компании Y&Y/Blue Sky Research (имеющихся в свободном доступе на CTAN).

8.6.7 Классы документов $\mathcal{AMS}$-LaTeX'а

Пакет $\mathcal{AMS}$-LaTeX поставляется с двумя классами документов amsart и amsbook, соответствующими LaTeX'овским article (статья) и book (книга). Первоначально они предназначались для подготовки рукописей, представляемых для опубликования в Американское математическое общество, но ничто не мешает их применению в других целях. При использовании этих классов документов автоматически подключается пакет amsmath, так что вы можете начать ваш документ просто при помощи команд `\documentclass{amsart}` или `\documentclass{amsbook}`.

8.7 Примеры многострочных формул

На нескольких следующих страницах мы приводим множество взятых из «реальной жизни» примеров окружений выравнивания, которые обсуждались выше. Тонкие линии на полях, между которыми набраны формулы, не являются частью окружений выравнивания, а добавлены для более наглядного представления кромок полей.

8.7.1 Окружение split

Окружение split не является независимым и должно использоваться внутри какого-нибудь другого окружения типа equation или align.

Если для формулы, набираемой с помощью окружения split, не хватает места, то ее номер будет сдвинут вверх на предыдущую строку в том случае, когда номера формул ставятся слева; ее номер будет сдвинут вниз на последующую строку, если номера формул ставятся справа.

Если вы не хотите, чтобы формула нумеровалась, используйте для нее окружение equation*.

$$\begin{aligned} f_{h,\varepsilon}(x,y) &= \varepsilon \mathbf{E}_{x,y} \int_0^{t_\varepsilon} L_{x,y_\varepsilon(\varepsilon u)}\varphi(x)\,du \\ &= h \int L_{x,z}\varphi(x)\rho_x(dz) \\ &\quad + h\Bigg[\frac{1}{t_\varepsilon}\Bigg(\mathbf{E}_y \int_0^{t_\varepsilon} L_{x,y^x(s)}\varphi(x)\,ds - t_\varepsilon \int L_{x,z}\varphi(x)\rho_x(dz)\Bigg) \\ &\qquad + \frac{1}{t_\varepsilon}\Bigg(\mathbf{E}_y \int_0^{t_\varepsilon} L_{x,y^x(s)}\varphi(x)\,ds - \mathbf{E}_{x,y} \int_0^{t_\varepsilon} L_{x,y_\varepsilon(\varepsilon s)}\varphi(x)\,ds\Bigg)\Bigg] \end{aligned} \tag{8.46}$$

Это было набрано следующим образом (TEX'овская команда \phantom используется для того, чтобы оставить пробел, равный ширине ее аргумента):

```
\begin{equation}
\begin{split}
f_{h,\varepsilon}(x,y)
  &=\varepsilon\mathbf{E}_{x,y}\int_0^{t_\varepsilon}
    L_{x,y_\varepsilon(\varepsilon u)}\varphi(x)\,du        \\
  &= h\int L_{x,z}\varphi(x)\rho_x(dz)                      \\
  &\quad+h\biggl[\frac{1}{t_\varepsilon}\biggl(\mathbf{E}_{y}
    \int_0^{t_\varepsilon}L_{x,y^x(s)}\varphi(x)\,ds
    -t_\varepsilon\int L_{x,z}\varphi(x)\rho_x(dz)\biggr)  \\
  &\phantom{{=}+h\biggl[}+\frac{1}{t_\varepsilon}
    \biggl( \mathbf{E}_{y} \int_0^{t_\varepsilon}L_{x,y^x(s)}
      \varphi(x)\,ds - \mathbf{E}_{x,y}\int_0^{t_\varepsilon}
     L_{x,y_\varepsilon(\varepsilon s)}
     \varphi(x)\,ds\biggr)\biggr]\\
\end{split}
\end{equation}
```

Если опция centertags включена в список опций пакета amsmath, то номера формул из окружения split будут вертикально центрированы относительно общей высоты формул, что и показано в примере ниже.

$$\begin{aligned} |I_2| &= \left| \int_0^T \psi(t) \left\{ u(a,t) - \int_{\gamma(t)}^a \frac{d\theta}{k(\theta,t)} \int_a^\theta c(\xi) u_t(\xi,t)\,d\xi \right\} dt \right| \\ &\le C_6 \left\| f \int_\Omega \left| \widetilde{S}_{a,-}^{-1,0} W_2(\Omega,\Gamma_l) \right| \right\| |u| \overset{\circ}{\to} W_2^{\widetilde{A}}(\Omega;\Gamma_r,T) \right\| . \end{aligned} \tag{8.47}$$

Это набирается следующим образом:

```
\begin{equation}
\begin{split}
 |I_2| &=\left| \int_{0}^T \psi(t)
        \left\{ u(a,t) - \int_{\gamma(t)}^a \frac{d\theta}{k(\theta,t)}
            \int_{a}^\theta c(\xi)u_t(\xi,t)\,d\xi \right\}dt \right| \\
  & \le C_6 \left| \left| f\int_\Omega
    \left| \widetilde{S}^{-1,0}_{a,-} W_2(\Omega,\Gamma_l) \right| \right|
    \left| |u|\overset{\circ}\to W_2^{\widetilde{A}}
          (\Omega;\Gamma_r,T) \right| \right| .
\end{split}
\end{equation}
```

Еще один пример затрагивает окружения **split** и **align**. Чтобы получить формулы без номеров, вместо последнего окружения используйте **align***.

$$
\begin{aligned}
|I_1| &= \left| \int_\Omega gRu \, d\Omega \right| \\
&\le C_3 \left[\int_\Omega \left(\int_a^x g(\xi,t)\, d\xi \right)^2 d\Omega \right]^{1/2} \\
&\quad \times \left[\int_\Omega \left\{ u_x^2 + \frac{1}{k} \left(\int_a^x c u_t \, d\xi \right)^2 \right\} c\Omega \right]^{1/2} \\
&\le C_4 \left\| f \left| \widetilde{S}_{a,-}^{-1,0} W_2(\Omega,\Gamma_l) \right| \right\| \left\| |u| \overset{\circ}{\to} W_2^{\widetilde{A}}(\Omega;\Gamma_r,T) \right\|.
\end{aligned} \tag{8.48}
$$

$$
\begin{aligned}
|I_2| &= \left| \int_0^T \psi(t) \left\{ u(a,t) - \int_{\gamma(t)}^a \frac{d\theta}{k(\theta,t)} \int_a^\theta c(\xi) u_t(\xi,t)\, d\xi \right\} dt \right| \\
&\le C_6 \left\| f \int_\Omega \left| \widetilde{S}_{a,-}^{-1,0} W_2(\Omega,\Gamma_l) \right| \right\| \left\| |u| \overset{\circ}{\to} W_2^{\widetilde{A}}(\Omega;\Gamma_r,T) \right\|.
\end{aligned} \tag{8.49}
$$

Приведенные формулы набраны так:

```
\begin{align}
\begin{split}
|I_1| &= \left| \int_\Omega gRu \,d\Omega \right|                    \\
      &\le C_3 \left[ \int_\Omega \left( \int_{a}^x
         g(\xi,t) \,d \xi \right)^2d \Omega \right]^{1/2}           \\
      &\quad\times \left[ \int_\Omega \left\{ u^2_x + \frac{1}{k}
        \left( \int_{a}^x cu_t \, d\xi \right)^2 \right\}
        c \Omega \right]^{1/2}                                      \\
      &\le C_4 \left| \left| f \left| \widetilde{S}^{-1,0}_{a,-}
        W_2(\Omega,\Gamma_l) \right| \right|
        \left| |u| \overset{\circ} \to W_2^{\widetilde{A}}
         (\Omega;\Gamma_r,T) \right| \right|.
\end{split}\label{eq:A}                                             \\
\begin{split}
|I_2| &= \left| \int_{0}^T \psi(t) \left\{ u(a,t)
          -\int_{\gamma(t)}^a \frac{d\theta}{k(\theta,t)}
          \int_{a}^\theta c(\xi) u_t(\xi,t) \,d \xi \right\} dt
        \right|                                                     \\
      &\le C_6 \left| \left| f \int_\Omega
          \left| \widetilde{S}^{-1,0}_{a,-}
             W_2(\Omega,\Gamma_l)  \right| \right|
          \left| |u| \overset{\circ} \to W_2^{\widetilde{A}}
            (\Omega;\Gamma_r,T) \right| \right|.
\end{split}
\end{align}
```

8.7.2 Окружение multline

Нумеруемый вариант окружения multline:

$$\int_a^b \left\{ \int_a^b [f(x)^2 g(y)^2 + f(y)^2 g(x)^2] - 2f(x)g(x)f(y)g(y)\,dx \right\} dy = \int_a^b \left\{ g(y)^2 \int_a^b f^2 + f(y)^2 \int_a^b g^2 - 2f(y)g(y) \int_a^b fg \right\} dy \quad (8.50)$$

Это было получено посредством следующего ввода:

```
\begin{multline}\label{eq:E}
  \int_a^b\biggl\{ \int_a^b [ f(x)^2g(y)^2 + f(y)^2 g(x)^2 ]
     -2f(x) g(x) f(y) g(y) \,dx \biggr\} \,dy              \\
  =\int_a^b \biggl\{ g(y)^2 \int_a^b f^2 + f(y)^2
   \int_a^b g^2 - 2f(y) g(y) \int_a^b fg \biggr\} \,dy
\end{multline}
```

Вариант без номера той же формулы набирается точно так же, если окружение multline заменить на multline*.

$$\int_a^b \left\{ \int_a^b [f(x)^2 g(y)^2 + f(y)^2 g(x)^2] - 2f(x)g(x)f(y)g(y)\,dx \right\} dy = \int_a^b \left\{ g(y)^2 \int_a^b f^2 + f(y)^2 \int_a^b g^2 - 2f(y)g(y) \int_a^b fg \right\} dy$$

А теперь ненумеруемый вариант окружения multline, пронумерованный при помощи команды \tag*.

$$\int_a^b \left\{ \int_a^b [f(x)^2 g(y)^2 + f(y)^2 g(x)^2] - 2f(x)g(x)f(y)g(y)\,dx \right\} dy = \int_a^b \left\{ g(y)^2 \int_a^b f^2 + f(y)^2 \int_a^b g^2 - 2f(y)g(y) \int_a^b fg \right\} dy \quad [a]$$

Это было порождено так:

```
\begin{multline*}\tag*{[a]}   ...  \end{multline*}
```

Ниже приводится эта же формула, когда параметр \multlinegap установлен в значение нуль. Заметьте, что пробел справа во второй строке не изменяется из-за присутствия в ней номера уравнения, тогда как первая строка прижимается к левому полю страницы.

$$\int_a^b \left\{ \int_a^b [f(x)^2 g(y)^2 + f(y)^2 g(x)^2] - 2f(x)g(x)f(y)g(y)\,dx \right\} dy$$
$$= \int_a^b \left\{ g(y)^2 \int_a^b f^2 + f(y)^2 \int_a^b g^2 - 2f(y)g(y) \int_a^b fg \right\} dy \quad [a]$$

Это было набрано так:

```
{\setlength{\multlinegap}{0pt}
\begin{multline*}\tag*{[a]}  ...  \end{multline*}}
```

8.7.3 Окружение gather

Вот как используется нумеруемый вариант gather с командой \notag во второй строке:

$$D(a,r) \equiv \{z \in \mathbf{C} \colon |z-a| < r\}, \tag{8.51}$$
$$\operatorname{seg}(a,r) \equiv \{z \in \mathbf{C} \colon \Im z = \Im a,\ |z-a| < r\},$$
$$c(e,\theta,r) \equiv \{(x,y) \in \mathbf{C} \colon |x-e| < y\tan\theta,\ 0 < y < r\}, \tag{8.52}$$
$$C(E,\theta,r) \equiv \bigcup_{e\in E} c(e,\theta,r). \tag{8.53}$$

Это было набрано так:

```
\begin{gather}
  D(a,r) \equiv \{ z\in\mathbf{C}\colon |z-a|<r \},      \\
  \operatorname{seg}(a,r) \equiv \{z\in\mathbf{C}:
      \Im z= \Im a,\ |z-a|<r\},                    \notag \\
  c(e,\theta,r)\equiv\{(x,y)\in\mathbf{C}:
      |x-e|<y \tan \theta, \ 0<y<r  \},                  \\
  C(E,\theta,r) \equiv \bigcup_{e \in E} c(e,\theta,r).
\end{gather}
```

8.7.4 Окружение align

Нумеруемый вариант:

$$\gamma_x(t) = (\cos tu + \sin tx, v), \tag{8.54}$$
$$\gamma_y(t) = (u, \cos tv + \sin ty), \tag{8.55}$$
$$\gamma_z(t) = \left(\cos tu + \frac{\alpha}{\beta} \sin tv, -\frac{\beta}{\alpha} \sin tu + \cos tv \right). \tag{8.56}$$

Это было получено посредством следующего набора:

```
\begin{align}
  \gamma_x(t) &= (\cos tu + \sin tx, v),                          \\
  \gamma_y(t) &= (u, \cos tv + \sin ty),                          \\
  \gamma_z(t) &= \left( \cos tu + \frac\alpha\beta \sin tv,
                 - \frac\beta\alpha \sin tu + \cos tv \right).
\end{align}
```

Вариант без номеров:

$$\begin{aligned}
\gamma_x(t) &= (\cos tu + \sin tx, v),\\
\gamma_y(t) &= (u, \cos tv + \sin ty),\\
\gamma_z(t) &= \left(\cos tu + \frac{\alpha}{\beta} \sin tv, -\frac{\beta}{\alpha} \sin tu + \cos tv \right).
\end{aligned}$$

Он был получен так:

```
\begin{align*}  ...  \end{align*}
```

8.7.5 Использование окружений align и split внутри gather

При использовании окружения `align` внутри окружения `gather` одно из них или оба обязаны присутствовать как ненумеруемые варианты (т. е. варианты со звездочкой), поскольку одновременная нумерация и для внешнего и для внутреннего окружения бессмысленна.

Вот пример нумеруемого варианта `gather` вместе со `split` и `align*`:

$$\begin{aligned}
\varphi(x,z) &= z - \gamma_{10}x - \sum_{m+n\geq 2} \gamma_{mn} x^m z^n\\
&= z - Mr^{-1}x - \sum_{m+n\geq 2} Mr^{-(m+n)} x^m z^n\\
\zeta^0 &= (\xi^0)^2,\\
\zeta^1 &= \xi^0\xi^1
\end{aligned} \tag{8.57}$$

Здесь окружение `split` нумеруется посредством внешнего окружения `gather`, а номера отдельных строк в `align*` опущены, так как используется вариант со звездочкой.

```
\begin{gather}
  \begin{split}
    \varphi(x,z)
      &= z - \gamma_{10} x - \sum_{m+n\ge2} \gamma_{mn} x^m z^n \\
      &= z - M r^{-1} x - \sum_{m+n\ge2} M r^{-(m+n)} x^m z^n
  \end{split}                                                    \\[6pt]
  \begin{align*}
    \zeta^0   &= (\xi^0)^2,      \\
    \zeta^1   &= \xi^0 \xi^1
    \end{align*}
  \end{gather}
```

Ниже показан пример варианта со звездочкой окружения gather вместе с окружением align без звездочки:

$$\begin{aligned}\varphi(x,z) &= z - \gamma_{10}x - \sum_{m+n\ge2} \gamma_{mn} x^m z^n \\ &= z - Mr^{-1}x - \sum_{m+n\ge2} Mr^{-(m+n)} x^m z^n\end{aligned}$$

$$\zeta^0 = (\xi^0)^2, \tag{8.58}$$

$$\zeta^1 = \xi^0 \xi^1 \tag{8.59}$$

Последнее было набрано следующим образом:

```
\begin{gather*}
    \begin{split} ...\end{split}            \\[6pt]
    \begin{align}...\end{align}
\end{gather*}
```

8.7.6 Использование окружений alignat

Нумеруемый вариант:

$$V_i = v_i - q_i v_j, \qquad X_i = x_i - q_i x_j, \qquad U_i = u_i, \qquad \text{for } i \neq j\ ; \tag{8.60}$$

$$V_j = v_j, \qquad X_j = x_j, \qquad U_j u_j + \sum_{i\neq j} q_i u_i. \tag{8.61}$$

Этот пример был набран при помощи следующих команд:

```
\begin{alignat}{3}
V_i &= v_i - q_i v_j,  &  \qquad X_i &= x_i - q_i x_j,
  & \qquad U_i &= u_i, \qquad \text{for $i\ne j\,$;}\label{eq:B}  \\
V_j &= v_j,            &  \qquad X_j &= x_j,
  & \qquad U_j u_j &+ \sum_{i\ne j} q_i u_i.
\end{alignat}
```

Вариант без номеров:

$$\begin{aligned} V_i &= v_i - q_i v_j, & X_i &= x_i - q_i x_j, & U_i &= u_i, & \text{for } i \neq j\ ; \\ V_j &= v_j, & X_j &= x_j, & U_j u_j &+ \sum_{i \neq j} q_i u_i. \end{aligned}$$

Он был получен так:

```
\begin{alignat*}{3}  ...  \end{alignat*}
```

Обычно окружение `alignat` используется для набора формул вида:

$$\begin{aligned} x &= y & &\text{в силу (8.48)} && (8.62)\\ x' &= y' & &\text{в силу (8.60)} && (8.63)\\ x + x' &= y + y' & &\text{по аксиоме 1.} && (8.64) \end{aligned}$$

Этот пример был получен при помощи команд:

```
\begin{alignat}{2}
 x      &= y     && \qquad \text{в силу (\ref{eq:A})}\label{eq:C} \\
 x'     &= y'    && \qquad \text{в силу (\ref{eq:B})}\label{eq:D} \\
 x + x' &= y+y'  && \qquad \text{по аксиоме~1.}
\end{alignat}
```

А вот расширенный вариант `flalign`:

$$\begin{aligned} x &= y & &\text{в силу (8.62)} && (8.65)\\ x' &= y' & &\text{в силу (8.63)} && (8.66)\\ x + x' &= y + y' & &\text{по аксиоме 1.} && (8.67) \end{aligned}$$

Это было порождено следующим образом:

```
\begin{flalign}  ...  \end{flalign}
```

8.8 Расширения для окружения theorem

Пакет theorem, разработанный Франком Миттельбахом [48], представляет собой расширение LaTeX'овского механизма для создания «теорем» [$\mathcal{L}$ **58, 174**], [$\mathcal{A}$ **239–41**]; он дает возможность управлять оформлением «теорем» посредством указания стиля.

В данном контексте слово «теорема» относится к любому типу формулировок (каких-либо высказываний) с заголовками, часто выделяемых из основного текста

дополнительными пробелами и изменением шрифта. Теоремы, следствия, гипотезы, определения, замечания — все это примеры «теорем». Заголовок теоремы состоит из названия (скажем, ТЕОРЕМА или ЗАМЕЧАНИЕ) и номера, позволяющего отличать друг от друга «теоремы» с одним и тем же названием.

Чтобы отвечать требованиям различных математических журналов, часто бывает необходимо надлежащим образом приспособить окружение, определяющее оформление теорем. В дополнение к этому, чтобы различать «тип теорем», возможно, потребуются различные форматы: например, тексты замечаний и определений должны набираться прямым шрифтом, а курсив применяется для формулировок настоящих теорем.

8.8.1 Как определять новые окружения типа теоремы

Как и в исходной версии LaTeX'a, команда `\newtheorem` определяет новое окружение типа теоремы. Два обязательных ее аргумента присваивают имя новому окружению и предоставляют текст (для заголовка теоремы), который будет печататься всякий раз, когда используется это окружение, а необязательный аргумент определяет, каким образом новое окружение будет нумероваться:

`\newtheorem{`*env-name*`}{`*label-text*`}`

Здесь команда `\newtheorem` определяет окружение с именем *env-name* и текст *label-text*, который будет всякий раз печататься (в заголовке). В этом окружении используется свой собственный счетчик.

`\newtheorem{`*env2-name*`}[`*env-name*`]{`*label-text2*`}`

Эта команда `\newtheorem` определяет окружение с именем *env2-name* и название заголовка *label-text2*. Она использует тот же самый счетчик, что и *env-name*[11].

`\newtheorem{`*env3-name*`}{`*label-text3*`}[`*section*`]`

Этот вариант команды `\newtheorem` определяет окружение *env3-name* и название заголовка *label-text3*. Здесь счетчик *label-text3* подчинен счетчику *section*, т. е. с каждым новым разделом `\section` нумерация начинается цифрой один, а сам номер состоит из номера раздела и номера самой «теоремы».

`\theoremstyle{`*style*`}`

При помощи команды `\theoremstyle` можно охарактеризовать оформление различных, если не всех, окружений типа теоремы. Следует отметить, что любое окружение типа теоремы, определенное при помощи команды `\newtheorem`, печатается в стиле `\theoremstyle`, который является текущим к моменту определения.

[11] Точнее, имя этого счетчика совпадает с именем *env-name*.— *Прим. перев.*

Таким образом, команды

```
\theoremstyle{break}          \newtheorem{Cor}{Corollary}
\theoremstyle{plain}          \newtheorem{Exa}{Example}[section]
```

приводят к тому, что окружение Cor оформляется в стиле break, а окружение Exa и все следующие за ним окружения оформляются в стиле plain, если только не указана еще одна команда \theoremstyle. Так как определения, данные при помощи команды \newtheorem, глобальны, то вы также можете локально ограничить действие команды \theoremstyle, используя фигурные скобки для группирования.

\theorembodyfont{*font-declarations*}

Выбор шрифта для формулировки (т. е. текста внутри теоремы) никак не зависит от выбранного стиля \theoremstyle; такое положение дел имеет свои преимущества. Например, команды

```
{\theorembodyfont{\rmfamily}          \newtheorem{Rem}{Remark}}
```

определяют окружение типа теоремы Rem, которое будет набираться шрифтом \rmfamily в текущем оформлении (которое в нашем примере есть стиль plain). Как и для \theoremstyle, выбранный шрифт формулировки \theorembodyfont является текущим к моменту применения \newtheorem. Если команда \theorembodyfont не указана или вы используете \theorembodyfont{}, то необходимый шрифт будет определен командой \theoremstyle.

\theoremheaderfont{*font-declarations*}

Можно также «заказать» шрифт для заголовков теорем. Но это будет носить глобальный характер, а потому в преамбуле документа должно быть не более одной команды \theoremheaderfont. Если же действительно требуются различные шрифты для заголовков, то вам придется определить новые стили теорем (позволяющие вставлять нужный шрифт).

Два дополнительных параметра влияют на вертикальные пробелы вокруг окружений типа теоремы — это \theorempreskipamount и \theorempostskipamount; они определяют соответственно вертикальный пробел до и вертикальный пробел после таких окружений. Эти параметры воздействуют на все окружения типа теоремы и могут быть изменены при помощи обычных макро, меняющих длины. Задаваемые этими параметрами длины могут растягиваться и сжиматься, и, значит, в этих командах можно применять ключевые слова plus и minus. Устанавливаются эти параметры при помощи команды \setlength.

Команды для определения окружений типа теоремы, рассмотренные в этом разделе, могут располагаться только в преамбуле документа или в пакетном файле.

Стили теорем, имеющиеся в настоящее время, приведены в табл. 8.22.

plain	Воспроизводит исходное LaTeX'овское определение, но дополнительно использует параметры \theorempreskipamount и \theorempostskipamount.
break	В этом стиле заголовок теоремы расположен на отдельной строке.
marginbreak	Номер теоремы печатается на левом поле и, как в стиле break, заголовок расположен на отдельной строке.
changebreak	Как стиль break, но название заголовка и его номер поменялись местами.
change	Номер заголовка и его название поменялись местами и формулировка продолжает строку, на которой расположен заголовок.
margin	Номер заголовка ставится слева и формулировка продолжает строку, на которой расположен заголовок.

Таблица 8.22. Список существующих стилей теорем

Все стили (кроме plain) в качестве \normalfont\slshape по умолчанию используют \theorembodyfont.

8.8.2 Примеры определений и использования теорем

Предположим, что преамбула содержит следующие команды (декларации):

```
\theoremstyle{break}              \newtheorem{Cor}{Следствие}
\theoremstyle{plain}              \newtheorem{Exa}{Пример}[section]
{\theorembodyfont{\rmfamily}      \newtheorem{Rem}{Замечание}}
\theoremstyle{marginbreak}        \newtheorem{Lem}[Cor]{Лемма}
\theoremstyle{change}
\theorembodyfont{\itshape}        \newtheorem{Def}[Cor]{Определение}

\theoremheaderfont{\scshape}
```

Тогда следующие ниже типичные примеры показывают, что получается в результате применения так определенных окружений:

СЛЕДСТВИЕ 1
Предложение, набранное в окружении типа теоремы Cor.

```
\begin{Cor}
   Предложение, набранное
   в окружении типа теоремы \Lenv{Cor}.
\end{Cor}
```

ПРИМЕР 8.8.1 *Предложение, набранное в окружении типа теоремы* Exa.

```
\begin{Exa}
   Предложение, набранное
   в окружении типа теоремы \Lenv{Exa}.
\end{Exa}
```

ЗАМЕЧАНИЕ 1 Предложение, набранное в окружении типа теоремы Rem.

```
\begin{Rem}
   Предложение, набранное
   в окружении типа теоремы \Lenv{Rem}.
\end{Rem}
```

2 Лемма (Ваня Пользователь)
Предложение, набранное в окружении типа теоремы Lem.

```
\begin{Lem}[Ваня Пользователь]
  Предложение, набранное
  в окружении типа теоремы \Lenv{Lem}.
\end{Lem}
```

3 Определение (Впечатляющее)
Предложение, набранное в окружении типа теоремы Def.

```
\begin{Def}[Впечатляющее]
  Предложение, набранное
  в окружении типа теоремы \Lenv{Def}.
\end{Def}
```

Последние два примера показывают, что дает необязательный аргумент в этих окружениях (его содержимое в круглых скобках печатается сразу за названием заголовка).

8.8.3 Некоторые специальные вопросы

Заголовок теоремы и ее формулировка рассматриваются в качестве единого целого. Это означает, что `\theoremheaderfont` унаследует характерные черты `\theorembodyfont`, если используется NFSS. Так, если, например, `\theorembodyfont` имеет курсивное начертание `\itshape` и `\theoremheaderfont` имеет полужирную насыщенность `\bfseries`, то шрифт для заголовка будет характеризоваться как «расширенный полужирный курсив». Если это нежелательно, вы должны написать нечто вроде `\theoremheaderfont{\normalfont\bfseries}`, т. е. вы явно должны указывать всю необходимую информацию, касающуюся шрифтов. Подробнее о том, как это делать, см. гл. 7.

8.9 Параметры, задающие математические стили

В этом разделе объясняется, каким образом можно глобально управлять стилем, в котором набираются ваши математические формулы, и как изменять размеры некоторых элементов формул.

8.9.1 Как управлять размерами символов

Буквы и другие математические символы иногда получаются меньшего размера, если они появляются в дробях, верхних или нижних индексах. В действительности TEX имеет восемь различных стилей, в которых он может обрабатывать формулы, а именно:

D, D'	`\displaystyle`	для выключных формул
T, T'	`\textstyle`	для формул в тексте
S, S'	`\scriptstyle`	для верхних или нижних индексов
SS, SS'	`\scriptscriptstyle`	для индексов к индексам

Символы с акцентами представляют так называемые *сжатые* (*cramped*) стили, которые аналогичны обычным стилям, но показатели степени не подняты так высоко. TEX также использует три различных типа размеров для математических символов, а именно: текстовый размер (text size), размер индекса (script size) и размер повторного индекса (scriptscript size).

Формула в тексте (заключенная между двумя знаками доллара $ или между \(и \)) набирается в текстовом стиле (стиль T). Выключная формула (на отдельных строках), расположенная, например, между \[и \], будет набрана в выключном стиле (стиль D). Размер различных частей формулы можно определить согласно следующей таблице:

Символ в стиле	будет набран в	(пример)
D, D', T, T'	текстовом размере	(text size)
S, S'	размере индекса	(script size)
SS, SS'	размере повторного индекса	(scriptscript size)

Тип стиля, используемый в математических формулах, выбирается следующим образом:

стиль	верхний индекс	нижний индекс	числитель	знаменатель
D	S	S'	T	T'
D'	S'	S'	T'	T'
T	S	S'	S	S'
T'	S'	S'	S'	S'
S, SS	SS	SS'	SS	SS'
S', SS'	SS'	SS'	SS'	SS'

Последние две колонки описывают стиль, который используется при наборе числителя и знаменателя у дроби. Пример употребления различных стилей виден в записи следующей непрерывной дроби (см. также разд. 8.3.16):

$$b^0 + \cfrac{a^1}{b_1 + \cfrac{a^2}{b_2 + \cfrac{a^3}{b_3}}}$$

```
\normalsize
\[ b^0 + \frac{a^1}{b_1 +
          \frac{a^2}{b_2 +
           \frac{a^3}{b_3}}}
\]
```

Здесь часть b формулы b^0 получена в стиле D, а 0 — в стиле S; части a и b формул a^1 и b_1 получены в стилях T и T' соответственно, верхний индекс 1 — в стиле S, а нижний индекс 1 — в стиле S'. Обе части a и b формул a^2 и b_2 записаны в стиле S', а верхний и нижний индексы 2 — в стиле SS'; наконец, каждый элемент формул a^3 и b_3 набран в стиле SS'.

Предыдущий пример можно представить более элегантно, если каждый раз самостоятельно решать, какой стиль использовать. Заметьте, что для сокращения набора мы определяем макро \D для команды \displaystyle.

$$b^0 + \cfrac{a^1}{b_1 + \cfrac{a^2}{b_2 + \cfrac{a^3}{b_3}}}$$

```
\newcommand{\D}{\displaystyle}
\normalsize
\[ b^0 +  \frac{a^1}{\D b_1 +
                \frac{a^2}{\D b_2 +
                 \frac{a^3}{b_3}}}
\]
```

8.9.2 Параметры математических стилей в LaTeX'е

Поскольку для математического набора LaTeX в большой мере использует механизм TeX'а [$\mathcal{L}$ 170], [$\mathcal{A}$ 199], мы вкратце опишем параметры, которые определяют математический стиль и которые используются LaTeX'ом для набора формул. Все они являются параметрами длины, которые вы можете переопределить при помощи команд \setlength или \addtolength (см. разд. А.1.4). Более того, две стандартные опции leqno и fleqn управляют нумерацией [$\mathcal{L}$ 82], [$\mathcal{A}$ 143] и выравниванием формул. С опцией fleqn формулы выравниваются слева на фиксированном расстоянии от левого поля (см. ниже параметр \mathindent), а не располагаются в центре строки.

При указании опции leqno номера формул, генерируемые окружениями equation и eqnarray, печатаются с левой стороны, а не с правой. Заметьте, что номера формул в LaTeX'е по умолчанию ставятся справа, а при загруженном пакете amsmath — слева. Так что, чтобы иметь номера формул справа при использовании пакета amsmath, вам нужно указать опцию reqno (см. разд. 8.6.6).

В приводимом ниже перечне параметров, задающих математические стили, все длины (за исключением \jot и \arraycolsep) являются растяжимыми. При указании опции fleqn четыре параметра длины displayskip приравниваются списку, определяющему длину \topsep, к которой, в случае, когда выделенная формула начинает абзац, добавляется значение \partopsep (см. рис. 3.5).

\arraycolsep Задает половину расстояния по горизонтали между соседними колонками в окружении array (значение по умолчанию равно 5pt, см. также разд. 5.3.2).

\jot Это дополнительный вертикальный пробел, добавляемый между строками в окружениях eqnarray или eqnarray* (значение по умолчанию равно 3pt).

\mathindent Определяет величину отступа выключных формул от левого поля для опции fleqn (значение по умолчанию равно отступу списка первого уровня, т. е. 2.5em, и определяется опцией fleqn).

\abovedisplayskip Определяет дополнительный вертикальный пробел над длинной выключной формулой, за исключением случая опции fleqn, когда

используется параметр \topsep. Длинная формула — это та, которая располагается ближе к левому полю, чем конец предыдущей строки (значение по умолчанию равно 12pt plus 3pt minus 9pt).

\belowdisplayskip Определяет дополнительный вертикальный пробел под длинной выключной формулой, за исключением случая опции fleqn, когда используется параметр \topsep (значение по умолчанию равно 12pt plus 3pt minus 9pt).

\abovedisplayshortskip Определяет дополнительный вертикальный пробел над короткой выключной формулой, за исключением случая опции fleqn, когда используется параметр \topsep. Короткая формула — это та, начало которой расположено правее, чем конец предыдущей строки (значение по умолчанию равно 0pt plus 3pt).

\belowdisplayshortskip Определяет дополнительный вертикальный пробел под короткой выключной формулой, за исключением случая опции fleqn, когда используется параметр \topsep (значение по умолчанию равно 7pt plus 3pt minus 4pt).

ГЛАВА 9

LaTeX в многоязычной среде

Эта глава начинается с краткого введения в многочисленные технические проблемы, которые должны быть решены для того, чтобы использовать (LA)TEX с языком, отличным от английского. Во втором разделе обсуждается система Babel, предоставляющая удобный путь для создания документов на различных языках. Затем в качестве примера более сложного пакета мы ознакомимся с пакетом french, разработанным Бернаром Голлем. Этот пакет хорошо подходит для придания документам вида в соответствии с французскими типографскими традициями.

9.1 TEX и языки, отличные от английского

Благодаря своей популярности в академической среде, TEX быстро распространился по всему миру и в настоящее время используется не только с различными языками, базирующимися на латинском алфавите, но также с китайским, японским, корейским, коптским[1], русским, таиландским, вьетнамским, несколькими языками Индии, персидским, арабским и ивритом. Это обстоятельство быстро сделало некоторые ограничения TEX'а версии 2.x более очевидными, особенно набор 7-битных символов для ввода и невозможность одновременно загружать образцы переноса слов для более чем одного языка. Поэтому на 10-й ежегодной конференции TUG в 1989 г. Дональд Кнут объявил, что в связи с задачами многоязычной поддержки будут созданы новые версии TEX'а и METAFONT'а. Эти версии, TEX3 и METAFONT2, были официально выпущены в марте 1990 г. Хотя они представляют собой большой шаг в правильном направлении, сами по себе они

[1] Афро-азиатский язык коптов, основанный на древнеегипетском языке; в настоящее время используется главным образом в ритуальных службах коптской церкви — местной христианской церкви Египта и Эфиопии.— *Прим. перев.*

не решают всех проблем, связанных с обеспечением удобной среды для использования LATEX'а с несколькими языками или языками, отличными от английского.

Для достижения удобства пользования TEX и связанные с ним программы должны быть сделаны по-настоящему международными, и усилия нужно направить на следующее:

1. Приспособить все программы к отдельно взятому языку (или языкам):
   - обеспечить поддержку набора в различных направлениях (например, TEX--XET) [39],
   - создать шрифты, содержащие национальные символы,
   - определить стандартные кодировки символов и
   - сгенерировать образцы для алгоритма переноса слов.
2. Обеспечить перевод названий разделов документа, создать национальные макеты для стандартных документов и заготовить файлы с TEX'овским кодом для автоматической обработки зависящих от языка типографских правил.
3. Обеспечить возможность обработки многоязычных (более чем один язык в одном документе) и международных (только один язык, но есть выбор из нескольких вариантов) документов. Например, составление предметного указателя и библиографических ссылок должно выполняться в соответствии с алфавитом данного языка или данной схемой упорядочения.

В то же время вам должно быть удобно редактировать, просматривать и печатать ваши документы, используя любой данный набор символов, а LATEX должен быть в состоянии успешно обработать созданные файлы. Имеется, однако, почти столько же различных схем кодировки символов, сколько существует языков (например, в персональных компьютерах IBM PC предусмотрены дюжины кодовых страниц). В дополнение к этому имеются несколько международных стандартов, таких как серия ISO-8859-x [98]. Поэтому следует продумать вопрос о совместимости и переносимости. Если документ должен быть воспроизводим в различных условиях, вопросы стандартизации становятся важными. В частности, пересылка документов в 8-битной кодировке по электронной почте часто вызывает проблемы, поскольку некоторые почтовые серверы опускают бит старшего разряда, так что получающийся документ нельзя обработать. Проблема с почтой, возможно, решится, когда отправители будут придерживаться стандарта MIME (Multipart Internet Mail Extensions), где использование конкретного стандарта кодировки (например, ISO-8859-x) явно указывается в заголовке почтового сообщения. Однако проблемы с кодировкой документов будут разрешены только тогда, когда новые стандарты, такие как Unicode [102] или ISO-10646 [100], будут приняты всеми.

Важность указанных проблем была осознана TEX'овским сообществом, и на Портлендской встрече организации TUG в июле 1992 г. Технический Совет TUG основал «Техническую Рабочую Группу по Координации Многоязычной Поддержки» (Technical Working Group on Multiple Language Coordination, сокращенно

	00	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
0	`	“	␣	0	@	P	‘	p	Ă	Ř	ă	ř	À	Đ	à	ð
1	´	”	!	1	A	Q	a	q	Ą	Ś	ą	ś	Á	Ñ	á	ñ
2	^	„	"	2	B	R	b	r	Ć	Š	ć	š	Â	Ò	â	ò
3	~	«	#	3	C	S	c	s	Č	Ş	č	ş	Ã	Ó	ã	ó
4	¨	»	$	4	D	T	d	t	Ď	Ť	ď	ť	Ä	Ô	ä	ô
5	″	.	%	5	E	U	e	u	Ě	Ţ	ě	ţ	Å	Õ	å	õ
6	°	—	&	6	F	V	f	v	Ę	Ű	ę	ű	Æ	Ö	æ	ö
7	ˇ		’	7	G	W	g	w	Ğ	Ů	ğ	ů	Ç	Œ	ç	œ
8	˘	ı	(	8	H	X	h	x	Ĺ	Ÿ	Í	ÿ	È	Ø	è	ø
9	¯	ȷ	)	9	I	Y	i	y	Ľ	Ź	ľ	ź	É	Ù	é	ù
A	˙	j	*	:	J	Z	j	z	Ł	Ž	ł	ž	Ê	Ú	ê	ú
B	¸	ff	+	;	K	[	k	{	Ń	Ż	ń	ż	Ë	Û	ë	û
C	˛	fi	,	<	L	\	l	|	Ň	IJ	ň	ij	Ì	Ü	ì	ü
D	‚	fl	-	=	M	]	m	}	Ŋ	İ	ŋ	¡	Í	Ý	í	ý
E	‹	ffi	.	>	N	^	n	~	Ő	đ	ő	¿	Î	Þ	î	þ
F	›	ffl	/	?	O	_	o	-	Ŕ	§	ŕ	£	Ï	SS	ï	ß

Таблица 9.1. Раскладка расширенного TEX'овского шрифта, принятая в г. Корк (Ирландия) в 1990 г.

TWGMLC) во главе с Яннисом Хараламбусом. Цель этой группы состоит в том, чтобы поддерживать и координировать стандартизацию и развитие TEX'овских программных продуктов, приспособленных к различным языкам. Для каждого языка или группы языков должен быть создан стилевой пакет, облегчающий набор текста. Такой пакет должен содержать информацию о шрифтах, правилах ввода, переносах, LaTeX'овский файл опций, совместимый с концепцией babel (см. разд. 9.2), возможно препроцессор и, конечно, документацию на английском языке и на языке, который будет использоваться.

В разд. 7.5.1 мы познакомились со шрифтами DC-EC, которые используют корковскую схему кодировки, показанную в табл. 9.1. Существуют стилевые файлы для отображения различных кодировок на 256 символов шрифта DC. Например, пакет isolatin1 преобразует файлы, закодированные в соответствии со стандартом ISO-8859-1 (который известен также под названием Latin-1). Этот пакет, к примеру, переводит код "ab (левую французскую кавычку '«' в стандарте Latin-1) в знак "13 кода DC (см. табл. 9.1), если шрифты DC загружены, или в противном случае в TEX'овскую команду $<<$, дающую похожий знак. Подобные же пакеты можно создать для других кодировок, таких как широко распространенная международная кодовая страница 850 (USA) для IBM PC.

9.1.1 Механизм виртуального шрифта

Удобным средством создания новых шрифтов является механизм виртуальных шрифтов [37]. Виртуальные шрифты дают хорошее решение задачи о перестройке шрифта, т. е. они могут переопределить соответствие между кодами и символами. Они также позволяют создавать сложные символы (например, символы с диакритическими знаками) или составные шрифты (комбинируя символы из различных шрифтов, например, светлый прямой из Computern Modern и строчной из Greek) без написания специальных TEX'овских макро. Виртуальные шрифты позволяют использовать обычный метод ввода, приспособленный для данного языка (например в стандарте ISO-8859 или в кодировке ISO-10646 в будущем).

Нужно, чтобы виртуальный шрифт существовал только в логическом, не обязательно в физическом, смысле. TEX может выполнять свою работу, не зная откуда берутся реальные символы. Это уже задача драйвера устройства — используя информацию в файле, отвечающем за виртуальный шрифт, собрать сведения об изображении реальных символов. Последние можно сдвигать, увеличивать или комбинировать с другими символами из множества различных шрифтов. Виртуальный шрифт может даже использовать символы из других виртуальных шрифтов, включая и себя. Виртуальные шрифты также делают возможной удобную замену символов при чтении корректуры, когда шрифты, созданные для одного устройства вывода, отсутствуют на другом устройстве. Механизм виртуального шрифта, таким образом, дает общий интерфейсный механизм для переключения между множеством видов шрифтов, предоставляемых различными поставщиками типографского оборудования.

Однако остается проблема переносимости `dvi`-файлов с платформы на платформу. Виртуальный шрифт может содержать локальную перекодировку некоторых реальных шрифтов. В результате запуска TEX'а получится `dvi`-файл, который не может быть воспроизведен на другой платформе. Чтобы решить эту проблему с `dvi`-файлами, содержащими виртуальные шрифты, можно обращаться различными способами:

- Использовать `dvi`-драйвер, который воспринимает виртуальные шрифты. Примерами таких драйверов являются драйвер `dvips` и драйверы из дистрибутива emTEX'а.
- Если `dvi`-драйвер не поддерживает виртуальные шрифты, то можно воспользоваться какой-либо программой конвертирования `dvi-в-dvi` (например, `dvicopy` Петера Брайтенлохнера или PosTeX Василия Константиновича Малышева), чтобы устранить ссылки на виртуальные шрифты из `dvi`-файла и заменить их ссылками на реально существующие шрифты. Это делает `dvi`-файлы переносимыми.

9.2 Babel — LaTeX владеет многими языками

LaTeX'овский дистрибутив содержит несколько стандартных классов документов, которые применяются большинством пользователей. Эти классы (article, report, book и letter) придают документам «американизированный» вид, и это не всем нравится. Более того, названия разделов документа, такие как Chapter (глава) и Table of Contents (оглавление), по умолчанию приводятся по-английски.

Пакет babel, разработанный Йоханнесом Брамсом [11], предоставляет набор файлов опций, позволяющих пользователю выбирать язык, на котором будет набираться документ. Этот пакет характеризуется следующим образом:

- Несколько языков могут использоваться одновременно.
- Образцы переноса слов, которые загружаются во время работы INITEX'а, могут быть определены динамически через внешний файл.
- Для более чем двадцати языков представлены переводы названий разделов документа и команд, упрощающие ввод текста.

Среда пользователя системы babel проста. Она состоит из двух команд: одна служит для выбора языка, а вторая — для того чтобы узнать, какой язык является текущим.

9.2.1 Среда пользователя

Любой язык, который вы используете в вашем документе, должен быть указан в команде `\usepackage` в качестве языковой опции[2]. Поддерживаемые в настоящее время опции перечислены в табл. 9.3. Например, следующее предписание подготавливает к набору на немецком (опция german[3]) и итальянском (опция italian) языках.

```
\usepackage[german,italian]{babel}
```

В начале документа языком по умолчанию будет язык, указанный последним в опциях команды `\usepackage`. В приведенном примере названия разделов документа, образцы переноса слов (если они были загружены для данного языка во время генерирования LaTeX'овского формата при помощи INITEX'а; см. обсуждение на с. 306) и, возможно, интерпретация некоторых зависящих от языка команд (таких как дата) будут приспособлены для итальянского языка с начала

[2] В принципе, поскольку язык(и), на котором написан документ, является глобальной характеристикой этого документа, разумно указать его в команде `\documentclass`, например, `\documentclass[polish]{...}`. Подробнее см. разд. 2.1.

[3] Для совместимости с версией babel из LaTeX 2.09 система babel распознает также опцию germanb.

документа и до того места, где вы выберете другой язык. Вы можете изменить язык при помощи команды \selectlanguage, имеющей структуру:

\selectlanguage{*language*}

Например, если вы хотите переключиться на немецкий язык, вам следует использовать команду \selectlanguage{german}. Аналогично можно переключаться на другие языки, если только они были указаны в начале документа в команде \usepackage.

Если вы используете более одного языка, то может возникнуть необходимость узнать, какой язык является активным в определенное время. Это можно проверить, обращаясь к команде \iflanguage, имеющей вид:

\iflanguage{*language*}{*true-clause*}{*false-clause*}

Первый аргумент *language* — это название языка; второй аргумент *true-clause* определяет, какие команды должны быть выполнены, если *language* является активным языком; и третий аргумент *false-clause* указывает команды, которые следует выполнить в противном случае.

9.2.2 Опция german

Здесь мы обсудим опцию german[4] как пример возможностей, предлагаемых системой babel.

Кроме перевода зависящих от языка названий разделов документа на немецкий язык, опция german предоставляет следующее:

- "a есть сокращение для \"a, что дает ä. Аналогичным образом умляуты могут быть добавлены и к другим гласным (см. рис. 9.2);
- "s для сдвоенных s: ßs; "S дает «SS» (см. рис. 9.2);
- "ck для «ck», что при переносе станет «k-k»;
- "ff для «ff», что переносится как «ff-f», и аналогично для двойных l, m, n, p и t;
- "‘ или \glqq для нижней и "’ или \grqq для верхней немецких кавычек („Gänsefüßschen“);
- \glq для нижней и \grq для верхней ‚одинарных кавычек‘;
- "< или \flqq для левой и "> или \frqq для правой французских кавычек («guillemets»);
- \flq для левой и \frq для правой ‹одинарных французских кавычек›, используемых для *цитат внутри цитат*;
- "| для предотвращения лигатур;

[4] Опция german системы babel базируется на пакете german [57], одобренном DANTE и осуществленном Берндом Райхлем.

\selectlanguage	Результат действия команды \today
english	31st January 1993
german	31. Januar 1993
austrian	31. Jänner 1993

Рис. 9.1. Даты и диалекты в системе babel

- "- для указания места переноса, подобно \-, но переносы до и после указанного места остаются возможными;
- "" также указывает место переноса, но знак переноса не печатается, если даже слово и разбито на части;
- \dq печатает знак ".

Примеры этих команд приводятся ниже.

Nachfolgend sind einige Beispiele für die Verwendung der deutschen Befehle:

Die schönste älteste Brücke
DIE SCHÖNSTE ÄLTESTE BRÜCKE
Straße oder STRASSE

„Ja, bitte!“, »Nein, danke!«
„Sag' immer ‚Ja, bitte!‘“
»Sag' immer ›Ja, bitte!‹«

Drucker getrennt: Druk-ker
Rolladen getrennt: Roll-laden
Auflage (statt Auflage)

```
Nachfolgend sind einige Beispiele f"ur
die Verwendung der deutschen Befehle:
\begin{flushleft}
Die sch"onste "alteste Br"ucke    \\
DIE SCH"ONSTE "ALTESTE BR"UCKE    \\
Stra"se oder STRA"SE              \\[3pt]
"'Ja, bitte!"', ">Nein, danke!"<  \\
"'Sag' immer \glq Ja, bitte!\grq"'\\
">Sag' immer \frq Ja, bitte!\flq"<\\[3pt]
Dru"cker  getrennt: Druk-ker      \\
Ro"lladen getrennt: Roll-laden    \\
Auf"|lage (statt Auflage)
\end{flushleft}
```

Различия между «диалектами» языка также учитываются, как видно из рис. 9.1, который показывает, как выглядят на печати даты.

На рис. 9.2 приводится исходный файл с текстом на немецком языке, который был введен с применением соглашений опции german системы babel. Особенно обратите внимание на использование LaTeX'овских команд для создания оглавлений и надписей к рисунку и таблице. Распечатка результата обработки файла приведена на с. 309, где видно, что названия разделов документа переведены на немецкий язык («Inhaltverzeichnis» вместо «Table of Contents», «Abbildung» вместо «Figure» и т. д.). Вам следует сравнить эти две страницы с рис. 9.3 и следующей страницей, где показаны LaTeX'овский файл, основной текст которого взят со с. 308 и переведен на французский язык, и результат его распечатки. Как видно, использованы одни и те же LaTeX'овские команды, но во втором случае, просто указывая опцию french вместо german в команде \usepackage, получаем все зависящие от языка названия разделов документа на французском языке.

9.2.3 Структура языковых стилевых файлов системы babel

Языковые файлы системы babel должны подчиняться некоторым соглашениям. Прежде всего они должны определять новый язык, а затем указывать информацию, характерную для него.

`\addlanguage{`*language*`}`

Макрокоманда `\addlanguage` определяет новый язык *language*.

`\adddialect{`*dialect-name*`}{`*language*`}`

Макрокоманду `\adddialect` можно применять, когда два языка могут (или должны) использовать одни и те же правила переноса слов. Это может быть полезным, когда вы хотите использовать язык, для которого в данном формате не загружены образцы переноса слов. В таком случае система babel по умолчанию ведет себя так, чтобы определить этот язык как «диалект» того языка, для которого правила переноса были загружены как `\language0`.

Информация, характерная для языка, поставляется посредством четырех макро.

`\captions`*lang*

Макрокоманда `\captions`*lang* определяет ряд команд, которые заключают в себе названия разделов документа на языке *lang* для замещения исходных английских названий разделов документа. Имена этих команд приведены в первой колонке табл. 9.2, а в оставшихся колонках показаны значения этих команд для английского и чешского языков.

`\date`*lang*

Макрокоманда `\date`*lang* определяет команду `\today` на языке *lang*, включая правильные написание даты и названий месяцев.

`\extras`*lang*

Макрокоманда `\extras`*lang* содержит все дополнительные определения, необходимые для данного языка *lang*, такие как типографские сокращения для проставления акцентов, умляутов, и задействование правил для знаков препинания. В некоторых случаях имеются разновидности одного и того же базового языка для различных сообществ пользователей (например, португальский язык в Бразилии и Португалии и немецкий язык в Германии и Австрии). Чтобы облегчить правильное использование типографских норм, для некоторых языков в файле опций определены дополнительные команды, например, для чешского языка (некоторые акценты), для голландского языка (типографские сокращения и правила написания с переносами), для французского языка (множество правил,

сокращений и специальных форматов перечней и библиографий), для немецкого языка (типографские сокращения и правила написания с переносами; примеры см. в разд. 9.2.2) и для испанского языка (типографские нормы).

Вы можете добавить ваши собственные определения команд, и, когда язык выбран, они вступят в силу, если вы укажете их в качестве аргументов команды `\addto`, например,

```
\addto\extrasdutch{мои дополнительные определения}
```

В приведенном примере все ваши дополнительные определения команд станут активными вместе с уже определенными командами системы babel, когда встретится команда `\selectlanguage{dutch}`.

`\noextras`*lang*

Когда вы закончили работу на языке *lang*, система babel должна вернуть вас в то состояние, в котором вы находились до выбора языка *lang*. Как указано выше, команда `\extras`*lang* фиксирует все определения команд для языка *lang*, но в то же время она сохраняет и те определения команд, которые она изменяет. Команда `\noextras`*lang* использует эти сохраненные значения, чтобы вернуть LaTeX в состояние, в котором он находился перед выполнением команды `\extras`*lang*.

Правила переноса слов для различных языков, поддерживаемых в конкретной конфигурации системы babel, не могут быть загружены во время работы LaTeX'а, однако они должны быть считаны во время генерирования LaTeX'овского формата при помощи INITEX'а. Для того чтобы контролировать загрузку правил переноса при запуске INITEX'а, система babel использует внешний файл `language.dat`, содержащий по одной строке для каждого языка, для которого имеются образцы переноса слов.

Вот пример файла `language.dat`:

```
% Имя: language.dat
% Использование: Соответствие: язык - файл правил переноса
english     ehyphen.tex   % английский
german      ghyphen3.tex  % немецкий
french      fr8hyph.tex   % французский
italian     italhyph.tex  % итальянский
spanish     spanhyph.tex  % испанский
```

Первый элемент на каждой строке указывает имя языка, а затем следует имя файла, содержащего образцы переноса слов. Для каждого языка в файле `language.dat` определена команда `\l@`*lang*, т. е. `\l@english` и т. д. При запуске LaTeX'а для данного языка *lang* система babel, проверяет определена ли команда `\l@`*lang*, и если да, то загружает соответствующие правила переноса; в противном случае она загружает правила переноса для языка по умолчанию language 0 (первого языка, загруженного INITEX'ом, т. е. английского в приведенном примере).

Команда	Английский	Чешский
\abstractname	Abstract	Abstrakt
\alsoname	see also	viz též
\appendixname	Appendix	Dodatek
\bibname	Bibliography	Literatura
\ccname	cc	cc
\chaptername	Chapter	Kapitola
\contentsname	Contents	Obsah
\enclname	encl	Příloha
\figurename	Figure	Obrázek
\headtoname	To (letter)	Komu
\indexname	Index	Index
\listfigurename	List of Figures	Seznam obrázků
\listtablename	List of Tables	Seznam tabulek
\pagename	Page	Strana
\partname	Part	Část
\prefacename	Preface	Předmluva
\seename	see	viz
\refname	References	Reference
\tablename	Table	Tabulka

Таблица 9.2. Примеры перевода названий LaTeX'овских разделов документа в системе babel

Имеющаяся в настоящее время версия системы babel распознает следующие опции:

american (вариант опции english),		austrian (вариант опции german),	
brazil (вариант опции portuges),		catalan,	croatian,
czech,	danish,	dutch,	english,
esperanto,	finnish,	french,	galician,
german,	italian,	magyar,	norsk,
nynorsk (вариант опции norsk),		polish,	portuges,
romanian,	russian,	slovak,	slovene,
spanish,	swedish,	turkish	

(а также francais и germanb).

Таблица 9.3. Опции, поддерживаемые системой babel

```
\documentclass[german]{article}% класс документа article и опция german
\usepackage{babel}              % загрузить пакет babel
\usepackage[dvips]{epsfig}      % загрузить пакет epsfig и использовать
                                % драйвер dvips
\usepackage{makeidx}            % загрузить пакет makeidx
\newcommand{\DQ}[1]{\texttt{\dq#1}\enspace"#1}% Напечатать аргумент,
                                              % которому предшествует "
\makeindex
\begin{document}              % Конец преамбулы и начало текста.
\begin{center}\Large Beispiel eines Artikels in deutscher Sprache\\
\today\end{center}
\tableofcontents
\listoffigures
\listoftables
\section{Eine EPS Abbildung}\index{Abschnitt}
Dieser Abschnitt zeigt, wie man eine PostScript-Abbildung\cite{bib-PS}
in ein \LaTeX{} Dokument einbinden kann.
Abbildung~\ref{Fpsfig} wurde mit dem Befehl
\verb!\epsfig{file=colorcir.eps,width=3cm}! in den Text aufgenommen.
\index{Abbildung}\index{PostScript}
\begin{figure}[hbt]
  \centering
  \begin{tabular}{c@{\qquad}c}
     \mbox{\epsfig{file=colorcir.eps,width=3cm}}&
     \mbox{\epsfig{file=tac2dim.eps,width=3cm}}
  \end{tabular}
  \caption{Zwei EPS Bilder}\label{Fpsfig}
\end{figure}
\section{Beispiel einer Tabelle}
Die Tabelle~\ref{tab:exa} auf Seite~\pageref{tab:exa}
zeigt, wie man die Umgebung \texttt{table} gebrauchen kann.
\begin{table}[hbt]
  \centering\begin{tabular}{ccccccc}
    \DQ{a}  & \DQ{A} & \DQ{o} & \DQ{O} & \DQ{u}  & \DQ{U} & \DQ{s}
  \end{tabular}
  \caption{Eingabe der deutschen Zusatzzeichen}\label{tab:exa}\index{Tabelle}
\end{table}
\begin{thebibliography}{99}
\index{Literaturverzeichnis}
  \bibitem{bib-PS} Adobe Inc. \emph{PostScript Handbuch (2. Auflage)}
                   Addison-Wesley (Deutschland) GmbH, Bonn, 1991
\end{thebibliography}
\printindex
\index{Index}
\end{document}              % Конец документа.
```

Рис. 9.2. Пример опции german системы babel (распечатка на следующей странице)

Beispiel eines Artikels in deutscher Sprache

11. Oktober 1993

Inhaltsverzeichnis

1 Eine EPS Abbildung **1**

2 Beispiel einer Tabelle **1**

Abbildungsverzeichnis

1 Zwei EPS Bilder 1

Tabellenverzeichnis

1 Eingabe der deutschen Zusatzzeichen 1

1 Eine EPS Abbildung

Dieser Abschnitt zeigt, wie man eine PostScript-Abbildung[1] in ein LaTeX Dokument einbinden kann. Abbildung 1 wurde mit dem Befehl `\epsfig{file=colorcir.eps,width=3cm}` in den Text aufgenommen.

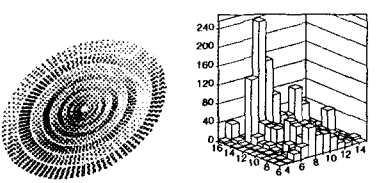


Abbildung 1: Zwei EPS Bilder

2 Beispiel einer Tabelle

Die Tabelle 1 auf Seite 1 zeigt, wie man die Umgebung `table` gebrauchen kann.

`"a`	ä	`"A`	Ä	`"o`	ö	`"O`	Ö	`"u`	ü	`"U`	Ü	`"s`	ß

Tabelle 1: Eingabe der deutschen Zusatzzeichen

Literatur

[1] Adobe Inc. *PostScript Handbuch (2. Auflage)* Addison-Wesley (Deutschland) GmbH, Bonn, 1991

Index

Abbildung, 1
Abschnitt, 1
Index, 1
Literaturverzeichnis, 1
PostScript, 1
Tabelle, 1

```
\documentclass{article}
\usepackage[french]{babel} % загрузить пакет babel и активизировать
                           % опцию french
\usepackage[textures]{epsfig}% загрузить пакет epsfig и использовать
                           % код драйвера textures
\usepackage{makeidx}       % загрузить пакет makeidx
\newcommand{\Lcs}[1]{\texttt{\symbol{'134}#1}\enspace}% перед аргументом
                                                      % добавить \
\makeindex
\begin{document}           % Конец преамбулы и начало текста.
\begin{center}\Large Exemple d'un article en fran\c{c}ais\\
\today\end{center}
\tableofcontents
\listoffigures
\listoftables
\section{Une figure EPS}
\index{section}
Cette section montre comment inclure une figure PostScript\cite{bib-PS}
dans un document \LaTeX. La figure~\ref{Fpsfig}
est ins\'er\'ee dans le texte \`a l'aide de la commande
\verb+\epsfig{file=colorcir.eps,width=3cm}+.
\index{figure}\index{PostScript}
\begin{figure}[hbt]
  \centering
  \begin{tabular}{c@{\qquad}c}
    \mbox{\epsfig{file=colorcir.eps,width=3cm}}&
    \mbox{\epsfig{file=tac2dim.eps,width=3cm}}
  \end{tabular}
  \caption{Deux images EPS}\label{Fpsfig}
\end{figure}
\section{Exemple d'un tableau}
Le tableau~\ref{tab:exa} \`a la page \pageref{tab:exa}
montre l'utilisation de l'environnement \texttt{table}.
\begin{table}[hbt]
  \centering\begin{tabular}{cccccc}
    \Lcs{primo} \primo & \Lcs{secundo} \secundo & \Lcs{tertio} \tertio &
    \Lcs{quatro} \quatro & 2\Lcs{ieme}\ 2\ieme
  \end{tabular}
  \caption{Quelques commandes de l'option \texttt{french} de \texttt{babel}}
  \label{tab:exa}\index{tableau}
\end{table}
\begin{thebibliography}{99}
  \bibitem{bib-PS} Adobe Inc.
  \emph{PostScript, manuel de r\'ef\'erence (2\ieme \'edition)}
  Inter\'Editions (France), 1992 \index{r\'ef\'erences}
\end{thebibliography}
\printindex
\index{index}
\end{document}             % Конец документа.
```

Рис. 9.3. Пример опции french системы babel (распечатка на следующей странице)

Exemple d'un article en français

11 octobre 1993

Table des matières

1 Une figure EPS 1

2 Exemple d'un tableau 1

Liste des figures

1 Deux images EPS 1

Liste des tableaux

1 Quelques commandes du style `francais` 1

1 Une figure EPS

Cette section montre comment inclure une figure PostScript[1] dans un document LaTeX. La figure 1 est insérée dans le texte à l'aide de la commande `\epsfig{file=colorcir.eps,width=3cm}`.

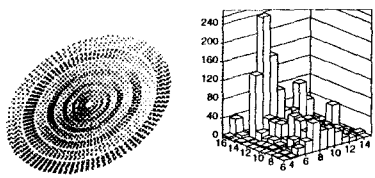


Figure 1: Deux images EPS

2 Exemple d'un tableau

Le tableau 1 à la page 1 montre l'utilisation de l'environnement `table`.

`\primo` 1° `\secundo` 2° `\tertio` 3° `\quatro` 4° 2`\ieme` 2e

Tableau 1: Quelques commandes du paquet `francais`

Références

[1] Adobe Inc. *PostScript, manuel de référence (2e édition)* InterÉditions (France), 1992

Index

figure, 1
index, 1
PostScript, 1
références, 1
section, 1
tableau, 1

9.3 Следование типографским нормам

Система babel, представленная в предыдущем разделе, хорошо справляется с переводом названий разделов документа и облегчает ввод. В этом разделе мы познакомимся со вторым классом пакетов, которые более продвинуты, поскольку пытаются приспособить LaTeX лучше отражать типографские традиции используемого языка.

Примером такого пакета является пакет french [18], который был создан Бернаром Голлем. Пакет french не ограничивается переводом зависящих от языка названий разделов документа и определением аббревиатур для проставления акцентов, а предоставляет также команды, облегчающие применение французских типографских норм, и добавляет/переопределяет некоторые LaTeX'овские команды, придающие документам, набираемым при помощи пакета french, вид действительно французских изданий.

9.3.1 Традиционные французские типографские нормы

В этом разделе собраны наиболее важные правила набора текстов на французском языке (см. работы [91, 94, 97]). Следуя этим правилам, вы извлечете наибольшую пользу от возможностей пакета french. Однако вы можете заставить этот пакет самому попытаться проделать всю работу — например, если вы не хотите оставлять пробелы перед составными знаками пунктуации и т. п., то вы можете указать команду `\untypedspaces`, которая выполнит нужное действие. Заметьте, однако, что такая автоматическая процедура не всегда может привести к ожидаемому результату.

- Пробел должен ставиться
  - перед составным знаком пунктуации (! ? : ;);
  - перед закрывающей кавычкой guillemets (»);
  - перед знаком % и вообще перед единицами мер, например 10 km, 1000 francs;
  - после « и, конечно, после » ; : , . ! ?
- Кавычки guillemets (« и ») вводятся как << и >>. Обычно во французском языке другие кавычки (‘ " ’ ` ´ “ „ ”) не используются.
- Многоточие (points de suspension) вводится как три обычные точки (...).
- В числах между целой и дробной частями ставится запятая, например 3,1415926.
- Латинские выражения (вполне обычное дело во французском языке) набираются *курсивом* посреди прямого текста (исключение составляют слова cf., etc. и латинские слова, ставшие французскими, как, например, слово critérium).
- Вот некоторые общепринятые аббревиатуры и способы их ввода:

```
c.-\‘a-d. / \emph{i.e.} / p.ex. / etc. / cf. / id. / p.i. / p.o.
 chap. / part. / vol. / paragr. /  R.S.V.P.  / T.S.V.P. /...
```

В действительности дистрибутив пакета french поставляется вместе с файлом `frabbrev`, содержащем наиболее важные сокращения французского языка.

- Между буквами сокращений торговых марок, названий компаний и т. п. не должно быть точек, а сами буквы обычно набираются капителью. Для этого случая предусмотрена команда `\lcs` — т. е. `\lcs{RATP}` и `\lcs{SnCf}` соответственно дадут RATP и SNCF, независимо от того, в каком регистре были введены аргументы.
- Считается, что прописные гласные уже имеют акценты.
- Только первая буква первого слова в названии должна быть прописной [1].
- В простом нумерованном перечне, т. е. когда любой элемент перечня содержит только одно предложение, каждый элемент этого перечня должен начинаться с en-тире, первое слово должно начинаться со строчной буквы и все предложения в перечне, кроме последнего, должны заканчиваться точкой с запятой.
- Фамилии пишутся капителью, а имена — прямым светлым шрифтом, например Donald KNUTH. Для этого случая предусмотрена команда `\fcs`; таким образом, обе команды `\fcs{KNUTH}` и `\fcs{knuth}` напечатают KNUTH.

9.3.2 Команды пакета french

Для облегчения процесса создания документов на французском языке добавлено несколько команд. Ниже приводятся лишь некоторые из них. За более подробной информацией обратитесь к исходной документации.

- `\begin{resume}` текст резюме `\end{resume}`;
- `\begin{order} ... \end{order}` порождает нумерованный перечень вида (1° ... 2° ...);
- `\sommaire[n]` используется для аннотации; `[n]` указывает уровень вложенности для заголовка;
- `\annexe` или `\annexes` используется для добавлений или приложений;
- `\glossaire` или `\glossaires` используется для глоссария (словаря, приложенного в конце книги);
- для основного текста предусмотрены следующие команды :
  - `\ier` для 1er (`1\ier{}`); `\iere` для 1re (`1\iere{}`); `\ieme` для 2^{e} (`2\ieme{}`);
  - `\primo`, `\secundo`, `\tertio`, `\quatro` напечатают 1°, 2°, 3°, 4°. Другие числа можно получать при помощи команды `\quando`, например, `\quando={11}` дает 11° ;
  - команда `\fup{`*text*`}` поднимает аргумент *text* и печатает его в меньшем размере, например, чтобы получить XVIe, нужно набрать `XVI\fup{e}`;
- различные специальные команды для набора личных и деловых писем, выглядящих «по-французски».

Английский		Французский	
ввод	вывод	ввод	вывод
`ellipsis\ldots`	ellipsis ...	`points de suspension...`	points de suspension...
`123,456.789`	123,456.789	`123 456,789`	123 456,789
`semicolon;`	semicolon;	`point-virgule ;`	point-virgule ;
`He said: Yes`	He said: Yes	`Il dit : oui`	Il dit : oui
`My god!`	My god!	`Mon Dieu !`	Mon Dieu !
`Why not?`	Why not?	`Pourquoi pas ?`	Pourquoi pas ?
``` ``I say'' ```	"I say"	`<< Je dis >>`	« Je dis »

Таблица 9.4. Сравнение французского и английского печатных текстов

В зависимости от точной комбинации указанных в опциях команд пакет french делает различные символы активными, а именно < ‘ " ’ > и : ; ! ?. Тем самым эти символы эквивалентны командным последовательностям, и их появление вызовет TEX'овские макрокоманды, связанные с их именами. Это может привести к затруднениям, если вы используете пакет french вместе с другими пакетами, в которых используются эти символы. Чтобы ввести эти символы в ваш файл без ущерба для него, используйте полные варианты этих команд:

`\inferieura`	вместо	<	`\pointvirgule`	вместо	;
`\superieura`	вместо	>	`\pointexclamation`	вместо	!
`\lq`	вместо	‘	`\pointinterrogation`	вместо	?
`\rq`	вместо	’	`\dittomark`	вместо	"
`\lqq`	вместо	‘‘	`\deuxpoints`	вместо	:
`\rqq`	вместо	’’			

9.3.3 Структура пакета french

Пакет french состоит из четырех частей: общие типографские нормы (для основного текста), макет полосы набора, перевод названий разделов документа и дополнительные определения команд. При загрузке этого пакета все четыре части активизируются. Каждая часть может быть задействована или отключена индивидуально при помощи команд:

`\frenchtypography`	...	`\nofrenchtypography`
`\frenchlayout`	...	`\nofrenchlayout`
`\frenchtranslation`	...	`\nofrenchtranslation`
`\frenchmacros`	...	`\nofrenchmacros`

Некоторые из основных различий между тем, как вводятся французские и английские тексты, и их печатные варианты показаны в табл. 9.4.

ГЛАВА 10

Графика в LaTeX'е

По всей видимости TeX обладает наилучшим алгоритмом для верстки абзацев и создания страниц из них. Но в наш век возрастающего обмена информацией все большее число публикаций не ограничивается текстом — важность графического материала значительно выросла. Сам по себе TeX не касается этой проблемы, поскольку он имеет дело только с расположением (черных) боксов на странице. Однако Дональд Кнут предусмотрительно ввел команду `\special`, предоставляющую возможности, которые отсутствуют в базовом языке. При верстке эта команда не влияет на выводимую страницу, но TeX расположит материал, указанный в качестве аргумента команды `\special`, в текущей точке `.dvi`-файла буквально[1]. Это уже `dvi`-драйвер должен обработать полученную информацию и вывести изображение соответствующим образом.

Множество статей посвящено тому, как наилучшим образом приспособить TeX для работы с графикой, и различные авторы нашли свои решения, которые зачастую зависят от создателя, а потому не годятся для всеобщего использования.

В своей обзорной статье «A Survey of TeX and Graphics» [63] Себастьян Ратц обсуждает шесть различных путей для воспроизводства графики в TeX'e[2]:

1. ASCII-рисование, такое как PiCTeX [13], предоставляет совершенный язык для черчения, где кривые получаются сочетанием очень большого числа маленьких точек. Для сложных чертежей это требует очень большой внутренней памяти для TeX'овской программы, а также массу вычислительного времени.
2. Шрифты с элементами картинок, как в LaTeX'овском окружении `picture`. Система XY-pic Кристофера Роуза [70] использует специальные шрифты для печати диаграмм.

---

[1] В некоторых случаях команда `\special` может изменить расположение элементов на странице, поскольку эта команда может произвести дополнительную точку разбивки, что помешает LaTeX'у воспринимать пробелы.

[2] См. также статью Малколма Кларка «Portable Graphics in TeX» [27].

3. Макропакеты для рисования, базирующиеся главным образом на окружении `picture` или на оригинальных TeX'овских командах для рисования линий. Среди прочего имеются пакеты для рисования фейнмановских диаграмм, химических формул и деревьев.
4. Шрифты с картинками, где каждый печатающийся символ — это одна, возможно большая, «буква» в шрифте. Для порождения этих картинок можно использовать METAFONT [82] или уже существующие растры и преобразовать их напрямую в `.pk`-файл. Программа `pbmtopk` Ангуса Даггана может преобразовать картинку из одного из pbm-форматов (portable bitmap) в формат `.pk`-файла; она ставит в соответствие буквам алфавита последовательность картинок. Программа BM2FONT конвертирует различные виды растровых шрифтов (bitmap-файлов) в TeX'овские шрифты и записывает входной файл для интегрирования этой графики в документы [73].
5. Полутоновые шрифты — это такие блоки, которые состоят из различной насыщенности серого цвета и которые могут быть скомбинированы обычным TeX'овским способом для создания картинок [26, 35].
6. Графический материал может быть включен при помощи команды `\special`. По определению такой подход зависит от устройства, поскольку он опирается на возможности `dvi`-драйвера и выводящего устройства. Однако с распространением дешевых программ предварительного просмотра (previewer) и принтеров, использующих язык PostScript, этот подход становится все более популярным. Пакеты `psfrag` и `pstricks` представляют собой примеры систем, использующих PostScript вместе с LaTeX'ом. См. также обсуждение в гл. 11.

Все эти методы имеют свои достоинства и недостатки. Хотя варианты 1–5 независимы от устройства, они лишены гибкости, особенно когда иллюстрации нужно масштабировать или поворачивать. Применение команды `\special` ограничено только возможностями используемого языка, и, хотя этот подход по определению зависим от устройства, в случае PostScript'а он позволяет использовать все команды этого богатого языка и включать материал, производимый при помощи множества пакетов, которые обеспечивают вывод в PostScript'е. Все это будет обсуждаться в следующей главе.

В этой главе главным образом обсуждаются встроенные в LaTeX графические средства, базирующиеся на окружении `picture`, а также пакеты, которые расширяют это окружение для того, чтобы порождать высококачественную не зависящую от устройства графику. Мы также рассматриваем и другие визуальные эффекты, такие, как «рамки с тенью», которые можно воспроизвести на разных устройствах. Используя портативные устройства, вы можете быть уверены, что получатель документа, имеющий то же самое LaTeX'овское окружение, что и его разработчик, на выходе получит результат, идентичный отправленному.

В первом разделе этой главы обсуждаются орнаменты, которые могут быть полезны при выделении важного материала. В следующем разделе рассматривается окружение `picture` и рассказывается о пакетах для рисования гистограмм и произвольных кривых. Затем мы уделяем внимание двум пакетам `epic` и `eepic`, которые расширяют окружение `picture` при помощи некоторого набора новых

команд. Все они детально описаны, а примеры показывают, как их использовать на практике. В конце мы обсуждаем два пакета, базирующихся на epic и eepic, для рисования двудольных графов и деревьев.

10.1 Орнаменты

LaTeX'овские рамки кратко рассматриваются в разд. А.2. Ниже мы представляем пакеты, расширяющие возможности обычных LaTeX'овских рамок.

10.1.1 Министраницы в рамке

Окружение boxedminipage, определенное в пакете boxedminipage (созданном Марио Волчко), ведет себя как стандартное окружение minipage, только еще заключает все в рамку. Толщина линий контролируется при помощи стилевых параметров \fboxrule и \fboxsep.

Это небольшой пример министраницы в рамке[a], которая, к тому же, имеет сноску.

[a] Очень простой пример.

```
\begin{boxedminipage}[t]{5.3cm}
Это небольшой  пример министраницы
в рамке\footnote{Очень простой пример.},
которая, к~тому же, имеет сноску.
\end{boxedminipage}
```

10.1.2 Рамки с тенью

Пакет shadow (разработанный Мауро Орландини) определяет команду \shabox, которая аналогична LaTeX'овской команде \fbox, с разницей лишь в том, что к нижней и правой сторонам рамки добавляется «тень».

Внешний вид этой рамки контролируется тремя параметрами (значения по умолчанию указаны в скобках):

\sboxrule толщина линий рамки (0.4pt);
\sboxsep расстояние между рамкой и текстом (10pt);
\sdim ширина тени (4pt).

Стандартная рамка с текстом в рамке, а затем рамка с тенью с текстом в рамке с тенью.

Целый абзац можно выделить, помещая его в parbox, вложенный в shabox.

```
Стандартная рамка
\fbox{с текстом в рамке},
а затем рамка с тенью
\shabox{с текстом в рамке с тенью}.
\par\bigskip
\setlength{\sdim}{3\fboxsep}
\shabox{\parbox{6cm}{%
Целый абзац можно выделить,
помещая его в parbox, вложенный
в \texttt{shabox}.}}
```

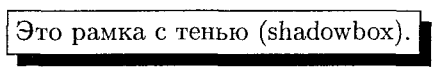
10.1.3 Декоративные рамки

Тимоти Ван Зандт создал пакет fancybox для использования внутри своего пакета seminar для изготовления слайдов. В нем вводится несколько новых команд для заключения в рамки информации в LaTeX'e. В этом разделе мы опишем только некоторые из наиболее важных команд. Более подробную информацию можно найти в документации, сопровождающей упомянутый выше пакет seminar.

Разновидности команды \fbox

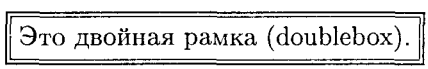
В пакете fancybox определяется четыре варианта команды \fbox. Как и в случае команды \fbox, расстояние между текстом и рамкой задается параметром длины \fboxsep (по умолчанию 3 pt). Другие параметры, регулирующие вид рамок, описываются ниже.

\shadowbox



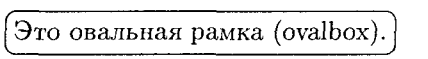
Толщина линий рамки задается параметром \fboxrule (тем же самым, что и для \fbox). Ширина тени задается при помощи \shadowsize (по умолчанию: 4 pt).

\doublebox



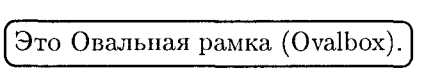
Толщина внутренней и внешней рамок соответственно равна .75\fboxrule и 1.5\fboxrule. Расстояние между рамками равно 1.5\fboxrule plus 0.5pt.

\ovalbox



Толщина рамки определяется командой \thinlines. Диаметр угловых дуг задается командой \cornersize. Команда \cornersize{*num*} устанавливает диаметр в значение *num* × минимум(ширина рамки, высота рамки); команда \cornersize*{*len*} полагает значение диаметра равным *len*. По умолчанию \cornersize{0.5}.

\Ovalbox



То же самое, что и \ovalbox, но толщина линий задается командой \thicklines.

Как определять окружения, рисующие рамки

Чтобы помочь вам при определении окружений, рисующих рамки, в пакете fancybox предусмотрено окружение Sbox, аналогичное команде \sbox. Оно сохраняет содержимое окружения в бункере памяти (storage bin), а команда \TheSbox извлекает это содержимое назад. Используя последнюю команду, вы можете перевоплотить окружение boxedminipage, о котором шла речь в разд. 10.1.1.

Пример министраницы в рамке, определенной посредством окружения Sbox.

```
\newenvironment{Boxedminipage}%
 {\begin{Sbox}\begin{minipage}}%
 {\end{minipage}\end{Sbox}\fbox{\TheSbox}}
\begin{Boxedminipage}{5cm}
 Пример министраницы в рамке, определенной
 посредством окружения \Lmenv{Sbox}.
\end{Boxedminipage}
```

В пакете fancybox предопределены следующие окружения, рисующие рамки:

- `Bcenter`, `Bflushleft` и `Bflushright` порождают соответственно заключенные в рамки окружения `center`, `flushleft` и `flushright`.
- `Bitemize`, `Benumerate` и `Bdescription` порождают соответственно заключенные в рамки окружения `itemize`, `enumerate` и `description`.
- `Beqnarray` дает окружение, заключенное в рамку, которое подобно `eqnarray`, но номера формул всегда ставятся справа. `Beqnarray*` аналогично `eqnarray*`, но размер рамки таков, чтобы только-только охватить все формулы.

Во всех этих командах вы сами можете добавлять рамку. Пример приводится ниже.

$$\begin{aligned} y &= x^2 \\ a^2+2ab+b^2 &= (a+b)^2 \\ \int_0^\infty e^{-ax}dx &= \frac{1}{a} \end{aligned}$$

```
\fbox{\begin{Beqnarray*}
                        y &=& x^2       \\
          a^2 + 2ab + b^2 &=& (a + b)^2 \\
\int_0^\infty e^{-ax} dx  &=& \frac{1}{a}
      \end{Beqnarray*}}
  \par\bigskip
\fbox{\begin{Beqnarray}
                        y &=& x^2       \\
          a^2 + 2ab + b^2 &=& (a + b)^2 \\
\int_0^\infty e^{-ax} dx  &=& \frac{1}{a}
      \end{Beqnarray}}
```

$$\begin{aligned} y &= x^2 && (10.1)\\ a^2+2ab+b^2 &= (a+b)^2 && (10.2)\\ \int_0^\infty e^{-ax}dx &= \frac{1}{a} && (10.3) \end{aligned}$$

В рассматриваемом пакете определяются также команды, рисующие рамку вокруг всей страницы, и обновляются некоторые команды для буквального воспроизведения текстов (см. также разд. 3.3). В частности, для буквального воспроизведения текста в рамке новое окружение `FrameVerb` может быть определено и использовано следующим образом:

```
\newenvironment{FramedVerb}%
 {\VerbatimEnvironment
   \begin{Sbox}\begin{minipage}{60mm}\begin{Verbatim}}%
 {\end{Verbatim}\end{minipage}\end{Sbox}
  \setlength{\fboxsep}{3mm}\fbox{\TheSbox}}
```

```
\newcommand{\Com}[1]{#1^a_b}
```

```
\begin{FramedVerb}
\newcommand{\Com}[1]{#1^a_b}
\end{FramedVerb}
```

10.2 Окружение picture

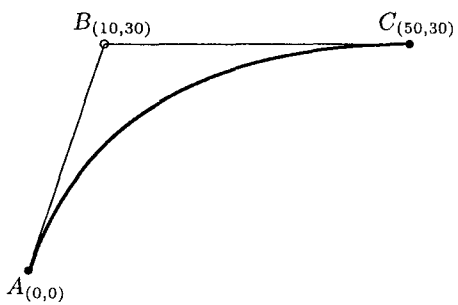
Исходное LaTeX'овское окружение picture [$\mathcal{L}$ 101–11], [$\mathcal{A}$172–81] может быть использовано во многих случаях для получения простой графики на выходе, и в этой книге многие линии нарисованы так же. В настоящем разделе обсуждаются различные пакеты, которые были созданы для того, чтобы расширить возможности окружения picture.

10.2.1 Аппроксимации Безье

LaTeX2$_\varepsilon$ позволяет строить достаточно сложные математические кривые, используя технику аппроксимаций сплайнами Безье. Отметим, что язык PostScript также использует кривые Безье (третьего порядка, т. е. кубические) как фундамент для своих действий при рисовании кривых.

`\qbezier[`*N*`](`*AX*`,`*AY*`)(`*BX*`,`*BY*`)(`*CX*`,`*CY*`)`

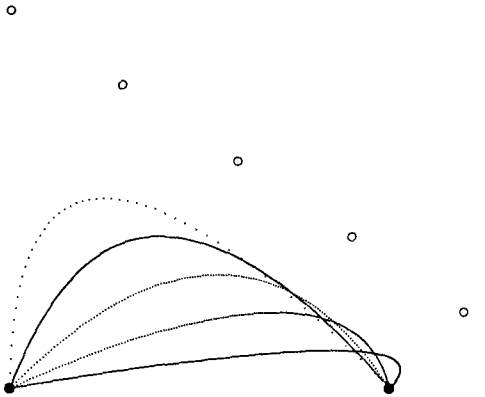
Эта команда определяет квадратичную кривую Безье с концами в точках (*AX*,*AY*) и (*CX*,*CY*) и контрольной точкой (*BX*,*BY*). Необязательный параметр *N*, если он указан, обозначает, что для аппроксимации кривой требуется поставить $N+1$ точку[3]. В приводимом ниже примере *A* и *C* — это концы кривой, а *B* — контрольная точка; количество рисуемых точек вычисляется автоматически.




```
\setlength{\unitlength}{1mm}
\begin{picture}(50,30)(-10,10)
  \linethickness{1pt}
  \qbezier(0,0)(10,30)(50,30)
  \thinlines
  \put(0,0){\line(1,3){10}}
  \put(50,30){\line(-1,0){40}}
  \put(0,0){\circle*{1}}
  \put(0,-1){
   \makebox(0,0)[t]{$A_{(0,0)}$}}
  \put(10,30){\circle{1}}
  \put(10,31){
   \makebox(0,0)[b]{$B_{(10,30)}$}}
  \put(50,30){\circle*{1}}
  \put(50,31){
   \makebox(0,0)[b]{$C_{(50,30)}$}}
\end{picture}
```

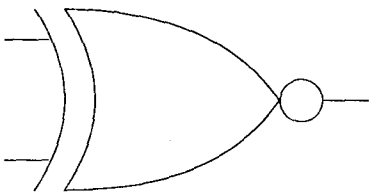
[3] В LaTeX'e 2.09 пакет Лесли Лэмпорта bezier определяет команду \bezier при помощи следующего соотношения: \bezier{N}(AX,AY)(BX,BY)(CX,CY)=\qbezier[N](AX,AY)(BX,BY)(CX,CY). Основное отличие состоит в том, что число точек, необходимых для создания однородной непрерывной кривой, при помощи \qbezier вычисляется автоматически, а значит, может больше не указываться.

Изменение числа точек на кривой и контрольной точки приводит к очевидному результату. В следующем примере две кривые нарисованы, исходя из числа точек по умолчанию, а для остальных число используемых точек указывается.



```
\setlength{\unitlength}{.5mm}
\begin{picture}(120,100)(-5,0)
 \linethickness{.5pt}
 \qbezier [50](0,0)(0,100)(100,0)
 \qbezier     (0,0)(30,80)(100,0)
 \qbezier[150](0,0)(60,60)(100,0)
 \qbezier[200](0,0)(90,40)(100,0)
 \qbezier     (0,0)(120,20)(100,0)
% mark the end points
 \put(0,0){\circle*{3}}
 \put(100,0){\circle*{3}}
% mark the control points
 \multiput(0,100)(30,-20){5}%
    {\circle{2}}
\end{picture}
```

Наконец, вот как вводится одна из диаграмм на рис. 10.5 (см. с. 335):



```
\setlength{\unitlength}{4mm}
\begin{picture}(12,10)(-2,0)
 \linethickness{0.4pt}
 \qbezier(2.00,6.00)(7.00,6.00)(9.00,3.00)
 \qbezier(2.00,0.00)(7.00,0.00)(9.00,3.00)
 \qbezier(2.00,6.00)(4.00,3.00)(2.00,0.00)
 \qbezier(1.00,6.00)(3.00,3.00)(1.00,0.00)
 \put(9.75,3.00){\circle{1.50}}
 \put(10.50,3.00){\line(1,0){1.50}}
 \put(0.00,5.00){\line(1,0){1.50}}
 \put(0.00,1.00){\line(1,0){1.50}}
\end{picture}
```

10.2.2 Как совмещать несколько рамок

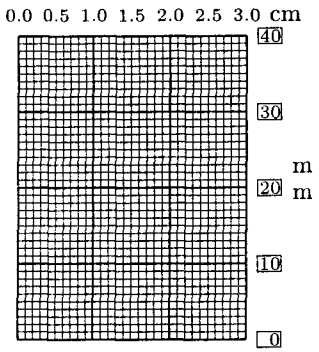
Пакет multibox (Брайана Гамильтона Келли) определяет следующие две команды для использования внутри окружения picture:

\multimake(*x,y*)(*dx,dy*){*n*}(*w,h*)[*pos*]{*text*$_1$}{*text*$_2$}...{*text*$_n$}
\multiframe(*x,y*)(*dx,dy*){*n*}(*w,h*)[*pos*]{*text*$_1$}{*text*$_2$}...{*text*$_n$}

Эти команды располагают n текстов, начиная с $text_1$ до $text_n$, соответственно внутри \makebox или \framebox. Нижний левый угол первой рамки находится в точке (x,y), а последующих рамок — в точке $(x+dx,y+dy)$, и так до точки $(x+(n-1)dx,y+(n-1)dy)$.

Каждая рамка имеет ширину и высоту, определяемые параметрами (w,h), а необязательный параметр *pos*, служащий для расположения рамки, применя-

ется ко всем генерируемым текстам. Ниже показан простой пример, в котором синтаксис можно сравнить с синтаксисом LaTeX'овской команды \multiput:




```
\setlength{\unitlength}{1mm}
\begin{picture}(40,50)(-10,0)
\linethickness{0.25mm}\scriptsize
 \multiput(0,0)(10,0){4}{\line(0,1){40}}
 \multiput(0,0)(0,10){5}{\line(1,0){30}}
\linethickness{0.1mm}
 \multimake(-2,42)(5,0){8}(4,3)[b]{0.0}{0.5}%
    {1.0}{1.5}{2.0}{2.5}{3.0}{\small cm}
 \multiframe(31.5,-1)(0,10){5}(3,2)[r]%
                    {0}{10}{20}{30}{40}
 \small
 \put(36,18.5){\shortstack{m\\ m}}
 \multiput(1,0)(1,0){29}{\line(0,1){40}}
 \multiput(0,1)(0,1){39}{\line(1,0){30}}
\end{picture}
```

10.2.3 Как рисовать бинарные и тернарные деревья

Пакет trees (Питера Ванруза) основан исключительно на окружении picture. Другой пакет для рисования деревьев обсуждается в разд. 10.5.2. В нем определяются макро, позволяющие рисовать (бинарное или тернарное) дерево любого размера. Для каждой внутренней вершины вы должны указать только нисходящие вершины при помощи команд \branch (бинарная вершина) или \tbranch (тернарная вершина).

Диаграммы деревьев создаются внутри окружения picture. В пакете trees имеются следующие команды:

\branchlabels{*labela*}{*labelb*}{*labelc*}

Эта команда определяет пометку, которой будет отмечаться каждое ребро.

\root(*x-coord*, *y-coord*)␣*rootid*.

(*x-coord*, *y-coord*) — это абсолютные координаты корня, на рисунке идентифицируемого как *rootid*. Заметьте, что пара круглых скобок, пробел и точка являются обязательными.

\branch{*steepness*}{*text*}{*branchid*}:*childa*,*childb*.

Параметр *steepness* определяет наклон ребер; это — целое число между 0 и 3. Параметр *text* указывает тот текст, который будет написан над вершиной разветвления. *branchid* — это идентификатор данного ребра, а *childa* и *childb* — это идентификаторы двух непосредственных потомков. Двоеточие, запятая и завершающая точка — обязательны.

\tbranch{*steepness*}{*text*}{*branchid*}:*childa*,*childb*,*childc*.

Эта команда — такая же, как и предыдущая, но она позволяет определить три, а не два идентификатора потомков.

\leaf{*toptext*}{*sidetext*}{*leafid*}.

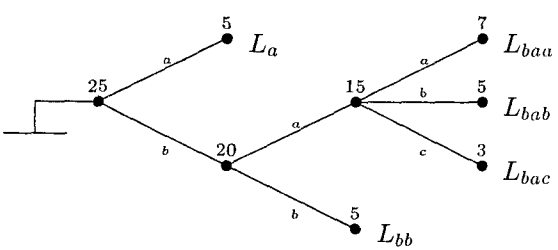
Параметры *toptext* и *sidetext* соответственно определяют текст, который будет написан над текущим листом и справа от текущего листа, имеющего идентификатор *leafid*.

Деревья строятся с пометками на ребрах (по умолчанию 0 и 1) и текстом (названием или значением) на вершинах.

Пример приводится ниже. (Внутренние) идентификаторы (0–7), используемые для пометки ветвей и листьев, могут быть заменены чем угодно.

Исходный текст

```
\setlength{\unitlength}{4pt}
\begin{picture}(50,30)
 \branchlabels abc          % по умолчанию 012
 \root(10,15) 0.            % пометка корня 0
        % вершина 0 имеет потомков 1 и 2
 \branch2{25} 0:1,2.
   \leaf{5}{$L_a$} 1.       % вершина 1 есть лист
   \branch2{20} 2:3,7.
        % ветвь к вершине 3 поднимается вверх она имеет пометку a
      \tbranch2{15} 3:4,5,6.
          \leaf{7}{$L_{baa}$}4.
          \leaf{5}{$L_{bab}$}5.
          \leaf{3}{$L_{bac}$}6.
   \leaf{5}{$L_{bb}$}7.
\end{picture}
```



Текст на выводе

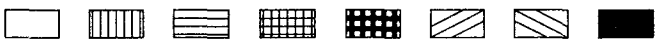
10.2.4 Как рисовать гистограммы

Пакет bar (Йоахима Блезера и Эдмунда Лэнга) может создавать гистограммы и полностью базируется на окружении `picture`. Диаграммы в виде столбиков (гистограммы) получаются при помощи окружения `barenv`.

Внутри окружения `barenv` возможны следующие команды:

`\bar{`*height*`}{`*hatch-index*`}[`*description*`]`

Столбик (bar) определяется указанием его высоты *height* и индекса штриховки *hatch-index* — цифры между 1 и 8 согласно следующей схеме:



1 2 3 4 5 6 7 8

Необязательный аргумент *description* — это текстовое описание столбика; оно набирается в соответствии с установками параметров `\setnumberpos` и `\setstyle`.

`\hlineon`

Эта команда рисует горизонтальные отрезки на заднем плане.

`\legend{`*hatch-index*`}{`*legend text*`}`

Эта команда ставит в соответствие стилю штриховки *hatch-index* поясняющий ее текст *legend text*.

`\setdepth{`*number*`}`

Эта команда определяет глубину для 3-мерной гистограммы (*number* ≥ 10).

`\sethspace{`*fraction*`}`

Эта команда определяет горизонтальный пробел, который нужно оставить между столбиками. Посредством параметра *fraction* он выражается как доля ширины столбика.

`\setlinestyle{`*style*`}`

Эта команда указывает стиль горизонтальных отрезков на заднем плане (активируемых командой `\hlineon`). Возможные значения стиля *style* — это `solid` (*сплошные линии*) и `dotted` (*линии, состоящие из точек*).

`\setnumberpos{`*position*`}`

Эта команда определяет положение надписей к столбикам. Возможные значения параметра *position* таковы:

`empty` Надписи не нужны.
`axis` Надписи приводятся под или над осью x.
`down` Надписи располагаются под столбиками.

`inside` Надписи приводятся внутри столбиков.
`outside` Надписи располагаются вне столбиков.
`up` Надписи приводятся над столбиками.

`\setprecision{`*digits*`}`

Определяет число разрядов *digits*, которые следует напечатать после десятичного знака.

`\setstretch{`*factor*`}`

Определяет масштабный множитель *factor* для размера диаграммы по вертикали.

`\setstyle{`*fontstyle*`}`

Определяет характеристику шрифта.

`\setwidth{`*number*`}`

Указывает ширину столбиков в пунктах (pt).

`\setxaxis{`*origin*`}{`*end*`}{`*step*`}`

Эта команда определяет разбиение оси x. Указанные три параметра обозначают начальное (*origin*) и конечное (*end*) значения и значения шага (*step*).

`\setxname{`*x-label*`}`

Определяет надпись *x-label* к оси x.

`\setxvaluetyp{`*type*`}`

Эта команда указывает надписи на отрезках оси x. По умолчанию используются числа, но вы можете использовать названия дней недели или месяцев. Таким образом, *type* в приведенном определении — это или `month` (месяц), или `day` (день). В этом случае начальное и конечное значения на оси x, указанные в команде `\setxaxis`, опираются на соответствие: 1 — январь, 2 — февраль и т. д., 12 — декабрь, 13 — снова январь и т. д. Для дней недели соответствие такое: 1 — понедельник, 2 — вторник и т. д., 7 — воскресенье, 8 — снова понедельник и т. д. (Однако в настоящей реализации пакета `bar` поддерживается только немецкий язык.)

`\setyaxis[`*offset*`]{`*origin*`}{`*end*`}{`*step*`}`

Эта команда определяет разбиение оси y. Три обязательных параметра — такие же, как для `\setxaxis`, а необязательный аргумент *offset* указывает значение,

которое добавляется к начальному и конечному значениям оси, при этом не изменяя разбиение оси: начальную точку, конечную точку и размер шага.

`\setyname{`*y-label*`}`

Определяет надпись *y-label* к оси y.

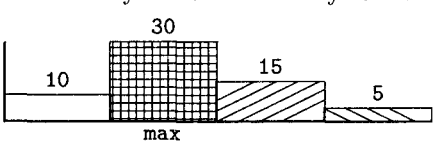
Общая структура последовательности команд, порождающих гистограмму, следующая:

```
\begin{barenv}
    Declarations (Установки)
       \bar{height}{hatch-index}
              .
              .
       \bar{height}{hatch-index}
\end{barenv}
    Legend (Текст, поясняющий индекс штриховки)
```

Отметим, что поскольку окружение `barenv` использует окружение `picture`, то все команды, которые можно применять в окружении `picture`, применимы и с окружением `barenv`.

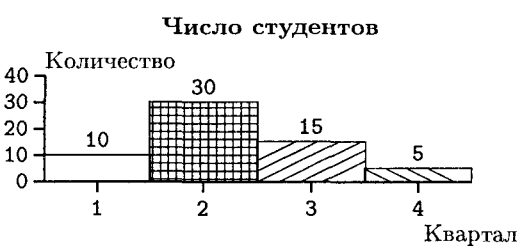
10.2.5 Примеры окружения `barenv`

Первый пример показывает простую гистограмму, порожденную окружением `barenv` с установками по умолчанию.



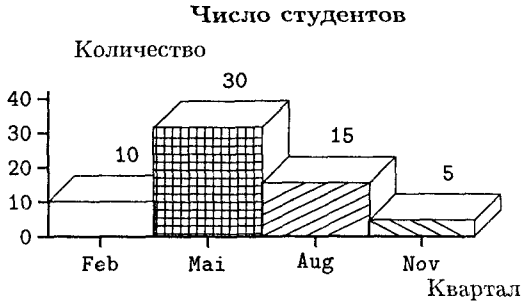

```
\begin{barenv}
\bar{10}{1}
\bar{30}{4}[\texttt{max}]
\bar{15}{6}
\bar{5}{7}
\end{barenv}
```

Надписи к осям и заголовок можно легко добавить:



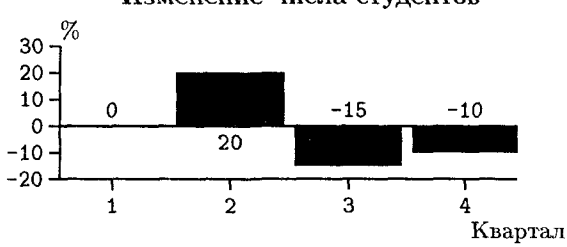

```
\begin{center}
\textbf{Число студентов}\\[5mm]
\end{center}
\begin{barenv}
\setxaxis{1}{4}{1}
   \setxname{Квартал}
\setyaxis{0}{40}{10}
   \setyname{Количество}
\bar{10}{1}   \bar{30}{4}
\bar{15}{6}   \bar{5}{7}
\end{barenv}
```

Возможно, мы предпочитаем видеть информацию в 3-мерном виде, и для того чтобы сделать различия более наглядными, мы растягиваем ось y. Чтобы информацию о кварталах сделать более точной, мы указываем средний месяц каждого трехмесячного периода.




```
\begin{center}
\textbf{Число студентов}\\[5mm]
\end{center}
\begin{barenv}
\setdepth{10}% 3-мерный эффект
\setstretch{1.4}% растянуть
                % измерение y
\setnumberpos{up}% числа
                  % над столбиками
\setxvaluetyp{month}% месяцы
                      % вдоль оси x
\setxaxis{2}{12}{3}
   \setxname{Квартал}
\setyaxis{0}{40}{10}
   \setyname{Количество}
\bar{10}{1} \bar{30}{4}
\bar{15}{6} \bar{5}{7}
\end{barenv}
```

Если мы хотим акцентировать внимание на изменениях числа студентов в кварталах, то мы можем использовать следующее изображение:

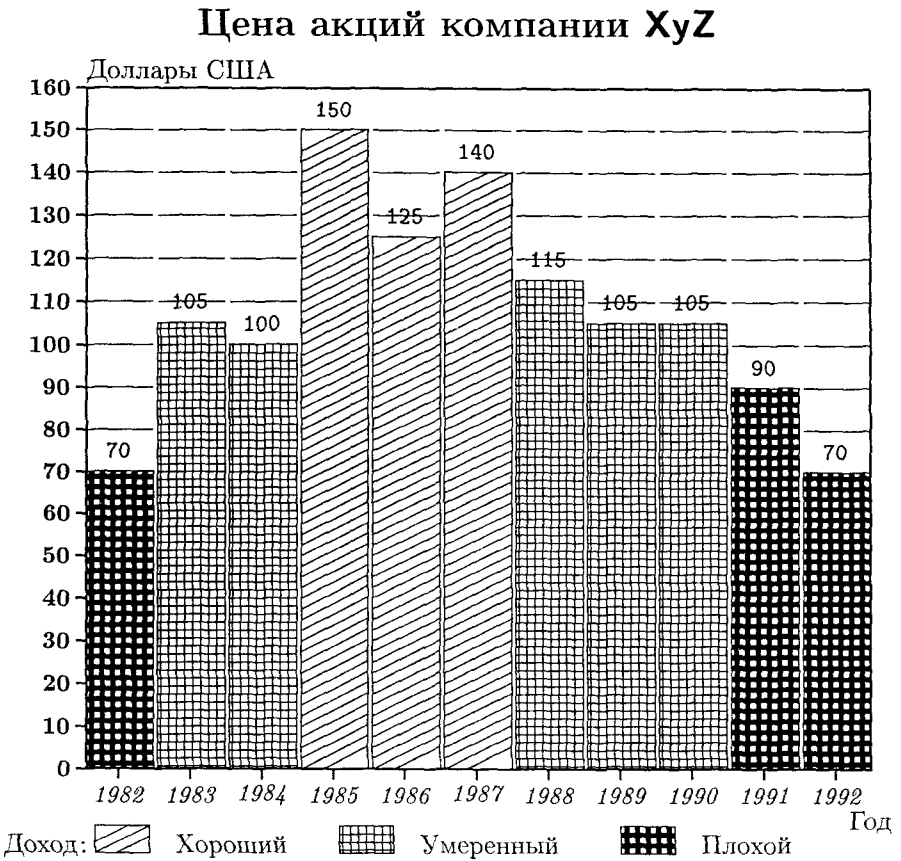



```
\begin{center}
\textbf{Изменение
        числа студентов}\\[5mm]
\end{center}
\begin{barenv}
\setxaxis{1}{4}{1}
   \setxname{Квартал}
\setyaxis{-20}{30}{10}
   \setyname{\%}
\sethspace{0.1}\setnumberpos{axis}
\bar{0}{1}    \bar{20}{8}
\bar{-15}{8}  \bar{-10}{8}
\end{barenv}
```

Наш последний и наиболее сложный пример показывает двумя способами изменение цены акций компании XyZ. Сначала это иллюстрируется плоской гистограммой (рис. 10.1), а затем — в виде объемной гистограммы (рис. 10.2).

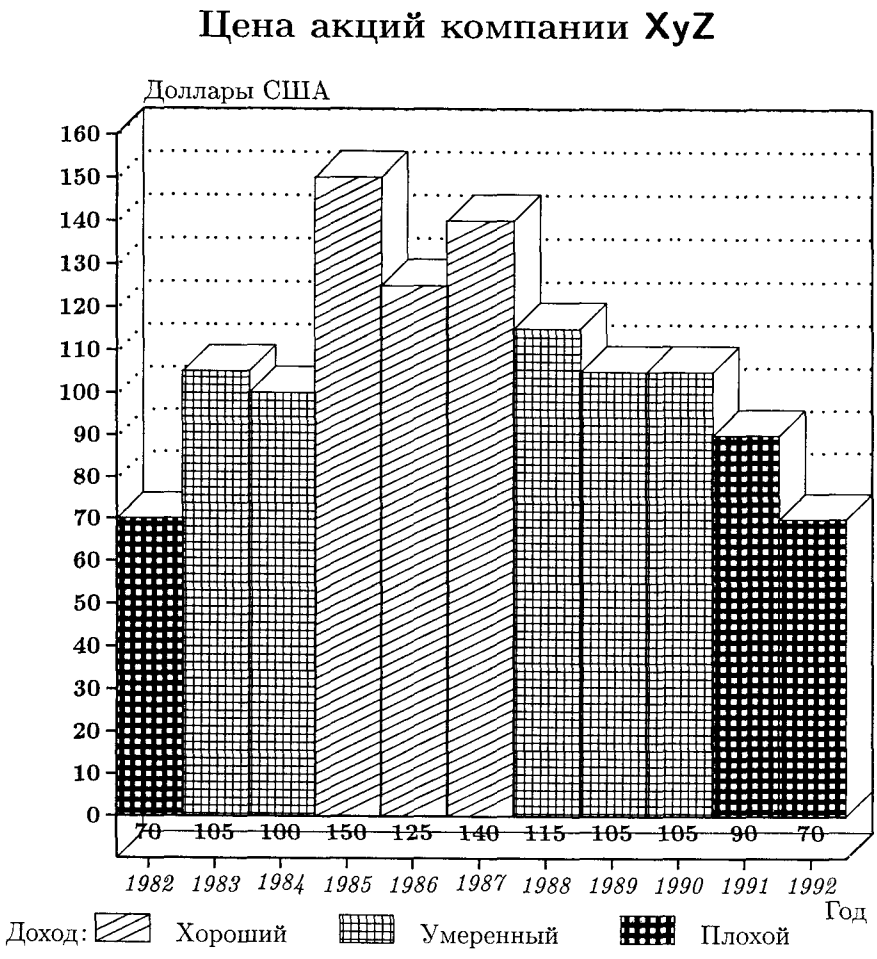
10.2.6 Как рисовать произвольные кривые

Пакет curves (И. Л. Маклэйна-кросса) расширяет возможности LaTeX'овского окружения picture, позволяя рисовать кривые, которые состоят из небольших наложенных друг на друга кругов, а размеры и число этих кругов подбираются так, чтобы получить наилучшую визуальную плавность кривой. Специальный подпакет curvesls заново реализует некоторые из наиболее поглощающих время и место частей кода команд \special, определенных в программе Эберхарда Маттеса emTeX для компьютеров, совместимых с IBM PC.




```
\begin{center}
  \Large\bfseries
  Цена акций компании
              \textsf{XyZ}\\[6mm]
  \begin{barenv}
    \setstretch{1.6}
    \setwidth{25}
    \sethspace{0.1}
    \setstyle{\bfseries}
    \setxname{Год}
    \setyname{Доллары США}
    \setstyle{\small\itshape}
    \setxaxis{1982}{1992}{1}
    \setstyle{\small\bfseries}
    \setyaxis{0}{160}{10}
    \setlinestyle{solid}
    \hlineon
    \setnumberpos{up}
    \setstyle{\small\ttfamily}
    \bar{70}{5}  \bar{105}{4}
    \bar{100}{4} \bar{150}{6}
    \bar{125}{6} \bar{140}{6}
    \bar{115}{4} \bar{105}{4}
    \bar{105}{4} \bar{90}{5}
    \bar{70}{5}
  \end{barenv}
\end{center}
\par\vspace{\baselineskip}
Доход: \legend6{Хороший}      \qquad
       \legend4{Умеренный} \qquad
       \legend5{Плохой}
```

Рис. 10.1. Пример гистограммы — плоский вариант




```
\begin{center}
  \textbf{\Large Цена акций компании
          \textsf{XyZ}}\\[5mm]
  \begin{barenv}
    \setstretch{1.6}
    \setnumberpos{down}
    \setwidth{25}\setdepth{10}
    \setstyle{\small\bfseries}
    \setxname{Год}
    \setyname{Доллары США}
    \setstyle{\small\itshape}
    \setxaxis{1982}{1992}{1}
    \setstyle{\small\bfseries}
    \setyaxis[10]{0}{160}{10}
    \setlinestyle{dotted}
    \hlineon
    \setnumberpos{axis}
    \bar{70}{5}  \bar{105}{4}
    \bar{100}{4} \bar{150}{6}
    \bar{125}{6} \bar{140}{6}
    \bar{115}{4} \bar{105}{4}
    \bar{105}{4} \bar{90}{5}
    \bar{70}{5}
  \end{barenv}
\end{center}
\par\vspace{\baselineskip}
Доход: \legend6{Хороший}      \qquad
       \legend4{Умеренный} \qquad
       \legend5{Плохой}
```

Рис. 10.2. Пример гистограммы — объемный вариант

Возможности этого пакета включают в себя:

- совместимую замену для пакета bezier;
- регулировку толщины кривой между 0.5 и 15 pt (т. е. 0.17 и 5.2 mm);
- вычерчивание кривых с непрерывно меняющимся тангенсом угла наклона;
- управление углом наклона в конечных точках при помощи `\tagcurve`;
- вычерчивание замкнутых кривых с непрерывно меняющимся тангенсом угла наклона с использованием команды `\closecurve`;
- вычерчивание больших окружностей посредством `\bigcircle` и дуг окружностей посредством команды `\arc`;
- независимое масштабирование абсциссы кривой и ординат для того, чтобы вычерчивать кривую по точкам;
- аффинное масштабирование с целью придания дугам и окружностям вида кусочков эллипса;
- нанесение символов и вычерчивание пунктирных линий.

Команды для вычерчивания кривых

В этом разделе рассматриваются наиболее важные команды для вычерчивания кривых, которые используются в приводимых ниже примерах. Нарисованные кривые состоят из дуг парабол, соединяющих точки с указанными координатами и имеющих в каждой точке касательную параллельную прямой, проходящей через прилежащие точки. Дуги в конечных точках кривой представляют собой параболу, проходящую через три точки, указанные последними.

`\arc[`*nbsymb*`](`x_1, y_1`){`*angle*`}`

Эта команда рисует дугу окружности с центром на текущей позиции, начиная с точки (x_1, y_1) и поворачиваясь против часовой стрелки на угол *angle* градусов. Необязательный аргумент *nbsymb* указывает число узоров или символов, которые следует нарисовать (по умолчанию — 0).

`\bigcircle[`*nbsymb*`]{`*diameter*`}`

Эта команда рисует окружность с диаметром, равным параметру *diameter*, умноженному на `\unitlength`. Необязательный аргумент *nbsymb* — такой же, что и описанный выше.

`\closecurve[`*nbsymb*`](`$x_1, y_1, x_2, y_2 ... x_n, y_n$`)`

Эта команда рисует замкнутую кривую с непрерывно меняющимися касательными во всех точках. Требуется указать по крайней мере шесть координат. Необязательный аргумент *nbsymb* — такой же, как и выше.

`\curve[`*nbsymb*`](`$x_1, y_1, x_2, y_2 ... x_n, y_n$`)`

Эта команда рисует кривую, проходящую через точки с указанными координатами. Две пары координат порождают отрезок прямой; три пары — дугу па-

раболы, проходящую через соответствующие точки. Необязательный аргумент *nbsymb* — такой же, как выше.

`\scaleput`(x_1, y_1)`{`*pict-object*`}`

Эта команда помещает рисунок *pict-object* в положение (x_1, y_1). Одновременно она производит аксонометрическую проекцию и поворот, определяемый масштабными множителями `\xscale`, `\xscaley`, `\yscale` и `\yscalex`, которые имеют начальные значения соответственно 1.0, 0.0, 1.0 и 0.0.

`\tagcurve[`*nbsymb*`]`$(x_1, y_1, x_2, y_2 ... x_n, y_n)$

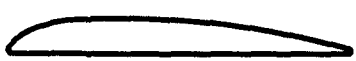
Эта команда рисует кривую без первой и последней дуг. Если указаны только три пары координат, то она рисует только последнюю дугу. Необязательный аргумент *nbsymb* — такой же, как и выше.

Примеры



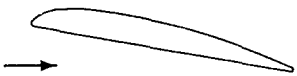
```
\tracingmacros=1
\setlength{\unitlength}{0.36pt}
\linethickness{0.7mm}
\begin{picture}(400,110)(-10,0)
  \tagcurve(80,0, 0,0, 40,100, 80,0, 0,0)
  \closecurve(150,0, 190,100, 230,0)
  \curve(300,0, 340,100, 380,0)
\end{picture}
```

Следующий пример показывает профиль крыла, который часто используется в исследованиях по аэродинамике. Команды `\arc` рисуют участки окружностей, соответствующие передней и задней кромкам крыла. Координаты, указанные в первой команде `\curve`, взяты из аэродинамических таблиц и соответствуют верхнему участку крыла; вторая команда `\curve`, содержащая две пары координат, чертит горизонтальную хорду в нижней части.



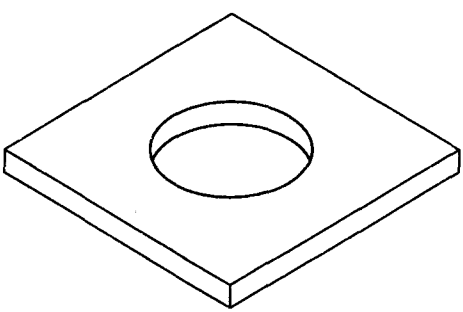
```
\newcommand{\RAFsixE}{%
 \scaleput(1.25,1.25){\arc(0,-1.25){-135}}
 \scaleput(0,0){\curve(0.36,2.13,
   1.25,3.19,2.5,4.42, 5.0,6.10, 7.5,7.24,
  10,8.09  ,15,9.28 ,20,9.90 ,30,10.3  ,
  40,10.22 ,50,9.80 ,60,8.98 ,70,7.70  ,
  80,5.91  ,90,3.79 ,95,2.58 ,99.24,1.52)}
 \scaleput(99.24,0.76){\arc(0,-0.76){180}}
 \scaleput(0,0){\curve(1.25,0, 99.24,0)}  }
\begin{center}
 \begin{picture}(100,20)
   \RAFsixE
 \end{picture}
\end{center}
```

Теперь мы можем повернуть этот профиль, используя масштабные параметры, описанные с командой \scaleput. В приводимом ниже примере мы выбираем поворот по часовой стрелке на угол 10°, так что диагональные элементы \xscale и \yscale соответствуют $\cos(10°) \approx 0.9848$, а недиагональные элементы соответствуют $\pm\sin(10°) \approx \pm 0.1736$.



```
\begin{picture}(120,50)(0,0)
    \renewcommand{\xscale}{0.9848}
    \renewcommand{\xscaley}{0.1736}
    \renewcommand{\yscale}{0.9848}
    \renewcommand{\yscalex}{-0.1736}
    \put(20,30){\RAFsixE}
    \thicklines
    \put(10,15){\vector(1,0){20}}
\end{picture}
```

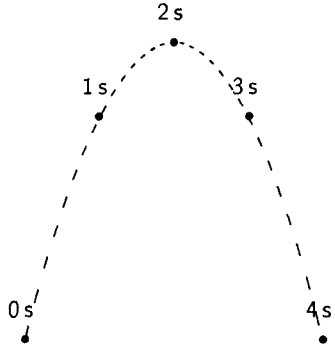
Аксонометрическая проекция является другим применением масштабирования. Окружности превращаются в эллипсы, а дуги окружностей становятся дугами эллипсов. Отметим, что в приводимом примере углы в команде \arc были найдены методом проб и ошибок.



```
\setlength{\unitlength}{1.5mm}
\begin{picture}(40,30)
    \thicklines
    \multiput(20,5)(20,12){2}{%
       \line(0,-1){2}\line(-5,3){20}}
    \multiput(20,5)(-20,12){2}{%
       \line(5,3){20}}
    \put(20,3){\line(5,3){20}}
    \put(20,3){\line(-5,3){20}}
    \put(0,15){\line(0,1){2}}
    \linethickness{1pt}
    \put(20,5){%
      \renewcommand{\xscale}{1}
      \renewcommand{\xscaley}{-1}
      \renewcommand{\yscale}{0.6}
      \renewcommand{\yscalex}{0.6}
      \scaleput(10,10){\bigcircle{10}}
      \put(0,-2){%
        \scaleput(10,10){\arc(5,0){121}}
        \scaleput(10,10){\arc(5,0){-31}}}
                }
\end{picture}
```

Добавить символы к вашим кривым довольно легко. Команда \curvesymbol определяет, какой символ будет использоваться. Если значение необязательного параметра *nbsymb* в одной из команд, рисующих кривые, отрицательно, то он фиксирует число символов для каждой дуги кривой. TeX'овский примитив \phantom используется для того, чтобы центрировать кружки диаметром 1 mm.

Следующий пример показывает траекторию (высота–дальность) объекта, брошенного в воздух, как функцию времени.




```
\thinlines
\setlength{\unitlength}{1mm}
\begin{picture}(80,46)(7,0)
% между черточками в 1mm вставлены пробелы в 2mm
 \curvedashes[1mm]{0,1,2}
 \put(10,0){\curve(0,0, 19.6,39.2, 39.2,0)}
 \curvedashes{}% восстановить черточки по умолчанию
 \newcounter{time}
 \curvesymbol{\textsf{\thetime\,s}
              \addtocounter{time}{1}}
 \put(10,4){\curve[-2](0,0, 19.6,39.2, 39.2,0)}
 \curvesymbol{\phantom{\circle*{1}}\circle*{1}}
 \put(10,0){\curve[-2](0,0, 19.6,39.2, 39.2,0)}
\end{picture}
```

10.2.7 Другие пакеты

В теоретической физике фейнмановский пакет Майкла Левина [43] стал весьма популярным. При помощи командного языка высокого уровня, основанного на командах окружения `picture`, он позволяет пользователю рисовать траектории движения различных частиц и группы вершин диаграмм Фейнмана. Рисунок 10.3 является типичным рисунком, получаемым при помощи этого пакета.

В области органической химии набор макро был создан Росвитой Хос и Кевином О'Кейном [84]. Их система ChemTEX позволяет рисовать сложные молекулы (см. рис. 10.4).

Набор символов, обозначающих элементы электронной схемы, был создан Адрианом Джонстоуном. Он имеется на CTAN как часть его системы `lcircuit`, которая используется с программой TEXcad (см. рис. 10.5). В этот набор входят все основные логические элементы в четырех ориентациях, полевые транзисторы, выводы источников питания, буферные элементы, конденсаторы, резисторы и монтажные Т-соединения. Указанный набор использует пакет bezier, который обсуждался ранее.

10.3 epic — усовершенствования к окружению `picture`

Пакет epic [61] (Сунила Подара) совершенствует графические возможности LATEX'а и предоставляет мощный, высокого уровня пользовательский интерфейс для окружения `picture`. Основная его цель — сократить количество вычислений вручную, требуемых для указания расположения *объектов*. Дополнительные команды пакета epic позволяют затрачивать меньше усилий на создание сложных рисунков, чем это было раньше.

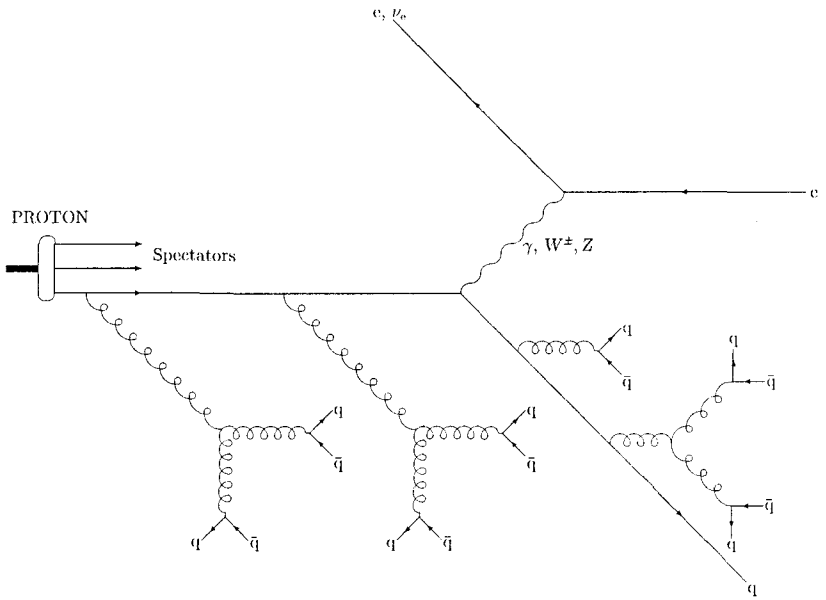


Рис. 10.3. Пример рисунка, созданного при помощи фейнмановского пакета

Большинство команд для вычерчивания (картинок) требуют явного указания координат для каждого *объекта*. Однако команды высокого уровня помогают сократить количество координат, которые нужно вычислить вручную. По существу для создания таких команд можно рассматривать два подхода:

- Набор объектов отбирается таким образом, что каждый элемент этого набора может быть нарисован при помощи указания одной или двух пар координат — команда `\shortstack` попадает в эту категорию.
- Заготавливаются команды, которые внутренне произведут большую часть вычислений и потребуют указания только простых пар координат — команда `\multiput` является примером такого подхода.

Очевидное преимущество в наличии команд, которые попадают в указанные выше категории, состоит в том, что их не только легче применять, но и что любое последующее изменение в расположении объекта требует минимального пересчета.

Часто используемый примитив `\line` имеет серьезные ограничения и недостатки. Его аргументы совсем не очевидны и требуют больших вычислений — зачастую процесс написания команды `\line` включает в себя:

1. вычисление координат двух точек на концах;
2. вычисление расстояния по горизонтали и по вертикали;

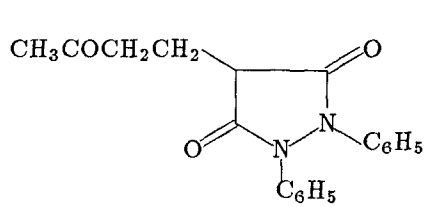


Рис. 10.4. Пример химической формулы

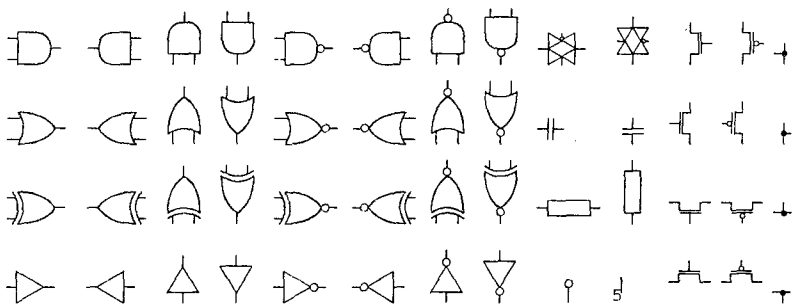


Рис. 10.5. Символы элементов электронной схемы, подготовленные при помощи команд окружения `picture`

3. запись сказанного выше в виде пары (x,y), определяющей угол наклона, и расстояния по горизонтали, определяющего длину отрезка;
4. определение наличия необходимого угла наклона и, если такого нет, повторение шагов 1—3 до нахождения удовлетворительного угла наклона.

Такой механизм рисования очень громоздок. Более того, в силу способа реализации команды `\line` длина возможного наикратчайшего отрезка при разных углах наклона неодинакова. В пакете epic вводятся команды для вычерчивания отрезков, которые обходят эти недостатки и в то же самое время имеют более простой синтаксис. Эти команды используют только координаты концов, устраняя тем самым шаги, связанные с указанием других данных об отрезке. На базе этого определяется также несколько новых команд высокого уровня. Поэтому пакет epic позволяет создавать сложные рисунки с меньшими усилиями, чем прежде.

10.3.1 Описание команд

`\multiputlist`$(x,y)(\Delta x,\Delta y)$`[`*pos*`]{`*item1*`,`*item2*`,`*item3*`,...,`*itemN*`}`

Эта команда представляет собой разновидность LaTeX'овской команды `\multiput`, которая позволяет помещать *один и тот же* объект в равномерно отстоящие точки. Команда `\multiputlist` аналогична, но объекты в ней могут

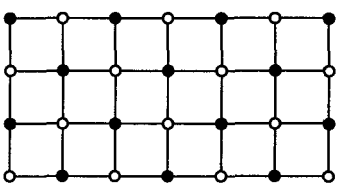
быть *разными*. При выполнении команды \multiputlist объекты, которые нужно поставить, выбираются из *списка объектов* (*item1*, ... ,*itemN*) в соответствии с приращением координат. (Первый объект *item1* ставится в первую позицию, второй объект *item2* — во вторую и т. д.) Например, вы можете надписать числа вдоль оси x на графике, если укажете команду:

```
\multiputlist(0,0)(10,0){1.00,1.25,1.50,1.75,2.00}
```

В сущности (виртуально) объекты в списке могут быть какими угодно, включая \makebox, \framebox или математические символы. Эта команда придает определенную регулярность и симметричность в расположение различных объектов на рисунке.

\matrixput(x,y)($\Delta x_1,\Delta y_1$){n_1}($\Delta x_2,\Delta y_2$){n_2}{*object*}

Команда \matrixput — это двумерный аналог LaTeX'овского примитива \multiput. Однако использовать команду \matrixput более эффективно, чем несколько команд \multiput. Эта команда особенно полезна для рисунков, в которых узор повторяется через равные интервалы в двух измерениях.



```
\setlength{\unitlength}{.7mm}
\begin{picture}(62,32)(-1,-1)
\thicklines
\matrixput(0,0)(10,0){7}(0,10){4}{\circle{2}}
\matrixput(10,0)(20,0){3}(0,20){2}{\circle*{2}}
\matrixput(0,10)(20,0){4}(0,20){2}{\circle*{2}}
\matrixput(1,0)(10,0){6}(0,10){4}{\line(1,0){8}}
\matrixput(0,1)(10,0){7}(0,10){3}{\line(0,1){8}}
\end{picture}
```

\grid(*width,height*)(Δ*width*, Δ*height*)[*initial-X-int,initial-Y-int*]

Команда \grid создает решетку размером *width* единиц шириной и *height* единиц высотой. Вертикальные линии рисуются с интервалами Δ*width*, а горизонтальные — с интервалами Δ*height*. Если указать третий (необязательный) аргумент, то края решетки будут помечены числами, начальные значения которых определяются целыми числами *initial-X-int* и *initial-Y-int* соответственно. Эти числа будут расположены вдоль осей на расстояниях Δ*width* и Δ*height* друг от друга.

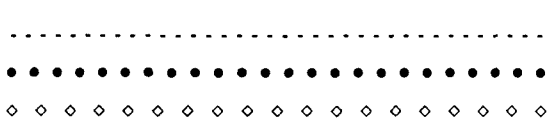
Команда \grid создает бокс. Поэтому его нужно поместить командой \put в точку с требуемыми координатами, например:

```
\put(0,0){\grid(50,100)(5,10)}
\put(0,0){\tiny \grid(100,100)(5,5)[-50,0]}% набрать числа в размере \tiny
```

Как рисовать отрезки разного вида

`\dottedline[`*dotchar*`]{`*dotgap*`}`$(x_1, y_1)(x_2, y_2)...(x_n, y_n)$

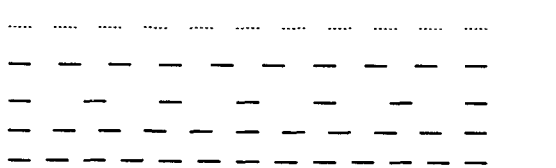
Команда `\dottedline` соединяет указанные точки, рисуя отрезок из точек между каждой парой из них. Должны быть определены по меньшей мере координаты двух точек. Линия из точек рисуется с промежутком между точками, который указывается во втором (первом обязательном) аргументе *dotgap* (в единицах, определяемых параметром `\unitlength`). Поскольку число изображаемых точек должно быть целым, промежутки между точками могут не получиться в точности такими, какие указаны. По умолчанию в виде точки используется маленький квадратик (`\picsquare`, описываемый ниже), что можно изменить с помощью необязательного аргумента, если выбрать другой символ *dotchar*. Когда используются точки по умолчанию, толщина точек контролируется текущими значениями параметров `\thinlines`, `\thicklines` или `\linethickness`. Отметим, что некоторые символы, такие, как «*» в романском шрифте, появляются на печати не по центру, хотя большинство других символов центрируется.



```
\begin{picture}(150,15)(0,0)
\thicklines
\dottedline{2}(0,10)(70,10)
\dottedline[$\bullet$]{3}(0,5)(70,5)
\dottedline[$\diamond$]{4}(0,0)(70,0)
\end{picture}
```

`\dashline[`*stretch*`]{`*dashlength*`}[`*dashdotgap*`]`$(x_1, y_1)(x_2, y_2)...(x_n, y_n)$

Команда `\dashline` соединяет указанные точки, рисуя пунктирный отрезок между каждой парой из них. По меньшей мере должны быть указаны координаты двух точек. Каждая черточка создается при помощи команды `\dottedline`. Обязательный параметр *dashlength* определяет длину черточки, а необязательный параметр *dashdotgap* задает промежуток между точками, из которых состоит черточка; оба параметра измеряются в единицах `\unitlength`. По умолчанию создается черточка, выглядящая сплошной.



```
\begin{picture}(70,22)(0,-2)
\dashline{3}[0.7](0,18)(63,18)
\thicklines
\dashline{3}(0,13)(63,13)
\dashline[-30]{3}(0,8)(63,8)
\dashline[+15]{3}(0,4)(63,4)
\dashline[+30]{3}(0,0)(63,0)
\end{picture}
```

В определении команды \dashline необязательный параметр *stretch* является целым числом между -100 и ∞. Он указывает процентное отношение, с которым число черточек «растянуто» или увеличено ($stretch > 0$), «стянуто» или уменьшено ($stretch < 0$). Если параметр *stretch* равен нулю, то используется минимальное число черточек, совместимое с приблизительно одинаковым размещением по отношению к пустому пространству между черточками. Идея процентного параметра *stretch* состоит в том, что если рисуется несколько пунктирных прямых с черточками разных длин, то все пунктирные прямые с одинаковыми значениями *stretch* будут внешне похожи. Значения по умолчанию процентного параметра *stretch* можно изменить командой \renewcommand, примененной к параметру \dashlinestretch:

```
\renewcommand{\dashlinestretch}{-50} % Разрешены только целые числа
```

Аргумент (-50 в примере) определяет процентное увеличение или уменьшение (в виде целого числа), которое будет применяться ко всем последующим командам \dashline, исключая те команды, в которых параметр *stretch* явно указан как первый необязательный аргумент.

\drawline[*stretch*]$(x_1, y_1)(x_2, y_2)...(x_n, y_n)$

Команда \drawline соединяет заданные точки отрезками прямой с наиболее близким углом наклона, имеющимся в шрифтах. Как минимум должны быть указаны координаты двух точек. Поскольку в шрифтах, отвечающих за прямолинейные отрезки, имеется только конечное число углов наклона, некоторые линии могут содержать дырки. В зависимости от установки параметров \thinlines или \thicklines, действующих в данный момент, команда \drawline может нарисовать тонкую или жирную линию; для таких линий это единственные два параметра, отвечающие за толщину линий. С точки зрения использования памяти и центрального процессора компьютера описываемая команда предоставляет самый эффективный путь для вычерчивания отрезков с произвольными углами наклона. Необязательный параметр *stretch* подобен такому же параметру, описанному с командой \dashline. Если *stretch* равен нулю, то это дает минимальное число черточек для того, чтобы линия выглядела сплошной, где каждая черточка «соединена» на концах. Если *stretch* больше нуля, то для построения линии используется большее число черточек, что придает линии менее зазубренный вид. Как и в случае параметра \dashlinestretch и команды \dashline, аналогичный параметр \drawlinestretch позволяет установить значение по умолчанию для процентного параметра *stretch* команды \drawline.

```
\renewcommand{\drawlinestretch}{20} % Разрешены только целые числа
```

Команды, рисующие несколько кривых

```
\jput(x,y){object}

\begin{dottedjoin}[dotchar]{dotgap}
.....        команды \jput, внутренне связанные с командой \dottedline
\end{dottedjoin}

\begin{dashjoin}[stretch]{dashlength}[dashdotgap]
.....        команды \jput, внутренне связанные с командой \dashline
\end{dashjoin}

\begin{drawjoin}[stretch]
.....        команды \jput, внутренне связанные с командой \drawline
\end{drawjoin}
```

Приведенные три окружения `dottedjoin`, `dashjoin` и `drawjoin` соответствуют трем командам для вычерчивания отрезков `\dottedline`, `\dashline` и `\drawline`. Их аргументы имеют тот же самый смысл, что и в соответствующих командах, а параметры `\dashlinestretch` и `\drawlinestretch` можно использовать для глобального переопределения параметра *stretch*. В этих окружениях применяется новая команда `\jput` (join and put, т. е. соединить и поместить), которая идентична обычной LaTeX'овской команде `\put` с той разницей, что она используется внутри этих трех окружений. Кроме того что все *объекты*, заключенные в любое из этих трех окружений с использованием команды `\jput`, будут нарисованы, они еще будут соединены линиями соответствующих им типов. Отметим, что пользователь сам решает, центрировать ли объекты в изображаемых точках.

Каждое из трех окружений типа `join` в отдельности определяет свою «кривую», поэтому любое множество точек, принадлежащее различным «кривым», должно быть заключено в свое окружение типа `join`. Первоначальная мотивация в создании окружений типа `join` состоит в том, чтобы иметь возможность вычерчивать чертежи с различным типом кривых и разнородных отрезков. Пример приведен на рис. 10.7.

`\picsquare`

Команда `\picsquare` создает маленькую квадратную точку, центр которой является ее точкой отсчета. Размер этого квадратика зависит от текущей установки команды `\thinlines`, `\thicklines` или `\linethickness`. Большинство команд пакета epic, рисующих маленькие точки, обращаются к этой команде, хотя первоначально она предназначалась для использования совместно с командой `\putfile`, описываемой ниже.

`\putfile{`*filename*`}{`*object*`}`

Команда `\putfile` аналогична LaTeX'овской команде `\put`, за исключением того, что координаты x и y, необходимые в команде `\put`, читаются из внешнего файла и в каждой из точек с этими координатами рисуется один и тот же объект *object*. Эта команда представлена потому, что TeX лишен возможности производить арифметические действия с плавающей точкой, которые необходимы, если вы хотите нарисовать параметрически заданную кривую, отличную от прямолинейного отрезка. Координаты точек на таких кривых могут быть легко получены при помощи программы на каком-либо языке программирования, а затем считаны TeX'ом. Внешний файл должен содержать координатные пары (x, y), по одной паре на каждой строке, и координаты в паре должны быть разделены пробелом. Знак % предназначается для комментария, но если комментарий находится на той же строке, что и данные, то после координаты y следует оставить по крайней мере один пробел, так как знак % скрывает символ начала новой строки.

Например, чтобы нарисовать гладкую кривую, проходящую через точки с заданными координатами, вы можете использовать следующую процедуру:

1. Создайте файл, содержащий координаты (x, y) данных точек, который вы, к примеру, назовете `plot.data`.
2. Если требуется, подладьте ваши данные.
3. В вашем LaTeX'овском файле внутри окружения `picture` наберите
   `\putfile{plot.data}{\picsquare}`

10.4 Расширение пакета epic

LaTeX предоставляет базовые, но ограниченные возможности рисования картинок, которые расширены за счет команд (чертящих сплошные отрезки, отрезки из точек, пунктирные) и новых окружений (пригодных для сложных рисунков), описанных в предыдущем разделе пакета epic.

Однако пакет epic все еще наследует многие LaTeX'овские ограничения при рисовании картинок. Как следствие, выполнение некоторых функций занимает много времени или рисунок на выходе получается не очень высокого качества.

Чтобы снабдить описание простых рисунков и графиков «естественным языком», несколько лет назад был создан язык программирования pic [24]. Такой gnu-препроцессор, как gpic, способен преобразовать его графические команды в результат, доступный средству форматирования troff системы UNIX. Более интересным для нас является то, что он может также генерировать TeX'овские команды `\special`, которые поддерживаются многими dvi-драйверами. Например, транслятор dvips из dvi-в-PostScript, описываемый в разд. 11.2, может интерпретировать эти команды.

Пакет eepic [41], написанный Конрадом Куоком, является расширением и LaTeX'а, и пакета epic; он снимает некоторые из ограничений в LaTeX'е, паке-

те epic и программе gpic, генерируя gpic'овские команды \special при помощи TEX'овских команд. Поскольку eepic содержит в себе epic как часть, вы можете применять этот пакет для обработки любого рисунка, использующего команды пакета epic, и получить результат, который выглядит лучше.

10.4.1 Расширения LATEX'а при помощи eepic

Для вычерчивания прямых и окружностей в LATEX'е используются специальные шрифты, поэтому имеется ограниченный набор возможностей. Функции пакета eepic позволяют пользователям вычерчивать отрезки под любым углом наклона и круги любого размера. Однако ограничение на углы наклона векторов остается прежним. Это означает, что можно обходиться только тангенсами углов наклона вида x/y, где x и y — целые числа в пределах $[-4, 4]$.

`\line(`*x*`,`*y*`){`*length*`}`

Синтаксис команды \line — такой же, как в LATEX'е [$\mathcal{L}$ 106], [$\mathcal{A}$ 175]. Но в данном случае x и y могут быть любыми целыми числами, приемлемыми в TEX'е. Кроме того, больше не существует ограничения снизу для параметра *length* (около 3.5 mm в стандартном LATEX'е).

`\circle{`*diameter*`}` `\circle*{`*diameter*`}`

Синтаксис команд \circle и \circle* для вычерчивания соответственно полых кругов (окружностей) и закрашенных кругов — такой же, как в LATEX'е [$\mathcal{L}$ 107], [$\mathcal{A}$ 176]. Но теперь параметр *diameter* может быть любым числом, приемлемым в TEX'е, и круг будет нарисован (в точности) того диаметра, который будет указан.

`\oval(`*x-dimen*`,`*y-dimen*`){`*part*`}`

Команда \oval была модифицирована так, что максимальный диаметр четвертинок окружностей «по углам» овала может быть установлен равным любому значению. Это можно сделать, присвоив переменной \maxovaldiam желаемое TEX'овское значение длины (dimension) по умолчанию 40 pt.

10.4.2 Расширения пакета epic при помощи eepic

Пакет epic порождает стандартные dvi-файлы и требует присутствия только стандартных LATEX'овских шрифтов. Пакет eepic как расширение epic дает лучший результат при вычерчивании линий, быстрее работает и требует меньше памяти. В нем переделываются команды \dottedline, \dashline и \drawline (см. с. 337) и соответствующие им окружения типа join: dottedjoin, dashjoin и drawjoin (см. с. 339).

10.4.3 Новые команды в пакете eepic

В пакете eepic определяется набор новых команд: кроме команды \path, другие команды не имеют эквивалентов в LaTeX'e и пакете epic. По поводу совместимости см. разд. 10.4.4.

\allinethickness{*dimension*}

Команда \allinethickness устанавливает толщину линий для всех рисующих линии команд, включая наклонные прямые, окружности, эллипсы, дуги, овалы и сплайны.

\Thicklines

После указания команды \Thicklines все рисуемые затем линии будут приблизительно в 1.5 раза толще, чем после указания \thicklines.

\path$(x_1,y_1)(x_2,y_2)...(x_n,y_n)$

Команда \path является быстрым вариантом команды \drawline. Необязательный аргумент *stretch* последней команды здесь недопустим, а потому команда \path рисует только сплошные линии. Команда \path используется главным образом для вычерчивания сложных траекторий.

\spline$(x_1,y_1)(x_2,y_2)...(x_n,y_n)$

Команда \spline рисует кривую Чайкина, которая проходит только через первую и последнюю точки. Все остальные точки являются всего лишь контрольными точками.

\ellipse{*x-diameter*}{*y-diameter*} \ellipse*{*x-diameter*}{*y-diameter*}

В полной аналогии с командами \circle и \circle* команды \ellipse и \ellipse* рисуют полый и закрашенный эллипсы, используя указываемые параметры *x-diameter* и *y-diameter*.

\arc{*diameter*}{*start-angle*}{*end-angle*}

Команда \arc рисует дугу окружности. Первый параметр *diameter* определяет диаметр в единицах \unitlength. Оба параметра *start-angle* (начальный угол) и *end-angle* (конечный угол) указываются в радианах; *start-angle* должен лежать в отрезке $[0, \frac{\pi}{2}]$, а *end-angle* может быть любым значением между *start-angle* и *start-angle* $+ 2\pi$. Дуги рисуются по часовой стрелке и угол 0 совпадает с положительным направлением оси x (указывая вправо по отношению к странице).

\filltype{*area-fill-type*}

Команда \filltype указывает тип насыщенности закрашивания для команд \circle* и \ellipse*. Сама по себе эта команда ничего не рисует. Она только меняет интерпретацию звездочки * в двух упомянутых выше командах. Возможные

значения для параметра *area-fill-type* — это `black` (черный; по умолчанию), `white` (белый) и `shade` (затененный); например, вы можете изменить тип насыщенности закрашивания на белый, написав `\filltype{white}`.

10.4.4 Совместимость

Пакет eepic не обязательно имеется на всех платформах, где есть LaTeX. Во избежание проблем с несовместимостью, которые могут возникнуть при его использовании, и в то же время чтобы извлечь выгоду из пакета eepic, дающего более качественный результат на печати, вам следует принять следующие меры предосторожности:

- Не пользуйтесь командой `\line`, а вместо нее используйте `\drawline`, так как команда `\line` в LaTeX'е поддерживает только конечное число углов наклона.
- Не используйте команду `\arc`. Если действительно нужно нарисовать сложную кривую, следует использовать команду `\spline`.
- При вычерчивании длинных пунктирных линий избегайте сплошных линий или линий с малым расстоянием между образующими линию точками, поскольку в исходной реализации пакета epic они требуют много TeX'овской памяти. Для пунктирных линий нужно использовать команду `\drawline` с отрицательным значением параметра *stretch*.

Если ваша инсталляция не поддерживает eepic, но вам нужно распечатать ваш документ, то следует использовать макро, эмулирующие пакет eepic и определенные в пакете eepicemu. Команды пакета eepic эмулируются следующим образом:

- Круги диаметром больше 40 pt рисуются при помощи команды `\oval`.
- Эллипсы рисуются при помощи команды `\oval`.
- Сплайны аппроксимируются при помощи команды `\drawline`.
- Команда `\path` замещается командой `\drawline`.
- Команда `\Thicklines` замещается командой `\thicklines`.
- Команда `\allinethickness` заменяется командами `\thicklinesи` `\linethickness`.

Поскольку в пакете eepic переопределяются некоторые команды пакета epic, указание пакета eepic должно следовать за указанием пакета epic, т. е.

```
\documentclass[...]{article}
\usepackage{epic}
...
\usepackage{eepic}
```

При использовании команд пакета eepic хорошей практикой является всегда включать и пакет epic, хотя это не строго обязательно. В любом случае эмулирующий eepic пакет eepicemu будет работать только тогда, когда оба пакета указаны.

10.4.5 Примеры

Рисунок 10.6 на с. 345 показывает ряд степенных функций. На нем вы можете видеть, как используется сетка для обозначения координатной системы и как различные типы отрезков (состоящие из точек, обычные, пунктирные), рассмотренные в предыдущих разделах, применяются для того, чтобы разные кривые отличались друг от друга. Каждая кривая снабжена меткой с дробным показателем, обозначающей функцию, по которой вычислялись точки кривой. Обратите также внимание на то, что линии ниже диагонали нарисованы с применением команды `\thinlines`, а те линии, что расположены в верхней части диаграммы, нарисованы с использованием команды `\thicklines`.

Рисунок 10.7 на с. 346 является возможным приложением из жизни. На нем показаны энергия возбуждения (•) и пороговая энергия (○) изотопов с атомными весами вблизи 235 (уран). В этом случае мы построили горизонтальную и вертикальную оси, используя команды `\multiputlist`. Текст вдоль вертикальной оси был напечатан при помощи команды `\shortstack`. Наконец, различные координаты точек были введены с использованием окружений `dottedjoin` и `dashjoin` пакета epic, а также соответствующих им команд `\jput` (см. с. 339 и дальше).

10.5 Пакеты, основанные на epic

10.5.1 Как рисовать двудольные графы

Пакет eclbip, написанный Хидеки Исодзака, использует функциональные возможности пакета epic для того, чтобы создавать двудольные графы. Двудольный граф — это линейная сеть, в которой вершины можно разделить на две группы, левую (*left*) и правую (*right*), так что начальные и конечные точки всех дуг лежат (начинаются) в противоположных группах.

Двудольный граф рисуется при помощи окружения `bipartite`.

`\begin{bipartite}{`*leftwd*`}{`*gapwd*`}{`*rightwd*`}{`*gapht*`}{`*labelwd*`}`

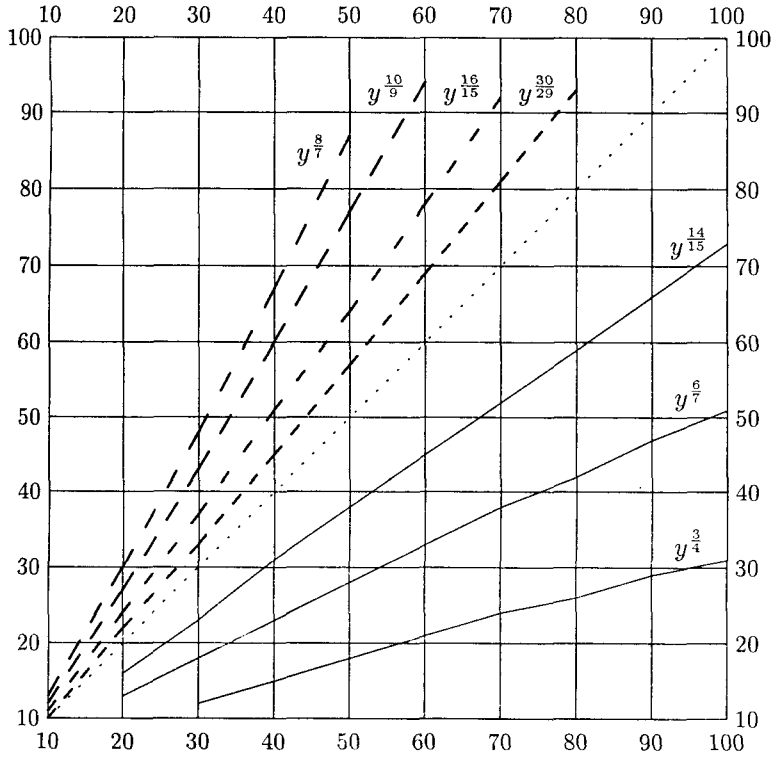
Окружение `bipartite` имеет пять аргументов:

leftwd Максимальная ширина меток в группе левых вершин.
gapwd Ширина зазора между группой левых вершин и группой правых вершин.
rightwd Максимальная ширина меток в группе правых вершин.
gapht Минимальная высота зазора по вертикали между метками вершин.
labelwd Зазор между вершиной (жирной точкой bullet) и ее меткой.

Внутри окружения `bipartite` можно применять следующие команды:

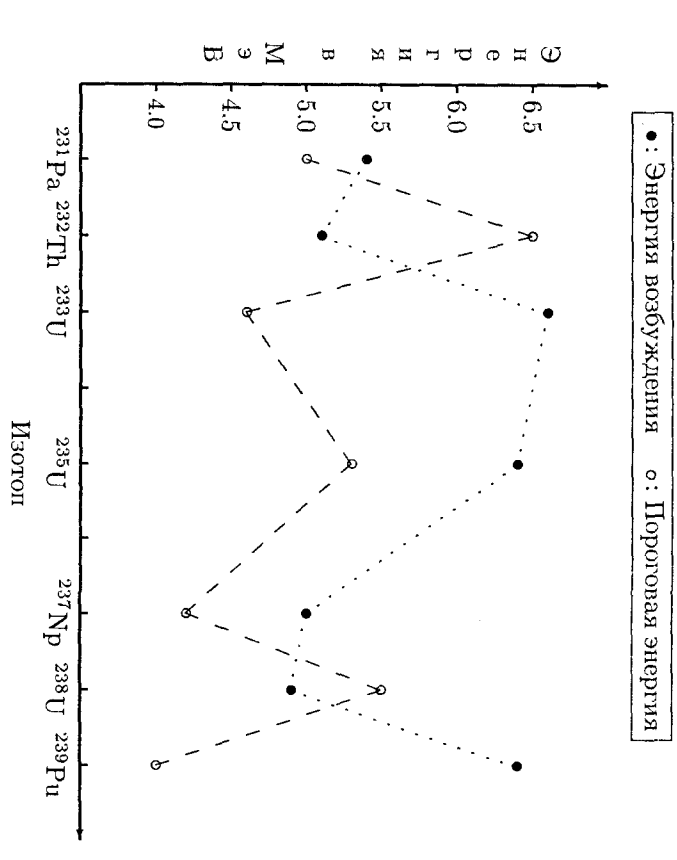
`\leftnode[`*short-label*`]{`*long-label*`}`

Это метка, которую нужно написать на левой вершине. Обязательный аргумент *long-label* будет напечатан, а (его) аббревиатуру *short-label* можно использовать в команде `\match` (см. ниже).




```
\newcommand{\Fr}[2]{$y^{\frac{#1}{#2}}$}
\unitlength = 1mm
\begin{picture}(100,105)(0,0)
  \thinlines
  \put(10,10){\small\grid(90,90)(10,10)[10,10]}
  \dottedline{2}(10,10)(100,100)
  \put(95,33){\makebox(0,0){\Fr{3}{4}}}
  \drawline(30,12)(40,15)(50,18)%
        (60,21)(70,24)(80,26)(90,29)(100,31)
  \put(95,53{\makebox(0,0){\Fr{6}{7}}}
  \path(20,13)(30,18)(40,23)(50,28)%
        (60,33)(70,38)(80,42)(90,47)(100,51)
  \put(95,73}{\makebox(0,0){\Fr{14}{15}}}
  \path(20,16)(30,23)(40,31)(50,38)%
        (60,45)(70,52)(80,59)(90,66)(100,73)
  \thicklines
  \put(45,85){\makebox(0,0){\Fr{8}{7}}}
  \dashline{3}(10,13)(20,30)(30,48)(40,67)(50,87)
  \put(55,93){\makebox(0,0){\Fr{10}{9}}}
  \dashline[+20]{3}(10,12)(20,27)(30,43)(40,60)%
        (50,77)(60,94)
  \put(65,93){\makebox(0,0){\Fr{16}{15}}}
  \dashline{2}(10,11)(20,24)(30,37)(40,51)%
        (50,64)(60,78)(70,92)
  \put(75,93){\makebox(0,0){\Fr{30}{29}}}
  \dashline[+20]{2}(10,10)(20,22)(30,33)(40,45)%
        (50,57)(60,69)(70,81)(80,93)
\end{picture}
```

Рис. 10.6. Команды для вычерчивания линий из пакетов epic и eepic




```
\newcommand{\CHo}{\makebox(0,0){$\bullet$}}
\newcommand{\CHc}{\makebox(0,0){$\circ$}}
\newcommand{\El}[2]{\makebox(0,0){${}^{#1}$#2}}
\setlength{\unitlength}{10mm}
\begin{picture}(11,9)(229,-1)
   \linethickness{1pt}
   \put(230,0){\vector(1,0){10}}
   \put(230,0){\vector(0,1){7}}
   \thicklines
   \multiput(230,0)(1,0){10}{\line(0,1){.1}}
   \multiput(230,0)(0,1){7}{\line(1,0){.1}}
   \multiputlist(231,-.3)(1.,0){\El{231}{Pa},%
         \El{232}{Th},\El{233}{U}, ,\El{235}{U},%
       ,\El{237}{Np},\El{238}{U},\El{239}{Pu}}
   \multiputlist(230.2,1)(0,1)[l]{\small
       4.0,4.5,5.0,5.5,6.0,6.5}
   \put(235,-.8){\makebox(0,0){Изотоп}}
   \put(229.6,4.){\makebox(0,0){\shortstack{%
       Э\\н\\е\\р\\г\\и\\я\\[2ex]в\\[2ex]М\\э\\В}}}
   \put(234.5,7.6){\makebox(0,0){\fbox{%
       \makebox(.3,.2)[lb]{\put(.2,.06){\CHo}}:
           Энергия возбуждения\quad
       \makebox(.3,.2)[lb]{\put(.2,.06){\CHc}}:
           Пороговая энергия}}}
   \thinlines
   \begin{dottedjoin}{.2}
     \jput(231,3.8){\CHo}\jput(232,3.2){\CHo}
     \jput(233,6.2){\CHo}\jput(235,5.8){\CHo}
     \jput(237,3.0){\CHo}\jput(238,2.8){\CHo}
     \jput(239,5.8){\CHo}\end{dottedjoin}
   \begin{dashjoin}{.2}
     \jput(231,3.0){\CHc}\jput(232,6.0){\CHc}
     \jput(233,2.2){\CHc}\jput(235,3.6){\CHc}
     \jput(237,1.4){\CHc}\jput(238,4.0){\CHc}
     \jput(239,1.0){\CHc}\end{dashjoin}
\end{picture}
```

Рис. 10.7. Графики, нарисованные при помощи пакетов epic и eepic

`\rightnode[`*short-label*`]{`*long-label*`}`

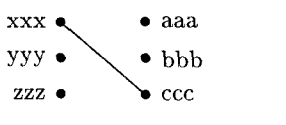
Это метка для правой вершины. Аргументы те же самые, что и в команде `\leftnode`.

`\match{`*leftnode*`}{`*rightnode*`}`

Эта команда соединяет вершины, специфицированные как *leftnode* (левая вершина) и *rightnode* (правая вершина).

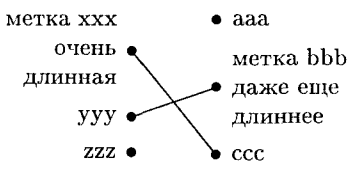
`\brush{`*draw-command*`}`

Эта команда указывает команду для рисования *draw-command*, которая будет использоваться с последующими командами `\match`.



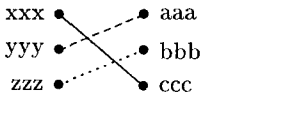

```
\begin{bipartite}{2cm}{1.5cm}{2cm}{3mm}{2mm}
  \leftnode{xxx} \leftnode{yyy} \leftnode{zzz}
  \rightnode{aaa}\rightnode{bbb}\rightnode{ccc}
  \match{xxx}{ccc}
\end{bipartite}
```

Использование длинных меток (*long-label*) и коротких меток (*short-label*) показано в следующем примере:




```
\begin{bipartite}{2cm}{1.5cm}{2cm}{3mm}{2mm}
  \leftnode[x]{метка xxx очень длинная}
  \leftnode{yyy} \leftnode{zzz}
  \rightnode{aaa}
  \rightnode[b]{метка bbb даже еще длиннее}
  \rightnode{ccc}
  \match{x}{ccc} \match{yyy}{b}
\end{bipartite}
```

Чтобы начертить отрезки, связывающие вершины, вы можете использовать команды для вычерчивания линий пакета epic (см. с. 337), указывая выбранную вами команду как аргумент команды `\brush`.




```
\begin{bipartite}{2cm}{1.5cm}{2cm}{3mm}{2mm}
  \leftnode{xxx} \leftnode{yyy} \leftnode{zzz}
  \rightnode{aaa}\rightnode{bbb}\rightnode{ccc}
  \match{xxx}{ccc}
  \brush{\dottedline{3}}   \match{zzz}{bbb}
  \brush{\dashline[50]{3}} \match{yyy}{aaa}
\end{bipartite}
```

10.5.2 Как рисовать деревья

Пакет ecltree, также написанный Хидеки Исодзака, использует функции пакета epic для рисования деревьев. Другой пакет для рисования бинарных и тернарных деревьев описывается в разд. 10.2.3.

Дерево можно нарисовать при помощи окружения bundle.

`\begin{bundle}{`*topnode*`}`

Окружение bundle имеет один аргумент *topnode*, указывающий метку верхней вершины.

Внутри окружения bundle можно использовать следующие команды:

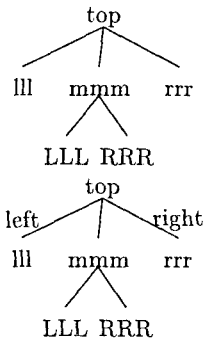
`\chunk[`*edge-text*`]{`*node-text*`}`

Команда \chunk специфицирует вершины дерева. Обязательный аргумент *node-text* — это метка вершины, а необязательный аргумент *edge-text*, если он указан, — это метка ребра.

`\drawwith{`*draw-command*`}`

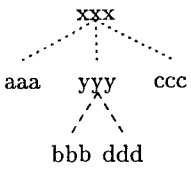
Эта команда контролирует атрибуты линий, используемых для рисования ребер. Ее аргументом является одна из команд пакета epic, описание которых начинается со с. 337.

Первый пример показывает, как используются вложенные окружения bundle, а второй — как используется необязательный аргумент *edge-text* для нанесения меток для ребер.



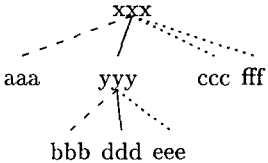

```
\begin{bundle}{top}
\chunk{lll}
\chunk{\begin{bundle}{mmm}
        \chunk{LLL} \chunk{RRR}
       \end{bundle}}
\chunk{rrr}
\end{bundle}

\begin{bundle}{top}
\chunk[left]{lll}
\chunk{\begin{bundle}{mmm}
        \chunk{LLL} \chunk{RRR}
       \end{bundle}}
\chunk[right]{rrr}
\end{bundle}
```

Атрибуты линий для ребер контролируются при помощи команды \drawwith. Заметьте, что аргумент команды \drawwith вычисляется при выходе из окружения bundle после команды \end{bundle}. Поэтому в следующем примере первая команда \drawwith игнорируется:




```
\begin{bundle}{xxx}
  \chunk{aaa}
  \chunk{\begin{bundle}{yyy}
          \drawwith{\drawline}%Игнорируется
          \chunk{bbb}
          \drawwith{\dashline[50]{3}}
          \chunk{ddd}
         \end{bundle}}
  \drawwith{\dottedline{3}}
  \chunk{ccc}
\end{bundle}
```

Вкладывая друг в друга команды `\drawwith`, для линий (ребер) можно установить несколько атрибутов, при этом команды `\drawwith` будут выполняться в обратном порядке.



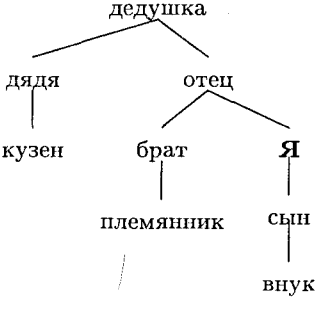

```
\drawwith{\drawwith{\drawwith{\dottedline{3}}%
                    \drawline}%
          \dashline{3}}
\begin{bundle}{xxx}
 \chunk{aaa}
 \chunk{\begin{bundle}{yyy}
         \chunk{bbb} \chunk{ddd} \chunk{eee}
        \end{bundle}}
 \chunk{ccc} \chunk{fff}
\end{bundle}
```

Пробелами внутри окружения `bundle` можно управлять при помощи трех параметров, значения которых должны быть установлены перед входом в окружение `bundle`.

`\GapDepth` Минимальная высота зазоров между прилежащими вершинами.

`\GapWidth` Минимальная ширина зазоров между прилежащими вершинами.

`\EdgeLabelSep` Отделение метки ребра от нижней вершины ребра.




```
\setlength{\GapDepth}{5mm}
\setlength{\GapWidth}{5mm}
\begin{bundle}{дедушка}
  \chunk{\begin{bundle}{дядя}
           \chunk{кузен}
         \end{bundle}}
  \chunk{\begin{bundle}{отец}
           \chunk{\begin{bundle}{брат}
                    \chunk{племянник}
                  \end{bundle}}
           \chunk{\begin{bundle}{\textbf{Я}}
                    \chunk{\begin{bundle}{сын}
                             \chunk{внук}
                           \end{bundle}}
                  \end{bundle}}
         \end{bundle}}
\end{bundle}
```

ГЛАВА 11

Как использовать PostScript

Как упоминалось в предыдущей главе, сложная графика, выраженная в командах языка PostScript, может быть включена в LaTeX'овские документы при помощи команд `\special`, которые интерпретируются `dvi`-драйвером, поддерживающим язык PostScript.

Эта глава вначале представляет краткое введение в язык PostScript, а затем знакомит с программой dvips — одним из наиболее популярных `dvi`-драйверов, поддерживающих PostScript. Последующие разделы рассматривают вопросы о том, как соединять графику и текст и как манипулировать (вращать, масштабировать, затенять и менять цвет) элементами документа, используя возможности языка PostScript и драйвера dvips. В последнем разделе еще одно обращение к NFSS покажет, каким образом при помощи виртуальных шрифтов исходные PostScript'овские шрифты могут быть использованы с LaTeX'ом.

11.1 Язык PostScript

11.1.1 Несколько слов о языке

PostScript [65, 66, 69, 72, 88, 89] — это язык описания страницы. Он представляет способ выразить посредством языка внешний вид печатной страницы, включая текст, строки и графику. PostScript является не зависящим от устройства и разрешения языком программирования, который описывает сразу всю страницу в единицу времени, а не отдельную строку, как в случае строчного принтера.

PostScript — это язык программирования высокого уровня со стековой организацией, использующий обратную польскую или постфиксную нотацию. Это гибкий язык, поскольку он включает в себя циклические конструкции, процедуры и операторы сравнения и поддерживает много видов данных: вещественные, булевы, массивы, цепочки символов и такие сложные объекты, как словари.

PostScript не зависит от устройства и его разрешающей способности, т. е. программное обеспечение не привязано к какому-либо конкретному аппаратному

обеспечению. Один и тот же ASCII-файл будет напечатан на обычном лазерном принтере с разрешением 300 dpi (dots per inch) и на фотонаборном устройстве с разрешением 2540 dpi. Этот файл можно также просмотреть на дисплее компьютера при помощи программ Display PostScript или программ предварительного просмотра, таких, как ghostscript/ghostview.

Хотя для данного приложения есть возможность откомпилировать PostScript, обычно PostScript'овский файл обрабатывается в принтере, так что приложения можно разработать естественным образом. Тем самым большинству пользователей никогда не придется сталкиваться с языком напрямую, но они должны осознавать, что используют PostScript всякий раз, когда они печатают что-либо на принтере, поддерживающем язык PostScript.

Язык PostScript обладает перечисленными ниже возможностями, которые можно использовать в любом сочетании.

- Из прямолинейных отрезков, дуг и кубических кривых можно построить любые фигуры. Эти фигуры могут самопересекаться и содержать несвязные куски и дыры.
- Примитивы, отвечающие за изображение, позволяют рисовать фигуры при помощи линий любой толщины, закрашивать любым цветом или использовать их как шаблоны для высечения любой другой графики.
- Текст полностью совмещается с графикой. Текстовые символы в PostScript'е трактуются как графические образы, на которые можно воздействовать любым из графических операторов языка. Это одинаково справедливо как для шрифтов Type 1, где образы символов определяются с использованием специальных процедур кодирования [90], так и для определенных пользователем шрифтов Type 3, где образы символов определяются как обычные процедуры языка PostScript. В настоящее время в формате PostScript имеются тысячи моделей шрифтов, включая шрифты и таких основных производителей типографского обеспечения в мире, как компании Linotype, Agfa-Compugraphic, Monotype, Autologic и Varityper. Вы можете подгрузить эти шрифты или шрифты вашего собственного изготовления с Макинтоша, PC или большой вычислительной машины на любой PostScript'овский принтер. Хотя можно использовать растровые шрифты, но, как правило, контурные шрифты предпочтительнее по следующим причинам:
  - они не зависят от устройства и разрешения;
  - они созданы при помощи математического моделирования;
  - использование кривых Безье дает выигрыш в точности и гибкости;
  - они определены в координатной системе 1000 на 1000 с символами размером 1 pt, которые можно по желанию масштабировать, поворачивать или наклонять (см., например, рис. 11.1).
- Для сложных языков с множеством тысяч символов (например, китайского или японского языков) можно использовать составные шрифты Type 0.
- Для изображений (таких как фотографии или синтетически созданные образы) можно получать пробы с любым разрешением и во множестве динамиче-

ски меняющихся диапазонов. PostScript предоставляет средства для контролирования при воспроизведении образов на выводящем устройстве.

- Поддерживается несколько цветных шаблонов, и возможно конвертирование из одного шаблона в другой.
  - RGB или аддитивный шаблон *Red Green Blue* (*Красный Зеленый Синий*) используется с дисплеями и кинокамерами.
  - HSB или шаблон *Hue Saturation Brightness* (*Оттенок Насыщенность Яркость*), где *hue* — это количество красного, зеленого и синего цветов; *saturation* — это количество цвета и тени; и *brightness* — это яркость или затемненность по отношению к полному цвету.
  - CMYK или субтрактивный шаблон *Cyan Magenta Yellow Black* (*Голубой Пурпурный Желтый Черный*) используется в индустрии книгопечатания.
  - CIE или международный стандарт используется в графическом искусстве, на телевидении и в индустрии книгопечатания в целях подготовки ссылок.
- Общее приспособление координатной системы поддерживает все комбинации линейных преобразований, включая масштабирование, поворот, отражение и наклон. Эти преобразования воздействуют равномерно на все элементы страницы, включая текст, графические образы и пробы.
- Вы можете создавать библиотеки цветов, шрифтов, форм, изображений, полутонов и узоров.
- Имеется несколько уплотняющих фильтров, таких, как JPEG и LZW.

В качестве иллюстрации некоторых из перечисленных выше свойств снова см. рис. 11.1.

11.1.2 Что такое инкапсулированный PostScript?

Рисунки в формате PostScript часто требуется включить в текст, подготовленный таким средством форматирования, как TEX. Компания Adobe разработала формат файла *Инкапсулированный PostScript*[1] (EPS или EPSF), который создается в соответствии с *соглашениями по структурированию документов в формате PostScript* (*PostScript Document Structuring Conventions*, см. приложения G и H в [89] или [79]). Формат EPS определяет стандартные правила для импортирования файлов на языке PostScript в различные среды. EPS-файлы должны «хорошо себя вести» при использовании некоторых операторов PostScript'а, манипулировании графическим состоянием, стеком интерпретатора и глобальными библиотеками, так что они не влияют пагубно на страницу, подготовленную средством форматирования текстов.

[1] Инкапсуляцией называется объединение в объекте кода и данных. — *Прим. перев.*

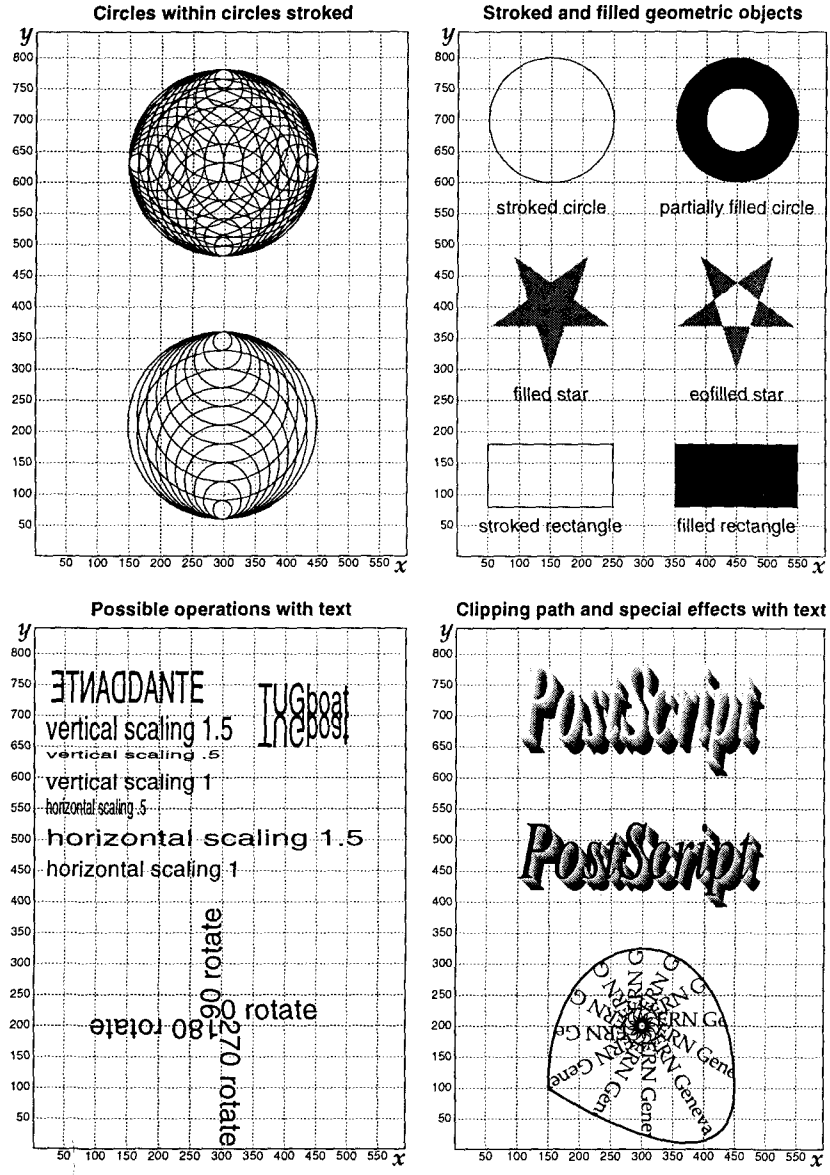


Рис. 11.1. Примеры возможностей языка PostScript

Числа на осях и пунктирные линии указывают значения координат.

Большинство современных графических приложений порождает файл в общепринятом формате EPS, который без труда может быть использован LaTeX'ом. Однако иногда вы можете столкнуться с чистым файлом в формате PostScript, который не содержит необходимой информации. Для использования с LaTeX'ом файл в формате PostScript не обязательно строго должен соответствовать структурным соглашениям, упомянутым выше. Если этот файл «хорошо ведет себя» (см. выше), то достаточно, чтобы PostScript'овский файл содержал размеры бокса, занимаемого рисунком. Эти размеры сообщаются LaTeX'у при помощи PostScript'овской закомментированной строки `%%BoundingBox` (дословно *обрамляющий бокс*), как показано ниже:

```
%!
%%BoundingBox: LLx LLy URx URy
```

Первая строка указывает, что мы имеем дело с нетрадиционным файлом в формате EPS. Заметьте, что символы `%!` должны занимать первые две позиции строки. Вторая строка, которая более важна для нашей цели, указывает размер содержащегося рисунка в «больших» PostScript'овских пунктах (bp); 72 таких пункта составляют один дюйм (см. табл. А.1). Четыре параметра на этой строке — это x- и y-координаты нижнего левого угла (`LLx` и `LLy`) и верхнего правого угла (`URx` и `URy`) нашего рисунка. Например, целая страница формата A4 (210 мм на 297 мм), нижний левый угол которой находится в начале координат, потребует следующей записи:

```
%!
%%BoundingBox: 0 0 595 842
```

Если рисунок начинается в нижней левой точке с координатами (100, 200) и заключен в квадрат со стороной 4 дюйма (288 пунктов), то утверждение будет таким:

```
%!
%%BoundingBox: 100 200 388 488
```

Чтобы быть уверенным, что рисунок будет включен целиком, хорошей практикой является добавить один или два пункта к координатам; это связано с ошибками округления при вычислениях, производимых интерпретатором.

11.2 dvips — преобразование dvi в PostScript

Большая часть TeX'овских документов в конкретной организации приспособлена для использования бумаги стандартного размера (например, формат letter в США или формат A4 в Европе). Программа dvips, транслятор из dvi в PostScript, разработанная Томашем Рокицки [68], настроена на эти размеры бумаги, но она также может быть приспособлена к любому формату в любой организации и для любого принтера. Программа dvips естественным образом поддерживает графику, позволяя вставлять графику в PostScript'е, автоматически масштабировать и размещать множеством способов. Команды PostScript'а можно

включать в команды `\special` буквальным образом, но такое их использование нежелательно.

Отсутствующие шрифты можно автоматически сгенерировать, если на компьютере имеется METAFONT. Если шрифт нельзя сгенерировать, то вместо него будет использован масштабированный вариант того же самого шрифта, но другого размера, а программа dvips выразит недовольство плохим качеством того, что получается на выходе.

Одной из важнейших черт программы dvips является то, что она поддерживает виртуальные шрифты, что позволяет использовать PostScript'овские шрифты с TEX'ом (подробнее об этом см. разд. 11.9). Шрифты в формате PostScript сопровождаются файлом с расширением `.afm` (Adobe font metric), таким, как `Times-Roman.afm`, который описывает характеристики данного шрифта. Чтобы использовать такие шрифты с TEX'ом, следует породить `.tfm`-файлы, которые содержат информацию о каждом символе. Программа afm2tfm, распространяемая вместе с dvips, извлекает необходимую информацию из `.afm`-файла и генерирует `.tfm`- и `.vf`-файлы. Она также позволяет использовать различные кодировки для PostScript'овского шрифта, которые в некоторых случаях оказываются удобными в пользовании.

Драйвер dvips имеет множество опций, используемых в командной строке. В табл. 11.1 представлена сводка этих опций.

11.3 Совмещение текста и графики в формате PostScript

В этом разделе будут описаны некоторые пакеты, которые используют особенности драйвера dvips и языка PostScript. В принципе некоторые из функций присущи и другим драйверам, таким, как dvitops или драйверам из дистрибутива emTEX'а (см. табл. 11.2). Само собой разумеется, что рисунки в формате PostScript будут видимы только с драйверами принтера или программами предварительного просмотра, которые поддерживают PostScript, как программа ghostview. Заметьте, что это мощное средство дает простой путь обнаружить на экране вашего компьютера *обрамляющий бокс* вашего PostScript'овского рисунка (см. разд. 11.1.2). Таблица 11.2 показывает, какие драйверы воспринимаются пакетами и до какой степени функциональные возможности, заложенные в пакете, могут быть осуществлены при использовании определенного драйвера. Последующие выпуски пакетов могут поддерживать дополнительные драйверы, так что вам следует обратиться к прилагаемой документации, содержащей обновленные списки драйверов. В табл. 11.2 также приводится специфическая для каждого пакета команда, указывающая драйвер, для которого LATEX должен сгенерировать команды `\special` в dvi-файле.

Обновленная версия этих пакетов для LATEX 2_ε будет поддерживать имена драйверов как опции к пакетам; например, вы можете написать что-то вроде

```
\documentclass[..,emtex]{article}
\usepackage{epsfig}   \usepackage{changebar}
```

a*	Сберечь память, а не время	x #	Подавить увеличение в dvi-файле
b #	Копии страниц, например, для объявлений	y #	Принять увеличение в dvi-файле
c #	Неупорядоченные копии	A	Напечатать нечетные (TEX) страницы
d #	Отладка	B	Напечатать четные (TEX) страницы
e #	Значение maxdrift	C #	Упорядоченные копии
f*	Обработать как фильтр	D #	Разрешение
h f	Добавить файл заголовка	E*	Попытаться создать EPSF
i*	Разложить файл по разделам	F*	Послать control-D в конце
k*	Напечатать высекающие пометки	K*	Вытолкнуть комментарии
l #	Последняя страница	M*	Не создавать шрифты
m*	Вывод вручную	N*	Без комментарий
n #	Максимальное число страниц	O c	Установить/изменить сдвиг бумаги
o f	Выводимый файл	P s	Загрузить config.$s
p #	Первая страница (p=# полностью)	R	Обработать без остановок
pp#	Только одну страницу от и до ppn_1 : n_2	S #	Максимальное количество страниц в разделе
q*	Обработать быстро	T c	Указать нужный размер страницы
r*	Обратный порядок страниц	U*	Невозможность сохранить параметры
s*	Вывод в режиме сохранить/ восстановить	X #	Разрешение по горизонтали
t s	Формат страницы	Y #	Разрешение по вертикали

# = число f = имя файла s = строка * = суффикс, 'O' — выключить
c = разделенная запятой пара TEX'овских длин (например, 3.2 in, -32.1 cm)

Таблица 11.1. Основные опции программы dvips

и пакеты epsfig и changebar воспримут глобальную опцию emtex для того, чтобы выбрать подходящий код для предполагаемого драйвера. Иначе, как видно из рис. 9.2 и рис. 9.3, вы можете также указать эту опцию (т. е. emtex) в самой команде \usepackage, но поскольку у вас может быть только один драйвер, обрабатывающий весь документ, указание этой опции в команде \documentclass является более экономичным. Ожидается, что другие пакеты LATEX 2_ε, которые также имеют зависящие от драйвера части, будут распознавать одинаковые имена опций, так что имена, указанные выше (и дополнительные имена), станут стандартными опциями для пакетов, имеющих дело с зависящим от драйвера кодом.

В настоящее время разрабатывается пакет под названием graphics, который в будущем будет включен в стандартный дистрибутив LATEX 2_ε. Этот пакет предназначен служить базой для других пакетов, имеющих графические возможности, предоставляя способы подключения графики и конструкции манипулирования

Опция	*Описание драйверов*	epsfig	rotating	changebar
ln	Принтеры Digital Corp. (например, LN03)	X	–	X
dvips	dvips Томаша Рокицки	X	X	X
dvitops	dvitops Джеймса Кларка	X	X	X
emtex	emTeX Эберхарда Маттеса	X	–	X
oztex	OzTeX Эндрю Треворрова	X	o	o
textures	TEXTURES компании Blue Sky Research	X	X	o
	Команда для указания драйвера:	`\psfigdriver`	`\rotdriver`	`\driver`
X означает поддерживается, – означает не или частично поддерживается, o означает отсутствует				

Таблица 11.2. Как поддерживаются dvi-драйверы различными пакетами

графикой (такие как повороты и масштабирование боксов). Как только этот пакет появится, посмотрите документацию, сопутствующую дистрибутиву LaTeX 2_ε.

Пакет epsfig (разработанный Себастьяном Ратцем и базирующийся на более ранней работе Тревора Даррелла) облегчает подключение в TeX'овские документы рисунков в формате EPS. Он извлекает информацию об обрамляющем боксе рисунка из файла и автоматически располагает его, правильно масштабируя на странице согласно пожеланиям пользователя и правильно расставляя пробелы. Всюду в вашем документе вы можете свободно использовать привычные символы, такие как «⊗» и «◒» (последний получается с помощью команды `\epsfig{file=cm.eps,height=3mm}`).

```
\epsfig{file=fn,height=ht,width=wd,clip=,angle=degrees,%
        silent=,bbllx=blx,bblly=bly,bburx=brx,bbury=bry}
```

`file` Этот параметр указывает имя файла *fn* в формате EPS (вы можете также использовать установку `figure=`).

`height` Этот параметр устанавливает желаемую высоту *ht* рисунка (в любых принятых в TeX'е единицах). Если этот параметр не указан, то рисунок будет напечатан в его «естественную» высоту, т. е. в высоту, указанную в строке `BoundingBox` внутри PostScript'овского файла. Когда указана только ширина (см. ниже) и не указана высота, последняя масштабируется в той же пропорции, что и ширина.

`width` Этот параметр устанавливает желаемую ширину *wd* рисунка (в любых принятых в TeX'е единицах). Если этот параметр не указан, то рисунок будет напечатан в его «естественную» ширину, т. е. в ширину, указанную в строке `BoundingBox` внутри PostScript'овского файла. Когда указана только высота (см. выше) и не указана ширина, последняя масштабируется в той же пропорции, что и высота.

`bbllx` Это x-координата *blx* нижнего левого угла обрамляющего бокса (BoundingBox).

`bblly` Это y-координата *bly* нижнего левого угла обрамляющего бокса.

`bburx` Это x-координата *brx* верхнего правого угла обрамляющего бокса.

`bbury` Это y-координата *bry* верхнего правого угла обрамляющего бокса.

`clip` Этот параметр гарантирует, что никакая часть рисунка не появится вне обрамляющего ее бокса. `clip=` — это переключатель; он не принимает никаких значений, но знак `=` должен присутствовать.

`angle` Этот параметр определяет угол поворота (в градусах *degrees*, отсчитываемый против часовой стрелки).

`silent` Этот переключатель заставляет работать команду `\epsfig` по умолчанию.

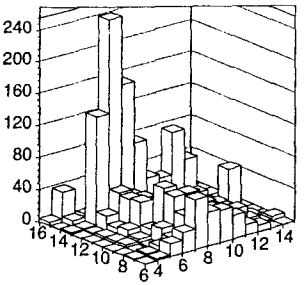
Обычно, когда вы имеете дело с файлами в формате EPS, вам не требуется указывать параметры обрамляющего бокса, поскольку они считываются командой `\epsfig`, указанной в файле. Заметьте, что когда вы указываете параметры обрамляющего бокса перед спецификатором `file=` в команде `\epsfig`, то параметры этого бокса внутри PostScript'овского файла игнорируются. Это оказывается полезным в том случае, когда параметры обрамляющего бокса отсутствуют в PostScript'овском файле или неверны. Такой метод не следует использовать для того, чтобы получить специфические эффекты масштабирования или сдвига на странице. Для этих целей нужно использовать параметры `width` или `height`. Макрокоманда `\epsfig` не чувствительна к пробелам внутри себя; если из-за каких-либо пробелов между аргументами вы обнаружите ошибки, то имеющаяся у вас версия пакета является устаревшей.

`\psfigdriver{`*driveroption*`}`

Эта команда определяет драйвер, для которого должны быть сгенерированы команды `\special`. Поддерживаемые значения параметра *driveroption* указаны в третьей колонке табл. 11.2 на с. 357. Значением по умолчанию является `dvips`. Как отмечалось ранее, версия этого пакета для LaTeX 2_ε будет альтернативно распознавать имя драйвера, указанного как опция документа.

11.3.1 Простые рисунки

Обычно при импортировании рисунка в формате EPS вы будете указывать желаемые высоту и ширину на выводимой странице (если вы не указываете никаких размеров, то будут браться «естественные» размеры рисунка, считанные со строки `BoundingBox` в файле; эти размеры соответствуют размерам рисунка, напечатанного отдельно на PostScript'овском принтере). Нижний край рисунка будет располагаться в том месте, где дана команда `\epsfig`. Изображение будет масштабировано на нужную ширину (или высоту) с одной и той же пропорциональностью по горизонтали и по вертикали, если указан хотя бы один из параметров `height` или `width`.




```
\begin{center}
\epsfig{file=tac2dim.eps,%
        height=3.75cm}
\end{center}
```

Рис. 11.2. Одиночный центрированный рисунок




```
\psdraft        % переключиться
                % в черновой режим
\begin{center}
  \epsfig{file=tac2dim.eps,height=2.5cm}
\end{center}
\psfull         % снова переключиться
                % в окончательный вариант
```

Рис. 11.3. Рисунок в черновом режиме

На рис. 11.2 показана гистограмма заранее заказанной высоты в 4 см. Она расположена по центру благодаря тому, что помещена в окружение `center`.

11.3.2 Черновые рисунки

Передача на принтер и печатание некоторых рисунков в формате PostScript могут занять довольно много времени; для таких рисунков предусмотрен черновой режим «draft», при использовании которого ускоряется печатание черновых вариантов документа. Рисунок, напечатанный в черновом режиме, будет выглядеть как рамка, внутри которой напечатано имя файла, отвечающего за рисунок (рис. 11.3). Если указать опцию draft, то все рисунки будут печататься в черновом режиме. Окончательная верстка, принятая по умолчанию, может быть явно указана при помощи опции final. Команда `\psdraft`, указанная в документе, переключает на черновой режим, и все последующие команды `\epsfig` будут давать черновые рисунки до тех пор, пока не встретится макрокоманда `\psfull`, которая отключает черновой режим. В черновом режиме не используется команда `\special`, поэтому черновой документ можно просматривать при помощи любого `dvi`-драйвера.

11.3.3 Более сложная организация рисунков

Рисунки 11.4, 11.5 и 11.6 показывают, как можно расположить графику на странице, используя окружение `minipage`.

```
\begin{figure}
  \begin{minipage}[b]{.40\linewidth}
    \centering\epsfig{figure=Europe.eps,width=\linewidth}
    \caption{Европа перед 1990\,г.} \label{fig:Europe}
  \end{minipage}\hfill
  \begin{minipage}[b]{.49\linewidth}
    \centering\epsfig{figure=CentralAmerica.eps,width=\linewidth}
    \caption{Центральная Америка}     \label{fig:CentralAmerica}
  \end{minipage}
  \centering\epsfig{figure=TheWorld.eps,width=\linewidth}
  \caption{Карта мира}                \label{fig:World}
\end{figure}
```

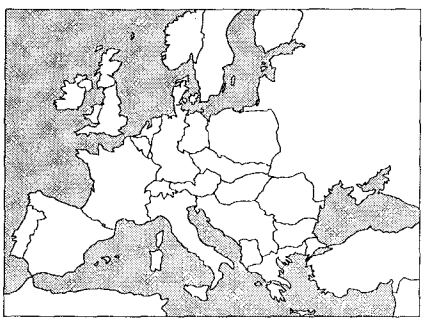


Рис. 11.4. Европа перед 1990 г.

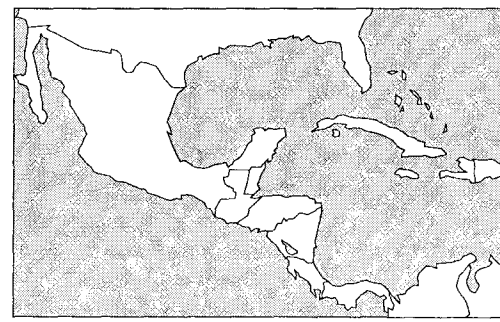


Рис. 11.5. Центральная Америка

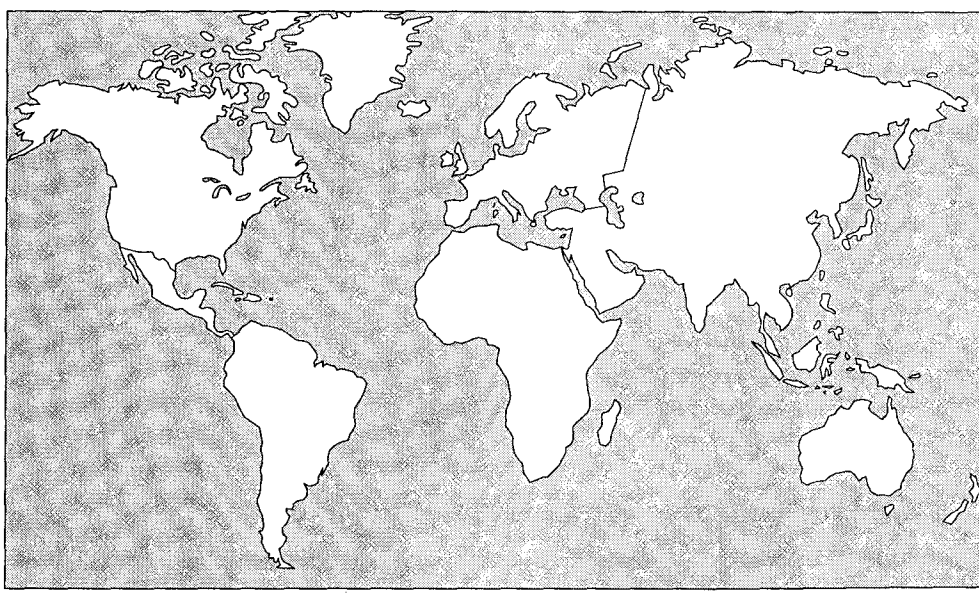


Рис. 11.6. Карта мира

11.4 Как повернуть материал

В пакете rotating Себастьяна Ратца и Леонор Баррока [64] определяются новые окружения, позволяющие легко поворачивать информацию в LaTeX'овских документах. Обновление этого пакета в LaTeX 2_ε будет распознавать имена драйверов из табл. 11.2 на с. 357, указанных в виде опций. Для обратной совместимости используемый драйвер можно указать при помощи команды

`\rotdriver{`*driveroption*`}`

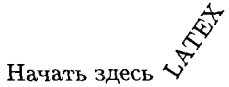
Эта команда определяет драйвер, для которого должны быть сгенерированы команды `\special`. Поддерживаемые значения аргумента *driveroption* приводятся в четвертой колонке табл. 11.2 на с. 357. Значением по умолчанию является драйвер `dvips`.

Окружение `rotate` предоставляет обобщенное окружение для поворотов, где текст поворачивается (по часовой стрелке, как принято в PostScript'е) на заданный угол, указываемый в качестве параметра к этому окружению. Однако специального пространства вокруг повернутого материала не предусмотрено.



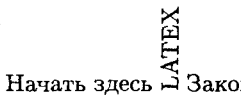

```
Начать здесь \begin{rotate}{56}
   LATEX
\end{rotate} Закончить здесь
```

Если пользователь желает, чтобы LaTeX оставил место для повернутого бокса, то следует использовать окружение `turn`:




```
Начать здесь \begin{turn}{-56}
   LATEX
  \end{turn} Закончить здесь
```

Частным случаем является окружение `sideways`, где поворот осуществляется на $-90°$ и для повернутого бокса оставляются правильные пробелы.




```
Начать здесь \begin{sideways}
   LATEX
\end{sideways} Закончить здесь
```

Если вам придется иметь дело с целым абзацем текста, то вскоре вы обнаружите, что TeX'овские боксы не такие уж простые, какими иногда кажутся: они имеют высоту *и* глубину. Повороты делаются вокруг точки на левом ребре бокса, где ребро пересекает базовую линию. Результаты могут получиться неожиданными, как показано на целом ряде поворотов абзацев на рис. 11.7. Если вы действительно хотите повернуть абзац так, чтобы это выглядело как поворот вокруг *реального* низа TeX'овского бокса, то вам следует подвинуть этот бокс при

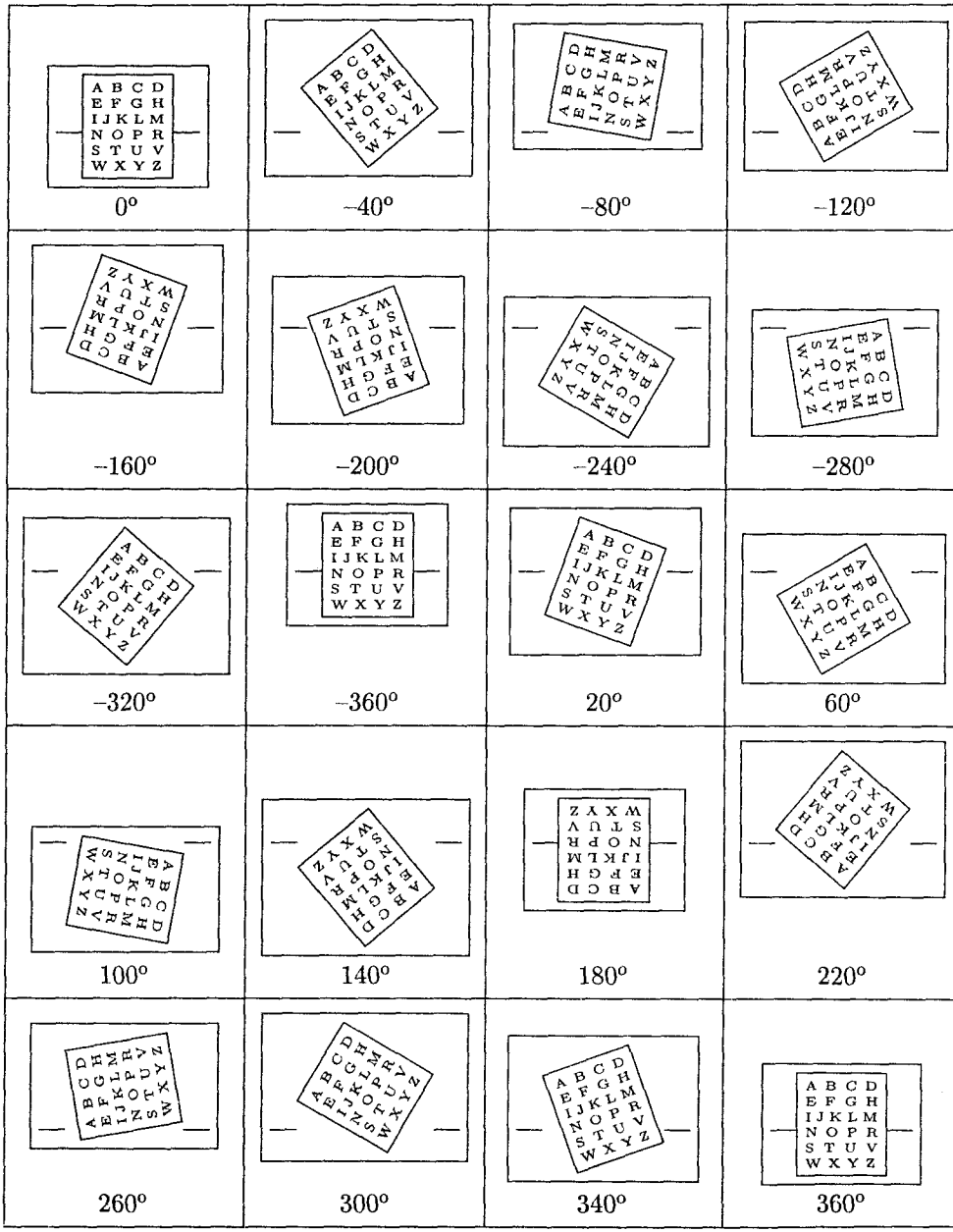


Рис. 11.7. Поворот абзацев

помощи (необязательных) параметров в LaTeX'овских командах, отвечающих за расположение:

Исходный текст

```
\newcommand{\T}{A B C D E F G H I J K L M N O P R S T U V W X Y Z}
Начать       \begin{turn}{-45} \parbox[t]{15mm}{\T} \end{turn}
Продолжить   \begin{turn}{-45} \parbox[b]{15mm}{\T} \end{turn}
Закончить
```

Текст на выводе

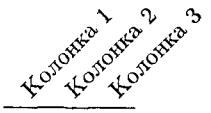
Начать 

Продолжить 

Закончить

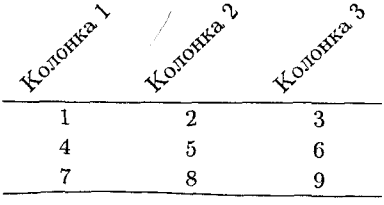
11.4.1 Как повернуть табличный материал

Описанным выше способом можно поворачивать и материал, находящийся в окружении `tabular`. Приводимые ниже примеры показывают, как управлять расстоянием между колонками и вертикальным расположением таблицы, используя линейки нулевой ширины или нулевой высоты.



Колонка 1	Колонка 2	Колонка 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{rrr}
\rule{0pt}{15mm}% вертикальное расположение
\begin{rotate}{-45}Колонка 1\end{rotate}&
\begin{rotate}{-45}Колонка 2\end{rotate}&
\begin{rotate}{-45}Колонка 3\end{rotate}\\
\hline 1& 2& 3\\ 4& 5& 6\\ 7& 8& 9\\ \hline
\end{tabular}
```



Колонка 1	Колонка 2	Колонка 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{ccc}
\begin{turn}{-45}Колонка 1\end{turn}&
\begin{turn}{-45}Колонка 2\end{turn}&
\begin{turn}{-45}Колонка 3\end{turn}\\
\hline 1& 2& 3\\ 4& 5& 6\\ 7& 8& 9\\ \hline
\end{tabular}
```

Колонка 1	Колонка 2	Колонка 3
1	2	3
4	5	6
7	8	9

```
\begin{tabular}{rrr}
\rule{0pt}{15mm}% вертикальное расположение
\begin{rotate}{-45}Колонка 1\end{rotate}
\rule{.5cm}{0pt}&
\begin{rotate}{-45}Колонка 2\end{rotate}
\rule{.5cm}{0pt}&
\begin{rotate}{-45}Колонка 3\end{rotate}
\rule{.5cm}{0pt}\\
\hline 1& 2& 3\\ 4& 5& 6\\ 7& 8& 9\\ \hline
\end{tabular}
```

Как показано ниже, повороты можно вкладывать друг в друга.

Слово	привет	до свидания
Частота	33	34

```
\begin{sideways}
\begin{tabular}{l@{\qquad}r}
\em Слово \rule{0pt}{1in}
            & \begin{rotate}{-90}%
Частота   \end{rotate}  \\[1mm]
\hline
привет & 33\\ до свидания & 34\\ \hline
\end{tabular}
\end{sideways}
```

Более сложный пример приведен в табл. 11.3. Содержимое окружения `tabular` повернуто при помощи окружения `sideways`. Обратите также внимание на окружение `rotate`, генерирующее вертикально расположенный текст (Классы форматов). Так как окружение `rotate` порождает бокс нулевой ширины, оно окружается двумя «невидимыми линейками» в 1 мм шириной каждая. Эта ширина добавляется к параметру `\tabcolsep` величиной в 2 мм и дает результат, показанный в таблице. Повернутую таблицу можно также создать при помощи окружения `sidewaystable`; в этом случае поворачиваются и таблица, и подпись к ней (см. табл. 11.4). Окружение `sidewaystable` использует ширину, равную `\textheight`, так что, когда поворачивается плавающий объект, он умещается по высоте. На самом деле это не очень хорошо, поскольку в действительности нужно, чтобы плавающие объекты занимали ровно столько места, сколько они занимают. Но подписи представляют проблему, так как они могут предшествовать рисунку или таблице. Как результат, их нельзя заключить в бокс нужной ширины (т. е. высоты ожидаемого объекта), потому что она еще неизвестна. Одно из возможных решений — это сделать так, чтобы окружение `sidewaystable` (и его эквивалент `sidewaysfigure`, обсуждаемый в следующем разделе) всегда заполняло целую страницу. Если же это не желательно, то вы можете создать бокс нужного размера и расположить материал и подпись внутри этого бокса.

11.4.2 Как повернуть рисунок

Рисунки можно повернуть при помощи тех же команд, что описаны выше. Рисунок 11.8 показывает, как можно по желанию повернуть EPS-файл при помощи команды `\epsfig`. Обратите внимание на положение базовой линии, отмеченной em-тире (которое определено командой `\HR`).

Форматы бумаги ISO (в мм)			
Классы форматов	Серия A	Серия B	Серия C
0	841×1189	1000×1414	917×1297
1	594×841	707×1000	648×917
2	420×594	500×707	458×648
3	297×420	353×500	324×458
4	210×297	250×353	229×324
5	148×210	176×250	162×229
6	105×148	125×176	114×162
7	74×105	88×125	81×114
8	52×74	62×88	57×81

```
\renewcommand{\arraystretch}{1.2}
\setlength{\tabcolsep}{2mm}
\begin{sideways}
\begin{tabular}{|l|l*3{r@{$\times$}l}|}
                                        \hline
\multicolumn{8}{|c|}{Форматы бумаги ISO
                                (в мм)}\\\hline
&&\multicolumn{2}{c}{Серия A}
 &\multicolumn{2}{c}{Серия B}
 &\multicolumn{2}{c|}{Серия C}\\\cline{2-8}
&0&841&1189&1000&1414&917&1297\\\cline{2-8}
&1&594&841 & 707&1000&648&917 \\\cline{2-8}
&2&420&594 & 500&707 &458&648 \\\cline{2-8}
&3&297&420 & 353&500 &324&458 \\\cline{2-8}
&4&210&297 & 250&353 &229&324 \\\cline{2-8}
&5&148&210 & 176&250 &162&229 \\\cline{2-8}
&6&105&148 & 125&176 &114&162 \\\cline{2-8}
&7& 74&105 &  88&125 & 81&114 \\\cline{2-8}
\rule{1mm}{0pt}
\begin{rotate}{-90}%
    \hspace*{8mm}Классы форматов%
\end{rotate}\rule{1mm}{0pt}
&8& 52&74  &  62&88  & 57&81   \\\hline
\end{tabular}
\end{sideways}
```

Таблица 11.3. Поворот информации в окружении tabular

Так же, как в случае таблицы и соответствующего ей окружения `sidewaystable`, можно повернуть целое окружение, содержащее рисунок и подпись к нему, если вместо окружения `figure` воспользоваться окружением `sidewaysfigure`.

11.4.3 Как повернуть только подпись к рисунку

Иногда поворот целого рисунка дает не вполне ожидаемый результат. Поэтому оказывается желательным повернуть подпись и содержимое плавающего объекта независимо. Подпись к рисунку можно повернуть отдельно на 90°, если вместо команды `\caption` использовать команду `\rotcaption`.

11.5 Отчеркивания на полях

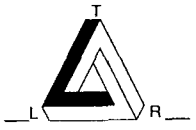
Пакет changebar, адаптированный к PostScript'у Йоханнесом Брамсом и базирующийся на более ранней работе Майкла Файна и Нила Уинтона, дает возможность отмечать подлежащие модификации части в LaTeX'овском документе, делая отчеркивания на полях в виде полосок. Этот пакет работает с большинством драйверов из-dvi-в-ps и, в частности, с драйвером dvips.

Заметьте, что как в случае со ссылками или метками, обычно вам придется обрабатывать документ дважды (а иногда и трижды) для того, чтобы отчеркива-

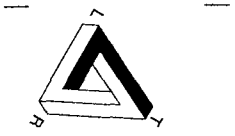
```
\begin{sidewaystable}
\centering
\begin{tabular}{|l|c|c|c|c|c|l|}
   .....
\end{tabular}
\caption[...]{....}
\label{tab:sidewaystable}
\end{sidewaystable}
```

Исходные единицы	Район изучения	Число районов				Принять или Отвергнуть Недейств. Гипотет.
		Всего	Пограничных			
			Набл.	Ожидаемых		
				От	До	
	Полная выборка	41	31	10.3	27.0	Отвергнуть
	Район выборки 1	23	16	4.3	16.7	Принять
	Район выборки 2	18	15	2.8	13.7	Отвергнуть
	Rushen	13	9	1.2	10.4	Принять
	Arbory	10	7	0.6	8.8	Принять
	Marown	10	8	0.4	8.6	Принять
	Santon	8	7	0.0	7.3	Принять

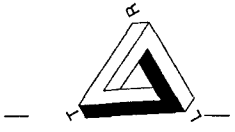
Таблица 11.4. Поворот информации в окружении `tabular` при помощи окружения `sidewaystable`
В этом случае видно, что повернуто все содержимое окружения, включая и подпись.



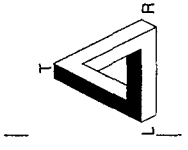

```
\newcommand{\HR}{\rule{1em}{.4pt}}%
\HR
  \epsfig{figure=Escher.eps,width=1.7cm}
\HR
```




```
\HR\begin{turn}{-240}
  \epsfig{figure=Escher.eps,width=1.7cm}
\end{turn}\HR
```




```
\HR\begin{turn}{240}
  \epsfig{figure=Escher.eps,width=1.7cm}
\end{turn}\HR
```




```
\HR\begin{sideways}
  \epsfig{figure=Escher.eps,width=1.7cm}
\end{sideways}\HR
```

Рис. 11.8. Варианты рисунка в естественном виде, в окружении `turn` и в окружении `sideways`

ния появились в нужном месте. Если требуется обработать файл еще раз, LaTeX выдаст предупреждение.

Отчеркивания можно вкладывать друг в друга. Любой уровень вложенности можно выделить полосками различной толщины. Их также можно вложить в другие окружения, включая плавающие объекты и сноски. Отчеркивается весь материал, отмеченный для выделения, включая плавающие объекты, независимо от того, куда эти объекты уплывут. Исключение этому составляют лишь плавающие объекты на полях. Полоски для корректуры могут переходить с одной страницы на другую.

11.5.1 Среда пользователя

`\driver{`*driver*`}`

Эта команда определяет драйвер, для которого должны быть порождены команды `\special`. Поддерживаемые значения параметра *driver* приводятся в пятой колонке табл. 11.2. Значением по умолчанию является `dvips`.

`\cbstart[`*barwidth*`]`

Команда `\cbstart` указывает начало той части документа, которую следует выделить полоской. Необязательный параметр *barwidth* задает ширину полоски. Если ширина не указана, то текущее значение этого параметра равно значению `\changebarwidth`. Последнюю команду можно переопределить в любой момент при помощи команды `\setlength`.

`\cbend`

Команда `\cbend` указывает конец той части документа, где будет нарисована полоска. Команды `\cbstart` и `\cbend` можно использовать где угодно, но они должны быть правильно вложены с плавающими объектами и сносками. Например, нельзя, чтобы один конец полоски находился внутри плавающей вставки, а другой — снаружи.

`\begin{changebar}`

Наряду с макрокомандами `\cbstart` и `\cbend` определено LaTeX'овское окружение `changebar`, которое дает тот же результат, что и пара `\cbstart` и `\cbend`. Преимущество в использовании этого окружения состоит в том, что LaTeX сам проделает всю работу по проверке правильности вложенности различных окружений.

`\cbdelete[`*barwidth*`]`

Команда `\cbdelete` печатает квадратную полоску на полях, указывая, что в этом месте документа некоторый текст был удален. Необязательный параметр *barwidth* задает ширину полоски. Когда аргумент не указан, используется текущее значение параметра `\deletebarwidth`. Последний параметр можно переопределить при помощи команды `\setlength`.

Это и есть текст первого абзаца. Это и есть текст первого абзаца.

Это текст второго абзаца. Это текст второго абзаца.

Это третий абзац.

Это абзац четвертый.

```
\cbstart
Это и есть текст первого абзаца.
Это и есть текст первого абзаца.\cbend

Это текст второго абзаца.
\cbdelete
Это  текст второго абзаца.

\begin{changebar}
Это  третий абзац.\par Это абзац четвертый.
\end{changebar}
```

`\nochangebars`

Команда `\nochangebars` отменяет команды, связанные с отчеркиванием.

11.5.2 Параметры команд пакета changebar

`\changebarwidth`

Ширина полосок при отчеркивании контролируется при помощи LaTeX'овского параметра длины `\changebarwidth`. Значение по умолчанию — 2pt. Это значение можно изменить при помощи команды `\setlength`. Изменение значения `\changebarwidth` повлияет на все последующие полоски в соответствии с пределами возможностей команды `\setlength`.

`\deletebarwidth`

Ширина квадратных полосок контролируется LaTeX'овским параметром длины `\deletebarwidth`. Значение по умолчанию — 4pt. Это значение можно изменить при помощи команды `\setlength`. Изменение значения `\deletebarwidth` повлияет на все последующие квадратные полоски в зависимости от пределов возможностей команды `\setlength`.

`\changebarsep`

Пространство между текстом и полосками определяется значением LaTeX'овского параметра длины `\changebarsep`. Значение по умолчанию — 35pt.

`changebargrey`

«Затемненность» полосок можно контролировать при помощи LaTeX'овского счетчика `changebargrey`. Команда типа `\setcounter{changebargrey}{85}` изменяет значение этого счетчика. Значение счетчика измеряется в процентах, где значение 0 означает черные полоски, а 100 означает белые полоски. Значение по умолчанию равно 65.

`outerbars`

Отчеркивания делаются на «внутренних» полях документа. Это означает, что они находятся на левой стороне страницы. Когда задействована опция twoside, полоски на четных страницах печатаются с правой стороны. Это положение можно изменить, включив в документ команду `\outerbarstrue`. (Эта команда будет помещать полоски на внешнем поле страницы.— *Перев.*)

11.5.3 Недостатки и неточности

- Существует предел — двадцать полосок на страницу.
- В настоящем виде пакет changebar не воспринимает печать в две колонки.
- Алгоритм отчеркиваний не работает со сносками, простирающимися на несколько страниц. Простейший путь обойти это — предотвратить разбиение сносок, но это может повлечь еще менее удовлетворительные разбивки страниц.
- Как правило, команда `\cbend` «прилипает» к символу, следующему за ней, а не к символу, который стоит перед ней. Это может привести к более

длинной полоске, чем требуется. Например, рассмотрим сочетание «word1 \cbend word2». Если между «word1» и «word2» произошел разрыв строки, то полоска неправильно растянется еще на одну строку. Данный конкретный случай можно поправить, указав «word1\cbend{}␣word2».

11.6 Обрамление и затенение

Пакет psboxit (Джерома Мэлло) помещает сгенерированный PostScript'ом бокс после TEX'овского бокса, который управляет его положением и размером. Для того чтобы задействовать PostScript-команды этого пакета, в начале запуска LATEX'а следует выполнить команду \PScommands.

\psboxit{*PScommands*}{*TEX material*}

Из первого аргумента *PScommands* команда \psboxit получает PostScript-код, который должен быть выполнен. Для получения специальных эффектов (см. примеры ниже) в рассматриваемом пакете определено несколько PostScript-процедур, таких как cartouche, rectcartouche (в переводе с французского — соответственно рамка и прямоугольная рамка) и roundedbox (в переводе с английского — закругленный бокс). Второй аргумент *TEX material* вначале набирается в боксе, прежде чем на него начнут воздействовать PostScript-команды *PScommands*.




```
---\psboxit{5 cartouche}{CCC}---
---\psboxit{rectcartouche}{\spbox{DDD}}---
---\psboxit{box .7 setgray fill}{\spbox{EEE}}---
```

Дополнительная команда \spbox работает так же, как команда \fbox, т. е. она помещает свой аргумент в бокс, но саму рамку не рисует; она всего-навсего добавляет дополнительный пробел вокруг естественных границ бокса, равный \fboxsep. Основное предназначение этой команды — затенить бокс, занимающий такую же прямоугольную область, какую дает ее эквивалент \fbox, рисующий прямоугольную рамку.

Можно легко определить более удобные команды, например:

```
\newcommand{\graybox}[1]{\psboxit{box .7 setgray fill}{\fbox{#1}}}
```

\begin{boxitpara}{*PScommands*}

Окружение boxitpara позволяет обращаться с более крупными кусками текста.

Это и есть весь текст абзаца. Это и есть весь текст абзаца. Это и есть весь текст абзаца. Это и есть весь текст абзаца.

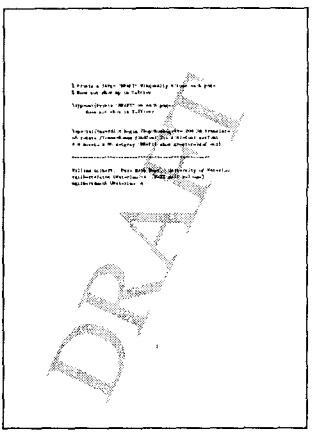
```
\begin{boxitpara}{box 0.9 setgray fill}
Это и есть весь текст абзаца.
Это и есть весь текст абзаца.
Это и есть весь текст абзаца.
Это и есть весь текст абзаца.
\end{boxitpara}
```

11.7 Цветной вывод

В прошлом цвета не были напрямую включены в LaTeX 2.09, но драйвер dvips поддерживает некоторые цветные шаблоны, имеющиеся в языке PostScript, посредством команд `\special` и пакета Джеймса Хэфнера colordvi.

В настоящее время для LaTeX 2_ε разрабатывается пакет color[2].

11.8 Наложение текста на выводимую страницу



DRAFT (Черновик)

Для включения пользовательского PostScript-кода в начале или конце документа или в конце каждой страницы (подробнее см. [68]) программа dvips предоставляет несколько приемов, например, `start-hook`, `end-hook`, `bop-hook` и `eop-hook`.

Это средство можно использовать, например, для того, чтобы вдоль каждой страницы было напечатано слово или какие-либо другие знаки. Примером может служить пакет draftcopy, который на каждой странице печатает по диагонали слово «DRAFT».

11.9 Еще одно обращение к NFSS

Совмещая возможности NFSS и механизма виртуальных шрифтов, в LaTeX'е относительно просто использовать PostScript-шрифты. У драйвера dvips имеется необходимый механизм, позволяющий использовать резидентные (в принтере) или загружаемые (с диска) PostScript'овские шрифты с LaTeX'ом.

11.9.1 Как называются эти тысячи шрифтов

Схема наименования шрифтов, которую можно использовать с TeX'ом, но которая вызывает дискуссии [50], была предложена Карлом Берри [5]. Он пытается классифицировать все имена файлов, отвечающих за шрифты, при помощи восьми символов из букв или цифр, причем регистр (нижний или верхний) не имеет значения. Это ограничение из восьми символов гарантирует, что одни и те же имена файлов можно использовать на всех компьютерных платформах (именно то, что оценят пользователи больших вычислительных машин IBM и операционной системы MS-DOS!). Принцип устройства этой схемы описан в

[2] Теперь он уже готов и лежит на CTAN:macros/latex/packages/graphics.— *Прим. ред.*

	F	TT	W	V	E	DD
	Foundry	Typeface name	Weight	Variant	Expansion	Design Size
	Происхождение шрифта	Название	Насыщенность	Начертание	Ширина	Размер
напр.:	p=PostScript	tm=Times	b=Bold	i=Italic	c=Condensed	10=10 point

Таблица 11.5. Схема классификации имен файлов, отвечающих за шрифты, согласно Карлу Берри

табл. 11.5, а табл. 11.6 показывает классификацию тридцати пяти «базовых» PostScript'овских шрифтов согласно схеме NFSS. Для каждого шрифта указывается его полное имя, данное в компании Adobe, и в скобках — соответствующее короткое имя файла по Карлу Берри. Эта схема предоставляет удобный путь связать длинные типографские имена с их укороченными эквивалентами, с которыми легко справляется драйвер dvips на различных платформах, где он установлен.

Большая часть или все шрифты из табл. 11.6 присутствуют в оперативной памяти обычного лазерного принтера. Однако еще существуют принтеры (часто старых или недорогих моделей), в которые встроены не все из этих шрифтов. Тем самым, если важна переносимость напечатанного документа, то разумно придерживаться трех базовых PostScript'овских шрифтов: `Times-Roman`, `Helvetica` и `Courier`.

Соответствие между расположением элементов PostScript'овского шрифта и кодировкой шрифта, необходимой TEX'у, осуществляется при помощи механизма виртуального шрифта, который воспринимает драйвер dvips. Соответствие между именами шрифтов внутри TEX'а и внешними (по Карлу Берри) именами файлов контролируется через файл `psfonts.map`. Драйвер dvips ищет этот файл с целью определить, следует ли включать шрифты в документ или они являются резидентными в принтере. Редактируя этот файл, можно, например, использовать исполненное в PostScript'е семейство шрифтов Computer Modern вместо растровых `.pk`-образов. Преимуществом является то, что ваш документ можно теперь напечатать на любом PostScript-принтере, не зависящем от разрешения или печатающего механизма. При больших разрешениях это особенно сильно сокращает дисковое пространство, необходимое для хранения образов шрифтов. Исполнение в PostScript'е шрифтов семейства Computer Modern было произведено компанией Blue Sky Research; компания Y&Y добавила LATEX'овские шрифты и шрифты семейств AMS и Euler.

11.9.2 Система PSNFSS

Система PSNFSS, разработанная Себастьяном Ратцем и базирующаяся на более ранней работе Крестена Торапа и Тимоти Ван Зандта, предоставляет набор файлов, обеспечивающих полную рабочую среду NFSS2, для использования со шриф-

Семейство	Насыщенность	Начертание	Полные названия
(T1, OT1)			*Семейства «с засечками»*
ptm	m	n, it	Times-Roman(ptmr), Times-Italic(ptmri)
	b	n, it	Times-Bold(ptmb), Times-BoldItalic(ptmbi)
ppl	m	n, it	Palatino-Roman(pplr), Palatino-Italic(pplri)
	b	n, it	Palatino-Bold(pplb), Palatino-BoldItalic(pplbi)
pnc	m	n, it	NewCenturySchlbk-Roman(pncr), NewCenturySchlbk-Italic(pncri)
	b	n, it	NewCenturySchlbk-Bold(pncb), NewCenturySchlbk-BoldItalic(pncbi)
pbk	m	n, it	Bookman-Light(pbkl), Bookman-LightItalic(pbkli)
	b	n, it	Bookman-Demi(pbkd), Bookman-DemiItalic(pbkdi)
(T1, OT1)			*Семейства «без засечек»*
phv	m	n, sl	Helvetica(phvr), Helvetica-Oblique(phvro)
	b	n, sl	Helvetica-Bold(phvb), Helvetica-BoldOblique(phvbo)
	c	n, sl	Helvetica-Narrow(phvrrn), Helvetica-Narrow-Oblique(phvron)
	bc	n, sl	Helvetica-Narrow-Bold(phvbrn), Helvetica-Narrow-BoldOblique(phvbon)
pag	m	n, sl	AvantGarde-Book(pagk), AvantGarde-BookOblique(pagko)
	b	n, sl	AvantGarde-Demi(pagd), AvantGarde-DemiOblique(pagdo)
(T1, OT1)			*Непропорциональный шрифт*
pcr	m	n, sl	Courier(pcrr), CourierOblique(pcrro)
	b	n, sl	Courier-Bold(pcrb), Courier-BoldOblique(pcrbo)
(U) [a]			*Особые выделительные шрифты*
psy	m	n	Symbol(psyr)
pzd	m	n	ZapfDingbats(pzdr)
pzc	m	n	ZapfChancery-MediumItalic(pzcmi)

[a] За исключением ZapfChancery, который доступен в кодировках T1 и OT1

Таблица 11.6. Классификация базовых PostScript-шрифтов согласно NFSS (в скобках — имена файлов по Карлу Берри)

Стилевой файл	Без засечек	С засечками	Пишущей машинки
times	Helvetica	Times	Courier
palatino	Helvetica	Palatino	Courier
helvet	Helvetica		
avant	AvantGarde		
newcent	AvantGarde	NewCenturySchoolbook	Courier
bookman	AvantGarde	Bookman	Courier
garamond	Optima	Garamond	Courier
basker	Univers	Baskerville	Courier
mtimes	Univers	Monotype Times	cmtt
bembo	Optima	Bembo	Courier
lucid	LucidaSans	Lucida	Courier
lucidbrb	LucidaSans	LucidaBright	LucidaSansTypewriter
lucidbry	LucidaSans	LucidaBright	LucidaSansTypewriter

Таблица 11.7. Шрифты, используемые пакетами системы PSNFSS

тами в формате PostScript. Всюду в системе PSNFSS применяется схема наименования шрифтов по Карлу Берри.

Для того чтобы изменить текстовые шрифты, используемые по умолчанию, для одного или более из типов шрифтов прямого светлого, рубленого и пишущей машинки, в стандартном режиме работы вам, вероятно, придется привлекать только один из пакетов times, newcent, helvet, palatino и т. д. В табл. 11.7 перечислены PostScript'овские шрифты, заменяющие каждый из этих трех категорий шрифтов.

В этой таблице показаны шрифтовые семейства, выбираемые для прямого светлого шрифта (шрифт документа по умолчанию), рубленого шрифта и шрифта пишущей машинки, кроме пакетов helvet и avant, которые заменяют только рубленый шрифт соответственно на шрифт Helvetica и шрифт AvantGarde. Пакеты в верхней части таблицы используют только шрифты, как правило имеющиеся в оперативной памяти лазерных PostScript-принтеров. Пакеты из нижней половины таблицы обращаются к шрифтам, которые обычно придется покупать (кроме шрифтов Courier и cmtt). Заметьте, что пакеты lucidbrb и lucidbry загружают одно и то же семейство шрифтов Lucida-Bright, но первый пакет использует имена шрифтовых файлов по Карлу Берри, а второй пакет опирается на исходные имена, определенные в дистрибутиве компании Y&Y.

Отметим, что математические шрифты не изменяются, если только не загружены другие подходящие шрифты. Если вы приобрели шрифты семейства Lucida Math fonts и получили необходимые метрические файлы, то, загрузив пакет lucmath, вы устраните из документа все обращения к шрифтам семейства cmr. Альтернативно вы можете приобрести шрифт Lucida Bright font и использовать пакет lucidbrb (lucidbry). Другим интересным пакетом является пакет pifont, в кото-

			32		33	✁	34	✂	35	✃	36	✄	37	☎	38	✆	39	✇
40	✈	41	✉	42	☛	43	☞	44	✌	45	✍	46	✎	47	✏	48	✐	49	✑
50	✒	51	✓	52	✔	53	✕	54	✖	55	✗	56	✘	57	✙	58	✚	59	✛
60	✜	61	✝	62	✞	63	✟	64	✠	65	✡	66	✢	67	✣	68	✤	69	✥
70	✦	71	✧	72	★	73	☆	74	✪	75	✫	76	✬	77	✭	78	✮	79	✯
80	✰	81	✱	82	✲	83	✳	84	✴	85	✵	86	✶	87	✷	88	✸	89	✹
90	✺	91	✻	92	✼	93	✽	94	✾	95	✿	96	❀	97	❁	98	❂	99	❃
100	❄	101	❅	102	❆	103	❇	104	❈	105	❉	106	❊	107	❋	108	●	109	❍
110	■	111	❑	112	❒	113	❑	114	❒	115	▲	116	▼	117	◆	118	❖	119	◗
120	|	121	❙	122	❚	123	❛	124	❜	125	❝	126	❞						
		161	❡	162	❢	163	❣	164	❤	165	❥	166	❦	167	❧	168	♣	169	♦
170	♥	171	♠	172	①	173	②	174	③	175	④	176	⑤	177	⑥	178	⑦	179	⑧
180	⑨	181	⑩	182	❶	183	❷	184	❸	185	❹	186	❺	187	❻	188	❼	189	❽
190	❾	191	❿	192	①	193	②	194	③	195	④	196	⑤	197	⑥	198	⑦	199	⑧
200	⑨	201	⑩	202	❶	203	❷	204	❸	205	❹	206	❺	207	❻	208	❼	209	❽
210	❾	211	❿	212	➔	213	→	214	↔	215	↕	216	➘	217	➙	218	➚	219	➛
220	➜	221	→	222	→	223	➡	224	➡	225	➡	226	➢	227	➢	228	➤	229	➥
230	➦	231	➧	232	➨	233	➩	234	➪	235	➫	236	➬	237	➭	238	➮	239	➯
		241	➱	242	➲	243	➳	244	➴	245	➵	246	➶	247	➷	248	➸	249	➹
250	➺	251	➻	252	➼	253	➽	254	➾										

Таблица 11.8. Символы PostScript-шрифта ZapfDingbats

ром определяются команды для использования с так называемыми Pi-шрифтами, т. е. с шрифтами со специальными символами, такими как ZapfDingbats или Symbol.

11.9.3 Как использовать Pi-шрифты из PostScript'а

Шрифты, содержащие наборы специальных символов, как правило не имеющихся в текстовых шрифтах, называются Pi-шрифтами. Один из таких шрифтов — шрифт ZapfDingbats из PostScript'а — станет доступным, если вы используете пакет pifont. Этот пакет является частью системы PSNFSS.

Символы из PostScript-шрифта ZapfDingbats, которые можно получить непосредственно, показаны в табл. 11.8. Нужный символ выбирается при помощи команды `\ding`. Ее аргументом является целое число, указанное в таблице рядом с нужным символом. Например, команда `\ding{38}` дает символ ✆.

Окружение dinglist является частным случаем перечня itemize. Аргументом этого окружения является число, соответствующее символу, который будет ставиться перед каждым элементом перечня.

➤ Первый элемент перечня.
➤ Второй элемент перечня.
➤ Третий элемент перечня.

```
\begin{dinglist}{228}
\item Первый элемент перечня.
\item Второй элемент перечня.
\item Третий элемент перечня.
\end{dinglist}
```

Данным символом можно заполнить целую строку, если применить команду \dingline, аргумент которой снова указывает на тот символ, который нужен. Чтобы некоторым символом заполнить оставшуюся часть строки, дайте команду \dingfill (с аргументом, смысл которого — тот же. — *Перев.*)

✂ ✂ ✂ ✂ ✂ ✂
слово слово слово слово ➽ ➽ ➽ ➽

```
\dingline{35}
\par\medskip
слово слово слово слово \dingfill{253}
```

Имеется также окружение dingautolist, которое позволяет создать нумерованный перечень, элементы которого нумеруются при помощи символов шрифта ZapfDingbats. В этом случае аргументом окружения является число, соответствующее первому нумерующему символу в списке. Последующие элементы перечня будут пронумерованы последовательно символами, следующими за указанным первым символом согласно табл. 11.8.

① Первый элемент списка.
② Второй элемент списка.
③ Третий элемент списка.

Ссылки на номера элементов списка выглядят так: ①, ②, ③

```
\begin{dingautolist}{192}
\item Первый элемент списка.
      \label{lst:zd1}
\item Второй элемент списка.
      \label{lst:zd2}
\item Третий элемент списка.
      \label{lst:zd3}
\end{dingautolist}
Ссылки на номера элементов списка
выглядят так: \ref{lst:zd1},
\ref{lst:zd2}, \ref{lst:zd3}
```

11.9.4 Общие команды в пакете pifont

Чтобы охватить Pi-шрифты, в пакете pifont предусмотрен общий механизм. Этот пакет предоставляет следующие общие команды, в которых первый аргумент *fontname* указывает (краткое по Карлу Берри) имя рассматриваемого Pi-шрифта (например, psy означает шрифт Symbol, а pzd означает шрифт ZapfDingbats; см. табл. 11.6).

\Pifont{*fontname*}

Эта команда переключает на шрифт с именем *fontname*.

`\Pisymbol{`*fontname*`}{`*numsym*`}`

Эта команда печатает символ из шрифта с именем *fontname*, находящийся в позиции, определяемой десятичным числом *numsym* (сравните с командой `\ding`).

`\Piline{`*fontname*`}{`*numsym*`}`

Эта команда печатает целую строку, состоящую из нескольких копий символа с десятичной позицией *numsym* из шрифта с именем *fontname* (сравните с командой `\dingline`).

`\Pifill{`*fontname*`}{`*numsym*`}`

Оставшаяся часть строки заполняется несколькими копиями символа с десятичной позицией *numsym* из шрифта с именем *fontname* (ср. с командой `\dingfill`).

`\begin{Pilist}{`*fontname*`}{`*numsym*`}`

Эта команда определяет окружение, порождающее именованный перечень, где перед каждым элементом перечня будет печататься символ с десятичной позицией *numsym* из шрифта с именем *fontname* (ср. с окружением `dinglist`).

`\begin{Piautolist}{`*fontname*`}{`*numsym*`}`

Эта команда определяет окружение, порождающее нумерованный перечень, где элементы перечня последовательно нумеруются символами из шрифта с именем *fontname*, первый из которых находится в десятичной позиции *numsym* (ср. это с окружением `dingautolist`).

11.9.5 Шрифт Symbol

Используя команды, описанные в предыдущем разделе, теперь легко получить доступ к символам шрифта Symbol, показанным в табл. 11.9. Например, `\Pisymbol{psy}{224}` дает ◊.

Когда требуются греческие буквы, можно воспользоваться командой `\Pifont` и обратиться к табл. 11.10 за соответствием с латинскими буквами, например,

ΑΛΦΑ ωμεγα αλφα. `{\Pifont{psy} ALFA\quad wmega\quad alfa}.`

Вы также можете создать перечень из символов шрифта Symbol следующим образом:

⇒ Первый элемент перечня.

⇒ Второй элемент перечня.

```
\begin{Pilist}{psy}{222}
  \item Первый элемент перечня.
  \item Второй элемент перечня.
\end{Pilist}
```

				32		33	!	34	∀	35	#	36	∃	37	%	38	&	39	∋
40	(	41	)	42	*	43	+	44	,	45	−	46	.	47	/	48	0	49	1
50	2	51	3	52	4	53	5	54	6	55	7	56	8	57	9	58	:	59	;
60	<	61	=	62	>	63	?	64	≅	65	A	66	B	67	X	68	Δ	69	E
70	Φ	71	Γ	72	H	73	I	74	ϑ	75	K	76	Λ	77	M	78	N	79	O
80	Π	81	Θ	82	P	83	Σ	84	T	85	Y	86	ς	87	Ω	88	Ξ	89	Ψ
90	Z	91	[	92	∴	93	]	94	⊥	95	_	96		97	α	98	β	99	χ
100	δ	101	ε	102	ϕ	103	γ	104	η	105	ι	106	φ	107	κ	108	λ	109	μ
110	ν	111	ο	112	π	113	θ	114	ρ	115	σ	116	τ	117	υ	118	ϖ	119	ω
120	ξ	121	ψ	122	ζ	123	{	124	|	125	}	126	~						
		161	ϒ	162	′	163	≤	164	/	165	∞	166	*f*	167	♣	168	♦	169	♥
170	♠	171	↔	172	←	173	↑	174	→	175	↓	176	°	177	±	178	″	179	≥
180	×	181	∝	182	∂	183	•	184	÷	185	≠	186	≡	187	≈	188	…	189	|
190	—	191	↵	192	ℵ	193	ℑ	194	ℜ	195	℘	196	⊗	197	⊕	198	∅	199	∩
200	∪	201	⊃	202	⊇	203	⊄	204	⊂	205	⊆	206	∈	207	∉	208	∠	209	∇
210	®	211	©	212	™	213	∏	214	√	215	·	216	¬	217	∧	218	∨	219	⇔
220	⇐	221	⇑	222	⇒	223	⇓	224	◊	225	⟨	226	®	227	©	228	™	229	∑
230	⎛	231	⎜	232	⎝	233	⎡	234	⎜	235	⎣	236	⎡	237	⎧	238	⎝	239	⎜
		241	⟩	242	∫	243	⌠	244	⎮	245	⌡	246	⎞	247	⎜	248	⎠	249	⎤
250	⎜	251	⎦	252	⎞	253	⎫	254	⎠										

Таблица 11.9. Символы PostScript-шрифта Symbol

a	α	b	β	c	χ	d	δ	e	ε	f	ϕ	g	γ	h	η	i	ι	j	φ	k	κ	l	λ	m	μ
n	ν	o	ο	p	π	q	θ	r	ρ	s	σ	t	τ	u	υ	v	ϖ	w	ω	x	ξ	y	ψ	z	ζ
A	Α	B	Β	C	Χ	D	Δ	E	Ε	F	Φ	G	Γ	H	Η	I	Ι	J	ϑ	K	Κ	L	Λ	M	Μ
N	Ν	O	Ο	P	Π	Q	Θ	R	Ρ	S	Σ	T	Τ	U	Υ	V	ς	W	Ω	X	Ξ	Y	Ψ	Z	Ζ

Таблица 11.10. Соответствие латинских и греческих символов в PostScript-шрифте Symbol

11.9.6 Как самостоятельно установить новые PostScript-шрифты

Если вы хотите установить новые PostScript-шрифты и создать необходимые `.fd`-файлы, то следуйте процедуре, описанной в разд. 7.7. Подходящие образцы для инсталлирования большого количества шрифтов можно найти в файле `psfonts.fdd` (охватывающем и старую TEX'овскую кодировку и EC-кодировку), а `.fd`-файл для одного шрифта легко написать вручную, если только вы знаете, какую кодировку шрифтов он использует. В качестве примера мы рассмо-

трим декларационный файл OT1ppl.fd для TEX-кодировки старого стиля шрифта Palatino (OT1):

```
% Первоначальные установки
\DeclareFontFamily{OT1}{ppl}{}
\DeclareFontShape{OT1}{ppl}{m}{n}{<->pplr}{}
\DeclareFontShape{OT1}{ppl}{m}{it}{<->pplri}{}
\DeclareFontShape{OT1}{ppl}{b}{n}{<->pplb}{}
\DeclareFontShape{OT1}{ppl}{b}{it}{<->pplbi}{}
\DeclareFontShape{OT1}{ppl}{m}{sc}{<->pplrc}{}
\DeclareFontShape{OT1}{ppl}{m}{sl}{<->pplro}{}
% Замещения
\DeclareFontShape{OT1}{ppl}{b}{sc}{<->sub * ppl/m/sc}{}
\DeclareFontShape{OT1}{ppl}{b}{sl}{<->sub * ppl/b/it}{}
\DeclareFontShape{OT1}{ppl}{bx}{n}{<->sub * ppl/b/n}{}
\DeclareFontShape{OT1}{ppl}{bx}{it}{<->sub * ppl/b/it}{}
\DeclareFontShape{OT1}{ppl}{bx}{sc}{<->sub * ppl/m/sc}{}
\DeclareFontShape{OT1}{ppl}{bx}{sl}{<->sub * ppl/m/sl}{}
\endinput
```

Как только мы указываем семейство шрифтов и кодировку, каждая комбинация насыщенности и стиля отображается на имя .tfm-файла. В случае PostScript-шрифтов нам не следует беспокоиться о том, какие имеются размеры (шрифтов), потому что эти шрифты можно масштабировать на любой размер (отсюда установки <-> в командах \DeclareFontShape). Во второй части файла приведены некоторые замещения шрифтов в тех случаях, когда шрифт отсутствует (полужирная капитель, полужирный наклонный или полужирный расширенный).

Если вы хотите создать свой собственный пакет, который можно использовать так же, как пакеты из PSNFSS, то вам следует просто воспользоваться стандартными конструкциями из NFSS2. Ниже мы приводим соответствующую часть файла times. После объявления шрифтов рубленого (Helvetica, phv), с засечками (Times-Roman, ptm) и шрифта пишущей машинки (Courier, pcr), мы полагаем, что полужирный (bold) есть «b» по умолчанию, а не «bx».

```
% Файл times.sty
\renewcommand{\sfdefault}{phv}% объявить рубленый шрифт
\renewcommand{\rmdefault}{ptm}% объявить шрифт с засечками
\renewcommand{\ttdefault}{pcr}% объявить шрифт пишущей машинки
\renewcommand{\bfdefault}{b}  % для полужирного использовать b
\endinput
```

11.9.7 Как заменить все TEX'овские шрифты PostScript-шрифтами

С помощью системы PSNFSS стало легко заменить все TEX'овские шрифты в документе на шрифты в формате PostScript.

Конечно, вы можете вначале заменить .pk-варианты семейства Computer Modern на их коммерческие эквивалентны Type 1, что легко сделать, редактируя

управляющий файл `psfonts.map` драйвера dvips. В этом случае результат можно напечатать на любых PostScript-устройствах без потери качества.

Другая возможность — это использовать шрифт типа Times-Roman как текстовый шрифт (загрузив, например, пакет times). Многие пользователи до сих пор сохраняют математическое семейство шрифтов Computer Modern, но их документы выглядят несбалансированными, поскольку многие типографские характеристики шрифтов Times-Roman, cmsy и cmmi совершенно различны (шрифты семейства cm выглядят слишком бледными и имеют также другую x-высоту). Чтобы документы выглядели визуально лучше, вы можете попробовать шрифтовой пакет *mathtime* Майкла Спивака [101]. Это семейство PostScript-шрифтов Type 1 было специально создано для набора математических текстов так, чтобы была совместимость со шрифтом Times-Roman. Алан Джеффри создал пакет pstimesm, который вместе с некоторыми макро самого Спивака замещает математические шрифты cm имеющимися в пакете *mathtime*.

Еще одно решение — это полностью заменить все TEX'овские шрифты шрифтами LucidaBright и LucidaNewMath, поставляемыми компанией Y&Y [105]. Эти шрифты были созданы Чарльзом Бигелоу и Крисом Холмсом, а сам пакет состоит из 22 шрифтов Type 1, содержащих более 4000 символов. Он включает в себя все символы для LATEX'а и $\mathcal{AMS}$-TEX'а (см. табл. 8.2–8.20, начиная со с. 255), множество дополнительных шрифтов, таких, как шрифт Calligraphic и шрифт Blackletter, ну и, конечно, TEX'овские символы. Пакет lucidbrb (или lucidbry) предоставляет все необходимые определения для полного набора вашего документа при помощи этого семейства шрифтов.

На рис. 11.9–11.11, начиная со с. 382, представлен один и тот же текст, набранный с учетом трех установок, описанных выше.

11.10 DCPS — корковская кодировка с PostScript-шрифтами

PostScript-шрифты компании Adobe не всегда содержат все символы, которые нужны для печати документов на всех языках, использующих латинский алфавит. В частности, большинство шрифтов не имеют некоторых символов корковской схемы (табл. 9.1).

Изначальная кодировка символов компании Adobe также отлична от кодировки, используемой в корковской схеме. В качестве примера рассмотрим таблицу PostScript-шрифта Helvetica (показанного в табл. 11.11). Как видно, большинство символов с акцентами, таких как à или ü, отсутствуют, поскольку они не закодированы в стандартном векторе кодировки компании Adobe [89, p. 598].

Чтобы соответствовать корковской таблице, PostScript-шрифты должны быть перекодированы. Наиболее просто это делается при помощи виртуальных шрифтов. Последние версии программ afm2tfm и dvips предоставляют средства перекодировки, так что вы можете напрямую перестроить доступные символы на свои места в расширенной таблице. Результат показан в табл. 11.12.

10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
		0	@	P	‘	p						—		
	!	1	A	Q	a	q			¡	–	̀		Æ	æ
	"	2	B	R	b	r			¢	†	́			
	#	3	C	S	c	s			£	‡	̂		ª	
	$	4	D	T	d	t			/	·	̃			
	%	5	E	U	e	u			¥		̄			ı
	&	6	F	V	f	v			ƒ	¶	̆			
	’	7	G	W	g	w			§	•	̇			
	(	8	H	X	h	x			¤	‚	̈		Ł	ł
	)	9	I	Y	i	y			'	„			Ø	ø
	*	:	J	Z	j	z			“	”	̊		Œ	œ
	+	;	K	[	k	{			«	»	̧		º	ß
	,	<	L	\	l	|			‹	…				
	-	=	M	]	m	}			›	‰	̋			
	.	>	N	^	n	~			fi		̨			
	/	?	O	_	o				fl	¿	̌			

ща 11.11. Исходная раскладка шрифта Helvetica компании Adobe

10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0
“	␣	0	@	P	‘	p	Ă	Ř	ă	ř	À	Đ	à	ð
”	!	1	A	Q	a	q	Ą	Ś	ą	ś	Á	Ñ	á	ñ
„	"	2	B	R	b	r	Ć	Š	ć	š	Â	Ò	â	ò
«	#	3	C	S	c	s	Č	Ş	č	ş	Ã	Ó	ã	ó
»	$	4	D	T	d	t	Ď	Ť	ď	ť	Ä	Ô	ä	ô
–	%	5	E	U	e	u	Đ	Ţ	đ	ţ	Å	Õ	å	õ
—	&	6	F	V	f	v	Ę	Ű	ę	ű	Æ	Ö	æ	ö
	’	7	G	W	g	w	Ğ	Ů	ğ	ů	Ç	Œ	ç	œ
	(	8	H	X	h	x	Ĺ	Ÿ	ĺ	ÿ	È	Ø	è	ø
ı	)	9	I	Y	i	y	Ľ	Ź	ľ	ź	É	Ù	é	ù
_	*	:	J	Z	j	z	Ł	Ž	ł	ž	Ê	Ú	ê	ú
ff	+	;	K	[	k	{	Ń	Ż	ń	ż	Ë	Û	ë	û
fi	,	<	L	\	l	|	Ň	IJ	ň	ij	Ì	Ü	ì	ü
fl	-	=	M	]	m	}		İ		¡	Í	Ý	í	ý
ffi	.	>	N	^	n	~	Ő	đ	ő	¿	Î	Þ	î	þ
ffl	/	?	O	_	o	-	Ŕ	§	ŕ	£	Ï	SS	ï	ß

11.12. TEX’овская раскладка DC-шрифта в применении к шрифту

1 Multiple integral signs

`\iiint`, and `\iiiint` give triple and quadruple multiple signs with the intermediate spacing nicely adjusted, in text style, for example, $\iiint_A f(x,y,z)\,dx\,dy\,dz$ and $\iiiint_A f(w,x,y,z)\,dw\,dx\,dy\,dz$, as well as in display style.

(1) $$\iiint_A f(x,y,z)\,dx\,dy\,dz$$

(2) $$\iiiint_A f(w,x,y,z)\,dw\,dx\,dy\,dz$$

2 Binomial expressions

For binomial expressions such as $\binom{n}{k}$ you have the commands `\binom`, `\dbinom`, and `\tbinom`. `\binom` is an abbreviation for `\fracwithdelims()[Opt]`.

(3) $$\begin{aligned}\sum_{\gamma\in\Gamma_C} I_\gamma &= 2^k - \binom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2} \\ &\quad + \cdots + (-1)^l\binom{k}{l}2^{k-l} + \cdots + (-1)^k \\ &= (2-1)^k = 1\end{aligned}$$

3 Split equations

The `split` environment provides no numbering because it is intended to be used only inside some other displayed equation structure, such as `equation`, `align`, or `gather`. These outer environments will provide the numbering.

(4) $$\begin{aligned}(a+b)^4 &= (a+b)^2(a+b)^2 \\ &= (a^2+2ab+b^2)(a^2+2ab+b^2) \\ &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\end{aligned}$$

Рис. 11.9. Пример распечатки страницы с использованием шрифтов Computer Modern

1 Multiple integral signs

`\iiint`, and `\iiiint` give triple and quadruple multiple signs with the intermediate spacing nicely adjusted, in text style, for example, $\iiint_A f(x, y, z)\,dx\,dy\,dz$ and $\iiiint_A f(w, x, y, z)\,dw\,dx\,dy\,dz$, as well as in display style.

(1) $$\iiint_A f(x, y, z)\,dx\,dy\,dz$$

(2) $$\iiiint_A f(w, x, y, z)\,dw\,dx\,dy\,dz$$

2 Binomial expressions

For binomial expressions such as $\binom{n}{k}$ you have the commands `\binom`, `\dbinom`, and `\tbinom`. `\binom` is an abbreviation for `\fracwithdelims()[Opt]`.

(3) $$\begin{aligned}\sum_{\gamma\in\Gamma_C} I_\gamma &= 2^k - \binom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2} \\ &\quad + \cdots + (-1)^l\binom{k}{l}2^{k-i} + \cdots + (-1)^k \\ &= (2-1)^k = 1\end{aligned}$$

3 Split equations

The `split` environment provides no numbering because it is intended to be used only inside some other displayed equation structure, such as `equation`, `align`, or `gather`. These outer environments will provide the numbering.

(4) $$\begin{aligned}(a+b)^4 &= (a+b)^2(a+b)^2 \\ &= (a^2+2ab+b^2)(a^2+2ab+b^2) \\ &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\end{aligned}$$

Рис. 11.10. Пример распечатки страницы с использованием шрифтов Mathtime

1 Multiple integral signs

\iiint, and \iiiint give triple and quadrupe multiple signs with the intermediate spacing nicely adjusted, in text style, for example, $\iiint_A f(x,y,z)\,dx\,dy\,dz$ and $\iiiint_A f(w,x,y,z)\,dw\,dx\,dy\,dz$, as well as in display style.

(1) $$\iiint_A f(x,y,z)\,dx\,dy\,dz$$

(2) $$\iiiint_A f(w,x,y,z)\,dw\,dx\,dy\,dz$$

2 Binomial expressions

For binomial expressions such as $\binom{n}{k}$ you have the commands \binom, \dbinom, and \tbinom. \binom is an abbreviation for \fracwithdelims()[Opt].

(3) $$\begin{aligned}\sum_{\gamma\in\Gamma_C} I_\gamma &= 2^k - \binom{k}{1}2^{k-1} + \binom{k}{2}2^{k-2} \\ &\quad + \cdots + (-1)^l\binom{k}{l}2^{k-l} + \cdots + (-1)^k \\ &= (2-1)^k = 1\end{aligned}$$

3 Split equations

The split environment provides no numbering because it is intended to be used only inside some other displayed equation structure, such as equation, align, or gather. These outer environments will provide the numbering.

(4) $$\begin{aligned}(a+b)^4 &= (a+b)^2(a+b)^2 \\ &= (a^2+2ab+b^2)(a^2+2ab+b^2) \\ &= a^4+4a^3b+6a^2b^2+4ab^3+b^4\end{aligned}$$

Рис. 11.11. Пример распечатки страницы с использованием шрифтов Lucida Math

ГЛАВА 12

Создание указателей

Чтобы найти материал по требуемой теме в большом документе, книге, справочнике, читатель обычно обращается к их оглавлению или, даже чаще, к указателю (предметному, именному и т. п.). Вследствие этого указатель является чрезвычайно важной частью документа и для большинства читателей поиск требуемой информации в нем начинается именно с просмотра указателя. Чтобы указатель отвечал такому назначению, необходимо спланировать его содержание, а затем сформировать его, обработав соответствующим образом текст документа. Эта операция выполняется при помощи приемов, описанных ниже, и состоит в размещении в соответствующих местах документа специальных команд, которые обеспечат согласованную печать требуемых ключевых слов в тексте и в получаемом указателе.

В данной главе вначале дается обзор основных команд формирования указателей, а также описываются средства, позволяющие получать такие указатели в виде, удобном для восприятия. В книге LaTeX book синтаксис команды `\index` описан недостаточно подробно [$\mathcal{L}$ **77–9**], [$\mathcal{A}$ **161–2**]. Ряд вопросов, связанных с построением указателей средствами TeX'а и LaTeX'а, рассматривался в статьях [3, 4, 22, 25, 77, 78, 85], помещенных в журнале *TUGboat*. Синтаксис команды `\index`, описываемой в разд. 12.1, относится к программе *MakeIndex* [47, 86], наиболее часто используемой для формирования указателей. Пользовательский интерфейс данной программы рассматривается в разд. 12.2 и 12.3.

В разд. 12.4 обсуждаются различные этапы подготовки указателя при наборе документа, а также некоторые детали форматов входных и выходных файлов, которые читает и записывает программа *MakeIndex*. Процессы интерпретации входного файла и форматирования выходного файла управляются при помощи стилевых параметров. Дается описание этих параметров, а также ряд простых примеров, показывающих, как изменение стилевых параметров влияет на вид получаемого указателя.

① При запуске LaTeX'а формируется полуфабрикат указателя (файл с расширением `.idx`).

② Полуфабрикат указателя, дополненный необязательной стилевой информацией (из стилевого файла с расширением `.ist`), используется в качестве входа для программы *MakeIndex*, создающей указатель, упорядоченный по алфавиту (файл с расширением `.ind`), и протокол (файл с расширением `.ilg`).

③ Указатель (файл с расширением `.ind`) читается LaTeX'ом при верстке окончательного варианта документа.

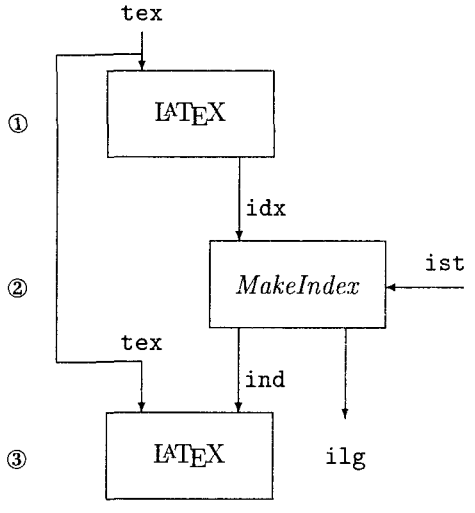


Рис. 12.1. Последовательность этапов формирования указателя и вспомогательные файлы, используемые при этом LaTeX'ом и программой *MakeIndex*

Завершающий раздел данной главы посвящен созданию нескольких указателей в одном документе; это демонстрируется на соответствующем примере.

Процесс формирования указателя схематически показан на рис. 12.1. Здесь представлены этапы работы, выполняемой LaTeX'ом и программой *MakeIndex*. Рисунок 12.2 на следующей странице дает пример различных шагов по преобразованию исходного файла в сверстанный указатель. Он показывает также несколько более детально, какие именно файлы вовлечены в процесс формирования указателя. В частности, из рис. 12.2(a) видно, в какой форме в исходный текст включаются команды `\index` (слева от этих команд указаны номера страниц, где размещены эти команды), а на рис. 12.2(b) показано, что им соответствует в полуфабрикате указателя (`.idx`-файле), формируемом LaTeX'ом. После обработки `.idx`-файла программой *MakeIndex* получается указатель с элементами, упорядоченными по алфавиту (`.ind`-файл), оформленный в виде последовательности LaTeX'овских команд (рис. 12.2(c)). Результат, полученный LaTeX'ом при завершающей верстке документа, представлен на рис. 12.2(d).

LaTeX совместно с программой *MakeIndex* использует некоторый набор соглашений, касающихся разметки документа, ориентированный на обеспечение управления видом получаемого указателя. В разд. 12.1, где описывается формат команды `\index`, для специальных (управляющих) символов везде используются значения по умолчанию (эти значения представлены в табл. 12.1 и табл. 12.2). В разделе 12.4.1 будет показано, как можно модифицировать эти символы и получаемый результат с тем, чтобы удовлетворить специфическим требованиям, возникающим у автора документа.

```
Страница vi:   \index{animal}
Страница 5:    \index{animal}
Страница 6:    \index{animal}
Страница 7:    \index{animal}
Страница 11:   \index{animalism|see{animal}}
Страница 17:   \index{animal@\emph{animal}}
               \index{mammal|textbf}
Страница 26:   \index{animal!mammal!cat}
Страница 32:   \index{animal!insect}
```

(a) Входной файл

```
\indexentry{animal}{vi}
\indexentry{animal}{5}
\indexentry{animal}{6}
\indexentry{animal}{7}
\indexentry{animalism|see{animal}}{11}
\indexentry{animal@\emph{animal}}{17}
\indexentry{mammal|textbf}{17}
\indexentry{animal!mammal!cat}{26}
\indexentry{animal!insect}{32}
```

(b) `.idx`-файл

```
\begin{theindex}
  \item animal, vi, 5-7
    \subitem insect, 32
    \subitem mammal
      \subsubitem cat, 26
  \item \emph{animal}, 17
  \item animalism, \see{animal}{11}
  \indexspace
  \item mammal, \textbf{17}
\end{theindex}
```

(c) `.ind`-файл

animal, vi, 5–7
    insect, 32
    mammal
        cat, 26
animal, 17
animalism, *see* animal

mammal, **17**

(d) Готовый указатель

Рис. 12.2. Шаги формирования указателя

12.1 Синтаксис описания элементов указателя

В этом разделе описывается синтаксис элементов указателя, принятый по умолчанию в LaTeX'е и в программе *MakeIndex*. Даются варианты описаний, возрастающие по уровню сложности, иллюстрируемые примерами соответствующих исходных файлов и получаемых результатов.

12.1.1 Простые описания элементов указателя

Каждая команда `\index` заставляет LaTeX поместить описание соответствующего элемента указателя в `.idx`-файл. Приведенный ниже пример демонстрирует ряд простых команд `\index` вместе с элементами указателя, порождаемыми ими. Номера страниц, упоминаемые в этом примере, показывают, где в документе содержалась соответствующая команда `\index`. Из этого примера видно, что если одна и та же команда (т. е. команда, указывающая на одно и то же ключевое слово) появляется на странице дважды, как это имеет место здесь для команды

`\index{stylistic}` на с. 23, в указателе ссылка на эту страницу будет дана лишь один раз.

style, 14	Страница iii:	`\index{style}`
style , 16	Страница xi:	`\index{Stylist}`
style, iii, 12	Страница 12:	`\index{style}`
style , 15		`\index{styles}`
style file, 34	Страница 14:	`\index{ style}`
styles, 12	Страница 15:	`\index{style }`
Stylist, xi	Страница 16:	`\index{ style }`
stylist, 34	Страница 23:	`\index{stylistic}`
stylistic, 23		`\index{stylistic}`
	Страница 34:	`\index{style file}`
		`\index{stylist}`

Обратите внимание на то, каким образом здесь осуществляется работа с пробелами. Пробелы внутри команд `\index` в выходной `.idx`-файл переписываются буквально. По умолчанию они рассматриваются программой *MakeIndex* как обычные символы, причем символ-пробел по порядку считается предшествующим всем буквам. В приведенном выше примере такое толкование пробела есть на с. 14, 15 и 16, где в команде `\index` имеется слово `style` с предшествующим (с. 14) и последующим (с. 16) пробелами. Слова с предшествующими пробелами помещаются в начало указателя, причем дважды, поскольку на с. 16 рассматриваемое слово имеет не только предшествующий пробел, но одновременно и завершающий, увеличивающий длину данного слова. Таким образом, слово `style` вошло в указатель четырежды, что едва ли можно признать желательным результатом. Вследствие этого важно исключить подобные избыточные пробелы из команд `\index`. Другой вариант состоит в том, что запуск программы *MakeIndex* следует выполнять с необязательным управляющим параметром-ключом `-c`. При таком варианте запуска начальные (предшествующие слову) и концевые (следующие за ним) пробелы будут подавляться (см. разд. 12.3.1). Еще одна часто встречающаяся ошибка состоит в том, что одно и то же слово упоминается в двух командах `\index` и при этом начинается в одном случае с прописной буквы, а в другом — со строчной (как слово `Stylist` на с. `xi` в приведенном выше примере). Если только это не является необходимым в самом деле, такие некорректные дублирования также надо исключать.

12.1.2 Формирование подчиненных элементов

LaTeX совместно с программой *MakeIndex* обеспечивает формирование иерархических (многоуровневых) указателей. В указателе может быть до трех уровней: верхний (main entry) и два последовательно подчиненных (subentry — первый подчиненный и subsubentry — второй подчиненный). Чтобы получить указатель такого вида, следует включить в команду `\index` как ключевое слово верхнего уровня, так и ключевые слова подчиненных уровней, разделенные знаком `!`.

Этот символ-разделитель можно переопределить в стилевом файле программы *MakeIndex* (см. табл. 12.1).

```
box, 21
    dimensions of, 33
    parameters, 5
dimensions
    figure, 12
    rule
      height, 12
      width, 3
    table, 9
```

```
Страница 3:  \index{dimensions!rule!width}
Страница 5:  \index{box!parameters}
Страница 9:  \index{dimensions!table}
Страница 12: \index{dimensions!rule!height}
             \index{dimensions!figure}
Страница 21: \index{box}
Страница 33: \index{box!dimensions of}
```

12.1.3 Диапазоны страниц и перекрестные ссылки

В указателе можно ссылаться не только на отдельные страницы, но и на диапазоны страниц. Для этой цели предназначена пара команд `\index{...|(}` и `\index{...|)}`, при помощи которых отмечается ключевое слово на первой и последней странице требуемого диапазона соответственно. Диапазоны можно задавать только для однородно пронумерованных страниц, т. е., например, для страниц, пронумерованных только арабскими или только римскими цифрами, смешивать такие нумерации нельзя. Программа *MakeIndex* умеет правильно работать в случаях, когда и начало и конец диапазона попадают на одну и ту же страницу, а также когда отдельно заданный элемент указателя попадает внутрь активного диапазона страниц.

В указателе кроме ссылок на страницы можно задавать также перекрестные ссылки одного элемента указателя на другой. Такая операция выполняется при помощи ключевого слова `see`. Поскольку элемент указателя, помеченный таким словом, не ссылается на номер какой-либо страницы, команды `\index{...|see{...}}` можно размещать в любом месте исходного файла после команды `\begin{document}`. По практическим соображениям все такие команды удобнее разместить в каком-либо одном месте документа.

```
fonts
    Computer Modern, 13–25
    math, see math, fonts
    PostScript, 5
table, ii–xi, 14
```

```
Страница ii:  \index{table|(}
Страница xi:  \index{table|)}
Страница 5:   \index{fonts!PostScript|(}
              \index{fonts!PostScript|)}
Страница 13:  \index{fonts!Computer Modern|(}
Страница 14:  \index{table}
Страница 17:  \index{fonts!math|see{math, fonts}}
Страница 21:  \index{fonts!Computer Modern}
Страница 25:  \index{fonts!Computer Modern|)}
```

12.1.4 Управление формой представления указателя

Иногда может потребоваться отсортировать элементы указателя по некоторому ключу в условиях, когда в набираемом тексте используются такие элементы, как греческие буквы, математические символы, а также фрагменты, выделяемые типографскими средствами (например, путем смены начертания или насыщенности символов в этих фрагментах). Формирование указателя с учетом этих условий возможно, если команда `\index` будет содержать выражения, построенные в соответствии со следующим шаблоном: *key@visual*, где *key* — ключ, определяющий алфавитную позицию соответствующего элемента указателя, а *visual* — текст этого элемента, помещаемый в указатель в позицию *key*.

delta, 14	Страница 5:	`\index{ninety-five}`
δ, 23	Страница 14:	`\index{delta}`
delta wing, 16	Страница 16:	`\index{delta wing}`
flower, 19	Страница 19:	`\index{flower@\textbf{flower}}`
ninety, 26	Страница 23:	`\index{delta@$\delta$}`
xc, 28		`\index{tabular@\texttt{tabular}`
ninety-five, 5		`environment}`
`tabular` environment, 23	Страница 26:	`\index{ninety}`
	Страница 28:	`\index{ninety@xc}`

В некоторых указателях отдельные номера страниц требуется выделить, в частности, использовать курсивные (например) номера, указывающие, что материал по данному вопросу, расположенный на данной странице, является основным (по сравнению с материалом по этой же теме в других местах документа), или же использовать букву *n* после номера страницы, обозначающую, что материал по данному вопросу на данной странице расположен в сноске. Программа *MakeIndex* позволяет оформлять номера страниц любым образом. Это выполняется при помощи символа-ограничителя `|`, используемого в команде `\index` (часть строки, задаваемой данной команде в качестве аргумента, после символа-ограничителя представляет собой форматирующую информацию, непосредственно не отображаемую в указателе). Например, команда `\index{keyword|xxx}` сформирует номер страницы в виде `\xxx{`*n*`}`, где *n* — номер соответствующей страницы. Аналогично, команда `\index{keyword|(xxx}` обеспечивает формирование диапазона страниц в виде `\xxx{`*n*–*m*`}`.

После символа-ограничителя `|` в тексте аргумента команды `\index` допускается применение как предопределенных команд (подобных команде `\textit` в примере, приводимом ниже), так и команд, определенных пользователем. Например, если в документе есть определение команды вида

```
\newcommand{\nn}[1]{#1n}
```

при формировании указателя будет получен следующий текст:

tabular, **ii**, *21*, 22n	Страница ii:	`\index{tabular|textbf}`
tabbing, 7, *34–37*	Страница 7:	`\index{tabbing}`
	Страница 21:	`\index{tabular|textit}`
	Страница 22	`\index{tabular|nn}`
	Страница 34:	`\index{tabbing|(textit}`
	Страница 37:	`\index{tabbing|)textit}`

Ограничитель see является частным случаем рассматриваемых средств, для которого команда \see определена в пакете makeidx.

12.1.5 Печать специальных символов

Для того чтобы в тексте элементов указателя можно было использовать символы, имеющие специальный статус с точки зрения программы *MakeIndex* (!, ", @ или |)[1], следует каждый такой символ предварять двойной кавычкой ". Если быть более точным, двойная кавычка " может предшествовать любому символу. Если при этом символ " является частью команды \", то такая запись будет соответствовать символу с умляутом. Символы !, @, " или | с двойной кавычкой " перед ними обрабатываются программой *MakeIndex* как обычные, теряя свою «специфичность». Перед тем как провести алфавитное упорядочение элементов указателя, двойная кавычка, предшествующая соответствующему символу, удаляется.

@ sign, 2		`\index{bar@\texttt{"|}|see{vertical bar}}`
|, *see* vertical bar	Страница 1:	`\index{quote (\verb+""+)}`
exclamation (!), 4		`\index{quote@\texttt{""} sign}`
Ah!, 5	Страница 2:	`\index{atsign@\texttt{"@} sign}`
Mädchen, 3	Страница 3:	`\index{maedchen@M\"{a}dchen}`
quote ("), 1	Страница 4:	`\index{exclamation ("!)}`
" sign, 1	Страница 5:	`\index{exclamation ("!)!Ah"!}`

12.1.6 Некоторые дополнительные замечания

Если команда \index используется непосредственно в тексте документа, ее аргумент раскрывается только при верстке указателя, а не тогда, когда формируется .idx-файл. Если же команда \index содержится в аргументе другой команды, символы, имеющие специальное значение в TEX'е, должны быть соответственно защищены, чтобы не произошло их некорректного использования при обработке аргумента команды \index. Это может стать определенной проблемой, если создаваемые элементы указателя содержатся в сноске или в командах, помещающих свой аргумент как в текст, так и в указатель (подобная ситуация обсуждается в следующем разделе). Даже в этом случае «нормальные» команды мож-

[1] Как уже отмечалось ранее, вместо этих символов, применяемых по умолчанию, можно использовать и другие. Способ их переопределения рассматривается на с. 404.

но размещать в @-части элемента указателя, как, например, в таком элементе: \index{rose@{\it rose}}, однако «хрупкие» команды следует защитить при помощи команды \protect.

Если в тексте указателя содержится команда, например, \index{Prog}, весьма вероятно, что формируемый ею элемент указателя будет отсортирован неверно, поскольку в основном тексте этот элемент будет сортироваться по ключу \Prog (со специальным символом \ в качестве начального в обрабатываемой строке), независимо от определения программы \Prog. С другой стороны, если \Prog использовать в качестве аргумента в другой команде, она будет раскрыта (т. е. выполнена соответствующая подстановка) еще до того, как она будет помещена в .idx-файл и содержимое указателя будет тогда зависеть от того, каким образом раскрыта команда \Prog.

Для аргумента любой из команд должно соблюдаться правило баланса скобок (равенство числа открывающихся скобок числу закрывающихся). Однако способность команды \index позволять использовать в ее аргументе символы, подобные символам % или \, в случае, когда эта команда размещена в основном тексте, может привести к возникновению аномальных ситуаций, поскольку скобки в командах \{ или \} будут учитываться при подсчете соответствующего баланса. Поэтому нельзя писать команды наподобие \index{\{}.

При сортировке программа *MakeIndex* предполагает, что страницы, пронумерованные римскими цифрами, предшествуют страницам, номера которых состоят из арабских цифр. Считается, что страницы, которые «пронумерованы» буквами, идут перед страницами, помеченными римскими цифрами. Этот порядок может быть изменен (см. элемент page-precedence в табл. 12.2).

Элементы указателя, в которых ключевые слова начинаются с символов, отличных от букв и цифр, программа *MakeIndex* размещает перед теми, которые начинаются с букв и цифр. Элементы с начальным символом, не являющимся алфавитно-цифровым, сортируются в порядке, задаваемом ASCII-кодами этих символов. При сортировке слов буквы в верхнем и нижнем регистрах (т. е. прописные и строчные соответственно) считаются идентичными, за исключением случая, когда есть два варианта одного и того же слова, одно из которых начинается с прописной буквы (этот элемент будет поставлен первым из двух рассматриваемых), а второе — со строчной. Числа сортируются в соответствии с порядком, принятым для числовых множеств. Пробелы обрабатываются как обычные символы во всех случаях, когда надо определить алфавитный порядок следования элементов указателя, а также в тех случаях, когда надо решить, совпадают ли между собой какие-либо два элемента (см. также пример на с. 388). Таким образом, если "␣" означает символ пробела, то команды \index{cat}, \index{␣cat} и \index{cat␣} сформируют три различных элемента указателя. При печати все эти три элемента будут выглядеть совершенно одинаково. Аналогично, команды \index{a␣space} и \index{a␣␣space} дадут два различных элемента указателя, которые на печати опять же будут выглядеть одинаково. Следовательно, важно уметь обнаруживать такие излишние пробелы, которые могут появиться, напри-

мер, если аргумент команды \index записан во входном файле с переносом с одной строки на другую.

Для улучшения вида элементов указателя и локализации возможных проблем полезен будет пакет showidx (автор — Лесли Лэмпорт). Он отображает все команды \index на полях печатаемой страницы. На рис. 12.3 и 12.4 показан исходной файл и соответствующий ему сформированный текст для небольшого LaTeX'овского документа. Здесь демонстрируются различные несложные возможности команд \index вместе с результатом обработки создаваемого документа пакетом showidx. С целью обеспечения согласованности элементов указателя (см. следующий раздел) были определены и использованы команды \Com и \Prog.

Для того чтобы результат верстки указателя разместить на одной странице, было переопределено окружение theindex, обеспечивающее формирование указателя (см. разд. 12.5, где показано, как выполняется эта операция).

12.1.7 Согласованность элементов указателя

Как уже отмечалось во введении, очень важно использовать одно и то же визуальное представление для идентичных имен или команд на протяжении всего документа, включая и указатель. Можно для этого определить пользовательские команды, которые обеспечат единство обозначений в тексте и указателе.

Например, можно определить команду \Index, аргумент которой будет помещен как в текст, так и в указатель.

```
\newcommand{\Index}[1]{#1\index{#1}}
```

Как объяснялось в начале предыдущего раздела, необходимо следить, чтобы аргумент таких команд не содержал раскрываемых элементов (обычно это управляющие последовательности) и избыточных пробелов. В простых случаях, например, для изолированных слов, этот подход вполне приемлем. Можно даже пойти дальше, определив некоторое специальное визуальное представление для элементов указателя, например, потребовать, чтобы он печатался шрифтом пишущей машинки.

```
\newcommand{\Indextt}[1]{\texttt{#1}\index{#1@\texttt{#1}}
```

Наконец, можно сгруппировать некоторые термины в указателе, определив команды, имеющие общее значение. Например, LaTeX'овские команды и имена программ можно было бы обрабатывать при помощи специальных команд, подобных таким, как

```
\newcommand{\bs}{\symbol{'134}}% print backslash
\newcommand{\Com}[1]{\texttt{\bs#1}\index{#1@\texttt{\bs#1}}}
\newcommand{\Prog}[1]{\texttt{#1}\index{#1@\texttt{#1} program}}
```

Команда \Com добавляет символ "\" к имени команды (аргумент команды \Com) как в основном тексте, так и в указателе, что упрощает работу наборщика.

В то же время команды, внесенные в указатель, будут упорядочены по их именам, игнорируя символ "\".

Команда `\Prog` позволяет сгруппировать имена программ в указателе отдельно. Это связано с тем, что имя программы в `\Prog` определено при помощи команды `\texttt`, а с точки зрения *MakeIndex* команды `\index{\texttt{key}}` и `\index{key}` (порождающие элементы указателя) формируют различные элементы указателя.

12.2 Подготовка указателя

12.2.1 Формирование полуфабриката указателя

После того как при помощи приемов, описанных выше, в текст документа внесены все необходимые команды `\index`, можно сформировать указатель, который будет включен в состав документа при последующих запусках LaTeX'а.

Пусть основной файл (исходный) с текстом документа поименован как `main.tex`. Для формирования указателя в этот файл необходимо внести следующие изменения:

- при помощи команды `\usepackage` подключить пакет makeidx;
- в преамбулу документа включить команду `\makeindex` [$\mathcal{L}$ 78], [$\mathcal{A}$ 161];
- вставить команду `\printindex` в то место документа, где должен появиться указатель — обычно это конец документа, непосредственно перед командой `\end{document}`.

Теперь надо запустить LaTeX для обработки всего документа в целом, в результате чего будет сформирован файл `main.idx`, который далее будем именовать idx-файлом.

12.2.2 Получение форматированного указателя

Чтобы получить форматированный указатель, следует запустить программу *MakeIndex* с именем соответствующего idx-файла в качестве параметра:

```
makeindex main.idx
```

В результате получим файл `main.ind`, который будем называть также ind-файлом. Если программа *MakeIndex* не выдала сообщений об ошибках, можно теперь запустить LaTeX, и в документе появится указатель. (Если переформатирование указателя в дальнейшем не предполагается, команду `\makeindex` из входного файла можно убрать.) Интерпретация сообщений об ошибках, которые могут быть получены от программы *MakeIndex*, см. в разд. 12.3.2.

При просмотре полученного указателя могут обнаружиться дополнительные ошибки. Их можно устранить, внося соответствующие исправления в команды `\index`, содержащиеся в тексте документа, и переформирования ind-файла.

Ниже приведен пример протокола использования программы *MakeIndex*. Упоминаемый здесь `.idx`-файл (`main.idx`) получен при первом запуске LaTeX'а для обработки исходного файла, показанного на рис. 12.3. Из протокола видно, что *MakeIndex* сформированы два файла, а именно, упорядоченный `.ind`-файл (`main.ind`) указателя, предназначенный для использования при последующем запуске LaTeX'а, а также `.ilg`-файл (`main.ilg`), в котором содержится протокол работы программы *MakeIndex*, который (в данном случае) содержит тот же самый текст, что и в выдаче на терминал, показанной выше. Если имели место ошибки, то в `.ilg`-файл будут записаны номера строк, вызвавшие эти сообщения, а также тексты соответствующих сообщений об ошибках. На рис. 12.4 показан результат запуска LaTeX'а с `.ind`-файлом, полученным после завершения программы *MakeIndex*.

```
> makeindex main
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file main.idx....done (8 entries accepted, 0 rejected).
Sorting entries....done (24 comparisons).
Generating output file main.ind....done (19 lines written, 0 warnings).
Output written in main.ind.
Transcript written in main.ilg.
```

12.3 Запуск программы *MakeIndex*

В предыдущем разделе были показаны примеры запуска программы *MakeIndex* с использованием значений по умолчанию для ее управляющих параметров. В этом разделе описываются параметры программы *MakeIndex*, их значения, а также возможности управления работой данной программы.

12.3.1 Опции программы *MakeIndex*

Синтаксис команды запуска программы *MakeIndex* определяется следующим выражением:

`makeindex [-ciglqr] [-o` *ind*`] [-p` *no*`] [-s` *sty*`] [-t` *log*`] [`*idx0 idx1 ...*`]`

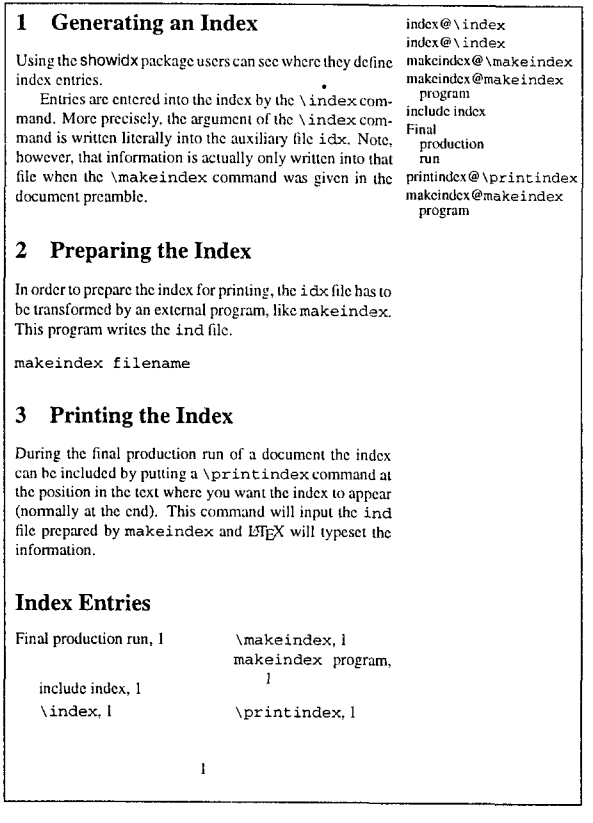
`-c` Включить режим подавления пробелов. По умолчанию каждый пробел считается значащим. При включенном параметре `-c` удаляются все предшествующие и последующие пробелы обрабатываемого ключевого термина; если в середине этого термина имеется несколько пробелов, идущих подряд, то они заменяются одним пробелом.

`-i` В качестве входного файла использовать стандартный входной файл (`stdin`). Если этот ключ активизирован, а ключ `-o` имеет значение `not`, выход программы *MakeIndex* записывается в стандартный выходной файл (`stdout`, имя выходного потока по умолчанию).

`-g` Использовать упорядочение элементов указателя в соответствии с нормами немецкого языка, описанными в стандарте ФРГ DIN 5007. В

```
\documentclass{article}
\usepackage{makeidx,showidx}
\newcommand{\bs}{\symbol{'134}}% print backslash
\newcommand{\Com}[1]{\texttt{\bs#1}%
                     \index{#1@\texttt{\bs#1}}}
\newcommand{\Prog}[1]{\texttt{#1}%
                      \index{#1@\texttt{#1} program}}
\makeindex
\begin{document}
\section{Generating an Index}
Using the \textsf{showidx} package users can see where they
define index entries.
\par Entries are entered into the index by the \Com{index}
command. More precisely, the argument of the \Com{index}
command is written literally into the auxiliary file \texttt{idx}.
Note, however, that information is actually only written into
that file when the \Com{makeindex} command was given in the
document preamble.

\section{Preparing the Index}
In order to prepare the index for printing, the \texttt{idx}
file has to be transformed by an external program, like
\Prog{makeindex}.\index{include index} This program writes the
\texttt{ind} file.
\begin{verbatim}
makeindex filename
\end{verbatim}

\section{Printing the Index}
\index{Final production run}
During the final production run of a document the index can be
included by putting a \Com{printindex} command at the position in
the text where you want the index to appear (normally at the end).
This command will input the \texttt{ind} file prepared by
\Prog{makeindex} and \LaTeX{} will typeset the information.
\printindex
\end{document}
```

Рис. 12.3. Пример размещения в тексте документа команд \index и вызова пакета showidx. Этот файл обрабатывается LaTeX'ом, затем запускается программа *MakeIndex*, после чего повторяется обработка LaTeX'ом



1 Generating an Index

Using the showidx package users can see where they define index entries.

Entries are entered into the index by the \index command. More precisely, the argument of the \index command is written literally into the auxiliary file idx. Note, however, that information is actually only written into that file when the \makeindex command was given in the document preamble.

2 Preparing the Index

In order to prepare the index for printing, the idx file has to be transformed by an external program, like makeindex. This program writes the ind file.

```
makeindex filename
```

3 Printing the Index

During the final production run of a document the index can be included by putting a \printindex command at the position in the text where you want the index to appear (normally at the end). This command will input the ind file prepared by makeindex and LaTeX will typeset the information.

Index Entries

Final production run, 1
include index, 1
\index, 1
\makeindex, 1
makeindex program, 1
\printindex, 1

1

index@\index
index@\index
makeindex@\makeindex
makeindex@makeindex program
include index
Final production run
printindex@\printindex
makeindex@makeindex program

Рис. 12.4. Указатель, сформированный при обработке текста, показанного на рис. 12.3. Все элементы указателя выведены на поле страницы, что облегчает их проверку, в частности обнаружение дублей

этом случае обычный порядок предшествования, принятый в программе *MakeIndex* (символы, отличные от цифр и букв; числа; прописные буквы; строчные буквы), будет заменен на порядок, соответствующий стандарту DIN 5007 (символы, отличные от цифр и букв; прописные буквы; строчные буквы; числа). Кроме того, при работе с активным ключом `-g` программа *MakeIndex* будет распознавать команды немецкой версии TEX'а (`"a`, `"o`, `"u`, `"s`; см. разд. 9.2.2) как `ae`, `oe`, `ue` и `ss` соответственно при выполнении операции упорядочения элементов указателя. Символ " (двойная кавычка) в этом случае необходимо переопределить в стилевом файле (см. с. 404), иначе будут выдаваться сообщения об ошибках и *MakeIndex* прервет свою работу. Следует отметить, что не все версии программы *MakeIndex* поддерживают работу с ключом `-g`.

`-l` Использовать побуквенное упорядочение; по умолчанию используется пословное упорядочение, при котором пробел считается символом, предшествующим всем буквам алфавита. При побуквенном упорядочении пробелы игнорируются. Например, элементы указателя `point in space` и `pointing` для этих двух видов упорядочения будут размещены поразному.

`-q` Работать в «тихом» режиме. В выходной поток для ошибок (`stderr`) в этом режиме сообщения выдаваться не будут. По умолчанию, сообщения о ходе работы программы, как и сообщения об ошибках, посылаются как в файл `stderr`, так и в файл протокола. Активизация ключа `-q` отключает выдачу сообщений в `stderr`.

`-r` Отключить неявное формирование диапазонов страниц. По умолчанию, три или более подряд идущих номера страниц автоматически заменяются в указателе на диапазон страниц (например, 1–5). Активизация ключа `-r` отключает эту операцию, оставляя только явное формирование диапазонов при помощи соответствующих операторов.

`-o` *ind* Использовать файл с расширением `.ind` в качестве выходного для формирования указателя. По умолчанию, в качестве имени выходного файла берется базовое имя *idx0*, которое сцепляется с расширением `.ind`.

`-p` *no* Задать номер начальной страницы выходного файла, содержащего указатели, равным *no*. Это может оказаться полезным, если файл указателя форматируется отдельно от основного текста документа. Кроме чисто числового варианта, параметр *no* имеет также три символьных значения: `any`, `odd` и `even`. При таких значениях данного параметра номер начальной строки определяется путем извлечения номера последней страницы из `.log`-файла, полученного при последнем запуске LATEX'а. Имя `.log`-файла определяется сцеплением базового имени файла для первого файла с полуфабрикатом указателя (*idx0*) и расширения `.log` . Последняя страница исходного текста определяется путем обратного поиска в `.log`-файле до того момента, как встретится номер, заключенный в квадратные скобки. Если такой номер не найден или же отсутствует `.log`-файл, попыт-

ки установления номера начальной страницы прекращаются. Каждый из упоминавшихся выше специальных случаев означает следующее:

`any` Номер начальной страницы указателя равен номеру последней страницы основного текста плюс один.

`odd` Номер начальной страницы указателя равен номеру первой нечетной страницы, следующей за последней страницей основного текста.

`even` Номер начальной страницы указателя равен номеру первой четной страницы, следующей за последней страницей основного текста.

`-s` *sty* Использовать *sty* в качестве стилевого файла. Значение по умолчанию для имени стилевого файла отсутствует. Переменная окружения `INDEXSTYLE` указывает, где размещен стилевой файл.

`-t` *log* Использовать *log* в качестве файла протокола работы программы *MakeIndex*. По умолчанию, в качестве имени протокольного файла берется базовое имя *idx0*, которое сцепляется с расширением `.ilg`.

12.3.2 Сообщения об ошибках

Программа *MakeIndex* выдает на терминал число прочитанных и выведенных строк, а также количество обнаруженных ошибок. Сообщения, идентифицирующие ошибки, записываются в файл протокола, который по умолчанию имеет расширение `.ilg`. Сообщения об ошибках при работе программы *MakeIndex* могут появиться как на этапе чтения `.idx`-файла, так и на этапе формирования `.ind`-файла. Каждое сообщение об ошибке состоит из текста, кратко описывающего суть ошибки, а также номера строки файла, где эта ошибка имела место. На этапе чтения данный номер строки относится к `.idx`-файлу.

Ошибки на этапе чтения

`Extra '!' at position ...`
Лишний '!' в

Аргумент команды `\index` содержит более одного символа ! без предшествующей ему двойной кавычки. По-видимому, надо поставить такую кавычку перед всеми «лишними» символами ! .

`Extra '@' at position ...`
Лишний '@' в

Аргумент команды `\index` содержит более одного символа @ без предшествующей ему двойной кавычки и разделяющего символа ! . По-видимому, надо поставить такую кавычку перед одним из символов @.

`Extra '|' at position ...`
Лишний '|' в

Аргумент команды `\index` содержит более одного символа | без предшествующей ему двойной кавычки. По-видимому, надо поставить такую кавычку перед всеми «лишними» символами | .

`Illegal null field`
Незаконное пустое поле
: Аргумент команды `\index` не имеет смысла, поскольку одна из строк, которые должны входить в него, является пустой. Такого типа ошибку даст команда `\index{!funny}`, поскольку в ней содержится описание подчиненного элемента "funny", но при этом отсутствует описание элемента указателя верхнего уровня. Аналогично, неверной будет и команда `\index{@funny}`, так как в качестве объекта алфавитного упорядочения указывает пустую строку.

`Argument ... too long (max 1024)`
Аргумент ... слишком длинный (max 1024)
: Документ содержит команду `\index` со слишком длинным аргументом. Не исключено, что просто забыта правая скобка, являющаяся признаком завершения аргумента данной команды.

Другие ошибки

Программа *MakeIndex* может выдавать и другие сообщения об ошибках, свидетельствующие о том, что в `.idx`-файле что-то не в порядке. Если появилось подобное сообщение, это скорее всего значит, что `.idx`-файл был каким-либо образом поврежден. Едва ли в такой ситуации причина заключается в неправильном формировании этого файла, если при его обработке LaTeX не выдал никаких сообщений об ошибках. Если же такие сообщения были, то следует проанализировать `.idx`-файл, чтобы выявить причину появления сообщения об ошибке.

Ошибки на этапе записи

`Unmatched range opening operator`
Нет парного к открывающему оператору
: Команда `\index{...|(}` не имеет парной и следующей за ней команды `\index{...|)}`. Часть аргумента "..." у обеих этих команд, стоящая перед символом-разделителем `|`, должна быть совершенно одинакова.

`Unmatched range closing operator`
Нет парного к закрывающему оператору
: Команда `\index{...|)}` не имеет предшествующей ей парной команды `\index{...|(}`.

`Extra range opening operator`
Лишний открывающий оператор
: В документе появились подряд две команды `\index{...|(}`, между которыми нет команды `\index{...|)}`.

`Inconsistent page encapsulator ... within range`
Несогласованность инкапсулятора страниц ... внутри диапазона
: Программа *MakeIndex* получила команду включить в указатель диапазон страниц для некоторого элемента и одновременно команду о форматиро-

вании этого же самого элемента на одной из страниц, попадающих в организуемый диапазон. Такая ситуация возникнет, например, если команду `\index{cat|see{animals}}` в тексте документа поместить между командами `\index{cat|(}` и `\index{cat|)}`.

`Conflicting entries`
Конфликтующие элементы

По мнению программы *MakeIndex*, ее заставляют напечатать один и тот же номер страницы дважды, задавая две разные команды, например, при помощи последовательности команд `\index{lion|see{...}}` и `\index{lion}`, расположенных на одной и той же странице.

12.4 Изменение вида указателя

Программа *MakeIndex* не ориентирована жестко на какой-либо фиксированный формат исходного и выходного файлов, этот формат можно менять, подстраивая его с учетом требований той или иной конкретной задачи. Такая подстройка осуществляется при помощи стилевого файла, управляющего работой *MakeIndex*; обычно этот файл имеет расширение `.ist` (см. рис. 12.1). Стилевой файл состоит из пар «ключевое слово / значение». Ключевые слова в стилевом файле разделяются на входные и выходные стилевые параметры. Входные стилевые параметры (они используются для программирования в исходном файле), а также их значения по умолчанию описываются в табл. 12.1. Из материала, содержащегося в данной таблице, видно, например, как можно изменить вид символа-разделителя уровней указателя (он задается значением параметра `level`, по умолчанию равным `!`).

Ключевые слова, показанные в табл. 12.2, управляют процессом трансляции входной информации в LaTeX'овские команды. Из этой таблицы видно, в частности, как задать способ форматирования различных уровней указателя (используя серии ключевых слов `item`). Более подробно познакомиться с тем, как на практике использовать различные входные и выходные ключевые слова, можно при помощи приводимых ниже примеров. В стилевых файлах программы *MakeIndex* для записи символьных строк используется синтаксис, принятый в системе UNIX, а именно, чтобы получить `\` в выходном файле, во входном файле должно быть записано `\\`.

12.4.1 Пример стилевых файлов указателя

В следующих разделах будет показано, каким образом можно изменить вид формируемого указателя, внося совсем небольшие изменения в значения ключевых слов, первоначально определенных по умолчанию.

Ключевое слово	*Значение по умолчанию*	*Описание*
`keyword` (*s*)	`"\\indexentry"`	Команда, сообщающая программе *MakeIndex*, что ее аргумент является элементом указателя.
`arg_open` (*c*)	`'{'`	Левый ограничитель для аргумента.
`arg_close` (*c*)	`'}'`	Правый ограничитель для аргумента.
`range_open` (*c*)	`'('`	Ограничитель, показывающий начало отсчета диапазона страниц.
`range_close` (*c*)	`')'`	Ограничитель, показывающий конец отсчета диапазона страниц.
`level` (*c*)	`'!'`	Ограничитель, отмечающий новый уровень подчиненного элемента указателя.
`actual` (*c*)	`'@'`	Символ, показывающий, что следующий за ним элемент должен появиться в реальном файле указателя.
`encap` (*c*)	`'|'`	Символ, показывающий, что остающаяся часть списка аргументов должна быть использована как команда для формирования номера страницы.
`quote` (*c*)	`'"'`	Символ, сохраняющий литеру, следующую за ним.
`escape` (*c*)	`'\\'`	Символ без специального значения, если только за ним не следует литера `quote`. В таком случае данная литера теряет свою специальную функцию и оба этих символа будут напечатаны. Такая возможность задействована вследствие того, что `\"` представляет собой в TEX'е команду для формирования умляута. Отмеченные два символа `quote` и `escape` должны быть отличны друг от друга.
`page_compositor` (*s*)	`"-"`	Ограничитель составных страниц.

(*s*) — атрибут типа «цепочка», (*c*) — атрибут типа «литера» (заключаются в двойные и в одинарные кавычки соответственно).

Таблица 12.1. Входные стилевые параметры для программы *MakeIndex*

Ключевое слово	*Значение по умолчанию*	*Описание*
Контекст		
`preamble` (*s*)	`"\\begin{theindex}\n"`	Команда начала формирования указателя.
`postamble` (*s*)	`"\n\n\\end{theindex}\n"`	Команда завершения формирования указателя.
Начальная страница		
`setpage_prefix` (*s*)	`"\n\\setcounter{page}{"`	Префикс для команды установки страницы.
`setpage_suffix` (*s*)	`"}\n"`	Суффикс для команды установки страницы.
Новая группа/буква		
`group_skip` (*s*)	`"\n\n\\indexspace\n"`	Промежуток по вертикали перед новой группой.
`heading_prefix` (*s*)	`""`	Префикс для заголовка новой буквенной группы.
`heading_suffix` (*s*)	`""`	Суффикс для заголовка новой буквенной группы.
`headings_flag` (*n*)	0	При значении `flag=0` между различными буквенными группами не вставляется ничего; значение `flag>0` (`<0`) включает прописные (строчные) варианты символов, характеризующих новую буквенную группу, префикс для которой задается параметром `heading_prefix`, а суффикс — параметром `heading_suffix`
Разделители для элементов указателя		
`item_0` (*s*)	`"\n\\item "`	Команда, которая должна быть вставлена перед элементом уровня 0.
`item_1` (*s*)	`"\n \\subitem "`	То же самое для элемента указателя уровня 1, начинающегося на уровне ≥ 1.
`item_2` (*s*)	`"\n \\subsubitem "`	То же самое для элемента указателя уровня 2, начинающегося на уровне ≥ 2.
`item_01` (*s*)	`"\n \\subitem "`	Команда перед элементом уровня 1, начинающегося на уровне 0.
`item_12` (*s*)	`"\n \\subsubitem "`	То же самое для элементов уровня 2, начинающихся на уровне 1.

окончание см. на след. стр.

Таблица 12.2. Выходные стилевые параметры для программы *MakeIndex*

окончание (см. предыдущую страницу)		
Ключевое слово	*Значение по умолчанию*	*Описание*
`item_x1` (*s*)	`"\n \\subitem "`	Команда, которая должна быть вставлена перед элементом уровня 1, если родительский уровень не имеет номеров страниц.
`item_x2` (*s*)	`"\n \\subsubitem "`	То же самое для уровня 2.
`delim_0` (*s*)	`", "`	Ограничитель (разделитель) между элементом указателя и первым номером страницы на уровне 0.
`delim_1` (*s*)	`", "`	То же самое для уровня 1.
`delim_2` (*s*)	`", "`	То же самое для уровня 2.
`delim_n` (*s*)	`", "`	Ограничитель (разделитель) между различными номерами страниц.
`delim_r` (*s*)	`","`	Описатель диапазона страниц.
Инкапсуляторы страниц		
`encap_prefix` (*s*)	`"\\"`	Префикс для использования перед инкапсулятором страниц.
`encap_infix` (*s*)	`"{"`	Инфиксный символ для инкапсулятора страниц.
`encap_suffix` (*s*)	`"}"`	Суффикс для инкапсулятора страниц.
Упорядочение номеров страниц		
`page_precedence` (*s*)	`"rRnaA"`	Порядок, в котором будут перечисляться номера страниц различных видов: `a`, `A` — строчные и прописные буквы; `n` — арабские цифры; `r`, `R` — строчные и прописные римские цифры.
Перенос строк		
`line_max` (*n*)	72	Максимально допустимая длина выходной строки.
`indent_space` (*s*)	`"\t\t"`	Команда включения отступа для перенесенных строк.
`indent_length` (*n*)	16	Величина отступа для перенесенных строк.

«\n» и «\t» — команды перехода на новую строку и табуляции, соответственно; (*s*) — атрибут имеет тип «строка»; (*n*) — атрибут числового типа.

Таблица 12.2.

12.4.2 Указатель, формируемый отдельно

Стилевой файл `mybook.ist`, показанный в примере, приводимом ниже, определяет указатель для книги, формируемый отдельно. Термин «формируемый отдельно» означает здесь, что указатель форматируется независимо от основного текста книги. Это может оказаться полезным, например, в случае, когда исходный файл с текстом книги «заморожен» (т. е. номера страниц меняться уже не будут) и требуется только переформатировать указатель.

```
% MakeIndex style file mybook.ist
preamble
"\\documentclass[12pt]{book}
\\begin{document}
\\begin{theindex}\n"
postamble
"\n\n\\end{theindex}
\\end{document}\n"
```

Пусть команды полуфабриката указателя находятся в файле `mybook.idx`, тогда команда вызова программы *MakeIndex* со стилевым файлом `mybook.ist` имеет вид

```
makeindex  -s mybook.ist -o mybookind.tex mybook
```

Имя выходного файла, отличное от его значения по умолчанию, используется здесь для того, чтобы избежать конфликта имен `.dvi`-файлов для основного текста книги (по умолчанию он будет иметь имя `mybook.dvi`) и указателя, поскольку, если указатель содержится в файле `mybook.ind`, `.dvi`-файл указателя получит имя `mybook.dvi`, что приведет к потере `.dvi`-файла для основного текста книги.

Если, кроме того, требуется чтобы нумерация страниц указателя правильно продолжала нумерацию основного текста книги (в примере, приводимом ниже, для этого необходимо, чтобы первая страница указателя имела номер 181), то команду вызова программы *MakeIndex* следует скорректировать следующим образом:

```
makeindex  -s mybook.ist  -o mybookind.tex -p 181  mybook
```

Для определения номера, который должна иметь первая страница указателя, программа *MakeIndex* может читать LaTeX'овский `.log`-файл (см. описание ключа `-p` на с. 395).

12.4.3 Изменение специальных символов

Следующий пример показывает, каким образом можно изменить интерпретацию специальных символов в исходном файле. Чтобы сделать это, надо задать в стилевом файле (например, в файле `myinchar.ist`, показанном ниже) новые специальные символы. Основываясь на данных из табл. 12.1, в следующем ниже примере

" sign, 1	Страница 1: `\index{\texttt{"} sign}`
= sign, 2	Страница 2: `\index{\texttt{@} sign}`
@ sign, 2	Страница 2: `\index{\texttt{!=} sign}`
Brücke, 5	Страница 3: `\index{Maedchen=M\"{a}dchen}`
Brücke, V	Страница c: `\index{Maedchen=M"adchen}`
Brücke, v	Страница v: `\index{Bruecke=Br"ucke}`
dimensions	Страница 5: `\index{Br"ucke}`
rule	Страница V: `\index{Br\"ucke}`
width, 3	Страница 3: `\index{dimensions>rule>width}`
exclamation (!), 4	Страница 4: `\index{exclamation (!!)}`
Ah!, 5	Страница 5: `\index{exclamation (!!)>Ah!!}`
Mädchen, c	
Mädchen, 3	

Рис. 12.5. Пример использования специальных символов при обращении к программе *MakeIndex*

изменим символ @ (см. с. 390) на = , маркер подчиненного уровня ! (см. с. 388) на >, а символ " (см. с. 391) на ! (являющийся по умолчанию маркером подчиненного уровня).

```
% MakeIndex style file myinchar.ist
actual '='      % = instead of default @
quote '!'       % !                    "
level '>'       % >                    !
```

В примере на рис. 12.5 (его следует выполнять при опции german в пакете babel) двойная кавычка " используется в качестве сокращенного обозначения для команды вида \", задающей нанесение диакритического знака (умляута) для ряда букв немецкого алфавита. Этот пример иллюстрирует некоторые особенности упорядочения элементов указателя, выполняемого программой *MakeIndex*, а именно то, что в данном случае ключевые слова, содержащие " и \", рассматриваются как различные элементы указателя (`Br"ucke` и `Br\"ucke`, `M"adchen` и `M\"adchen`, хотя в последнем случае ключевые слова считаются идентичными, т. е. `Maedchen`). Следовательно, важно использовать одни и те же соглашения в тексте всего документа.

12.4.4 Изменение выходного формата указателя

Имеется также возможность модифицировать выходной формат указателя. Первое, что мы попытаемся сделать, это внести в указатель заглавные буквы, которые предшествовали бы группам элементов указателя с ключевыми словами, начинающимися на соответствующую букву. Это делается при помощи стилевого

файла `myhead.ist`, показанного ниже (для получения более подробной информации см. табл. 12.2); получаемый результат показан на рис. 12.6.

```
% MakeIndex style file myhead.ist
heading_prefix "{\\bfseries\\hfil "     % Вставить перед буквой
heading_suffix "\\hfil}\\nopagebreak\n" % Добавить после буквы
headings_flag       1                   % Переключить на прописные
```

В указатель можно ввести еще одно усовершенствование, а именно выровнять номера страниц в указателе по правому краю столбца, в виде которого представляются элементы указателя, а промежуток между ключевым термином и номером страницы заполнять точками. Результат такой модификации указателя показан на рис. 12.7, а команды в стилевом файле, которые задают эту модификацию, имеют вид:

```
% MakeIndex style file myright.ist
delim_0   "\\dotfill "
delim_1   "\\dotfill "
delim_2   "\\dotfill "
```

LaTeX'овская команда `\dotfill`, использованная в рассмотренном примере, может быть заменена на другую (чтобы заполнять упомянутый выше промежуток не точками, а какими-либо другими символами), но принципиально при этом ничего не меняется.

12.4.5 Обработка нетрадиционных номеров страниц

Как уже отмечалось выше, программа *MakeIndex* позволяет работать с номерами страниц, составленными из арабских цифр, римских цифр (в прописном и строчном варианте), латинских букв (также в прописном и строчном варианте), причем эти элементы не должны смешиваться. Кроме этих основных видов нумерации страниц, можно также формировать и составные номера. В них используется символ-разделитель, видом которого можно управлять, используя в стилевом файле программы *MakeIndex* ключевое слово `page_compositor` (по умолчанию в качестве символа-разделителя принят дефис (`-`); см. табл. 12.1). При этом характер упорядочения различных видов номеров страниц в их списке определяется значением ключевого слова `page_precedence` (по умолчанию это значение равно `rRnaA`; см. табл. 12.2).

Начнем с простых номеров страниц и примем, что в страницах с номерами `ii`, `iv`, `1`, `2`, `5`, `a`, `c`, `A`, `C`, `II`, `IV` содержится команда `\index` со словом `style`. При значении параметра `page_precedence` по умолчанию (т. е. `rnaRA`) это слово в указателе будет отражено следующим образом:

style, ii, iv, c, 1, 2, 5, a, II, IV, C, A

Из этого примера видно, что номера страниц `c` и `C` были интерпретированы как римские цифры, а не как буквы латинского алфавита.

```
      Symbols
@ sign, 2
          B
box, 21
      dimensions of, 33
      parameters, 5
          D
dimensions
      figure, 17
      rule
        height, 12
        width, 3
      table, 9
          F
fonts
      Computer Modern, 21
      PostScript, 5
          R
rule
      depth, 33, 48
      width, 41
```

```
Страница 2:  {\texttt{"@} sign}
Страница 3:  {dimensions!rule!width}
Страница 5:  {box!parameters}
             {fonts!PostScript}
Страница 9:  {dimensions!table}
Страница 12: {dimensions!rule!height}
Страница 17: {dimensions!figure}
Страница 21: {box}
             {fonts!Computer Modern}
Страница 33: {box!dimensions of}
             {rule!depth}
Страница 41: {rule!width}
Страница 48: {rule!depth}
```

Рис. 12.6. Пример модификации выходного формата указателя

```
@ sign ..... 2
box ..... 21
      dimensions of .... 33
      parameters ..... 5
dimensions
      figure ..... 17
      rule
        height ..... 12
        width ..... 3
      table ..... 9
fonts
      Computer Modern 21
      PostScript ..... 5
rule
      depth .....33, 48
      width ..... 41
```

```
Страница 2:  {\texttt{"@} sign}
Страница 3:  {dimensions!rule!width}
Страница 5:  {box!parameters}
             {fonts!PostScript}
Страница 9:  {dimensions!table}
Страница 12: {dimensions!rule!height}
Страница 17: {dimensions!figure}
Страница 21: {box}
             {fonts!Computer Modern}
Страница 33: {box!dimensions of}
             {rule!depth}
Страница 41: {rule!width}
Страница 48: {rule!depth}
```

Рис. 12.7. Добавление отточий в указатель

Такой порядок можно изменить, если воспользоваться ключевым словом `page_precedence` и внести в стилевой файл `mypages.ist` команду

```
% MakeIndex style file mypages.ist
page_precedence "rRnaA"
```

В результате при обработке того же самого файла с заготовками элементов указателя в сформированный указатель будет внесена строка

style, ii, iv, c, II, IV, C, 1, 2, 5, a, A

На следующем шаге предположим, что в документе используются составные номера страниц. По умолчанию в качестве символа-разделителя принимается дефис. Предположим, что ссылка на слово `style` имеется в таком (неотсортированном) наборе страниц: `C-3`, `1-1`, `D-1-1`, `B-7`, `F-3-5`, `2-2`, `D-2-3`, `A-1`, `B-5`, `A-2`. После запуска программы *MakeIndex* в элементе указателя получим упорядочение последовательности страниц

style, 1-1, 2-2, A-1, A-2, B-5, B-7, C-3, D-1-1, D-2-3, F-3-5

Вид разделителя можно изменить, в частности, использовать точку вместо дефиса. Такое изменение осуществляется при помощи ключевого слова `page_compositor`. Например, можно сформировать такой стилевой файл `mypagsep.ist`:

```
% MakeIndex style file mypagsep.ist
page_compositor "."
```

Программа *MakeIndex* при запуске ее с этим стилевым файлом на тех же самых данных, что и в примере, приведенном выше, дает

style, 1.1, 2.2, A.1, A.2, B.5, B.7, C.3, D.1.1, D.2.3, F.3.5

12.4.6 Глоссарий

В LaTeX'е имеется команда `\glossary`, обеспечивающая формирование глоссария [*Л* **79**] (см. также [40, с. 398]). Команда `\makeglossary` организует файл с расширением `.glo`, подобный `.idx`-файлу для команд `\index`. LaTeX преобразует команды `\glossary` в элементы глоссария `\glossaryentry`, выполняя эту операцию точно так же, как и при трансляции команд `\index` в элементы указателя `\indexentry`.

Программа *MakeIndex* позволяет обрабатывать команды формирования глоссария, однако для этого в значения некоторых ключевых слов стилевого файла

требуется внести изменения. Пример таких изменений показан ниже в виде файла myglossary.ist.

```
% MakeIndex style file myglossary.ist
keyword   "\\glossaryentry"   % ключевое слово для элемента глоссария
preamble  "\n \\begin{theglossary}\n"  % Начать глоссарий
postamble "\n\n \\end{theglossary}\n"  % Завершить глоссарий
```

Кроме того, можно определить соответствующее окружение theglossary.

12.5 Изменение макета указателя

Можно переопределить окружение theindex [$\mathcal{L}$ 77], [$\mathcal{A}$ 160–1], используемое по умолчанию для печати указателя. Задаваемый этим окружением макет, а также определения команд \item, \subitem и \subsubitem приводятся в файлах классов article, book и report. В частности, для класса book эти определения имеют следующий вид:

```
\newenvironment{theindex}
          {\@restonecoltrue\if@twocolumn\@restonecolfalse\fi
           \columnseprule \z@  \columnsep 35\p@
           \twocolumn[\@makeschapterhead{\indexname}]%
           \@mkboth{\uppercase{\indexname}}{\uppercase{\indexname}}%
           \thispagestyle{plain}\parindent\z@
           \parskip\z@ \@plus .3\p@\relax
           \let\item\@idxitem}
          {\if@restonecol\onecolumn\else\clearpage\fi}
\newcommand\@idxitem  {\par\hangindent 40\p@}
\newcommand\subitem   {\par\hangindent 40\p@ \hspace*{20\p@}}
\newcommand\subsubitem{\par\hangindent 40\p@ \hspace*{30\p@}}
```

Имеется возможность сверстать указатель не в две колонки, а в три. Чтобы сделать это, следует воспользоваться пакетом multicol и окружением multicols.

```
\renewenvironment{theindex}{\newpage
   \addcontentsline{toc}{chapter}{Index Entries}%
   \pagestyle{plain}\let\item\@idxitem\setlength{\parindent}{0pt}
   \begin{multicols}{3}[{\Large\bfseries Index Entries}]
   \par\bigskip}%            definition of \begin{theindex}
  {\end{multicols}}%         definition of \end{theindex}
```

Из этого определения видно, что вначале выполняется переход на новую страницу, затем в .toc-файл (оглавление книги) вносится наименование раздела (Index Entries) как новой главы, после чего стиль оформления страницы меняется на plain и переопределяется команда \item. Следующий шаг — перегруппировка элементов указателя в три колонки при помощи окружения multicols.

12.5.1 Формирование нескольких указателей

Пакет multind (автор — Ф. У. Лонг) переопределяет команды \makeindex, \index и \printindex таким образом, чтобы можно было в одном документе формировать несколько указателей. Это достигается путем использования имени файла, содержащего указатель, в качестве дополнительного аргумента.

```
\makeindex{indexname}
\index{indexname}{entry}
\printindex{indexname}{indextitle}
```

Первоначально при обработке файла, показанного на рис. 12.8, LaTeX сформирует два файла с именами A.idx и B.idx. Преобразование каждого из этих файлов в .ind-форму требует отдельного запуска программы *MakeIndex*.

```
> makeindex A
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file A.idx....done (5 entries accepted, 0 rejected).
Sorting entries....done (12 comparisons).
Generating output file A.ind....done (18 lines written, 0 warnings).
Output written in A.ind.
Transcript written in A.ilg.
> makeindex B
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file B.idx....done (5 entries accepted, 0 rejected).
Sorting entries....done (14 comparisons).
Generating output file B.ind....done (21 lines written, 0 warnings).
Output written in B.ind.
Transcript written in B.ilg.
```

При следующем запуске LaTeX'а будет получен выходной файл, показанный на рис. 12.9. В этом примере было использовано окружение theindex, переопределенное так, как это было показано в разд. 12.5 при рассмотрении случая организации печати указателя в три колонки вместо стандартных двух. Заголовки для указателей задаются вторым аргументом переопределенной команды \printindex.

12.5.2 Модификация команд формирования указателей

Возможности, существующие в LaTeX'е для выполнения операций формирования указателей, расширяются по ряду направлений при помощи пакета index (автор — Дэвид Джоунз).

1. Поддерживается формирование нескольких указателей в одном документе.
2. Для формирования полуфабриката указателя (.idx-файла. по умолчанию) используется двухэтапный процесс, подобно тому как это делается при формировании файла оглавления (.toc-файла). На первом этапе элементы указателя записываются в .aux-файл, а затем, в конце сеанса обработки, они

```
\documentclass[12pt]{article}
\usepackage{multind}
\renewcommand{\printindex}[2]{% Redefine \printindex
 \input{#1.ind}\begin{center}\textbf{\large#2}\end{center}}
\renewenvironment{theindex}[1]{...}{...}
\newcommand{\bs}{\symbol{'134}}% print backslash
\newcommand{\Com}[1]{\texttt{\bs#1}%
                     \index{A}{#1@\texttt{\bs#1}}}
\newcommand{\Prog}[1]{\texttt{#1}%
                      \index{A}{#1@\texttt{#1} (program)}}
\makeindex{A}\makeindex{B}
\begin{document}
\section{Generating more than one Index}
Using the package \textsf{multind}%
\index{B}{multind package}%
users can enter information in more than one index.
The commands \Com{makeindex} and \Com{index} have been
modified to allow multiple indexes.
In both cases the first parameter is the index name.%
\index{B}{index name}

\section{New printindex command}
When you want to include the index in the document,
you should run the \Prog{makeindex} program on each file.
\index{B}{One more to B}

\begin{verbatim}
makeindex A
makeindex B
\end{verbatim}

A modified \Com{printindex} command lets you print multiple
indexes. The first parameter is the index name, the second
parameter is the index title (as printed). Some more
text\index{B}{entry index B}. The final text
\index{A}{Final to index A}\index{B}{Final to index B}.

\printindex{A}{Commands and programs}
\printindex{B}{Other stuff}
\end{document}
```

Рис. 12.8. Пример входного файла для случая нескольких указателей

1 Generating more than one Index

Using the package multind users can enter information in more than one index. The commands \makeindex and \index have been modified to allow multiple indexes. In both cases the first parameter is the index name.

2 New printindex command

When you want to include the index in the document, you should run the makeindex program on each file.

```
makeindex A
makeindex B
```

A modified \printindex command lets you print multiple indexes. The first parameter is the index name, the second parameter is the index title (as printed). Some more text. The final text.

Commands and programs

Final to index A, 1

\index, 1

\makeindex, 1
makeindex (program), 1

\printindex, 1

Other stuff

entry index B, 1

Final to index B, 1

index name, 1

multind package, 1

One more to B, 1

Рис. 12.9. Выходной файл с несколькими указателями

копируются в .idx-файл. При таком подходе, если производится формирование документа большого объема, организованного в виде набора файлов, подключаемых командой \include, в случае частичной переработки документа с использованием команды \includeonly не произойдет разрушения указателя.

3. Введен вариант «со звездочкой» (*) для команды \index. В таком варианте данная команда помещает свой аргумент не только в указатель, но и в *колонтитул*.

4. Для упрощения набора текста документа введена команда \shortindexingon, позволяющая пользоваться сокращенной записью для команд \index и \index*. Можно писать, например, ^{foo} вместо \index{foo} и _{foo} вместо \index*{foo}. Чтобы отключить эту возможность, следует воспользоваться командой \shortindexingoff. В математическом режиме работы LaTeX'а такую сокращенную запись применять нельзя, поскольку в нем символы "^" и "_" имеют специальное значение.

5. Пакет index реализует функциональные возможности, обеспечиваемые пакетом showidx. Команда \proofmodetrue реализует печать элементов указателя на полях документа. Размером и стилем шрифта, используемого при такой печати, можно управлять. Для этой цели предназначена команда \indexproofstyle, которая в качестве аргумента включает описание шрифта, например, \renewcommand{\indexproofstyle}{\footnotesize\itshape}.

Новые указатели могут быть объявлены при помощи команды \newindex, а переопределение существующих указателей осуществляется командой \renewindex.

\newindex{*tag*}{*raw-ext*}{*proc-ext*}{*indexname*}

Первый аргумент (*tag*) команды \newindex задает сокращенное имя, используемое для ссылок на указатель. Команды \index и \printindex переопределены в пакете index таким образом, что сокращенное имя входит в них в качестве необязательного аргумента, позволяя отмечать, к какому именно указателю относится данная команда. Если необязательный аргумент отсутствует, для сокращенного имени используется значение по умолчанию, соответствующее обычному указателю. Второй аргумент (*raw-ext*) в команде \newindex представляет собой расширение для файла, в который LaTeX будет записывать необработанные элементы указателя (по умолчанию этот параметр имеет значение .idx). Третий аргумент (*proc-ext*) задает расширение файла, в котором LaTeX будет искать набор обработанных элементов указателя (по умолчанию этот параметр имеет значение .ind). Наконец, четвертый параметр (*indexname*) команды \newindex содержит текст заголовка, который будет напечатан LaTeX'ом перед началом указателя.

Команды запуска программы *MakeIndex* и LaTeX'а при формировании нескольких указателей применительно к входному файлу, показанному на

```
\documentclass{book}
\usepackage{index}
\makeindex
\newindex{aut}{adx}{and}{Name Index}
\newindex{not}{ndx}{nnd}{List of Notation}
\shortindexingon
\proofmodetrue
\newcommand{\aindex}[1]{\index*[aut]{#1}}
\begin{document}
\tableofcontents
\newpage
\chapter{Here is a ^[aut]{chapter} title}
\section{Section header\index[aut]{section}}
\par Here is some text.\index{subject}
\par Here is \index[not]{notation}some
      more\index[not]{sin@$\sin$} text.
\par Here is some ^{more} _[not]{notation} text.
\par Here is yet more \aindex{text}.

\section{Another Section header _[aut]{section2}}

\par And here is some math: $x^1_b$.
\par Here is an ^[aut]{index} entry
      \fbox{inside an \index[not]{min@$\min$}fbox}
\par \fbox{Here is an ^[aut]{entry} in a box.}

\printindex[not]
\printindex[aut]
\printindex
\end{document}
```

Рис. 12.10. Входной файл для случая нескольких указателей

рис. 12.10, представлены на рис. 12.11 вместе с соответствующими протоколами. Видно, что программа *MakeIndex* обработала по отдельности три полуфабриката указателей, вначале стандартный, затем `.adx`-файл, что привело к формированию `.and`-файла, а после него `.ndx`-файл, с помещением результатов его обработки в `.nnd`-файл. После этого LaTeX был запущен повторно и прочитал перечисленные файлы в порядке, предписанном командой `\printindex` в тексте исходного файла (показанного на рис. 12.10). Результат этого (завершающего) этапа обработки документа показан на рис. 12.12.

```
> latex multindexa.tex
This is TeX, C Version 3.141
(multindexa.tex
LaTeX Version 2.09 <25 March 1992>
(/usr/local/lib/tex/macros/book.sty
Standard Document Style 'book' <14 Jan 92>.
(/usr/local/lib/tex/macros/bk10.sty)) (index.sty
Style-Option: 'index' v3.01 <19 July 1993> (dmj))
index.sty> Writing index file multindexa.idx
index.sty> Writing index file multindexa.adx
index.sty> Writing index file multindexa.ndx
No file multindexa.aux.
No file multindexa.toc.
[1] [2]
Chapter 1.
No file multindexa.nnd.
No file multindexa.and.
No file multindexa.ind.
[3] (multindexa.aux) )
Output written on multindexa.dvi (3 pages, 1556 bytes).
Transcript written on multindexa.log.
> makeindex multindexa
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file multindexa.idx....done
         (2 entries accepted, 0 rejected).
Sorting entries....done (2 comparisons).
Generating output file multindexa.ind....done
         (9 lines written, 0 warnings).
Output written in multindexa.ind.
Transcript written in multindexa.ilg.
> makeindex -o multindexa.and multindexa.adx
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file multindexa.adx....done
         (6 entries accepted, 0 rejected).
Sorting entries....done (20 comparisons).
Generating output file multindexa.and....done
         (22 lines written, 0 warnings).
Output written in multindexa.and.
Transcript written in multindexa.ilg.
> makeindex -o multindexa.nnd multindexa.ndx
This is makeindex, portable version 2.12 [26-May-1993].
Scanning input file multindexa.ndx....done
         (4 entries accepted, 0 rejected).
Sorting entries....done (9 comparisons).
Generating output file multindexa.nnd....done
         (13 lines written, 0 warnings).
Output written in multindexa.nnd.
Transcript written in multindexa.ilg.
> latex multindexa
This is TeX, C Version 3.141
(multindexa.tex
LaTeX Version 2.09 <25 March 1992>
(/usr/local/lib/tex/macros/book.sty
Standard Document Style 'book' <14 Jan 92>.
(/usr/local/lib/tex/macros/bk10.sty)) (index.sty
Style-Option: 'index' v3.01 <19 July 1993> (dmj))
index.sty> Writing index file multindexa.idx
index.sty> Writing index file multindexa.adx
index.sty> Writing index file multindexa.ndx
(multindexa.aux) (multindexa.toc) [1] [2]
Chapter 1.
(multindexa.nnd [3] [4]) (multindexa.and [5])
(multindexa.ind [6]) (multindexa.aux) )
Output written on multindexa.dvi (6 pages, 2776 bytes).
Transcript written on multindexa.log.
```

Рис. 12.11. Формирование нескольких указателей — запуск LaTeX'а и пакета *MakeIndex*

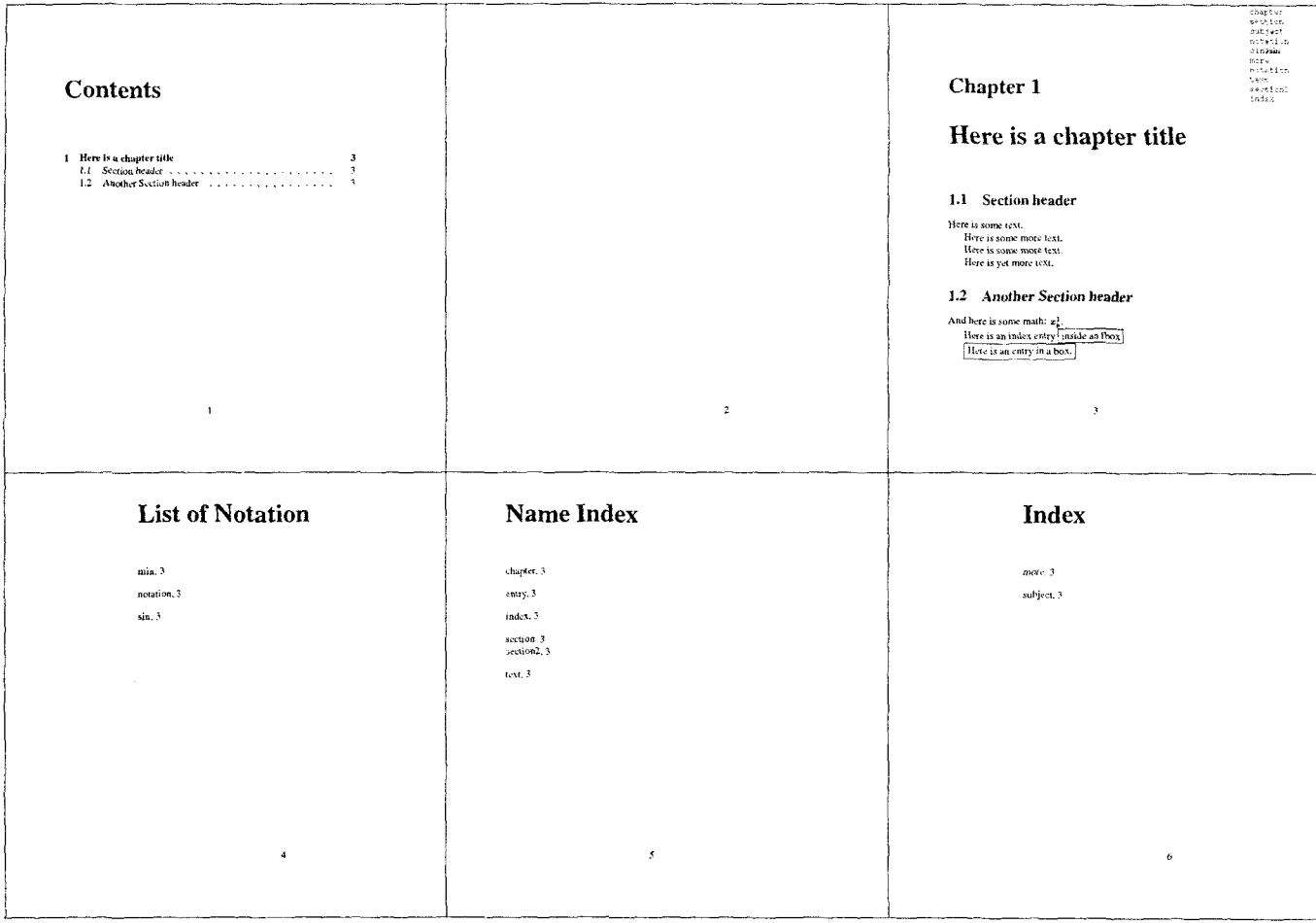


Рис. 12.12. Формирование нескольких указателей – пример вывода

ГЛАВА 13

Создание списка литературы

В отличие от оглавления (см. разд. 2.4) и указателя (см. гл. 12), которые помогают читателю ориентироваться в книге, библиографические ссылки создаются для того, чтобы по ним можно было продолжить изучение предмета. Чтобы эта цель достигалась, ссылки должны быть точными и содержать достаточную информацию для отыскания соответствующих публикаций.

Стили, которые могут использоваться для оформления списка литературы, весьма разнообразны. При этом в различных областях науки утвердились собственные достаточно строгие стандарты. Хороший обзор существующих способов оформления списка литературы имеется в книге *The Chicago Manual of Style* [103].

Обычно при оформлении библиографии авторам приходится следовать правилам, принятым в конкретном издательстве, и создание списка литературы, точно соответствующего этим правилам, можно считать одной из весьма важных задач при подготовке оригинал-макета книги или статьи.

При использовании традиционных способов подготовки списка литературы «вручную», без компьютерной автоматизации этого процесса, приходится сталкиваться со следующими весьма неприятными проблемами:

- Правильное оформление библиографических ссылок, в особенности на работы большого числа соавторов, является довольно трудным делом. Приходится одновременно следить, например, за тем, чтобы личные имена всюду указывались либо полностью, либо в виде одних лишь инициалов (с точками или без них), за тем, чтобы названия книг или статей выделялись курсивом либо заключались в кавычки, чтобы не было разнобоя при указании редакторов публикаций (либо «ed.», либо «Ed.», либо «Editor»), чтобы единообразной была нумерация томов журналов и т. п.
- Чрезвычайно большие трудности могут возникнуть в том случае, когда по требованию издательства нужно изменить формат списка литературы (на-

пример, перейти от упорядочения по фамилиям авторов в алфавитном порядке плюс год публикации к нумерации в порядке упоминания в тексте).

- Со значительными сложностями связан и подход, при котором из многочисленных литературных ссылок создается единая большая база данных, используемая при подготовке различных документов.

Подход к проблеме генерирования списков литературы, описанный в настоящей главе, основан на использовании LaTeX'а совместно с программой BibTeX, автором которой является Орен Паташник. За годы, прошедшие с момента появления этой программы, были созданы десятки стилей для форматирования библиографии с ее помощью (см. табл. 13.1), поэтому стиль, отвечающий требованиям конкретного издательства, как правило, бывает несложно подобрать (возможно, слегка модифицировав).

В первом разделе главы объясняется, как следует организовывать ссылки в тексте. Затем описан механизм совместной работы LaTeX'а и BibTeX'а и рассмотрены некоторые BibTeX'овские стили. На примере показано, как, используя один и тот же исходный tex-файл и BibTeX'овскую базу данных, можно без труда изменять как вид библиографических ссылок в тексте, так и оформление самого списка литературы.

Последующие разделы главы содержат уточненный и расширенный вариант приложения B [$\mathcal{L}$ **140–7**] из книги LaTeX book, в котором подробно рассмотрены *приемы построения библиографической базы данных для* BibTeX'а (*см. также* [40, с. 372–88]).

В заключительных разделах приведен формат BibTeX'овских стилевых файлов и дан краткий обзор используемых в них команд и специфических процедур. Здесь же описана общая структура файлов документации `btxbst.doc`. Наконец, показано, каким образом имеющийся стилевой файл может быть адаптирован к требованиям конкретного издательства или же того или иного иностранного языка.

13.1 Создание библиографических ссылок

Команда, создающая в тексте LaTeX'овского документа ссылку на литературный источник, имеет вид [$\mathcal{L}$ **73–4**], [$\mathcal{A}$ **159**]

`\cite[`*text*`]{`*cite_key-list*`}`

Команда `\cite` устанавливает соответствие между разделенными запятой элементами параметра *cite_key-list* (условными обозначениями для публикаций, на которые производится ссылка) и аргументами команд `\bibitem` в окружении `thebibliography`; результаты записываются в файл с расширением `.aux`. Как и для всех LaTeX'овских имен, прописные и строчные буквы в условных обозначениях публикаций считаются различными.

Необязательный параметр *text* служит для включения в ссылку дополнительной информации, которая будет напечатана вместе с текстом, генерируемым командой `\cite`. Если программа BibTeX не используется, нумерация ссылок определяется тем порядком, в котором условные обозначения входят в команды `\bibitem` внутри окружения `thebibliography`.

```
\begin{thebibliography}{widest_entry}
\bibitem[label1]{cite_key1} bibliographic information
\bibitem[label2]{cite_key2} bibliographic information
      ...
\end{thebibliography}
```

Как уже говорилось, соответствие между командой `\cite` и публикациями, перечисленными в списке литературы (одной или несколькими) устанавливается при помощи аргумента *cite_key-list*. Вид, который ссылка будет иметь в готовом документе, определяется выбранным библиографическим стилем.

При использовании BibTeX'а (см. ниже) возможен следующий вариант команды `\cite` [*Л* **74**] (см. также [40, с. 378]):

`\nocite{`*cite_key-list*`}`

Эта команда ничего не вносит в текст документа; действие команды состоит в том, что список ее аргументов *cite_key-list* записывается в файл `.aux`, так что соответствующая библиографическая информация оказывается внесенной в список литературы. Команда вида `\nocite{*}` вводит в список литературы все записи BibTeX'овской базы данных.

13.1.1 Придание ссылкам требуемого вида

За форматирование ссылок реально отвечает внутренняя LaTeX'овская команда `\@cite`. Возможные форматы ссылок задаются следующим образом (синтаксис команды `\ifthenelse` см. в разд. А.5 Приложения А):

- ссылки, заключенные в квадратные скобки (формат, используемый по умолчанию):

```
\renewcommand{\@cite}[2]{%
          [{#1\ifthenelse{\boolean{@tempswa}}{,#2}{}}]}
```

- ссылки, записываемые как показатель степени:

```
\renewcommand{\@cite}[2]{%
          {\textsuperscript{#1}\ifthenelse{\boolean{@tempswa}}{,#2}{}}}
```

Для понимания этих определений нужно знать, что LaTeX присваивает вспомогательной булевой переменной `\if@tempswa` значение `true`, если у команды `\cite` имеется необязательный аргумент.

Если ссылка содержит последовательности из трех или более последовательных номеров, расположенных один за другим, то все такие последовательности

можно автоматически заменить соответствующими диапазонами значений. Для этого достаточно подключить пакет cite, автором которого является Дональд Арсено. Например, ссылку [4,5,6,7,9,8,6] данный пакет преобразует в [4–7,9,8,6]. Дополнительно в пакете имеется команда `\citen`, позволяющая получать в ссылке (и в `.aux`-файле) номера без скобок и без дальнейшего форматирования. Так, набрав «`See also ref.~\citen{junk}.`», на печати получим «See also ref. 9».

Пакет citesort, принадлежащий Йену Грину, делает несколько больше, а именно прежде чем заменить последовательности диапазонами, упорядочивает по возрастанию все номера в ссылке. В результате для предыдущего примера получится [4–9].

Другой пакет Дональда Арсено, overcite, работает так же, как cite, с тем отличием, что ссылки оформляются как верхние индексы, отделенные друг от друга запятыми и маленькими пробелами. Три или более последовательных номера, идущих подряд, заменяются диапазоном, но упорядочивание не производится и наибольшее сжатие ссылки не достигается. Если одна из ссылок содержит дополнительный текст, весь список будет напечатан так, как если бы использовался пакет `cite.sty` (см. выше). Знаки препинания (`.` `,` `;` `:`) автоматически помещаются перед номером ссылки, даже если ссылке предшествует выражение в кавычках, знак препинания не выносится на позицию перед кавычкой. Так, например, `''The {\TeX}book'' \cite{Knuth}.` дает на печати "The TEXbook".[8], что не всегда хорошо[1].

Во многих стилях для создания списка литературы (см. следующий раздел) имеются вспомогательные команды, при помощи которых можно эффективно управлять видом ссылок на печати. Так, например, в пакете chicago, реализующем рекомендации руководства *The Chicago Manual of Style* [103], определены следующие команды:

`\cite{`*key*`}`	полный список авторов и год, например, (Brown 1978; Jarke, Turner, Stohl, et al. 1985)
`\citeA{`*key*`}`	только полный список авторов, например, (Brown; Jarke, Turner and Stohl)
`\citeN{`*key*`}`	полный список авторов и год, используемые в тексте, например, Shneiderman (1978) states that ...
`\shortcite{`*key*`}`	сокращенный список авторов и год
`\shortciteA{`*key*`}`	сокращенный список авторов
`\shortciteN{`*key*`}`	сокращенный список авторов и год, используемые в тексте
`\citeyear{`*key*`}`	только год в скобках

Для каждой из этих команд имеется версия, когда скобки не ставятся. В этом случае к имени команды присоединяется `NP`), например, `\citeNP`.

[1] В английских изданиях нормой по отношению к точке и запятой (но не точке с запятой!) является порядок, когда знак препинания предшествует закрывающей кавычке.— *Прим. перев.*

Пэтрик Дэйли, автор программы makebst (см. разд. 13.9), разработал стиль natbib BibTeX, который надлежит использовать вместе с пакетом natbib. В нем реализованы различные библиографические форматы типа «автор — год», наподобие тех, которые приведены выше. Данный стиль разом заменяет все стили apalike, astron, authordate, а также harvard, named и newapa BibTeX, описанные в табл. 13.1. Для управления пунктуацией внутри ссылок предусмотрена команда `\bibpunct`.

13.1.2 Выбор формата меток

Окружение `thebibliography` реализовано как перечень общего вида [$\mathcal{L}$ **187**], [$\mathcal{A}$ **160**]. По умолчанию, метка элемента списка литературы определяется так:

```
\newcommand{\@biblabel}[1]{[#1]}
```

В тех библиографические стилях, где в ссылках используются не номера, а фамилии авторов (к таковым относятся, например, стили apalike и chicago), команда `\@biblabel` переопределена так, что ничего не производит на печати,— в этих стилях метка строится при помощи иной, более сложной конструкции. Отступы для элементов списка литературы определяются величиной `\bibhang`, которая по умолчанию равна 2 em.

Различные части библиографической информации (автор, заглавие и т. д.) отделяются друг от друга в пределах одного элемента списка (`\bibitem`) командами `\newblock`. Обычно все эти «блоки» вместе образуют на печати один абзац. Впрочем, иногда список литературы делают «открытым», т. е. таким, что каждый «блок» начинается с новой строки, а последующие строки того же блока имеют отступ от левого края (этот отступ определяется величиной `\bibindent` и по умолчанию равен 1.5 em) [$\mathcal{L}$ **160**]. Такой вид списка литературы получается при использовании пакета openbib.

13.2 Совместное использование BibTeX'а и LaTeX'а

LaTeX умеет генерировать перекрестные ссылки. Одна из форм этой общей процедуры используется для создания списка литературы. В предыдущем разделе говорилось, что библиографические ссылки в тексте создаются командами вида `\cite{`*cite_key*`}`, где *key* — ключевое слово, отождествляемое с определенным элементом перечня `thebibliography`, т. е. списка литературы. При небольшом числе ссылок список литературы вполне можно составить вручную, однако удобнее иметь возможность считывать информацию из библиографической базы данных. В этом случае то же самое ключевое слово *cite_key* однозначно определяет запись в базе данных, и в процессе обработки документа LaTeX'ом библиографическая ссылка в тексте автоматически разворачивается в элемент списка литературы. Процесс совместной работы LaTeX'а и BibTeX'а схематически представлен на рис. 13.1.

① Прогон LaTeX'а, генерирующий список ссылок вида `\cite` во вспомогательном файле `.aux`.

② Прогон BibTeX'а, при котором вспомогательный файл считывается, производится поиск соответствующих записей в некоторой библиографической базе данных (могут использоваться несколько `.bib` файлов), после чего создается файл (с расширением `.bbl`), содержащий ссылки, отформатированные в соответствии с библиографическим форматом, определенным в стилевом файле `.bst`. Предупреждения и сообщения об ошибках записываются в протокол (файл с расширением `.blg`). Отметим, что BibTeX никогда не читает исходный LaTeX'овский файл.

③ Еще один прогон LaTeX'а, при котором считывается файл библиографии `.bbl`.

④ Третий прогон LaTeX'а, при котором ссылки в тексте принимают свой окончательный вид.

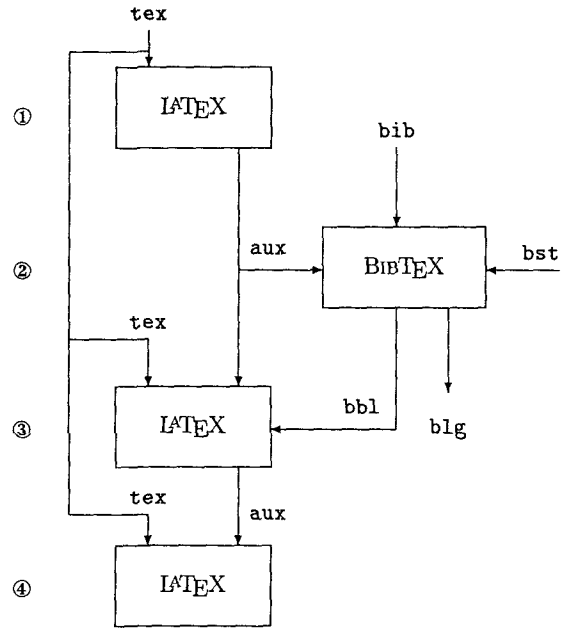


Рис. 13.1. Обмен данными при работе BibTeX'а и LaTeX'а

Точное описание формата исходных данных для BibTeX'а будет дано в разд. 13.5. Тем не менее, скажем об этом формате вкратце уже сейчас, для того чтобы сделать более понятными примеры, рассматриваемые далее в настоящем разделе. Структура данных, подаваемых на вход BibTeX'а, содержит три основных элемента:

1. *тип данных (публикации)*, например: `book`, `article`, `inproceedings`, `phdthesis`;
2. *ключевое слово, выбираемое пользователем* и используемое как идентификатор данной публикации. Аргумент *cite_key* команды `\cite`, используемой в тексте документа для ссылок на публикацию, должен совпадать с этим ключевым словом (при этом строчные и прописные буквы считаются различными!).
3. *определенный набор полей*, каждое из которых состоит из имени поля (например, `author`, `journal`, `title`) и собственно данных, заключенных в кавычки или фигурные скобки.

Для построения ключевых слов, идентифицирующих элементы библиографической базы данных, существует несколько довольно удобных схем. Весьма популярна так называемая гарвардская схема, согласно которой ключевое слово образуется из фамилии автора (написанной строчными буквами) и года публикации, разделенных двоеточием. Это выглядит вот так: `smith:1987`. Другие варианты «имен» для элементов списка литературы приведены на рис. 13.4, представляющем собой пример BibTeX'овской базы данных.

Информация о публикациях считывается BibTeX'ом из библиографической базы данных (`.bib`-файл), а форматирование этой информации определяется используемым библиографическим стилем (`.bst`-файл), который содержит набор команд, написанных на языке со стековой структурой. Эти команды и интерпретируются BibTeX'ом (см. разд. 13.7 и последующие разделы).

Для любого типа публикации BibTeX знает, какие поля являются обязательными, какие — необязательными, а какие следует проигнорировать (см. табл. 13.2). Если какая-то важная информация отсутствует, BibTeX выдает предупреждение типа «`author name required`». Стилевой файл управляет как видом, который будет иметь на печати ссылка в основном тексте, так и тем, как будет выглядеть соответствующий элемент списка литературы (т. е. окружения `thebibliography`).

Отметим, что сгенерировать список литературы и расставить надлежащим образом ссылки в тексте можно, не используя BibTeX, а пользуясь одним лишь LaTeX'ом. В этом случае данные о публикациях для списка литературы нужно вводить вручную. Кроме того, список литературы, созданный BibTeX'ом, можно вручную отредактировать — это может понадобиться в нестандартных случаях. Наконец, если нужно, чтобы LaTeX'овский документ был «замкнутым», в него можно включить содержимое `.bbl`-файла (см. описание утилиты aux2bib на с. 440).

13.2.1 Список стилевых файлов для BibTeX'а

Различными организациями и частными лицами создано немало стилевых файлов, поддерживающих фирменный стиль тех или иных журналов или издательств. Большое число BibTeX'овских стилей имеется в коллекции, собранной Нельсоном Биби (см. табл. 13.1). К каждому стилю в этой коллекции прилагается пример, демонстрирующий особенности данного стиля[2]. Некоторые BibTeX'овские стили, например, authordate⟨*i*⟩, jmb и named, должны, как указано в табл. 13.1, использоваться в сочетании со специальными LaTeX'овскими пакетами — только в этом случае получается требуемый эффект.

Заметим, что можно попытаться создать свой собственный стиль списка литературы путем модификации одного из стилей, указанных в таблице (о том, как это делается, см. разд. 13.8), или же при помощи программы makebst (см. разд. 13.9).

[2] По поводу того, как заполучить эти файлы из TeX'овских архивов, см. приложение В.

Название стиля	*Описание*
abbrv.bst	Стандартный BibTeX'овский стиль
abstract.bst	Модифицированный стиль alpha с ключевым словом `abstract`
acm.bst	Стиль Association for Computing Machinery
agsm.bst	Стиль Australian Government publications
alpha.bst	Стандартный BibTeX'овский стиль
amsalpha.bst	Стиль типа alpha для $\mathcal{AMS}$-TeX'а
amsplain.bst	Стиль типа plain для $\mathcal{AMS}$-TeX'а (с числовыми метками)
annotate.bst	Модифицированный стиль alpha с ключевым словом `annote`
annotation.bst	Модифицированный стиль plain с ключевым словом `annote`
apa.bst	Стиль American Psychology Association
apalike.bst	Вариант стиля apa
apalike.sty	LaTeX'овский пакет для apalike.bst
apalike2.bst	Вариант стиля apalike
astron.bst	Стиль журнала Astronomy
authordate*i*.bst	*i*=[1,4]. Набор стилей, создающих список литературы типа автор–год.
authordate1-4.sty	LaTeX'овский пакет для authordate*i*.bst
bbs.bst	Стиль журнала Behavioral and Brain Sciences
cbe.bst	Стиль для Council of Biology Editors (в том числе для журналов American Naturalist, Evolution и др.)
cell.bst	Набор незначительных модификаций стиля jmb
dcu.bst	Стиль для Design Computing Unit (Sidney University)
harvard.sty	LaTeX'овский пакет для гарвардских стилей (agsm, dcu, kluwer)
humanbio.bst	Стиль для Human Biology
humannat.bst	Стиль журналов Human Nature и American Anthropologist
ieeetr.bst	Стиль журнала Transactions of the Institute of Electrical and Electronic Engineers
is-abbrv.bst	abbrev BibTeX style with ISSN and ISBN keyword added
is-alpha.bst	Стиль alpha с добавленными ключевыми словами ISSN и ISBN
is-plain.bst	Стиль plain с добавленными ключевыми словами ISSN и ISBN
is-unsrt.bst	Стиль unsrt с добавленными ключевыми словами ISSN и ISBN
jmb.bst	Стиль журнала Journal of Molecular Biology
jmb.sty	LaTeX'овский пакет для jmb.bst
jtb.bst	Стиль журнала Journal of Theoretical Biology
kluwer.bst	Стиль издательства Kluwer Academic Publishers
named.bst	Стиль со ссылками типа «автор–год»
named.sty	LaTeX'овский пакет для named.bst
namunsrt.bst	Вариант стиля unsrt, аналогичный Named
nar.bst	Стиль журнала Nucleic Acid Research
nar.sty	LaTeX'овский пакет для nar.bst

окончание см. на след. стр.

Таблица 13.1. Подборка BibTeX'овских стилевых файлов

окончание (см. предыдущую страницу)	
Название стиля	*Описание*
natbib.bst	Общий BibTeX'овский стиль, реализующий различные форматы типа «автор–год»
natbib.sty	LaTeX'овский пакет для natbib.bst
nature.bst	Стиль журнала Nature
nature.sty	LaTeX'овский пакет для nature.bst
newapa.bst	Модификация стиля apalike.bst
newapa.sty	LaTeX'овский пакет для newapa.bst
phaip.bst	Стиль журналов издательства American Institute of Physics
phcpc.bst	Стиль журнала Computer Physics Communications
phiaea.bst	Стиль изданий Conferences of the International Atomic Energy Agency
phjcp.bst	Стиль журнала Journal of Computational Physics
phnf.bst	Стиль журнала Nuclear Fusion
phnflet.bst	Стиль журнала Nuclear Fusion Letters
phpf.bst	Стиль журнала Physics of Fluids
phppcf.bst	Версия стиля apalike для журналов по физике
phreport.bst	Стиль журнала Internal physics reports
phrmp.bst	Стиль журнала Reviews of Modern Physics
plain.bst	Стандартный BibTeX'овский стиль
plainyr.bst	Стиль plain BibTeX с упорядочением по годам
siam.bst	Стиль Society of Industrial and Applied Mathematics
unsrt.bst	Стандартный BibTeX'овский стиль

Таблица 13.1.

13.2.2 Примеры BibTeX'овских стилей

В этом разделе на примерах показывается, каким может быть эффект от использования разных библиографических стилей при неизменных исходном файле и файле библиографических данных.

На рис. 13.2 приведен LaTeX'овский файл, содержащий набор ссылок на наиболее часто встречающиеся типы публикаций. Данный файл использует библиографический стиль plain и ссылается на BibTeX'овскую базу данных `bsample`. На рис. 13.3 показана вся последовательность действий, необходимых для получения готового документа. Первым делом этот файл обрабатывается LaTeX'ом. Затем сгенерированный на первом шаге `.aux`-файл обрабатывается BibTeX'ом, причем информация о требуемых публикациях считывается из базы данных, которая в полном виде воспроизведена на рис. 13.4. Используемый библиографический стиль определяет формат, в котором публикации вносятся в `.bbl`-файл для последующей обработки LaTeX'ом; этот стиль задается командой [$\mathcal{L}$ 74] (см. также [40, с. 376]) `\bibliographystyle` в исходном файле. Затем LaTeX запускается еще дважды, в результате чего все ссылки принимают окончательный вид.

В данном случае, используя «стандартный» BibTeX'овский стиль plain вместе с исходным файлом, приведенным на рис. 13.2, мы получим текст, показанный

```
\documentclass{article}
\pagestyle{empty}
\begin{document}
\section*{Пример ссылок типа \texttt{plain}}
Вот ссылка на обычную книгу~\cite{Eijkhout:1991},
а вот на книгу, изданную редактором~\cite{Roth:postscript}.
Так ссылаются на статью одного автора~\cite{Felici:1991},
а так "--- на статью нескольких авторов~\cite{Mittelbach/Schoepf:1990}.
Ссылка на статью в сборнике~\cite{Yannis:1991}.
Теперь сошлемся на учебник~\cite{Dynatext} и на технический
отчет~\cite{Knuth:WEB}.
Вот ссылка на неопубликованную работу~\cite{EVH:Office}.
Ссылаемся на главу в книге~\cite{Wood:color} и
на диссертацию~\cite{Liang:1983}. А вот пример ссылки на несколько
работ~\cite{Eijkhout:1991,Roth:postscript}.

\bibliographystyle{plain}
\bibliography{bsample}
\end{document}
```

Рис. 13.2. Пример LaTeX'овского файла, предназначенного для использования BibTeX'а

в левой колонке на стр. 428. Еще пять вариантов оформления того же списка литературы показаны на рис. 13.5–13.7, начиная с правой колонки на стр. 428. Они соответствуют оставшимся трем [**Л 74,75**] (см. также [40, с. 377]) стандартным BibTeX'овским стилям и еще двум библиографическим стилям; эти стили должны быть указаны в исходном файле вместо plain, после чего файл должен быть обработан заново в той же последовательности, что и раньше.

plain Стандартный BibTeX'овский стиль. Источники нумеруются в алфавитном порядке, ссылки используют номера.

unsrt Стандартный BibTeX'овский стиль. Отличается от стиля plain тем, что публикации нумеруются в порядке цитирования.

alpha Стандартный BibTeX'овский стиль. Отличается от стиля plain тем, что метки публикаций образуются из имени автора и года публикации.

abbrv Стандартный BibTeX'овский стиль. Отличается от стиля plain тем, что информация о публикациях дается в более компактной форме: для личных имен, месяцев и названий журналов используются сокращения.

acm Альтернативный BibTeX'овский стиль, используемый в журналах, издаваемых под эгидой Association for Computing Machinery. Фамилия и имя автора печатаются капителью, для ссылок используются номера.

apalike Альтернативный BibTeX'овский стиль, используемый в журналах, издаваемых American Psychology Association. Используется вместе с

```
$ latex bsample  %%%%%%%%%%% 1-й прогон LaTeX'a
This is TeX, C Version 3.141
(bsample.tex
LaTeX Version 2.09 <06 August 1993> with NFSS2
(/usr/local/lib/tex/macros/article.sty
Standard Document Style 'article' <14 Jan 92>.
(/usr/local/lib/tex/macros/art10.sty)) (bsample.aux)
LaTeX Warning: Citation 'Eijkhout:1991' on page 1 undefined on input line 9.
LaTeX Warning: Citation 'Roth:postscript' on page 1 undefined on input line 10.
LaTeX Warning: Citation 'Felici:1991' on page 1 undefined on input line 11.
LaTeX Warning: Citation 'Mittelbach/Schoepf:1990' on page 1 undefined on input line 12.
LaTeX Warning: Citation 'Yannis:1991' on page 1 undefined on input line 13.
LaTeX Warning: Citation 'Dynatext' on page 1 undefined on input line 14.
LaTeX Warning: Citation 'Knuth:WEB' on page 1 undefined on input line 15.
LaTeX Warning: Citation 'EVH:Office' on page 1 undefined on input line 16.
LaTeX Warning: Citation 'Wood:color' on page 1 undefined on input line 17.
LaTeX Warning: Citation 'Liang:1983' on page 1 undefined on input line 18.
LaTeX Warning: Citation 'Eijkhout:1991' on page 1 undefined on input line 19.
LaTeX Warning: Citation 'Roth:postscript' on page 1 undefined on input line 19
No file bsample.bbl.
[1] (bsample.aux) )
Output written on bsample.dvi (1 page, 904 bytes).
Transcript written on bsample.log.

$ bibtex bsample                        %%%%%%%%%%% Прогон BibTeX'a
This is BibTeX, C Version 0.99c
The top-level auxiliary file: bsample.aux
The style file: plain.bst
Database file #1: bsample.bib
Warning--empty volume in Wood:color's crossref of Roth:postscript
(There was 1 warning)

$ latex bsample                         %%%%%%%%%%% 2-й прогон LaTeX'a
This is TeX, C Version 3.141
(bsample.tex
(bsample.aux)   ...    %%%%% Ссылки в документе все еще остаются непроставленными

$ latex bsample                         %%%%%%%%%%% 3-й прогон LaTeX'a
This is TeX, C Version 3.141
(bsample.tex (bsample.aux)  ...  %%%%% Все ссылки проставляются в окончательном виде
```

Рис. 13.3. Как LaTeX работает с BibTeX'овской библиографической базой данных

LaTeX'овским пакетом apalike. Источники упорядочиваются по фамилиям авторов в алфавитном порядке, каждый элемент списка литературы оформляется в виде абзаца с отрицательным абзацным отступом, номера не используются.

При использовании BibTeX'овских стилей вид ссылок и элементов списка литературы иногда выходит за рамки возможностей стандартного LaTeX'a. В этих случаях помимо задания используемого BibTeX'овского стиля должен быть посредством команды \usepackage подгружен некий дополнительный LaTeX'овский пакет. Так обстоит дело, например, со стилем apalike, фигурирующем в последнем примере.

```
% BiBTeX sample data base

%% bibtexfile{
%%  author   = "Michel Goossens",
%%  version  = "1.12",
%%  date     = "15 November 1993",
%%  filename = "bsample.bib",
%%  address  = "CN Division, CERN
%%              CH1211, Geneva 23
%%              Switzerland",
%%  email    = "<goossens at node cern.ch>"  }

@Preamble{{\input{bibnames.sty}}
 # {\hyphenation{Post-Script Sprin-ger}}
}

@String{AW = {{Ad\-di\-son-Wes\-ley}}}
@String{AW:adr = {Reading, Massachusetts}}
@String{j-TUGboat = {TUGboat}}

@manual{Dynatext,
 key          ={Dynatext},
 title        ={{Dynatext, Electronic Book
                 Indexer/Browser}},
 organization={Electronic Book
               Technology Inc.},
 address      ={Providence, Rhode Island},
 year         =1991                          }

@Book{Eijkhout:1991,
 author       ={Victor Eijkhout},
 title        ={{\TeX{} by Topic, a
                 {\TeX}nicians Reference}},
 publisher    =AW,
 address      =AW:adr,
 year         =1991,                         }

@techreport{EVH:Office,
 author       ={Eric van Herwijnen},
 title        ={{Future Office Systems
                 Requirements}},
 institution ={CERN DD Internal Note},
 year         =1988, month = nov             }

@Article{Felici:1991,
 author       ={James Felici},
 title        ={{PostScript versus TrueType}},
 journal      ={Macworld},
 volume       =8,pages={195--201},
 month        =sep, year = 1991              }

@techreport{Knuth:WEB,
 title        ={{The \textsf{WEB} System of
                 Structured Documentation}},
 month        =sep, year = 1983,
 author       ={Donald E. Knuth},
 address      ={Stanford, CA 94305},
 number       ={STAN-CS-83-980},
 institution ={Department of Computer
               Science, Stanford University} }

@phdthesis{Liang:1983,
 author       ={Franklin Mark Liang},
 month        =jun, year = 1983,
 school       ={Stanford University},
 address      ={Stanford, CA 94305},
 title        ={{Word Hy-phen-a-tion by
                 Com-pu-ter}},
 note         ={Also available as Stanford
               University, Department of
               Computer Science Report
               No. STAN-CS-83-977}            }

@Article{Mittelbach/Schoepf:1990,
 author       ={Frank Mittelbach and
               Rainer Sch{\"o}pf},
 title        ={{The New Font Selection "--- User
                 Interface to Standard \LaTeX}},
 journal      =j-TUGboat,
 volume       =11, number = 2,
 pages        ={297--305},
 year         =1990                          }

@Inbook{Wood:color,
 author       ={Pat Wood},
 crossref     ={Roth:postscript},
 booktitle    ={{Real World PostScript}},
 title        ={PostScript Color Separation},
 pages        ={201--225}                    }

@Book{Roth:postscript,
 editor       ={Stephen E. Roth},
 title        ={{Real World PostScript}},
 publisher    =AW,
 address      =AW:adr,
 year         =1988,

 ISBN         ={0-201-06663-7}               }
@Inproceedings{Yannis:1991,
 title = {{\TeX} and those other languages},
 author= {Yannis Haralambous},
 pages = {539--548},
 booktitle ={1991 Annual Meeting Proceedings,
             Part 2, \TeX{} Users Group,
             Twelfth Annual Meeting, Dedham,
             Massachusetts, July 15--18, 1991},
 editor = {Hope Hamilton},
 organization ={{\TeX} Users Group},
 year     =   {1991},
 address ={Providence, Rhode Island},
 journal =j-TUGboat,
 volume  = 12, month = dec                    }
```

Рис. 13.4. Пример BibTEX'овской базы данных

Этот файл использовался при обработке LATEX'овского файла, приведенного на рис. 13.2; полученные в результате этого тексты показаны на рис. 13.5–13.7 (см. стр. 428 и далее).

Example of citations of kind `plain`

Citation of a normal book [1] and an edited book [8]. Now we cite an article written by a single [3] and by multiple authors [7]. A reference to an article inside proceedings [4]. We refer to a manual [2] and a technical report [5]. A citation of an unpublished work [9]. A reference to a chapter in a book [10] and to a PhD thesis [6]. An example of multiple citations [1, 8].

References

[1] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[2] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[3] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[4] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

[5] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[6] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[7] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[8] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[9] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[10] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [8], 1988.

Example of citations of kind `unsrt`

Citation of a normal book [1] and an edited book [2]. Now we cite an article written by a single [3] and by multiple authors [4]. A reference to an article inside proceedings [5]. We refer to a manual [6] and a technical report [7]. A citation of an unpublished work [8]. A reference to a chapter in a book [9] and to a PhD thesis [10]. An example of multiple citations [1, 2].

References

[1] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[2] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[3] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[4] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[5] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

[6] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[7] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[8] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[9] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [2], 1988.

[10] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

Рис. 13.5. Примеры использования BIBTEX'овских стилей plain и unsrt

Example of citations of kind `alpha`

Citation of a normal book [Eij91] and an edited book [Rot88]. Now we cite an article written by a single [Fel91] and by multiple authors [MS90]. A reference to an article inside proceedings [Har91]. We refer to a manual [Dyn91] and a technical report [Knu83]. A citation of an unpublished work [vH88]. A reference to a chapter in a book [Woo88] and to a PhD thesis [Lia83]. An example of multiple citations [Eij91, Rot88].

References

[Dyn91] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[Eij91] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[Fel91] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[Har91] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

[Knu83] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[Lia83] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[MS90] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[Rot88] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[vH88] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[Woo88] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [Rot88], 1988.

Example of citations of kind `abbrv`

Citation of a normal book [1] and an edited book [8]. Now we cite an article written by a single [3] and by multiple authors [7]. A reference to an article inside proceedings [4]. We refer to a manual [2] and a technical report [5]. A citation of an unpublished work [9]. A reference to a chapter in a book [10] and to a PhD thesis [6]. An example of multiple citations [1, 8].

References

[1] V. Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[2] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[3] J. Felici. PostScript versus TrueType. *Macworld*, 8:195–201, Sept. 1991.

[4] Y. Haralambous. TEX and those other languages. In H. Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, Dec. 1991. TEX Users Group.

[5] D. E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, Sept. 1983.

[6] F. M. Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[7] F. Mittelbach and R. Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[8] S. E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[9] E. van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, Nov. 1988.

[10] P. Wood. *PostScript Color Separation*, pages 201–225. In Roth [8], 1988.

Рис. 13.6. Примеры использования BIBTEX'овских стилей alpha и abbrv

Example of citations of kind `acm`

Citation of a normal book [1] and an edited book [8]. Now we cite an article written by a single [3] and by multiple authors [7]. A reference to an article inside proceedings [4]. We refer to a manual [2] and a technical report [5]. A citation of an unpublished work [9]. A reference to a chapter in a book [10] and to a PhD thesis [6]. An example of multiple citations [1, 8].

References

[1] EIJKHOUT, V. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[2] ELECTRONIC BOOK TECHNOLOGY INC. *Dynatext, Electronic Book Indexer/Browser*. Providence, Rhode Island, 1991.

[3] FELICI, J. PostScript versus TrueType. *Macworld 8* (Sept. 1991), 195–201.

[4] HARALAMBOUS, Y. TEX and those other languages. In *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991* (Providence, Rhode Island, Dec. 1991), H. Hamilton, Ed., vol. 12, TEX Users Group, pp. 539–548.

[5] KNUTH, D. E. The WEB System of Structured Documentation. Tech. Rep. STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, Sept. 1983.

[6] LIANG, F. M. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[7] MITTELBACH, F., AND SCHÖPF, R. The New Font Selection — User Interface to Standard LATEX. *TUGboat 11*, 2 (1990), 297–305.

[8] ROTH, S. E., Ed. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts. 1988.

[9] VAN HERWIJNEN, E. Future Office Systems Requirements. Tech. rep., CERN DD Internal Note, Nov. 1988.

[10] WOOD, P. *PostScript Color Separation*. In Roth [8], 1988, pp. 201–225.

Example of citations of kind `apalike`

Citation of a normal book (Eijkhout, 1991) and an edited book (Roth, 1988). Now we cite an article written by a single (Felici, 1991) and by multiple authors (Mittelbach and Schöpf, 1990). A reference to an article inside proceedings (Haralambous, 1991). We refer to a manual (Dynatext, 1991) and a technical report (Knuth, 1983). A citation of an unpublished work (van Herwijnen, 1988). A reference to a chapter in a book (Wood, 1988) and to a PhD thesis (Liang, 1983). An example of multiple citations (Eijkhout, 1991; Roth, 1988).

References

Dynatext (1991). *Dynatext, Electronic Book Indexer/Browser*. Electronic Book Technology Inc., Providence, Rhode Island.

Eijkhout, V. (1991). *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts.

Felici, J. (1991). PostScript versus TrueType. *Macworld*, 8:195–201.

Haralambous, Y. (1991). TEX and those other languages. In Hamilton, H., editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island. TEX Users Group.

Knuth, D. E. (1983). The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305.

Liang, F. M. (1983). *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

Mittelbach, F. and Schöpf, R. (1990). The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305.

Roth, S. E., editor (1988). *Real World PostScript*. Addison-Wesley, Reading, Massachusetts.

van Herwijnen, E. (1988). Future Office Systems Requirements. Technical report, CERN DD Internal Note.

Wood, P. (1988). *PostScript Color Separation*, pages 201–225. In (Roth, 1988).

Рис. 13.7. Примеры использования BIBTEX'овских стилей `acm` и `apalike`

13.3 Документы с несколькими списками литературы

В больших документах, состоящих из нескольких независимых разделов, в сборниках трудов конференций, содержащих большое число различных статей, а также в книгах, где разные главы написаны разными авторами, обычно требуется иметь отдельный список литературы для каждой структурной единицы. В этом разделе мы рассмотрим два LaTeX'овских пакета, chapterbib и bibunits, предназначенных для решения этой проблемы. Заметим (это будет видно из дальнейшего), что указанные пакеты нельзя напрямую использовать с операционными системами типа MS-DOS, налагающими жесткие ограничения на структуру имен файлов.

13.3.1 Пакет chapterbib

Пакет chapterbib, разработанный Нилом Кемпсоном, позволяет создавать несколько списков литературы в одном LaTeX'овском документе, причем одни и те же публикации могут входить в разные списки литературы. Тем не менее без ограничений не обходится, а именно:

1. Команды `\bibliography` и `\bibliographystyle` в корневом файле использовать нельзя, а можно только в файлах, подгружаемых при помощи команды `\include`. В корневом файле команды, работающие с библиографией, игнорируются. Более того, в каждом подгружаемом файле команда `\bibliography` должна встречаться не более одного раза.
2. Если требуется использовать в корневом файле команду `\cite`, то для восстановления ссылок по ключевым словам этой команды нужно, чтобы внутри корневого файла имелось отдельное окружение `thebibliography` (см. рис. 13.8).

Для большей наглядности мы включили в LaTeX'овский файл на рис. 13.8 ссылки на два разных BibTeX'овских стиля (в реальной жизни такая ситуации вызвала бы недоумение).

Поскольку программа BibTeX умеет в каждый момент времени работать только с одним стилем (т. е. воспринимает только одну команду `\bibliographystyle`), BibTeX следует запускать отдельно для каждого из `.aux`-файлов `bs1.aux` и `bs2.aux`, в результате чего будут получены, соответственно, файлы `.bbl`, `bs1.bbl` и `bs2.bbl`. Вся процедура имеет вид:

```
$ latex chapterbibexa          %%%% 1-й прогон LaTeX'а
      ...
$ bibtex bs1                   %%%% Прогон BibTeX'а с 1-м файлом
This is BibTeX, C Version 0.99c
The top-level auxiliary file: bs1.aux
The style file: plain.bst
Database file #1: bsample.bib
$ bibtex bs2                   %%%% Прогон BibTeX'а со 2-м файлом
This is BibTeX, C Version 0.99c
```

```
%%%%%%%%%%%%%%%   Корневой файл chapterbibexa.tex   %%%%%%%%%%%%%%%
\documentclass{article}
\usepackage{chapterbib}
\begin{document}
\include{bs1}
\include{bs2}
\section*{Пример ссылок в корневом файле}
Ссылка на обычную книгу~\cite{Eijkhout:1991} в корневом файле.
\begin{thebibliography}{1}
\bibitem{Eijkhout:1991}
  Victor Eijkhout, \emph{\TeX{} by Topic, a
  {\TeX}nicians Reference}, Addison-Wesley (1991).
\end{thebibliography}
\end{document}

%%%%%%%%%%%%%%%%%   Подгружаемый файл bs1.tex    %%%%%%%%%%%%%%%%%

\section{Пример ссылок в подгружаемой главе}
\subsection{Сперва с использованием стиля plain}
Вот ссылка на обычную книгу~\cite{Eijkhout:1991},
а вот "--- на книгу, изданную редактором~\cite{Roth:postscript}.
Так ссылаются на статью одного автора~\cite{Felici:1991},
а так "--- на статью нескольких авторов~\cite{Mittelbach/Schoepf:1990}.
Ссылка на статью в сборнике~\cite{Yannis:1991}.
Теперь сошлемся на учебник~\cite{Dynatext} и на технический
отчет~\cite{Knuth:WEB}.
Вот ссылка на неопубликованную работу~\cite{EVH:Office}.
Ссылаемся на главу в книге~\cite{Wood:color} и
на диссертацию~\cite{Liang:1983}. А вот пример ссылки на несколько
работ~\cite{Eijkhout:1991,Roth:postscript}.
\bibliographystyle{plain}
\bibliography{bsample}

%%%%%%%%%%%%%%%%%    Подгружаемый файл bs2.tex    %%%%%%%%%%%%%%%%%

\section{Пример ссылок в подгружаемой главе}
\subsection{На этот раз используем стиль alpha}
      ..... Тот же текст .....
\bibliographystyle{alpha}
\bibliography{bsample}
```

Рис. 13.8. Корневой файл и два подгружаемых файла с самостоятельными списками литературы

1 Example of citations in an included chapter

1.1 First in plain style

Citation of a normal book [1] and an edited book [8]. Now we cite an article written by a single [3] and by multiple authors [7]. A reference to an article inside proceedings [4]. We refer to a manual [2] and a technical report [5]. A citation of an unpublished work [9]. A reference to a chapter in a book [10] and to a PhD thesis [6]. An example of multiple citations [1, 8].

References

[1] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[2] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*. 1991.

[3] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[4] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

[5] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[6] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[7] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[8] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[9] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[10] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [8], 1988.

1

2 Example of citations in an included chapter

2.1 Then in alpha style

Citation of a normal book [Eij91] and an edited book [Rot88]. Now we cite an article written by a single [Fel91] and by multiple authors [MS90]. A reference to an article inside proceedings [Har91]. We refer to a manual [Dyn91] and a technical report [Knu83]. A citation of an unpublished work [vH88]. A reference to a chapter in a book [Woo88] and to a PhD thesis [Lia83]. An example of multiple citations [Eij91, Rot88].

References

[Dyn91] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*. 1991.

[Eij91] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[Fel91] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[Har91] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

[Knu83] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[Lia83] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[MS90] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[Rot88] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[vH88] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[Woo88] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [Rot88], 1988.

2

Example of citations in the root file

Citation of a normal book [1] in the root file.

References

[1] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley (1991).

3

Рис. 13.9. Несколько списков литературы в одном файле (готовый документ)

Страницы, которые вы видите, получены путем обработки LATEX'ом и BIBTEX'ом файла, приведенного на рис. 13.8. Видно, что каждый из подгружаемых файлов был обработан отдельно и породил текст, оформленный по-своему. Мы не рекомендуем использовать в одном документе несколько различных команд `\bibliographystyle`, но в принципе, если запускать BIBTEX по отдельности с каждым файлом `.aux`, использование разных библиографических стилей возможно.

```
The top-level auxiliary file: bs2.aux
The style file: alpha.bst
Database file #1: bsample.bib
$ latex chapterbibexa                    %%%% 2-й прогон LaTeX'a
      ...
$ latex chapterbibexa                    %%%% 3-й прогон LaTeX'a
      ...
```

Окончательный результат показан на рис. 13.9.

13.3.2 Пакет bibunits

Пакет bibunits, созданный Хосе Альберто Фернандесом, позволяет порождать отдельные списки литературы для различных структурных единиц текста — глав, разделов (параграфов), а также для специального окружения `bibunit`[3]. Пакет собирает ссылки каждого блока в отдельный файл, который затем обрабатывается BibTeX'ом. Это не мешает LaTeX'у создавать общий список литературы для всего документа в целом, причем каждая публикация может фигурировать сразу в двух списках, «локальном» и «глобальном». Команда

`\bibliographyunit[`*unit*`]`

указывает, для каких блоков должны создаваться списки литературы: для каждой главы (*unit*=`\chapter`) или для каждого параграфа (*unit*=`\section`). Если необязательный аргумент отсутствует, команда `\bibliographyunit` инициирует лишь создание общего списка литературы для документа в целом.

При использовании команды `\bibliographyunit` дополнительные команды `\bibliographystyle` и `\bibliography` определяют BibTeX'овский стиль и BibTeX'овскую библиографическую базу данных, которые будут по умолчанию использоваться при генерации этих локальных списков литературы. Команды `\bibliography*` и `\bibliographystyle*` определяют стиль и базу данных по умолчанию только для данного локального блока и не влияют на глобальный список литературы.

Создать блок с самостоятельным списком литературы можно и не используя команду `\bibliographyunit`. Для этого имеется специальное окружение `bibunit`. В соответствующей команде

`\begin{bibunit}[`*style*`]`

необязательный параметр *style* определяет стиль для списка литературы, который будет использоваться в блоке вместо стиля, принятого по умолчанию. При этом до того, как закрыть данное окружение, необходимо при помощи команды

`\putbib[`*bibtex-files*`]`

указать то место в тексте, где будет размещен сгенерированный список литературы. В отсутствие необязательного параметра *bibtex-files* команда `\putbib` вос-

[3] Общий термин *unit* применяется для обозначения любой из этих структурных единиц; в качестве русского эквивалента мы будем использовать слово «блок».— *Прим. перев.*

```
\documentclass{article}
\usepackage{bibunits}
\begin{document}

\section{Пакет bibunits}
\begin{bibunit}[unsrt]%%%%%%%%%%%%%%%%%%%%%% Начало первого блока
\subsection{Начнем со стиля unsrt}
Ссылка на диссертацию~\cite{Liang:1983}.
Вот ссылка на статью, написанную несколькими
авторами~\cite{Mittelbach/Schoepf:1990}.
А вот "--- на статью в сборнике трудов~\cite{Yannis:1991}.
\putbib[bsample]
\end{bibunit}

\subsection{Теперь воспользуемся стилем abbrv}
\begin{bibunit}[abbrv]%%%%%%%%%%%%%%%%%%%%%% Начало второго блока
Сошлемся на учебник~\cite{Dynatext} и на технический
отчет~\cite{Knuth:WEB}.
Ссылка на главу в книге~\cite{Wood:color} и на книгу,
изданную редактором~\cite{Roth:postscript}.
\putbib[bsample]
\end{bibunit}

\subsection{В завершение используем стиль alpha}
\begin{bibunit}[alpha]%%%%%%%%%%%%%%%%%%%%%% Начало третьего блока
Ссылка на обычную книгу~\cite{Eijkhout:1991}
и на статью, написанную одним автором~\cite{Felici:1991}.
\putbib[bsample]
\end{bibunit}
\end{document}
```

Рис. 13.10. Пример исходного файла для пакета bibunits

пользуется библиографическими файлами (базами данных), принятыми по умолчанию.

При работе с пакетом bibunits требуется запускать BibTeX для каждого блока поочередно, поскольку для каждого блока генерируется отдельный файл `jobname.i.aux`, где `i` — порядковый номер блока.

Пример, иллюстрирующий сказанное, состоит из исходного файла, приведенного на рис. 13.10, последовательности команд, показанной на рис. 13.11, и результата, воспроизведенного на рис. 13.12.

Из двух рассматриваемых пакетов, chapterbib и bibunits, первый в большей мере опирается на интуитивные представления. Следует, однако, иметь в виду, что блоки, для которых требуются отдельные списки литературы, могут пред-

```
$ latex bibunitsexa      %%%%%%%%%%%%%  1-й прогон LaTeX'а
This is TeX, Version 3.141
(bibunitsexa.tex
No file bibunitsexa.aux.   ...     %%% Ссылки не определены
No file bibunitsexa.1.bbl. ...     %%% Данные о публикациях не готовы
No file bibunitsexa.2.bbl. ...     %%%    то же самое
No file bibunitsexa.3.bbl. ...     %%%    то же самое
[1] (bibunitsexa.aux) )
Output written on bibunitsexa.dvi (1 page, 1032 bytes).
Transcript written on bibunitsexa.log.
$ bibtex bibunitsexa.1  %%%%%%%%%%%%%  Обработка BibTeX'ом 1-го файла
This is BibTeX, C Version 0.99c
The top-level auxiliary file: bibunitsexa.1.aux
The style file: unsrt.bst
Database file #1: bsample.bib
$ bibtex bibunitsexa.2  %%%%%%%%%%%%%  Обработка BibTeX'ом 2-го файла
This is BibTeX, C Version 0.99c
The top-level auxiliary file: bibunitsexa.2.aux
The style file: abbrv.bst
Database file #1: bsample.bib
$ bibtex bibunitsexa.3      %%%%%%%%%%%%%  Обработка BibTeX'ом 3-го файла
This is BibTeX, C Version 0.99c
The top-level auxiliary file: bibunitsexa.3.aux
The style file: alpha.bst
Database file #1: bsample.bib
$ latex bibunitsexa          %%%%%%%%%%%%%   2-й прогон LaTeX'а
This is TeX, C Version 3.141
(bibunitsexa.tex
(bibunitsexa.aux)   ...
(bibunitsexa.1.bbl) ...    %%% Данные о публикациях все еще не готовы
(bibunitsexa.2.bbl) ...    %%%     то же самое
(bibunitsexa.3.bbl) ...    %%%     то же самое
[1] (bibunitsexa.aux) )
$ latex bibunitsexa    %%%%%%%%%%%%%  3-й, последний прогон LaTeX'а
This is TeX, Version 3.141
(bibunitsexa.tex
      ...
(bibunitsexa.aux)
(bibunitsexa.1.bbl) (bibunitsexa.2.bbl) (bibunitsexa.3.bbl)
[1] (bibunitsexa.aux) )
Output written on bibunitsexa.dvi (1 page, 3232 bytes).
```

Рис. 13.11. Создание нескольких списков литературы при помощи стиля bibunits

Данная последовательность команд представляет собой процесс обработки файла, приведенного на рис. 13.10; результат показан на рис. 13.12.

1 The bibunits package

1.1 First in bibstyle unsrt

Citation of a PhD thesis [1]. Now we cite an article written by multiple authors [2]. A reference to an article inside proceedings [3].

References

[1] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[2] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LaTeX. *TUGboat*, 11(2):297–305, 1990.

[3] Yannis Haralambous. TeX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TeX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TeX Users Group.

1.2 Let us continue in bibstyle abbrv

We refer to a manual [1] and a technical report [2]. A reference to a chapter in a book [4] and to an edited book [3].

References

[1] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[2] D. E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, Sept. 1983.

[3] S. E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[4] P. Wood. *PostScript Color Separation*, pages 201–225. In Roth [3], 1988.

1.3 And finish in bibstyle alpha

Citation of a normal book [Eij91] of an article written by a single [Fel91] author.

References

[Eij91] Victor Eijkhout. *TeX by Topic, a TeXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[Fel91] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

Рис. 13.12. Результат, полученный для примера, приведенного на рис. 13.10

ставлять собой отдельные файлы, подгружаемые при помощи команды \include. Пакет bibunits мощнее, однако работа с ним требует большой аккуратности: создание большого числа блоков, возможно находящихся в различных файлах, может привести к путанице. Кроме того, поскольку bibunits генерирует имена файлов вида jobname.i.aux, состоящие из трех элементов, его непосредственное использование возможно только на UNIX'е, но не на таких операционных системах, как MS-DOS, CMS или VMS.

13.4 Средства управления библиографическими базами данных

При работе с библиографией полезно иметь распечатку всех записей BibTeX'овской базы данных, упорядоченных тем или иным образом. Для получения такой распечатки существуют разнообразные средства, возможности которых примерно одинаковы, так что выбор того или иного из них является *преимущественно делом вкуса*.

Назовем, прежде всего, LaTeX'овский пакет biblist, разработанный Йоахимом Шродом и позволяющий получать распечатки BibTeX'овских файлов достаточно большого объема. Дело в том, что при работе с большими BibTeX'овскими файлами, в особенности содержащими длинные ключевые слова, может происходить переполнение той области TeX'овской памяти, которая отводится для хранения переменных типа «цепочка литер». В этом случае, вообще говоря, требуется использовать BigTeX. Однако пакет biblist в большинстве случаев сам решает эту проблему и позволяет обойтись «малыми» версиями TeX'а.

Чтобы воспользоваться пакетом, нужно подготовить LaTeX'овский документ с использованием класса article и пакета biblist. При этом допустимо применение опций и пакетов типа twoside, german или, скажем, a4, однако ни twocolumn, ни multicol работать не будут.

В аргументе команды \bibliography должны быть указаны имена всех BibTeX'овских баз данных, которые требуется распечатать. С помощью команды \bibliographystyle можно задать определенный библиографический стиль. По умолчанию, распечатаны будут все записи базы данных. Однако, если в документе присутствуют команды \nocite, в распечатку войдут только помеченные ими записи.

Вот как выглядит файл, при помощи которого была получена распечатка, показанная на рис. 13.13:

```
\documentclass{article}
\usepackage{a4}
\usepackage{biblist}
\begin{document}
  \bibliographystyle{is-alpha}
  \bibliography{bsample}
\end{document}
```

bsample.bib (August 22, 1993)

References

Dynatext .....
Electronic Book Technology Inc., Providence, Rhode Island.
Dynatext, Electronic Book Indexer/Browser, 1991.

Eijkhout:1991 .....
Victor Eijkhout.
TEX by Topic, a TEXnicians Reference.
Addison-Wesley, Reading, Massachusetts, 1991.

Felici:1991 .....
James Felici.
PostScript versus TrueType.
Macworld, 8:195–201, September 1991.

Yannis:1991 .....
Yannis Haralambous.
TEX and those other languages.
In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

Knuth:WEB .....
Donald E. Knuth.
The WEB System of Structured Documentation.
Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

Liang:1983 .....
Franklin Mark Liang.
Word Hy-phen-a-tion by Com-pu-ter.
PhD thesis, Stanford University, Stanford, CA 94305, June 1983.
Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

Mittelbach/Schoepf:1990 .....
Frank Mittelbach and Rainer Schöpf.
The New Font Selection — User Interface to Standard LATEX.
TUGboat, 11(2):297–305, 1990.

Roth:postscript .....
Stephen E. Roth, editor.
Real World PostScript.
Addison-Wesley, Reading, Massachusetts, 1988.
ISBN 0-201-06663-7.

EVH:Office .....
Eric van Herwijnen.
Future Office Systems Requirements.
Technical report, CERN DD Internal Note, November 1988.

Wood:color .....
Pat Wood.
PostScript Color Separation, pages 201–225.
In Roth [Roth:postscript], 1988.
ISBN 0-201-06663-7.

1

Рис. 13.13. Распечатка записей базы данных `bsample.bib`, полученная при помощи biblist

При этом последовательно запускались LaTeX, BibTeX и снова LaTeX. Запускать LaTeX дважды после того, как проработает BibTeX, не требуется, если только сами библиографические записи не содержат ссылок на другие записи (типа `note = {reviewed in \cite{..}}`). Есть еще целый ряд интересных BibTeX'овских утилит. Основным автором приведенного ниже набора утилит является Дэвид Котц. Утилиты написаны для UNIX'а, но лежащие в их основе идеи носят общий характер и потому можно взять имеющиеся сценарии программ (скрипты) за основу для переноса утилит на другие операционные системы.

aux2bib Этот скрипт на языке perl создает по имеющемуся `.aux`-файлу мобильный `.bib`-файл. Это бывает полезным в тех случаях, когда LaTeX'овские файлы нужно куда-то передать.

bibkey Этот скрипт на языке C-shell использует утилиты sed, egrep и awk для создания перечня всех записей, содержащих в поле `keyword` данное ключевое слово.

Команда запуска имеет вид `bibkey keyword file`

Те символы в параметре `keyword`, которые имеют специальное значение в регулярных выражениях, используемых утилитами sed и egrep, должны быть снабжены дополнительно префиксом \ (например, символ \ нужно заменить на, например, \\). При построении выборки строчные и прописные буквы отождествляются. Допустимым считается любое выражение, допустимое для egrep, например, можно производить поиск по нескольким ключевым словам: `bibkey 'jones|smith' foo.bib`

looktex При запуске этого скрипта, написанного на C-shell, из BibTeX'овской базы данных выбираются все записи, содержащие данное ключевое слово. Данный скрипт является обобщением скрипта bibkey, и все сказанное в предыдущем пункте сохраняет силу и в этом случае.

makebib Этот скрипт, написанный на C-shell, создает из заданного набора `.bib`-файлов и некоторого (необязательного) списка публикаций один мобильный `.bib`-файл.

Команда запуска: `makebib file.bib... [citekey]...`

Результат записывается в файл `subset.bib`. Если `citekey` отсутствует, в `.bib`-файл включаются все публикации.

printbib Этот скрипт, написанный на C-shell, создает `.dvi`-файл из заданного `.bib`-файла. При этом производится сортировка по ключевым словам и в документ включаются поля `keyword` и `abstract`.

Команда запуска: `printbib bibfile...`

При этом генерируется файл `abstract.dvi`, который можно распечатать при помощи `dvi`-драйвера. На рис. 13.14 показан результат применения этого скрипта к базе данных `bsample.bib`, приведенной на рис. 13.4.

Bibliography files
bsample
August 22, 1993

References

[Dynatext] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

[EVH:Office] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

[Eijkhout:1991] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

[Felici:1991] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

[Knuth:WEB] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

[Liang:1983] Franklin Mark Liang. *Word Hy-phen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

[Mittelbach/Schoepf:1990] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

[Roth:postscript] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988.

[Wood:color] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [?], 1988.

[Yannis:1991] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

Рис. 13.14. Распечатка базы данных `bsample.bib`, полученная при помощи `printbib`

Автор второго набора инструментов для работы с BibTeX'овскими базами данных — Нельсон Биби из университета штата Юта (Utah University). Опишем вкратце каждый из элементов этого набора.

bibclean Данная программа одновременно служит для получения хорошо оформленных распечаток BibTeX'овских баз данных, для синтаксической проверки записей и проведения их лексического анализа [6]. Программа написана для платформ UNIX, Vax/VMS и MS-DOS, имеет массу опций, но в принципе, чтобы использовать ее, достаточно набрать:

```
bibclean < bibfile1 bibfile2,... > outfile
```

bibextract Извлекает из списка BibTeX'овских файлов те записи, которые соответствуют паре заданных регулярных выражений и направляет их в *stdout* вместе со всеми командами `@Preamble` и `@String`. Первое из двух регулярных выражений, которые должны быть заданы, формирует выборку по ключевым словам (если это выражение не задано, берутся все записи), а второе служит для извлечения из нее подвыборки по тем или иным значениям других полей. В регулярных выражениях допускаются только строчные буквы.

Так, например, следующая команда извлечет все записи, у которых хотя бы в одном из полей встречается слово «PostScript»:

```
bibextract "" "postscript" bibfile(s)>new-bibfile
```

а команда, приведенная далее, выделит среди всех записей те, у которых в поле `author` или в поле `organization` встречается слово `Adobe`:

```
bibextract "author|organization" "adobe" bibfile(s)>new-bibfile
```

Отметим, что, для того чтобы программа bibextract правильно произвела выборку, может потребоваться вначале «почистить» `.bib`-файлы при помощи программы bibclean.

citefind и citetags Иногда бывает нужно выделить из нескольких больших BibTeX'овских баз данных те публикации, на которые реально имеются ссылки в данной публикации. Решению этой задачи служат скрипты citefind и citetags, написанные на Bourne shell и использующие утилиты awk и sed. Вначале citetags извлекает BibTeX'овские ссылочные метки из LaTeX'овского исходного файла или из `.aux`-файла и посылает их в файл стандартного вывода *stdout*. Там они подхватываются программой citefind, которая пытается отыскать заданные ключевые слова в имеющихся `.bib`-файлах. Получаемый в результате новый библиографический файл записывается в *stdout*. Это выглядит, например, так:

```
citetags *.aux | citefind - mybib1.bib mybib2.bib > newbib.bib
```

bibsort.sh Программа citefind выводит на печать данные о публикациях в том порядке, в котором они фигурируют в `.bib`-файлах. В то же время с точки зрения удобства для пользователя может оказаться полезным переупорядочить публикации. Это можно сделать при помощи shell-скрипта bibsort.sh, который использует утилиту sort и поддерживает большинство ее опций.

bibview Для тех, кто использует графический интерфейс X-window, работа с BibTeX'овскими базами данных упростится благодаря программе bibview.

Большое число BibTeX'овских баз данных для TeX'а создал и поддерживает Нельсон Биби. Кроме того, он является автором многих разработок в области графики, а также ряда стилей, уже упоминавшихся в табл. 13.1. В том, что касается TeX'а, к наиболее интересным из созданных им .bib-файлов относятся `texbook1.bib` и `texbook2.bib` (книги о TeX'е, METAFONT'е и программах-спутниках), `gut.bib` (содержание французского журнала *Cahiers Gutenberg*), `komoedie.bib` (содержание немецкого журнала *Die TEXnische Komödie*), `texgraph.bib` (о том, как научить TeX работать с графикой), `texjourn.bib` (список журналов, предлагающих авторам готовить рукописи своих статей в TeX'е), `tugboat.bib` (полный список статей, напечатанных в *TUGboat*), `type.bib` (статьи и книги о книгопечатании) и `standard.bib` (стандарты в области программного обеспечения).

Помимо всего перечисленного, Нельсон Биби создал пакеты bibmods и showtags. Понять, как работают эти пакеты и как выглядят генерируемые ими документы, можно из рис. 13.15, на котором показан результат применения связки LaTeX-BibTeX к следующему ниже файлу с использованием базы данных с рис. 13.4:

```
\documentclass[twocolumn]{article}

%%%%% Использование пакетов bibmods, bibnames и showtags %%%%%%
\usepackage{bibmods,bibnames,showtags}

\begin{document}
  \nocite{*}
  \bibliographystyle{is-alpha}
  \bibliography{bsample}
\end{document}
```

13.5 Формат .bib-файлов

В этом разделе дается детальное описание формата записей в BibTeX'овской базе данных. Материал приложения B книги LaTeX book [**Л 140–7**] обновляется, благодаря описанию версии 0.99c BibTeX'а [58] (автор — Орен Паташник).

13.5.1 Общая структура записей в BibTeX'овской базе данных

Запись в BibTeX'овской базе данных состоит из трех основных элементов: указатель *типа*, за которым следует *ключевое слово*, и, наконец, сами *данные* о публикации [**Л 140–1**]. Последний элемент представляет собой набор *полей*, записи в

References

Dynatext

[Dyn91] Electronic Book Technology Inc., Providence, Rhode Island. *Dynatext, Electronic Book Indexer/Browser*, 1991.

Eijkhout:1991

[Eij91] Victor Eijkhout. *TEX by Topic, a TEXnicians Reference*. Addison-Wesley, Reading, Massachusetts, 1991.

Felici:1991

[Fel91] James Felici. PostScript versus TrueType. *Macworld*, 8:195–201, September 1991.

Yannis:1991

[Har91] Yannis Haralambous. TEX and those other languages. In Hope Hamilton, editor, *1991 Annual Meeting Proceedings, Part 2, TEX Users Group, Twelfth Annual Meeting, Dedham, Massachusetts, July 15–18, 1991*, volume 12, pages 539–548, Providence, Rhode Island, December 1991. TEX Users Group.

Knuth:WEB

[Knu83] Donald E. Knuth. The WEB System of Structured Documentation. Technical Report STAN-CS-83-980, Department of Computer Science, Stanford University, Stanford, CA 94305, September 1983.

Liang:1983

[Lia83] Franklin Mark Liang. *Word Hyphen-a-tion by Com-pu-ter*. PhD thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Stanford University, Department of Computer Science Report No. STAN-CS-83-977.

Mittelbach/Schoepf:1990

[MS90] Frank Mittelbach and Rainer Schöpf. The New Font Selection — User Interface to Standard LATEX. *TUGboat*, 11(2):297–305, 1990.

Roth:postscript

[Rot88] Stephen E. Roth, editor. *Real World PostScript*. Addison-Wesley, Reading, Massachusetts, 1988. ISBN 0-201-06663-7.

EVH:Office

[vH88] Eric van Herwijnen. Future Office Systems Requirements. Technical report, CERN DD Internal Note, November 1988.

Wood:color

[Woo88] Pat Wood. *PostScript Color Separation*, pages 201–225. In Roth [Rot88], 1988. ISBN 0-201-06663-7.

Рис. 13.15. Распечатка базы данных `bsample.bib`, полученная при помощи showtags и bibmods

которых могут иметь одну или две формы. Приведем общее схематическое представление формата и пример:

```
@type_specifier{key_identifier,
  field_name_1 = "field_text_1",
  field_name_2 = {field_text_2},
        . . .
  field_name_n = {field_text_n}
           }
```

```
@book{lamport86,
  author = "Leslie Lamport",
  title  = "{\LaTeX{}} A Document
            Preparation system",
  publisher = {Addison-Wesley},
  year   = 1986                }
```

Запятая служит разделителем полей. Пробелы вокруг знаков равенства или запятых игнорируются. В текстовой части поля (заключаемой в двойные кавычки либо в фигурные скобки) может находиться любая последовательность символов, но при условии сбалансированности правых и левых фигурных скобок. Кавычки или фигурные скобки необязательны, если текст целиком состоит из цифр (как `year` в приведенном примере)

BibTeX *не делает различия* между прописными и строчными буквами в указателе типа, ключевом слове и именах полей. Необходима, однако, определенная аккуратность в отношении ключевого слова: поскольку LaTeX различает верхний и нижний регистр в ключевых словах, заданных как аргумент команды `\cite`, ключевое слово в базе данных должно быть идентичным соответствующему ключевому слову в LaTeX'овском файле (см. разд. 13.1).

13.5.2 Текстовая часть поля

Текстовые части полей в BibTeX'овской базе данных заключаются в парные двойные кавычки или фигурные скобки. Мы говорим, что часть текста *заключена в фигурные скобки*, если она находится внутри сбалансированной пары фигурных скобок, отличной от той пары, в которую заключена вся запись.

Структура имен

Поля `author` и `editor` содержат [$\mathcal{L}$ 141–2] (см. также [40, с. 379–80]) перечисление имен [4]. Формат, в котором имена появляются на печати, определяется библиографическим стилем. Запись же в `.bib`-файле просто сообщает BibTeX'у точное имя. Имя следует набирать в точности в том виде, в котором оно фигурирует в цитируемой публикации, даже если в другой работе оно выглядит немного иначе. Вот пример такой ситуации:

```
author = "Donald E. Knuth"          author = "D. E. Knuth"
```

Если вы уверены, что эти два автора — одно и то же лицо, можете в обоих случаях использовать одну и ту же форму записи, например, ту, которую пред-

[4] Разумеется, имеются в виду не только личные имена, но и, в первую очередь, фамилии.— *Прим. перев.*

почитает сам автор (в данном случае — Donald E. Knuth). Тем не менее во втором случае вы обязательно должны при этом указать, что в оригинале имя автора написано по-другому.

Встретив запись вида

```
author = "D[onald] E. Knuth"
```

BibTeX при упорядочении публикаций по авторам в алфавитном порядке проигнорирует текст в квадратных скобках, так что никакой путаницы не возникнет.

Большинство имен можно набирать в двух эквивалентных форматах:

```
"John Chris Smith"                    "Smith, John Chris"
"Thomas von Neumann"                  "von Neumann, Thomas"
```

Вторым форматом (содержащим запятую) следует всегда пользоваться в тех случаях, когда имя автора состоит из нескольких частей, каждая из которых начинается с прописной буквой. Вот пример:

```
"Lopez Fernandez, Miguel"
```

Если набрать `"Miguel Lopez Fernandez,"` BibTeX воспримет `"Lopez"` как второе имя, что в данном случае неверно. Если с прописной буквы пишется только одна часть полного имени (например `Johann von Bergen` или же `Pierre de la Porte`), то эта проблема не возникает.

Чтобы сгруппировать несколько слов, входящих в полное имя, достаточно заключить их в фигурные скобки: как видно из следующих ниже примеров, BibTeX рассматривает текст, заключенный в фигурные скобки, как одно имя. Вот один пример:

```
{{Boss and Friends, Inc.} and {Snoozy and Boys, Ltd.}}
```

В данном случае фигурные скобки не позволяют BibTeX'у ошибочно истолковать `Inc.` и `Ltd.` как личные имена.

Вообще говоря, BibTeX'овское имя может состоять из четырех различных частей, которые называются `First`, `von`, `Last` и `Jr` (имя, частица, фамилия, указание старшинства). Каждая часть в общем случае представляет собой последовательность элементов-имен, причем любая из них, за исключением `Last`, может не содержать ни одного элемента.

Таким образом, два следующих имени считаются различными:

```
"von der Schmidt, Alex"          "{von der Schmidt}, Alex"
```

Первое из них содержит части `von`, `Last` и `First`, тогда как второе — только `First` (`Alex`) и `Last` (`von der Schmidt`). Эта разница может сказаться на том, на какое место в списке попадет соответствующая публикация в результате сортировки.

С частью указания старшинства может быть связана некоторая трудность. В большинстве случаев люди, имеющие в составе своего имени «Jr.», отделяют его от фамилии запятой, вот так:

```
"Smith, Jr., Robert"
```

Однако некоторые предпочитают не пользоваться запятой. В таком случае «Jr.» трактуется как часть фамилии:

```
"{Lincoln Jr.}, John P."        "John P. {Lincoln Jr.}"
```

Аналогичным образом трактуется и «Miguel Lopez Fernandez» (см. выше):

```
"Lopez Fernandez, Miguel"
```

Здесь часть `First` содержит единственный маркер «Miguel», часть `Last` содержит два маркера, «Lopez» и «Fernandez», а части `von` и `Jr` пусты.

А вот пример посложнее:

```
"Johannes Martinus Albertus van de Groene Heide"
```

Здесь полное имя содержит три маркера в части `First`, два — в части `von` и два — в части `Last`. BibTeX знает, где заканчивается одна часть и начинается другая, так как маркеры в части `von` (в данном случае это — `van de`) начинаются со строчных букв.

Вообще, у маркеров части `von` первая буква на нулевом уровне групповой вложенности — строчная. Несколько забегая вперед, отметим следующую техническую деталь: содержимое «специального символа» всегда находится на нулевом уровне вложенности (см. с. 449). Благодаря этому можно воздействовать на восприятие того или иного маркера BibTeX'ом, используя искусственный специальный символ, первую букву которого, следующую за TeX'овской командой, можно сделать прописной или строчной в зависимости от того, что требуется. Например, при использовании записи

```
Maria {\uppercase{d}e La} Cruz
```

частицы De La (которые на печати начинаются с прописных букв!) воспринимаются BibTeX'ом как часть `von`, поскольку первая буква, следующая за командой, — строчная. В результате BibTeX в сочетании со стилем `abbrev` даст правильное сокращение M. De La Cruz, тогда как без использования вышеуказанного приема получился бы ошибочный вариант M. D. L. Cruz.

BibTeX правильно обрабатывает имена, содержащие дефис. Например, при использовании стиля `abbrev` из записи вида

```
author = "Maria-Victoria Delgrande",
```

на печати получается M.-V. Delgrande.

При наличии у публикации нескольких авторов их «имена» следует отделять друг от друга словом **«and»**, которое при этом не должно находиться внутри фигурных скобок.

```
author = "Frank Mittelbach and Rowley, Chris"
editor = "{Lion and Noble, Ltd.}"
```

Согласно этой записи авторов двое — Франк Миттельбах и Крис Роули, а редактор один, так как "and" находится внутри фигурных скобок. Если авторов или редакторов у публикации так много, что полностью перечислить их затруднительно, можно ограничиться сокращенным списком, который должен заканчиваться словами "and others,". При использовании стандартных стилей вместо этих слов на печати появится хорошо всем знакомое *«et al.»*.

Итак, BibTeX умеет работать с именами, заданными в одном из следующих трех форматов (во всех трех случаях можно использовать как двойные кавычки, так и фигурные скобки):

```
"First von Last"          например, {Johan van der Winden}
"von Last, First"         например, "von der Schmidt, Alexander"
"von Last, Jr, First"     например, {de la Porte, Fils, {\'Emile}}
```

Первый формат можно использовать почти всегда. Однако он неприменим при наличии части Jr, а также в том случае, когда часть Last состоит из нескольких маркеров, а часть von отсутствует.

Формат заглавий

Будут ли использоваться прописные буквы при написании заглавий в списке литературы — зависит, вообще говоря, от выбранного библиографического стиля [$\mathcal{L}$ **142–3**]. Обычно в заглавиях книг слова (кроме служебных) начинаются с прописных букв, тогда как в заглавиях статей все слова пишутся строчными буквами[5]. Заглавие следует набирать, в точности следуя его написанию в оригинале, например, так:

```
TITLE = "A Manual of Style"
TITLE = "Hyphenation patterns for ancient Greek and Latin"
```

В различных языках и библиографических стилях действуют свои правила употребления прописных букв в заглавиях. Если требуется отступить от принятого в используемом стиле формата, следует прибегнуть к помощи фигурных скобок, заключая в них то, что должно на печати выглядеть так же, как в исходном тексте. Следующие две записи эквивалентны:

```
TITLE = "The Towns and Villages of {Belgium}"
TITLE = {The Towns and Villages of {B}elgium}
```

[5] Речь идет, разумеется, о публикациях на английском языке. — *Прим. перев.*

Акцентированные и специальные символы

BibTeX воспринимает акцентированные символы. Например, при использовании библиографического стиля alpha из записи с полями

```
author = "Kurt G{\"o}del",
year = 1931,
```

будет сгенерирована надлежащая метка [Göd31]. Приведенный пример иллюстрирует следующее важное обстоятельство: весь текст, относящийся к акцентированному символу, должен быть заключен в фигурные скобки. В данном случае наряду с {\"o} можно использовать запись {\"{o}}. При этом скобки, выделяющие акцентированный символ, не должны сами находиться внутри каких-либо других фигурных скобок за исключением разве что тех, в которые может быть заключено целиком поле или целиком вся запись; текст внутри скобок, представляющий акцентированный символ, должен начинаться с \. Иначе говоря, не допустимо ни {G{\"{o}}del}, ни {G\"{o}del}.

При помощи описанного подхода обеспечивается корректная работа BibTeX'а со стандартными акцентированными и диакритическими символами, используемыми в LaTeX'овских документах, а также разнообразными «акцентами» и «диакритиками», которые могут вводиться пользователем. Для правильного подсчета числа букв в метках BibTeX рассматривает содержимое фигурных скобок как одну букву.

Для BibTeX'а акцентированный символ — это частный случай так называемого «специального символа». По определению, специальным символом является весь текст, начиная с левой фигурной скобки самого верхнего уровня, за которой сразу же идет знак \, и заканчивая парной ей правой фигурной скобкой. Например, запись

```
author = {\OE{le} {\'{E}mile} {Ren\'{e}} van R{\i\j}den}
author = "\OE{le} {\'{E}mile} {Ren\'{e}} van R{\i\j}den"
```

содержит два специальных символа — {\'{E}mile} и {\i\j}.

Вообще говоря, BibTeX не обрабатывает TeX'овские или LaTeX'овские команды, находящиеся внутри специального символа, однако обрабатывает весь остальной текст. Так, например, при использовании стиля, переводящего все заглавия в нижний регистр (т. е. печатающего их строчными буквами), BibTeX преобразует запись

```
The {\TeX BOOK\NOOP} Saga в The {\TeX book\NOOP} saga.
```

При этом артикль The в преобразованном тексте по-прежнему начинается с прописной буквы, так как является первым словом заглавия.

Как уже говорилось, на идее специальных символов основана работа BibTeX'а с акцентированными символами. О том, как при этом заставить BibTeX надлежащим образом производить упорядочение по алфавиту, см. описание команды

\SortNoop 451. Кроме того, поскольку BibTeX считает специальный символ одной буквой, данный подход позволяет увеличивать число символов в метках.

13.5.3 BibTeX и аббревиатуры

При заполнении текстовых полей в BibTeX'овском файле можно пользоваться сокращениями [$\mathcal{L}$ **143–4**] (см. также [40, с. 382]). Аббревиатура — это последовательность символов, начинающаяся с буквы и не содержащая пробелов, а также ни одного из следующих десяти символов:

```
" # % ' ( ) , = { }
```

Аббревиатуры, используемые в текстовых полях BibTeX'овских записей, не следует заключать в фигурные скобки или кавычки. Если для

```
Communications of the ACM
```

ввести аббревиатуру `cacm`, то поле `journal` можно будет заполнять следующими двумя эквивалентными способами:

```
journal = "Communications of the ACM"
journal = cacm
```

Аббревиатуры определяются пользователем в `.bib`-файле при помощи команды `@STRING`. Вот некоторые примеры того, как это делается:

```
@string{AW          = "Addison--Wesley Publishing Company"}
@STRING{CACM        = "Communications of the ACM"}
@String{pub-AW      = {{Ad\-di\-son-Wes\-ley}}}
@String{pub-AW:adr  = "Reading, MA, USA"}
@String{TUG         = "\TeX{} Users Group"}
@String{TUG:adr     = {Providence, RI, USA}}
```

BibTeX не делает различия между прописными и строчными буквами в аббревиатурах (т. е. отождествляет `CACM` и `cacm`), однако выдает предупреждение, когда прописные буквы используются вперемешку со строчными. Сама команда `@STRING` может быть записана строчными буквами, прописными или и теми и другими в любой комбинации. Команды `@STRING` могут размещаться где угодно в `.bib`-файле, лишь бы определение аббревиатуры предшествовало ее использованию. Можно рекомендовать записать все команды `@STRING` в начале `.bib`-файла или же завести для них отдельный `.bib`-файл. Команды `@STRING` в `.bib`-файле имеют приоритет перед определениями в стилевых файлах.

Несколько символьных последовательностей (или несколько аббревиатур, введенных командами `@STRING`) можно соединить в одну (конкатенировать) при

помощи оператора конкатенации #. Например, для аббревиатуры, определенной следующим образом:

```
@STRING{TUB = {TUGboat }}
```

удобно при помощи этого оператора производить модификацию поля journal для различных публикаций:

```
@article(tub-86,
  journal = TUB # 1986,
  . . .
@article(tub-87,
  journal = TUB # 1987,
  . . .
```

В большинстве библиографических стилей содержатся готовые наборы аббревиатур. Принято использовать трехбуквенные сокращения для месяцев: jan, feb, mar и т. д. В целях поддержки единого стандарта следует использовать именно эти сокращения, а не писать названия месяцев полностью. При необходимости указать точную дату лучше всего, используя конкатенацию, включить число в поле month, вот так:

```
month = apr # "~1,"
```

Большинство библиографических стилей содержит также аббревиатуры названий наиболее известных журналов в соответствующей области — их можно найти в прилагаемой к данному стилю документации. Чтобы пополнить эти наборы своими собственными аббревиатурами, достаточно поместить в свой библиографический файл соответствующие команды @STRING, не забыв указать этот файл в качестве одного из аргументов LaTeX'овской команды \bibliography.

13.5.4 BibTeX'овская преамбула

В BibTeX'е имеется команда @PREAMBLE. Она имеет примерно тот же синтаксис, что команда @STRING, но, в отличие от последней, не содержит ни имен-идентификаторов, ни знаков равенства, а одни лишь последовательности символов. Вот пример:

```
@preamble{ "\providecommand{\SortNoop}[1]{} "
        # "\providecommand{\OneLetter}[1]{#1} "
        # "\providecommand{\SwapArgs}[2]{#2#1} " }
```

Здесь команда @PREAMBLE при помощи все того же символа # производит конкатенацию нескольких определений. При использовании стандартных библиографических стилей аргумент команды @PREAMBLE без каких-либо изменений переносится в .bbl-файл, и когда LaTeX затем прочитывает этот файл, он усваивает и данные определения.

Так, например, определенную выше команду \SortNoop можно использовать для управления BibTeX'овским алгоритмом сортировки. Обычно результаты работы этого алгоритма вполне приемлемы, но в некоторых случаях может потребоваться заменить BibTeX'овскую сортировку своей собственной, осуществляемой по некоторому другому правилу. Такая потребность может возникнуть при работе с языками, отличными от английского, где действуют свои правила сортировки, или же в ситуации, когда нужно перечислить отдельные тома некоторой книги в порядке, определяемом датой первоначальной публикации каждого тома, вне зависимости от дат его последующих переизданий.

Предположим, например, что первое издание первого тома книги появилось в 1986 г., второе издание — в 1991 г., а второй том вышел в 1990 г. Воспользуемся следующей конструкцией:

```
volume=1, year = "{\SortNoop{86}}1991"
  .  .  .
volume=2, year = "{\SortNoop{90}}1990"
```

Согласно данному выше определению команды \SortNoop, LaTeX просто отбросит ее аргумент, и в списке литературы появятся только годы последнего издания каждого тома. Однако с точки зрения BibTeX'а команда \SortNoop определяет «акцентированный символ», значениями поля year оказываются числа 861991 и 901990, и в результате сортировки по значениям этого поля том 1 будет указан в списке литературы раньше, чем том 2, что и требовалось.

13.5.5 Перекрестные ссылки

Между элементами BibTeX'овской базы данных можно устанавливать перекрестные ссылки. Допустим, что в основном документе имеется ссылка \cite{Wood:color}, а в библиографической базе данных есть следующие две записи:

```
@Inbook{Wood:color, author = {Pat Wood}, crossref={Roth:postscript},
        title = {PostScript Color Separation}, pages={201--225}}
@Book{Roth:postscript, editor = {Stephen E. Roth}, title=
      {{Real World PostScript}} , booktitle={{Real World PostScript}},
      publisher=AW , address=AW:adr , year=1988, ISBN={0-201-06663-7}}
```

У первой записи, которая относится к публикации Wood:color, имеется специальное поле crossref. Находящаяся в нем ссылка сообщает BibTeX'у, что за содержимым полей, отсутствующих в этой записи, нужно обратиться ко второй записи, т. е. к публикации Roth:postscript. Если окажется, что такая «перекрестная» ссылка на Roth:postscript есть у двух или большего числа публикаций, на которые в основном документе имеются ссылки вида \cite или \nocite (т. е., помимо Wood:color, хотя бы еще один цитируемый в документе источник ссылается на Roth:postscript), то публикация Roth:postscript будет автома-

тически внесена в список литературы, даже если непосредственной ссылки на него ни одна из команд `\cite` или `\nocite` не содержит.

Публикация, на которую имеются перекрестные ссылки, должна располагаться в базе данных «дальше», чем любая из публикаций, ссылающаяся на нее. Целесообразно, например, размещать публикации, на которые есть перекрестные ссылки, в самом конце базы данных. Существенно, что публикация, на которую имеется перекрестная ссылка, не может в свою очередь содержать ссылку на другую публикацию.

Некоторые поля записей BibTeX'овской базы данных могут содержать LaTeX'овскую команду `\cite`. Это дает возможность, например, вводить дополнительную информацию типа замечаний:

```
note = "See Eijkhout~\cite{Eijkhout:1991} for more details"
```

Следует, однако, отдавать себе отчет в том, что использование этого приема может потребовать дополнительных прогонов LaTeX'а и BibTeX'а. Нужда в них возникает, если ссылка, перемещаемая из BibTeX'овской базы данных в `.bbl`-файл, содержит ключевое слово, на которое не было ссылок в основном документе. В этом случае LaTeX не сможет оформить данную ссылку за один прогон.

13.5.6 Дополнительные замечания

Источники с одинаковыми ключами сортировки будут перечислены в списке литературы в порядке их упоминания в основном документе. Ключ сортировки конструируется стилевым библиографическим файлом и обычно содержит информацию об авторе, год издания работы и ее название.

LaTeX'овский символ комментария % не является таковым внутри файлов библиографических баз данных (`.bib`-файлов).

Для того чтобы не дать возможности BibTeX'у перевести некоторый текст в нижний регистр (т. е. переписать его строчными буквами), вообще говоря, следует заключить этот текст в фигурные скобки. Указанный прием, однако, не срабатывает, если первым символом, стоящим сразу же после левой фигурной скобки, оказывается \ (см. с. 449).

13.6 Формат записей базы данных (подробное описание)

Как уже говорилось в разд. 13.2, каждый библиографический источник должен быть отнесен к определенному типу, а информация о нем должна быть распределена между вполне определенными полями.

Итак, первым делом надо решить, к какому типу относится данная публикация. Хотя в конечном счете никакая классификация не является исчерпывающей,

можно научить BibTeX достаточно хорошо справляться с самыми разными типами публикаций. Для совсем уж нестандартных случаев можно посоветовать не придавать большого значения BibTeX'овским предупреждающим сообщениям (см. ниже).

В большинстве BibTeX'овских стилей публикация может быть отнесена к одному из тринадцати типов [Л 144–6] (см. также [40, с. 384–6]), перечисленных в табл. 13.2. Для каждого типа требуется свой набор данных о публикации: ссылка на журнальную статью должна содержать сведения о томе и номере журнала, а применительно к книге этих данных, естественно, не требуется. Как следствие, разным типам записей в базе данных отвечают разные наборы полей. Однако независимо от типа публикации поля бывают трех категорий:

обязательные (required) Если такое поле оказалось незаполненным, BibTeX выдаст предупреждающее сообщение и, вполне вероятно, неправильно отформатирует соответствующий элемент списка литературы. Если запрашиваемая информация на самом деле несущественна, значит, неправильно определен тип публикации. Если же требуемая информация важна, но, например, содержится в другом поле, предупреждение можно проигнорировать.

необязательные (optional) В случае заполненности поля содержащиеся в нем данные будут использованы, но данное поле вполне можно и не заполнять, не опасаясь за формат публикации в списке литературы. Рекомендуется добавлять необязательные поля, если соответствующая информация может быть полезна читателю.

игнорируемые (ignored) Информация, содержащаяся в таком поле, не используется. BibTeX игнорирует все поля, не входящие в первые две категории, поэтому в принципе в записи в `.bib`-файле можно вводить дополнительно любые поля. Представляется разумным включать в соответствующую данной публикации запись в `.bib`-файле всю имеющуюся о нем информацию, даже ту, которая не имеет шансов попасть когда-нибудь в список литературы. Так, например, аннотацию статьи вполне имеет смысл включить в поле `abstract` записи, относящейся к данной статье. Можно сказать, что как место хранения аннотаций `.bib`-файл не хуже любого другого, а с другой стороны, существуют специальные библиографические стили для получения распечаток аннотаций (например, стиль abstract, упомянутый в табл. 13.1).

В табл. 13.2 представлены стандартные типы публикаций, принятые в стандартных библиографических стилях, с перечислением обязательных и необязательных полей для каждого типа. Поля внутри каждой из категорий (обязательные и необязательные) перечислены в том порядке, в котором данные из этих полей следуют друг за другом на печати. Заметим, что для некоторых типов публикаций алфавитный порядок следования публикаций в списке литературы может слегка нарушаться в случае незаполненности некоторых полей. Значения названий полей разъясняются в табл. 13.3. В нестандартных библиографических стилях содержимое некоторых необязательных полей может при создании списка

`article`	Статья в журнале. *Обязательные*: `author`, `title`, `journal`, `year`. *Необязательные*: `volume`, `number`, `pages`, `month`, `note`.
`book`	Книга (при наличии данных об издательстве). *Обязательные*: `author` или `editor`, `title`, `publisher`, `year`. *Необязательные*: `volume` или `number`, `series`, `address`, `edition`, `month`, `note`.
`booklet`	Отпечатанный и переплетенный текст без указания издательства или организации-спонсора. *Обязательные*: `title`. *Необязательные*: `author`, `howpublished`, `address`, `month`, `year`, `note`.
`inbook`	Часть книги (глава, раздел и т. п. и/или указанный диапазон номеров страниц). *Обязательные*: `author` или `editor`, `title`, `chapter` и/или `pages`, `publisher`, `year`. *Необязательные*: `volume` или `number`, `series`, `type`, `address`, `edition`, `month`, `note`.
`incollection`	Часть книги, имеющая отдельное заглавие. *Обязательные*: `author`, `title`, `booktitle`, `publisher`, `year`. *Необязательные*: `editor`, `volume` или `number`, `series`, `type`, `chapter`, `pages`, `address`, `edition`, `month`, `note`.
`inproceedings`	Статья в сборнике трудов конференции. *Обязательные*: `author`, `title`, `booktitle`, `year`. *Необязательные*: `editor`, `volume` или `number`, `series`, `pages`, `address`, `month`, `organization`, `publisher`, `note`.
`manual`	Техническая документация. *Обязательные*: `title`. *Необязательные*: `author`, `organization`, `address`, `edition`, `month`, `year`, `note`.
`mastersthesis`	Диссертация на соискание Master's Degree. *Обязательные*: `author`, `title`, `school`, `year`. *Необязательные*: `type`, `address`, `month`, `note`.
`misc`	Рекомендуется использовать этот тип, когда никакой другой не подходит. Предупреждение будет выдано, только если все необязательные поля пусты (т. е. если пусты вообще все поля). *Обязательные*: — *Необязательные*: `author`, `title`, `howpublished`, `month`, `year`, `note`.
`phdthesis`	Диссертация на соискание Philosophical Degree. *Обязательные*: `author`, `title`, `school`, `year`. *Необязательные*: `type`, `address`, `month`, `note`.
`proceedings`	Сборник трудов конференции. *Обязательные*: `title`, `year`. *Необязательные*: `editor`, `volume` или `number`, `series`, `address`, `publisher`, `note`, `month`, `organization`.
`techreport`	Отчет, выпущенный учебным заведением или другой организацией (как правило, с указанием его номера в соответствующей серии). *Обязательные*: `author`, `title`, `institution`, `year`. *Необязательные*: `type`, `number`, `address`, `month`, `note`.
`unpublished`	Документ, имеющий автора и заглавие, но формально не считающийся опубликованным. *Обязательные*: `author`, `title`, `note`. *Необязательные*: `month`, `year`.

Таблица 13.2. Список типов публикаций, принятых в большинстве BibTeX'овских стилей

`address`	Обычно — адрес издательства (`publisher`) или другой организации. Если издательство известное, достаточно указать город, если небольшое — не лишним будет его полный адрес.
`annote`	Аннотация. Не используется в стандартных библиографических стилях, однако используется в стилях, предназначенных для создания аннотированной библиографии (например, `annote`). Текст в этом поле всегда должен начинаться с прописной буквы, поскольку начинает новое предложение.
`author`	Имя(-ена) автора(-ов) в BibTeX'овском формате имен (см. разд. 13.5.2).
`booktitle`	Заглавие книги, на некоторую часть которой имеется ссылка (см. разд. 13.5.2). В случае, когда тип публикации — `book`, заглавие пишется в поле `title`.
`chapter`	Номер (главы, раздела и т. п.).
`crossref`	Ключевое слово той записи в базе данных, на которую имеется перекрестная ссылка (см. разд. 13.5.5).
`edition`	Порядковый номер издания книги, например «Second». Представляет собой порядковое числительное, начинающееся с прописной буквы (как в данном случае). В стандартных стилях при необходимости прописная буква будет заменена на строчную.
`editor`	Имя(-ена) редактора(-ов) в BibTeX'овском формате имен. В случае, если присутствует также поле `author`, в данном поле указывается редактор книги или сборника, куда входит цитируемая работа.
`howpublished`	Для тех случаев, когда издание представляет из себя что-то необычное.
`institution`	Организация-спонсор при издании технического отчета.
`journal`	Название журнала. Для многих журналов используются аббревиатуры (см. разд. 13.5.3).
`key`	Вспомогательное ключевое слово, используемое для упорядочения по алфавиту, для создания перекрестных ссылок, а также для создания меток при незаполненных полях `author` и `editor`. Не следует путать содержимое этого поля с ключевым словом, фигурирующим в команде `\cite` и в самом начале записи в базе данных.
`month`	Месяц как часть даты публикации или же, если работа не опубликована, даты написания. Из соображений единообразия следует использовать трехбуквенные сокращения: `jan`, `feb`, `mar` и т. д. (см. разд. 13.5.3).
`note`	Любая дополнительная информация, которая может быть полезна читателю.
`number`	Порядковый номер журнала, технического отчета или работы, изданной в рамках некоторой серии публикаций. Обычно журнальные выпуски различаются томом и номером, отчет, как правило, имеет номер, а иногда номера присваиваются и книгам, выходящим в серии, имеющей отдельное название.
`organization`	Организация, финансировавшая проведение конференции или издание технической документации (типа `manual`).
`pages`	Номера страниц. Допускается перечисление (одного или нескольких) номеров, а также указание некоторого диапазона. Например, возможны такие варианты: `42--111`, или `7,41,73--97`, или же `43+` (знак '`+`' используется, если номера страниц не образуют сплошной интервал).
`publisher`	Название издательства.
`school`	Название учебного заведения, в котором была написана диссертация.

окончание см. на след. стр.

Таблица 13.3. Список стандартных полей в BibTeX'овском файле

окончание (см. предыдущую страницу)	
`series`	Название книжной серии или многотомника. При ссылках на книгу ее собственное название берется из поля `title`, а в необязательном поле `series` может быть указано название серии или многотомника, в которые входит цитируемая книга.
`title`	Название работы. При его наборе следует придерживаться правил, описанных в разд. 13.5.2.
`type`	Название типа (разновидности) отчета, например Research Note. Это название появится вместо принятого по умолчанию Technical Report. Для публикаций типа `phdthesis` можно использовать название «Ph.D. dissertation», написав: `type = "{Ph.D.} dissertation."` Аналогично, для публикаций типа `inbook` и `incollection` можно получить на печати «section 1.2» вместо стандартного «chapter 1.2» при помощи указания `chapter = "1.2," type = "Section."`
`volume`	Номер тома журнала или многотомника.
`year`	Год публикации, а для непубликованных работ — год написания. В простейшем случае записывается при помощи четырех цифр (например, 1984), но, вообще говоря, стандартные стили воспринимают в поле `year` любой текст, в котором четыре последних символа, не являющиеся знаками пунктуации, — цифры (например, about 1984).

Таблица 13.3.

литературы игнорироваться. Следует помнить, что в `.bib`-файле названия типов публикаций должны начинаться с символа `at` (см. рис. 13.4).

Помимо полей, перечисленных в табл. 13.2, у записи каждого типа имеется необязательное поле `key`, которое используется некоторыми стилями для упорядочения по алфавиту, работы с перекрестными ссылками, а также для формирования метки `\bibitem`. Поле `key` следует вводить в те записи, в которых отсутствует информация об авторе публикации. Как правило, данные об авторе содержатся в поле `author`, но некоторые стили работают с полем `editor` или полем `organization`. Вот ситуация, в которой поле `key` оказывается полезным:

```
organization = "The Association for Computing Machinery",
key = "ACM"
```

Если бы поля `key` не было, то BibTeX при использовании стиля `alpha` построил бы метку по первым трем буквам текста, содержащегося в поле `organization`. Хотя в стиле и предусмотрено, что артикль `The` должен быть при этом отброшен, в результате была бы получена невразумительная метка [Ass86]. Благодаря полю `key` метка получается гораздо более информативной: [ACM86].

Перейдем к рассмотрению полей [$\mathcal{L}$ **146–7**], с которыми работают стандартные библиографические стили. Эти «стандартные» поля перечислены в табл. 13.3. Прочие поля, наподобие `abstract`, могут понадобиться, если используется какой-либо из нестандартных «расширенных» стилей, приведенных в табл. 13.1. Поскольку нестандартные поля игнорируются при использовании стандартных BibTeX'овских стилей, их можно использовать для включения в библиографическую запись «комментариев»: соответствующий текст должен быть заключен в квадратные скобки и помещен в одно из нестандартных полей.

Что касается названий полей у типов публикаций, приведенных в табл. 13.2, их следует трактовать в самом широком смысле, чтобы иметь возможность использовать их в максимально широком множестве ситуаций. Нужно также иметь в виду, что самые сложные проблемы часто удается решить, используя надлежащим образом поле `note`.

13.7 Как устроены BibTEX'овские стили

В этом разделе в сжатой форме дано введение в язык, используемый BibTEX'овскими стилями. Содержащиеся здесь сведения достаточны для того, чтобы производить небольшие модификации существующих стилевых файлов. Для более глубокого изучения этого вопроса рекомендуется обратиться к основополагающей статье Орена Паташника «Designing BibTEX Styles» [59].

13.7.1 Общее представление о BibTEX'овских стилевых файлах

В BibTEX'овских стилях используется постфиксный стековый язык (типа PostScript), при помощи которого BibTEX'у сообщается, каким должен быть формат элементов будущего списка литературы. Язык содержит десять команд (см. табл. 13.4), предназначенных для работы с объектами языка — константами, переменными, функциями, стеком и перечнем.

BibTEX'у известны функции двух типов: встроенные, являющиеся частью самого BibTEX'а (см. табл. 13.5), и пользовательские, определяемые при помощи одной из команд `MACRO` или `FUNCTION`.

Внутри пары двойных кавычек, ограничивающих постоянные цепочки литер, можно употреблять любые символы. Хотя BibTEX, вообще говоря, нечувствителен к регистру, т. е. не делает различия между строчными и прописными буквами, внутри строк чувствительность к регистру поддерживается. Все пробелы внутри постоянных цепочек литер учитываются, а переход на новую строку внутри постоянной цепочки литер недопустим.

Имена переменных и функций не должны начинаться с цифры и не должны содержать ни один из десяти символов, приведенных на с. 450.

BibTEX нечувствителен к регистру (не различает строчные и прописные буквы) в именах переменных, функций и макрокоманд.

В качестве констант и переменных могут выступать либо целые числа, либо цепочки литер (булевы значения «истинно» и «ложно» записываются как 1 или 0 соответственно).

Есть три типа переменных:

глобальные переменные (global variables) Могут быть либо целочисленными, либо цепочками литер; декларируются при помощи команд `INTEGERS` или `STRINGS` соответственно.

переменные на уровне записи (entry variables) Могут быть либо целочисленными, либо цепочками литер; декларируются при помощи команды ENTRY. Каждая переменная этого типа принимает определенное значение для каждой записи из перечня, прочитанного в BIBTEX'овской базе данных.

поля (fields) Переменные в виде цепочек литер, доступные только для чтения; содержат информацию, взятую из базы данных. Значения этих переменных устанавливаются при помощи команды READ. Как и переменные на уровне записи, каждая переменная поля принимает определенное значение для каждой записи.

13.7.2 Команды BibTEX'овского стилевого файла

Таблица 13.4 содержит краткое описание десяти BIBTEX'овских команд. То, что имена команд даны прописными буквами, не имеет значения, поскольку BIBTEX не различает верхний и нижний регистры.

Рекомендуется (хотя и не является обязательным) оставлять по крайней мере одну пустую строку между командами и не оставлять пустых строк внутри команд. Соблюдение этого правила помогает BIBTEX'у разобраться с встретившимися ему синтаксическими ошибками.

13.7.3 Встроенные функции

В табл. 13.5 дана краткая сводка тридцати семи BIBTEX'овских встроенных функций (более подробные сведения о них можно почерпнуть в [59]). Нетрудно заметить, что если имя команды содержит буквы, то в конце его обязательно стоит знак $.

13.7.4 Стилевой файл-документация btxbst.doc

Согласно схеме, реализованной Ореном Паташником, в основании стандартных BIBTEX'овских стилевых файлов abbrv, alpha, plain и unsrt лежит «обобщенный» файл btxbst.doc, который в то же время представляет собой подробную документацию и может использоваться для углубленного изучения механизма работы BIBTEX'овских стилей.

В стандартных стилях предусмотрены два базовых формата для меток: *alphabetic*, когда метка может иметь вид [Lam84] и *Numeric* для меток вида [34].

Имеется три возможности для перечисления публикаций в списке литературы.

Упорядоченный список, метки в формате *alphabetic* Источники указываются в алфавитном порядке, вначале по меткам, затем по автору(-ам) (или содержимому заменяющего его поля), затем по году публикации, затем по заглавию.

Упорядоченный список, метки в формате *Numeric* Источники перечисляются в алфавитном порядке, вначале по автору(-ам) (или содержимому заменяющего его поля), затем по году публикации, затем по заглавию.

Неупорядоченный список Источники перечисляются в том же порядке, в котором они цитируются в тексте.

Управление при работе со стилевым файлом осуществляют следующие команды, находящиеся в конце файла `btxbst.doc`:

```
EXECUTE {begin.bib}          % Преамбула и команда \begin{thebibliography}
EXECUTE {init.state.consts}  % Инициализация параметров состояния
ITERATE {call.type$}         % Циклический просмотр публикаций для
                             % построения списка литературы
EXECUTE {end.bib}            % Команда \end{thebibliography}
```

Точный смысл этих команд можно уяснить из табл. 13.4 и 13.5.

Стилевой файл начинается с декларации надлежащих полей, которая производится командой `ENTRY`, и переменных цепочек литер, используемых при построении меток.

Процедура обработки очередной записи начинается с вызова функции `output.bibitem`, записывающей соответствующую команду `\bibitem` и ее аргументы в `.bbl`-файл. Затем производится форматирование полей и «печать» их содержимого; это делает функция `output` либо функция `output.check`, которая следит за правильностью использования разделителей (запятых, точек, команд `\newblock`). Наконец, вызывается функция `fin.entry`, которая ставит заключительную точку и этим завершает процедуру.

Далее идут функции, осуществляющие форматирование отдельных частей записи. Для каждого из основных полей есть соответствующая функция. Примером может служить функция `format.names`, которая устанавливает для имен авторов представление «First Von Last, Junior», а также расставляет запятые и пишет «and» перед именем последнего автора (если перечень авторов заканчивается словом `others`, в списке литературы будет напечатано «et al.»). Имеется также функция `format.authors` для списка авторов, функция `format.editors` для списка редакторов (последняя после перечисления редакторов вписывает «, editor» либо «, editors») и т. п.

Следующая часть стилевого файла содержит функции, в которых определяются различные типы публикаций, которые могут встретиться в `.bib`-файле. Речь идет о функциях типа `article` или `book`. Фактически, эти функции и генерируют из данной записи в базе данных тот текст, который записывается в `.bbl`-файл. Они должны предшествовать команде `READ`. При разработке стилевого файла необходимо предусмотреть также функцию `default.type` для неизвестных типов публикаций.

Следующий раздел файла `btxbst.doc` содержит макроопределения для названий месяцев и для ряда известных журналов. В зависимости от стиля используются либо полные, либо сокращенные названия. Раздел завершается командой `READ`, которая вводит записи в `.bib`-файл.

ENTRY {*field_list*} {*integer_variable_list*} {*string_variable_list*}
Декларирует поля и переменные на уровне записи. BibTeX автоматически декларирует одно дополнительное поле crossref, используемое для создания перекрестных ссылок, и дополнительную цепочку литер на уровне записи sort.key$, используемую командой SORT. В каждом стилевом файле должна быть лишь одна команда ENTRY. Например, в стилях alpha и plain эта команда выглядит, соответственно, следующим образом:
`ENTRY { address author booktitle ... } {} { label extra.label sort.label }`
`ENTRY { address author booktitle ... } {} { label }`

EXECUTE {*function_name*} Выполняет некоторую (одну) процедуру. `EXECUTE {begin.bib}`

FUNCTION {*function_name*} {*definition*}
Определяет новую процедуру (функцию). Определение команды FUNCTION не может быть изменено вне стилевого файла.
`FUNCTION {end.bib}`
`{ newline$ "\end{thebibliography}" write$ newline$ }`

MACRO {*macro_name*} {*definition*} Определяет макрокоманду в виде цепочки литер. Определение MACRO может быть изменено вне стилевого файла. `MACRO {feb} {"February"}`
INTEGERS {*global_integer_variable_list*} Декларирует глобальные целочисленные переменные. `INTEGERS { longest.label.width last.extra.num }`
STRINGS {*global_string_variable_list*} Декларирует глобальные переменные — цепочки литер. `STRINGS { longest.label last.sort.label next.extra }`
ITERATE {*function_name*} Однократно выполняет некоторую процедуру для каждой публикации из списка в порядке их перечисления в данном списке. `ITERATE {longest.label.pass}`
REVERSE {*function_name*} Однократно выполняет некоторую процедуру для каждой публикации из списка в порядке, обратном порядку их перечисления в данном списке. `REVERSE {reverse.pass}`
READ Считывает из файла базы данных значения полей для каждой публикации, входящей в список. В каждом стилевом файле должна быть лишь одна команда ENTRY. Команды ENTRY и MACRO должны предшествовать команде READ.
SORT Упорядочивает список публикаций, используя цепочки литер sort.key$.

Таблица 13.4. Список команд BibTeX'овского стилевого файла

$\mathcal{I}_1$ $\mathcal{I}_2$ >		($\mathcal{I}$)	1 (если $\mathcal{I}_1$>$\mathcal{I}_2$) или 0 (в противном случае)
$\mathcal{I}_1$ $\mathcal{I}_2$ <		($\mathcal{I}$)	1 (если $\mathcal{I}_1$<$\mathcal{I}_2$) или 0 (в противном случае)
$\mathcal{I}_1$ $\mathcal{I}_2$ =		($\mathcal{I}$)	1 (если $\mathcal{I}_1$=$\mathcal{I}_2$) или 0 (в противном случае)
$\mathcal{S}_1$ $\mathcal{S}_2$ =		($\mathcal{I}$)	1 (если $\mathcal{S}_1$=$\mathcal{S}_2$) или 0 (в противном случае)
$\mathcal{I}_1$ $\mathcal{I}_2$ +		($\mathcal{I}_1$+$\mathcal{I}_2$)	сложить два целых числа
$\mathcal{I}_1$ $\mathcal{I}_2$ −		($\mathcal{I}_1$-$\mathcal{I}_2$)	вычесть одно целое число из другого
$\mathcal{S}_1$ $\mathcal{S}_2$ *		($\mathcal{S}_1\mathcal{S}_2$)	конкатенация двух цепочек
$\mathcal{L}$ $\mathcal{V}$:=			присвоить переменной $\mathcal{V}$ значение $\mathcal{L}$
$\mathcal{S}$	`add.period$`	($\mathcal{S}$.)	добавить точку к цепочке (не производится, если цепочка заканчивается *одним из символов* '.', '?' *или* '!')
	`call.type$`		выполнить процедуру, имя которой совпадает с названием типа публикации, например, `book`
$\mathcal{S}$ "t"	`change.case$`	($\mathcal{S}$)	перевести все литеры цепочки $\mathcal{S}$, кроме первой, в нижний регистр
$\mathcal{S}$ "l"	`change.case$`	($\mathcal{S}$)	перевести все литеры цепочки $\mathcal{S}$ в нижний регистр
$\mathcal{S}$ "u"	`change.case$`	($\mathcal{S}$)	перевести все литеры цепочки $\mathcal{S}$ в верхний регистр
$\mathcal{S}$	`chr.to.int$`	($\mathcal{I}$)	заменить литеру в цепочке ее номером в таблице ASCII
	`cite$`	(*cite_string*)	поместить на стек аргумент команды `\cite`
$\mathcal{L}$	`duplicate$`	($\mathcal{L}$ $\mathcal{L}$)	создать копию
$\mathcal{L}$	`empty$`	($\mathcal{I}$)	1 (если $\mathcal{L}$ — отсутствующее поле или пустая цепочка) или 0 (в противном случае)
$\mathcal{S}_1$ $\mathcal{I}$ $\mathcal{S}_2$	`format.name$`	($\mathcal{S}$)	отформатировать $\mathcal{I}$ имен $\mathcal{S}_1$ в соответствии с принятым форматом $\mathcal{S}_2$
$\mathcal{I}$ $\mathcal{F}_1$ $\mathcal{F}_2$	`if$`		если $\mathcal{I}$>0, выполнить $\mathcal{F}_1$; в противном случае выполнить $\mathcal{F}_2$
$\mathcal{I}$	`int.to.chr$`	($\mathcal{S}$)	преобразовать целое число в соответствующий символ таблицы ASCII
$\mathcal{I}$	`int.to.str$`	($\mathcal{S}$)	поместить на стек цепочку литер — эквивалент целого числа
$\mathcal{L}$	`missing$`	($\mathcal{I}$)	1 (если $\mathcal{L}$ — отсутствующее поле) или 0 (в противном случае)
	`newline$`		перейти к новой строке в `.bbl`-файле
$\mathcal{S}$	`num.names$`	($\mathcal{I}$)	число имен в $\mathcal{S}$
$\mathcal{L}$	`pop$`		отбросить верхний элемент стека
	`preamble$`	($\mathcal{S}$)	поместить на стек конкатенацию всех цепочек `@PREAMBLE`, считанных в файлах баз данных
$\mathcal{S}$	`purify$`	($\mathcal{S}$)	удалить не алфавитно-цифровые символы
	`quote$`	($\mathcal{S}$)	поместить на стек цепочку — символ двойных кавычек
	`skip$`		не делать ничего
	`stack$`		снять со стека и напечатать все содержимое стека

окончание см. на след. стр.

Таблица 13.5. Список встроенных функций BibTeX'овского стилевого файла

В приведенной таблице встроенным функциям предшествуют переменные, используемые функциями на стеке. В круглых скобках указаны результаты, оставляемые функциями на стеке. «Литерал» $\mathcal{L}$ используется для обозначения элемента стека, который может быть целым числом $\mathcal{I}$, строкой $\mathcal{S}$, переменной $\mathcal{V}$, функцией $\mathcal{F}$ либо специальным значением, обозначающим отсутствующее поле. Если снимаемый со стека литерал имеет неверный тип, BibTeX выдает сообщение об ошибке и, в зависимости от типа результата, возвращаемого функцией, помещает на стек либо число 0, либо нулевую строку.

окончание (см. предыдущую страницу)		
$\mathcal{S}$ $\mathcal{I}_1$ $\mathcal{I}_2$ `substring$`	($\mathcal{S}$)	подстрока строки $\mathcal{S}$, начинающаяся с позиции $\mathcal{I}_1$ и имеющая длину $\mathcal{I}_2$
$\mathcal{L}_1$ $\mathcal{L}_2$ `swap$`	($\mathcal{L}_2$ $\mathcal{L}_1$)	поменять литералы местами
$\mathcal{S}$ `text.length$`	($\mathcal{I}$)	число «текстовых» символов
$\mathcal{S}$ $\mathcal{I}$ `text.prefix$`	($\mathcal{S}$)	*первые* $\mathcal{I}$ *литер цепочки* $\mathcal{S}$
$\mathcal{L}$ `top$`		снять со стека и напечатать верхний элемент стека
`type$`	($\mathcal{S}$)	поместить на стек тип публикации для текущей записи, например, `book` (или "", если тип неизвестен
$\mathcal{S}$ `warning$`		снять со стека и напечатать верхний литерал и предупреждающее сообщение
$\mathcal{F}_1$ $\mathcal{F}_2$ $\mathcal{I}$ `while$`		выполнять $\mathcal{F}_2$, пока для значения $\mathcal{I}$ функции $\mathcal{F}_1$ соблюдается условие $\mathcal{I}$>0
$\mathcal{S}$ `width$`	($\mathcal{I}$)	поместить на стек значение длины $\mathcal{S}$ (в TEX'овских единицах)
$\mathcal{S}$ `write$`		записать $\mathcal{S}$ в буфер вывода

Таблица 13.5.

После этого строятся метки для элементов списка литературы. Какие именно поля используются для построения первой части метки, зависит от типа публикации.

Затем производится подготовка меток к сортировке. Ключ сортировки строится путем применения к каждой публикации функции `presort`. В случае алфавитного формата меток для того, чтобы обеспечить их единственность и тем самым единственность упорядочения (отсортированного списка литературы), может потребоваться дописать к меткам дополнительные буквы a, b и т. д. При этом потребуются и два дополнительных запуска процесса сортировки. При числовом формате меток публикации могут входить в список литературы либо отсортированными (в алфавитном порядке), либо в порядке цитирования. В обоих случаях необходимо зафиксировать метку наибольшей длины — этого требует окружение `thebibliography`.

Наконец, в результате циклического перебора публикаций и выполнения для каждой из них процедуры `call.type$` генерируется `.bbl`-файл.

13.8 Модификация стилевых файлов

Требования к оформлению, предъявляемые тем или иным издательством, нередко вызывают потребность в небольшом изменении имеющегося стилевого файла. Тому, как это делается, и посвящен настоящий раздел.

В качестве первого примера покажем, как добиться, чтобы названия работ не переводились в нижний регистр. Этот перевод характерен для большинства BIBTEX'овских стилевых файлов, но во многих случаях является нежелательным. Более естественно, чтобы названия оставались в том же регистре, в котором они были записаны в базе данных. Для достижения поставленной цели можно создать

разновидность стиля unsrt. Поскольку это будет другой стилевой файл, отличный от исходного, ему следует дать самостоятельное имя, например, myunsrt. Аналогичным образом можно модифицировать и другие стили.

Как видно из табл. 13.5, за изменение регистра отвечает функция `change.case$`. Произведя поиск вызова этой функции в стилевом файле (для этого можно использовать текстовый редактор), обнаружим, что для решения поставленной задачи нужно модифицировать функцию `format.title`. Искомая модификация выглядит следующим образом:

```
FUNCTION {format.title}
{ title empty$
    { "" }
    { title "t" change.case$ }
  if$
}
```

Исходное определение

```
FUNCTION {format.title}
{ title empty$
    { "" }
    { title } % <== modified
  if$
}
```

Модифицированное определение

Суть этих определений вполне очевидна (см. табл. 13.5). А вот как выглядит аналогичная модификация функции `format.edition`:

```
FUNCTION {format.edition}
{ edition empty$
   { "" }
   { output.state mid.sentence =
      { edition "l" change.case$ " edition" * }
      { edition "t" change.case$ " edition" * }
     if$
   }
 if$
}
```

Исходное определение

```
FUNCTION {format.edition}
{ edition empty$
   { "" }
   { output.state mid.sentence =
      { edition  " edition" * }%<== changed
      { edition  " edition" * }%<== changed
     if$
   }
 if$
}
```

Модифицированное определение

Подобным же образом модифицируются определения функций `format.chapter.pages`, `format.thesis.type` и `format.tr.number`.

13.8.1 Добавление нового поля

Иногда бывает нужно добавить к имеющимся полям еще одно. Предположим, например, что требуется ввести дополнительное поле annotation. Сделать это можно двумя способами. Первый способ используется в стиле annotate, второй — в стиле annotation. Рассмотрим вначале второй способ как более простой. В стиле annotation (который основан на стиле plain) добавление поля осуществляется следующим образом. Прежде всего к списку определений ENTRY добавляется поле annote. Обработка нового поля поручается функции fin.entry (т. е. вводится в ее определение). Напомним, что функция fin.entry вызывается в конце выполнения каждой функции, определяющей тип публикации (см. выше).

```
FUNCTION {fin.entry}
{ add.period$
  write$
  newline$
}
```

Исходное определение

```
FUNCTION {fin.entry}
{ add.period$
  write$
  newline$
  "\begin{quotation}\noindent{\sc Key:\ }" cite$ * write$
    annote missing$
    'skip$
    { "\\{\sc Annotation:\ }" write$  annote  write$ }
  if$
  "\end{quotation}" write$
  newline$
}
```

Модифицированное определение

Очевидно, модификация состоит в том, что внутрь окружения quotation выводится ссылка, а затем, вслед за словом «Annotation», которое начинает новую строку, записывается текст аннотации. В отсутствие поля annote никаких дополнительных действий не производится: после проверки условия annote missing$ в команде if$ выбирается ветвь skip$.

Более сложный путь реализован в библиографическом стиле annotate, основанном на стиле alpha.

После добавления к списку полей команды ENTRY элемента annotate создается функция format.annotate, форматирующая добавленное поле. Логика этой функции примерно такая же, как у предыдущей:

```
FUNCTION {format.annotate}
{ annotate empty$
  { "" }
  { " \begin{quotation}\noindent "
    annotate
    * " \end{quotation} " *
  }
    if$
}
```

Далее, для любого из типов публикаций, перечисленных в табл. 13.2, форматирующая программа содержит вспомогательную строку `format.annotate write$`, которая следует непосредственно за обращением к `fin.entry`.

Есть еще несколько дополнительных полей, используемых в тех или иных стилевых файлах. Так, например, поля `isbn` и `issn`, содержащие номера данной публикации по международной классификации, вводятся в стилевых файлах is-abbrv, is-alpha, is-plain и is-unsrt. Стиль abstract имеет три дополнительных поля: `abstract`, `keyword` и `comment`. Эти поля вводятся по схеме, описанной выше для поля `annotate`.

Продолжая обсуждать возможности адаптации стилей, перейдем к вопросу об оформлении собственно элементов списка литературы. Здесь несомненный интерес представляет стиль chicago, уже упоминавшийся в разд. 13.1.1. В данном стиле предусмотрен ряд новых команд для библиографических ссылок, и при этом список литературы оформляется с учетом рекомендаций книги *The Chicago Manual of Style* [103]. Другой интересный пример — стили authordate1–authordate4, разработанные Дэвидом Ридом. В них использованы многие рекомендации стандартов BS 1629 и BS 5605, принятых British Standards, а также Оксфордского и Кембриджского университетов и *The Chicago Manual of Style*. В соответствующих файлах ясно видно, где именно были сделаны изменения, поэтому, изучив их, можно понять, как модифицируются библиографические стилевые файлы в общем случае.

13.8.2 Поддержка языков, отличных от английского

Чтобы адаптировать тот или иной BibTeX'овский стиль к языку, отличному от английского, необходимо, во всяком случае, перевести на этот язык английские слова, «зашитые» в стилевой файл, типа слова «edition» (см. пример, приведенный ниже в настоящем разделе).

Используя текстовый редактор, все подобные слова в стилевом файле нужно заменить их адекватным переводом. Если предполагается использовать лишь один язык, то при этом можно учесть и полиграфические нормы этого языка. Примером реализации такого подхода может служить стиль nederlands, который был разработан Веренфридом Шпитом и представляет собой адаптацию стиля harvard к голландскому языку с учетом рекомендаций Ван Дейла (Van Dale, 1982).

Приведем два примера функций, подвергшихся при этом модификации.

Как можно понять из рис. 13.16, при оформлении библиографии на голландском языке безразлично, имеет ли публикация одного редактора или редакторов несколько. Нидерландское слово `redactie` охватывает оба случая.

Пример, приведенный на рис. 13.17, показывает, что если предполагается работать с одним-единственным языком, то адаптация стиля может быть достаточно кардинальной, как в отношении оформления, так и в смысле перевода. В данном случае речь идет о поле «edition». В рассматриваемом примере голландские термины используются для указания порядкового номера издания вплоть до третьего. Для последующих изданий (с номером, большим или равным четырем)

```
FUNCTION {format.editors}
{ editor empty$
    { "" }
    { editor format.names
      editor num.names$ #1 >
        { " (eds)" * }
        { " (ed.)" * }
      if$
    }
  if$
}
```

Исходное определение

```
FUNCTION {format.editors}
{ editor empty$
    { "" }
    {  editor format.names
       ", redactie" *
    }
  if$
}
```

Модифицированное определение

Рис. 13.16. Адаптация BibTeX'овского стиля к голландскому языку

```
FUNCTION {format.edition}
{ edition empty$
  { "" }
  { output.state mid.sentence =
    { edition "l" change.case$ " edition" * }
    { edition "t" change.case$ " edition" * }
    if$
  }
  if$
}
```

Исходное определение

```
FUNCTION {format.edition}
{ edition empty$
  { "" }
  { edition "1" =
    { "Eerste" }
    { edition "2" =
      { "Tweede" }
      { edition "3" =
        { "Derde" }
        { edition "$^{\mathrm{e}}$ " * }
        if$
      }
      if$
    }
    if$
    output.state mid.sentence =
      { "l" change.case$ " druk" * }
      { "t" change.case$ " druk" * }
    if$
  }
  if$
}
```

Модифицированное определение

Рис. 13.17. Создание BibTeX'овского стиля для языка, отличного от английского

```
FUNCTION {sort.format.title}
{ 't :=
  "A " #2
    "An " #3
      "The " #4 t chop.word
    chop.word
  chop.word
  sortify
  #1 global.max$ substring$
}
```

Исходное определение

```
FUNCTION {sort.format.title}
{ 't :=
  "De " #3
    "Een " #4 t chop.word
  chop.word
  sortify
  #1 global.max$ substring$
}
```

Модифицированное определение

Рис. 13.18. Устранение влияния артиклей на результат BibTeX'овской сортировки

используется общее обозначение $\mathtt{i}^{\mathtt{e}}$, где `i` — порядковый номер издания. Другие поучительные моменты данного примера — вложенные операторы `if$`, а также применение команды `change.case$`, изменяющей регистр.

Разумеется, при адаптации должны быть заменены названия месяцев. Кроме того, могут быть введены новые аббревиатуры для некоторых названий или словосочетаний используемого языка.

```
MACRO {jan} {"januari"}
MACRO {feb} {"februari"}
MACRO {mar} {"maart"}
    ...
MACRO {UvA} {"Universiteit van Amsterdam"}
MACRO {RUG} {"Rijksuniversiteit te Groningen"}
MACRO {RUL} {"Rijksuniversiteit te Leiden"}
MACRO {TUD} {"Technische Universiteit Delft"}
    ...
MACRO {NTN} {"Nederlands tijdschrift voor natuurkunde"}
```

При этом для того чтобы обеспечить правильный порядок следования отсортированных имен и названий, необходимо дополнить программу сортировки `sort.format.names` правилами, учитывающими специфику данного языка.

В частности, в большинстве языков имеются артикли или другие служебные слова, которые полагается игнорировать, например, при сортировке заглавий.

Из рис. 13.18 ясно, что роль функции `chop.word` как раз и состоит в удалении заданных слов из строки, находящейся на стеке, в данном случае — определенных (`A`, `An`, `De`) и неопределенных (`The`, `Een`) артиклей.

Более общий подход был реализован группой Delphi Group (Йорг Хайткёттер с сотрудниками) из Дортмундского университета (Германия). Разработанный этой группой набор стилей Delphi BibStyles collection устроен таким образом, что в базовый файл `btxbst.doc` добавлен условный код, активизируемый вспомогательными препроцессорными переменными. Имеются версии для английского,

Переменные	ABBRV	ALPHA	PLAIN	UNSRT
ENGLISH	abbrv	alpha	plain	unsrt
FRENCH	fabbrv	falpha	fplain	funsrt
GERMAN	gabbrv	galpha	gplain	gunsrt

Таблица 13.6. BibTeX'овские стилевые файлы системы Delphi

французского и немецкого языков, из которых можно сгенерировать стилевые файлы, перечисленные в табл. 13.6.

Чтобы расширить эту систему вводом других языков, нужно пополнить файл `btxbst.doc` надлежащими управляющими переменными и необходимыми словосочетаниями на соответствующих языках.

Подход, предложенный Гансом-Германом Боде из университета г. Оснабрюк, основан на так называемом «адаптирующемся» BibTeX'овском семействе, которое отчасти напоминает систему Babel, описанную в разд. 9.2. Идея предложенного подхода состоит в том, что все элементы стилевого файла, связанные с выбором конкретного языка, рассматриваются как параметры, благодаря чему становится возможным отделить собственно BibTeX'овский стиль от языковой «начинки», которая теперь хранится в отдельном файле [9].

Файл `btxbst.doc` и здесь используется в качестве базового, но английские текстовые фрагменты заменяются макроопределениями, значения которых зависят от выбора языка. В настоящее время система поддерживает только английский и немецкий языки. Переключение с одного языка на другой происходит иначе, чем в системе Babel. Впрочем, оснастить рассматриваемую систему интерфейсом, аналогичным Babel'евскому, по-видимому, несложно.

Хотя обе эти системы вполне успешно справляются с задачей замещения текста при переключении на другой язык, они не решают проблему в целом так же хорошо, как это делает рассмотренный выше стиль nederlands. В частности, они совершенно не обращают внимания ни на пунктуацию, ни на особенности сортировки, а фактически лишь обеспечивают перевод терминов.

13.9 Пакет makebst для модификации библиографических стилей

TeX'овская программа makebst, написанная Пэтриком Дейли, дает возможность на основе некоторого «базового» библиографического стилевого файла достаточно легко создавать свои собственные BibTeX'овские `.bst`-файлы. При этом уточнение формата элементов библиографии производится в интерактивном режиме, а получаемый в результате файл преобразуется в BibTeX'овский стилевой файл при помощи программы DOCSTRIP, описанной в следующей главе (см. разд. 14.3). Технически это реализовано следующим образом. «Базовый» стилевой файл со-

держит многовариантный программный код, отвечающий всевозможным опциям программы DOCSTRIP. В результате выбора в интерактивном меню того или иного пункта (см. пример, рассмотренный ниже) соответствующая часть программного кода активизируется, и тем самым обеспечивается требуемая настройка стиля.

13.9.1 Работа с программой makebst

Следующий пример воспроизводит начальную и заключительную фазы прогона TEX'а в сочетании с системой makebst. Запускаемая в конце этого процесса программа DOCSTRIP генерирует искомый BIBTEX'овский стилевой файл (в данном случае — mytest.bst).

```
tex makebst
This is TeX, C Version 3.141
(makebst.tex
**********************************
* This is Make Bibliography Style *
**********************************
It makes up a docstrip batch job to produce
a customized .bst file for running with BibTeX.
Do you want a description of the usage? (NO)

\yn=y

Enter the FULL name of the MASTER file (def=genbst.mbs)

\mfile=genbst.mbs

Name of the final OUTPUT .bst file?

\ofile=mytest.bst

Give a comment line to include in the style file.
Something like for which journals it is applicable.

\ans=Test of the generic bst making program makebst
(genbst.mbs

STYLE OF CITATIONS:
(*) Numerical as in standard LaTeX
(a) Author-year with some non-standard interface
(c) Cite key (special for listing contents of bib file)
  Select:

\ans=a
  You have selected: Author-year
```

```
AUTHOR-YEAR SUPPORT SYSTEM:
(*) Natbib for use with natbib.sty
(l) Apalike for use with apalike.sty
(h) Harvard system with harvard.sty
(a) Astronomy system with astron.sty
(c) Chicago system with chicago.sty
(d) Author-date system with authordate1-4.sty
  Select:

\ans=h
  You have selected: Harvard

ORDERING OF REFERENCES:
(*) Alphabetical by all authors
(l) By label (Jones before Jones and James before Jones et al)
  Select:
                        ..... И еще куча разных вопросов и ответов ....
Finished!!
Batch job written to file 'mytest.drv'
Shall I now run this batch job? (NO)

\yn=y
(a.drv (/usr/local/lib/tex/macros/docstrip.tex
Utility: 'docstrip' 2.0r <92/08/17>
English documentation <92/08/17>

************************************************************
* This program converts documented macro-files into fast *
* loadable files by stripping off (nearly) all comments! *
************************************************************

Generating file ./mytest.bst

Processing File genbst.mbs (ay,har,seq-lab,nm-rev,nmlm,x3,m3,nmft-sc,
dt-beg,yr-par,tit-it,atit-u,volp-com,edby,blk-com,pp) -> mytest.bst
```

ГЛАВА 14

Средства документирования макропакетов

Пакет doc, описанный в этой главе, предлагает удобный путь решения задачи сопровождения LaTeX'овских макропакетов. Центральная идея используемого подхода состоит в том, что в один и тот же файл записываются и сами LaTeX'овские макроопределения, и комментарии к ним, причем так, что при помощи некоторой стандартной процедуры из этого файла можно получить как документацию, так и «очищенный» командный файл (один или несколько). В главе описана структура, которую должны иметь подобные файлы, и показано, как при помощи программы DOCSTRIP можно создавать процедуры для автоматического получения собственно LaTeX'овских макропакетов и сопутствующей документации.

14.1 Макропакеты с документацией

Идея интегрированной документации использовалась Дональдом Кнутом при создании TeX'а. Программа TeX была написана им при помощи системы WEB, представляющей собой симбиоз программного метакода на языке, напоминающем Паскаль, и документации. Именно этот подход позволил довольно легко перенести TeX вместе с сопутствующими программами практически на все существующие компьютерные платформы.

В последнее время авторы LaTeX'овских макропакетов также начали осознавать, насколько важно документировать создаваемые ими программные коды. Многие из распространяемых ныне пакетов LaTeX'овских макрокоманд оформлены с помощью пакета doc (автор — Франк Миттельбах) и сопутствующей ему утилиты DOCSTRIP (авторы — Йоханнес Брамс, Дени Дюшье и Франк Миттельбах).

При таком подходе как сами LaTeX'овские макрокоманды, так и комментарии к ним содержатся в одном и том же TeX'овском файле. С помощью комментариев пользователю удается быстрее разобраться в тех или иных последовательностях сложных команд LaTeX'а, и это является очевидным преимуществом данной схе-

мы. Кроме того, становится проще выполнять обновление текущей версии пакета, поскольку все изменения вносятся в один-единственный файл.

Все, что требуется сделать, — это откомпилировать в LaTeX'е специальный «инсталляционный файл»; при этом будут автоматически сгенерированы все необходимые файлы, включая документацию.

Кроме того, знание основ системы doc позволяет без особого труда объединять различные файлы, относящиеся к одному и тому же пакету, в единый файл, содержащий к тому же и документацию. Получаемый «упакованный» файл вместе с соответствующим инсталляционным файлом — наиболее удобная форма распространения пакета.

14.2 Пользовательский интерфейс пакета doc

14.2.1 Основные понятия

Всякий LaTeX'овский файл, предназначенный для обработки пакетом doc, состоит из *документационных фрагментов* и *командных фрагментов*.

Каждая строка документационного фрагмента начинается с символа процента (%) в крайней левой позиции. В строке могут содержаться любые команды TeX'а или LaTeX'а, однако символ % не может быть использован в качестве символа комментария. Пользовательские комментарии могут вноситься в строку при помощи символа `^^A`. Более длинные куски текста можно превращать в комментарии при помощи конструкции `%␣\iffalse ... % \fi`, где вместо многоточия должен стоять требуемый текст. Все прочие фрагменты файла относятся к числу командных. В них содержатся определения макрокоманд (или отдельные фрагменты этих определений), комментарии к которым приводятся в документационных фрагментах.

При чтении файла-макропакета LaTeX игнорирует все документационные фрагменты и получает файл, состоящий из одних лишь командных фрагментов, причем макроопределения, которые в исходном файле были разбиты на отдельные куски, оказываются склеенными.

С другой стороны, если требуется сгенерировать документацию к пакету, то нужно, чтобы командные фрагменты файла-макропакета воспроизводились буквально (в стиле verbatim). Это достигается погружением командных фрагментов в окружение `macrocode`, для чего каждый командный фрагмент должен начинаться строкой

```
%␣␣␣␣\begin{macrocode}
```

и заканчиваться строкой

```
%␣␣␣␣\end{macrocode}
```

При этом необходимо, чтобы число пробелов между % и `\end{macrocode}` *в точности* равнялось четырем.

Внутри командного фрагмента допустимы любые TEX'овские команды, в частности, символ процента в конце строки можно использовать для подавления пробела, связанного с переходом на новую строку.

Вместо окружения `macrocode` можно использовать и окружение `macrocode*`. Отличие последнего лишь в том, что при распечатке документации пробелы изображаются символом ␣.

14.2.2 Описание новых макрокоманд и окружений

Описание новой макрокоманды всегда должно начинаться с команды `\DescribeMacro`, которая помечает начало описания в теле документа.

`\DescribeMacro{`*macro name*`}`

Аргумент этой команды будет напечатан на полях в соответствующем месте документа. Кроме того, имя описываемой макрокоманды может быть включено в предметный указатель. Начало описания может выглядеть, например, вот так:

```
% \DescribeMacro{\DocInput} \DescribeMacro{\IndexInput}
% И, наконец, фрагмент \meta{input commands} ...
```

Аналогичная команда `\DescribeEnv` указывает, что начинаются пояснения к LATEX'овскому окружению.

`\DescribeEnv{`*environment name*`}`

Определение новой макрокоманды дается при помощи окружения `macro`.

`\begin{macro}{`*macroname*`}`

Это окружение имеет один аргумент — имя новой макрокоманды, которое также печатается на полях и вносится в предметный указатель. В итоге в предметном указателе создаются самостоятельные ссылки на определение макрокоманды и на пояснения к ее использованию, что дает дополнительные удобства.

```
% \begin{macro}{\MacroTopsep}
%    Here is the default value for the \verb+\MacroTopsep+
%    parameter used above.
%    \begin{macrocode}
\newlength{\MacroTopsep}
\setlength{\MacroTopsep}{7pt plus 2pt minus 2pt}
%    \end{macrocode}
% \end{macro}
```

Документирование окружений, вводимых при помощи команды `\newenvironment`, может осуществляться при помощи окружения `environment`. Оно работает так же, как окружение `macro`, и требует сообщить ему имя окружения в качестве аргумента.

14.2.3 Перекрестные ссылки между используемыми макрокомандами

Имя каждой команды, встречающееся внутри окружений `macrocode` или `macrocode*`, порождает ссылку в предметном указателе. Благодаря этому можно узнать, где именно используется та или иная макрокоманда. Однако в ситуации, когда TEX'у приходится обрабатывать большой массив ссылок, его работа существенно замедляется. В связи с этим предусмотрена возможность отключения режима обработки ссылок. Такое отключение производит команда `\DisableCrossrefs`, внесенная в управляющий файл. Для того чтобы вновь включить режим создания ссылок, достаточно дать команду `\EnableCrossrefs`.

Программа *MakeIndex* позволяет создавать предметный указатель в двух различных модификациях. Выбор типа указателя производится путем добавления в преамбулу управляющего файла одной из двух приведенных ниже деклараций (если не добавлена ни одна декларация, указатель не генерируется). При наличии декларации `\PageIndex` все элементы указателя снабжаются ссылками на номера страниц; при добавлении вместо нее декларации `\CodelineIndex` элементы указателя, порожденные командами `\DescribeMacro` и `\DescribeEnv`, по-прежнему ссылаются на номера страниц, а элементы, порожденные окружениями `macro` и `macrocode`, — на номера строк в листинге соответствующего программного кода.

14.2.4 Заключительные процедуры создания указателя

Как следует из сказанного выше, некоторые макрокоманды пополняют предметный указатель новыми элементами. Необходимая сортировка этих элементов должна производиться внешней программой, например, программой *MakeIndex* (см. гл. 12). Следует запустить *MakeIndex* с ключом `-s` (см. разд. 12.4.2) и надлежащим стилевым файлом, например, с входящим в пакет `doc` файлом `gind.ist`.

Для того чтобы считать и распечатать готовый указатель, нужно в конце имеющегося документированного файла-макропакета, например, вслед за командами обработки библиографических ссылок написать команду `\PrintIndex`.

14.2.5 Дополнительные возможности

Документированный файл-макропакет может быть разделен на две части: одну, содержащую общее описание макрокоманд, и другую, в которой подробно разъясняется, как их следует использовать. Вообще говоря, пользователь должен иметь возможность отказаться от второй части.

```
\StopEventually{final text}
\Finale
```

Такую возможность предоставляет команда `\StopEventually`, помещаемая между первой и второй частями. Аргументом этой команды является текст, который необходимо выдать на печать в том случае, если документацию решено ограничить только первой частью (например, список литературы, кото-

рый обычно находится в самом конце документа). Если в управляющем файле имеется декларация `\OnlyDescription`, LaTeX обработает аргумент команды `\StopEventually`, после чего остановится. В противном случае (в отсутствие декларации `\OnlyDescription`) аргумент будет передан макрокоманде `\Finale`, которая поместит его в надлежащий раздел документа (как правило, в самый конец) в случае, когда документ генерируется полностью. Благодаря этому приему отпадает необходимость хранить две копии одного и того же заключительного текста.

Для документирования модификаций файла используется команда `\changes`, помещаемая во фрагмент, содержащий описание модифицированных макрокоманд.

`\changes{`*version*`}{`*date*`}{`*text*`}`

Соответствующий вспомогательный файл можно создать на основе информации, сообщаемой команде `\changes`, используя LaTeX'овский механизм `\glossary`. После надлежащего форматирования этот файл может быть распечатан. Запись информации инициируется включением в управляющий файл команды `\RecordChanges`. Чтобы считать и распечатать историю модификаций файла, достаточно в каком-либо подходящем месте файла-макропакета поместить команду `\PrintChanges`. Упорядочение записей о модификациях производится программой *MakeIndex*, которая применяется к неупорядоченному файлу-глоссарию с использованием подходящего стиля, например, стиля `gglo.ist`, входящего в дистрибутив системы doc (более подробно о том, как *MakeIndex* работает с глоссариями, написано в разд. 12.4.6).

`\MakeShortVerb{\`*character*`}`

В тех случаях, когда требуется многократно использовать стиль verbatim, например, при буквальном воспроизведении на печати имен TeX'овских команд, нерационально набирать каждый раз `\verb+ ... +`. Вместо этого можно воспользоваться следующим механизмом введения аббревиатуры, который предусмотрен в макропакете doc. Выбирается какой-либо символ *c*, который в набираемом тексте не встречается или встречается крайне редко. Этот символ и используется в качестве ограничителя буквально воспроизводимого текста. Часто выбор падает на символ ", однако если он оказывается занят (например, если в тексте часто встречаются немецкие умляуты), можно отдать предпочтение символу «|». Действие команды `\MakeShortVerb{\`*c*`}` состоит в том, что у записи `\verb`*c*`text`*c* появляется более короткий эквивалент *c*`text`*c*. Если в дальнейшем потребуется использовать символ *c* в его обычном значении, достаточно будет написать `\DeleteShortVerb{\`*c*`}`. Переход от употребления *c* в качестве аббревиатуры для `\verb` к его обычному использованию, а также обратный переход можно произвести сколько угодно раз. Следует помнить, что символ-аббревиатура *c*, равно как и сама команда `\verb`, не должен входить в аргументы других команд, однако его вполне можно использовать внутри окружений `verbatim` и `macrocode`.

Перечень всех пользовательских команд пакета doc приведен в табл. 14.1.

Преамбула и команды ввода
`\CharacterTable{`*character table*`}`
Пользовательский интерфейс проверки таблицы символов. См. пример на рис. 14.6.
`\CheckSum{`*checksum*`}`
Пользовательский интерфейс проверки контрольной суммы (числа символов \ в коде).
`\CheckModules`
Специальным образом форматировать модульные директивы DOCSTRIP (принимается по умолчанию).
`\CodelineIndex`
В указателе использовать ссылки на номера строк.
`\CodelineNumbered`
Нумеровать строки кода, но не нумеровать команды создания указателя.
`\DisableCrossrefs`
Не создавать элементов указателя для команд, находящихся внутри кода.
`\DocInput{`*file*`}`
Считать в *file* в соответствии с требованиями системы doc.
`\DontCheckModules`
Не форматировать специальным образом модульные директивы DOCSTRIP.
`\EnableCrossrefs`
Разрешить создание элементов указателя для команд, находящихся внутри кода.
`\IndexInput{`*file*`}`
Считать в *file*, напечатать его в стиле *verbatim* и создать указатель для содержащихся в нем команд.
`\OnlyDescription`
Не форматировать код; остановиться по команде `\StopEventually`.
`\PageIndex`
В указателе использовать ссылки на номера страниц.
`\RecordChanges`
Сгенерировать историю модификаций файла.
Команды, относящиеся к структуре документа
`\bslash`
Команда, создающая на печати символ \.
`\DeleteShortVerb`
Отмена предшествующего определения `\MakeshortVerb`.
`\DescribeEnv{`*env*`}`
Флаг, устанавливаемый в том месте текста, где описано окружение *env*.
`\DescribeMacro{`*cmd*`}`
Флаг, устанавливаемый в том месте текста, где описана макрокоманда *cmd*.
продолжение на следующей странице

Таблица 14.1. Перечень команд пакета doc

продолжение (см. предыдущую страницу)

`\begin{environment}{`*env*`}` (окружение)

Окружение, внутри которого находится описание окружения *env*.

`\Finale`

Команда, выполняемая в самом конце документа (см. `\StopEventually`).

`\begin{macro}{`*cmd*`}` (окружение)

Окружение, внутри которого находится описание макрокоманды *cmd*.

`\begin{macrocode}` (окружение)

Окружение, внутри которого находится TEX'овский командный код.

`\begin{macrocode*}` (окружение)

Аналогично окружению `macrocode` с той разницей, что для изображения пробелов используется символ ␣.

`\MakeShortVerb{`*char*`}`

Определяется символ-аббревиатура *char* для команды `\verb`.

`\meta{`*arg*`}`

Аргумент этой команды записывается как мета-предложение (по умолчанию как ⟨*arg*⟩).

`\PrintChanges`

В этом месте печатается листинг истории модификаций.

`\PrintIndex`

В этом месте печатается листинг указателя.

`\SpecialEscapechar{`*char*`}`

Определяется новый командный префикс, который используется вместо \.

`\StopEventually{`*cmds*`}`

В аргументе этой команды *cmds* содержатся те команды, которые должны быть выполнены в конце документа (они передаются команде `\Finale`).

`\begin{verbatim}` (окружение)

Слегка измененная версия стандартного LATEX'овского окружения `verbatim`: в тексте, охватываемом данным окружением, символ процента в первом столбце игнорируется.

`\begin{verbatim*}` (окружение)

Аналогично окружению `verbatim` с той разницей, что для изображения пробелов используется символ ␣.

Команды, относящиеся к указателю

`\actualchar`

Символ, используемый как разделитель между ключевым словом и собственно текстом элемента указателя (по умолчанию используется =).

`\DoNotIndex{`$cmd_1, \ldots, cmd_n$`}`

Перечисляются команды, которые не должны включаться в указатель.

продолжение на следующей странице

Таблица 14.1.

продолжение (см. предыдущую страницу)

`\encapchar`

Символ, используемый как разделитель между текстом элемента указателя и командой форматирования номера страницы, на который производится ссылка (по умолчанию используется |).

`\IndexMin`

Минимальная высота незанятой части страницы, при которой на этой странице разрешается начать указатель (по умолчанию 80pt).

`\IndexParms`

Макрокоманда, задающая формат столбцов указателя.

`\IndexPrologue{`*text*`}`

Текст, помещаемый в начало указателя взамен принятого по умолчанию.

`\levelchar`

Символ, используемый как разделитель между различными уровнями вложенности для элементов указателя (по умолчанию используется >).

`\main{`*number*`}`

Задается стиль для номеров страниц или строк кода в ссылках на основную информацию (по умолчанию цифры подчеркиваются).

`\quotechar`

Символ, используемый в указателе в тех случаях, когда следующий за ним символ был ранее определен как специальный; подавляет это специальное значение (по умолчанию используется *).

`\SortIndex{`*key*`}{`*entry*`}`

Создать элемент указателя для *entry* и отсортировать его по ключевому слову *key*.

`\SpecialEnvIndex{`*entry*`}`

Создать элемент указателя для окружения *entry*.

`\SpecialIndex{`*cmd*`}`

Создать элемент указателя для команды (используя стиль *verbatim* при внесении имени команды в указатель).

`\SpecialMainIndex{`*cmd*`}`

Создать элемент указателя для команды с добавлением ссылки на страницу, содержащую основную информацию (`\main`).

`\SpecialUsageIndex{`*cmd*`}`

Создать элемент указателя для команды с добавлением ссылки на описание использования команды (`\usage`).

`\usage{`*number*`}`

Задается стиль для номеров страниц в ссылках указателя на описание использования команд (по умолчанию цифры пишутся курсивом).

`\verbatimchar`

Символ, используемый как ограничитель текста, являющегося аргументом команды `\verb` (по умолчанию используется +).

продолжение на следующей странице

Таблица 14.1.

продолжение (см. предыдущую страницу)

История модификаций
`\changes{`*version*`}{`*date*`}{`*reason*`}`
Сведения о произведенной модификации, которые будут использованы при создании листинга истории модификаций.
`\docdate`
Сообщает дату последнего обновления документации.
`\filedate`
Сообщает дату последнего обновления макрокода.
`\filename`
Сообщает имя исходного файла.
`\fileversion`
Сообщает номер версии исходного файла.
`\GlossaryMin`
Минимальная высота незанятой части страницы, при которой на этой странице разрешается начать запись истории модификаций (по умолчанию 80pt).
`\GlossaryParms`
Макрокоманда, задающая формат столбцов при печати истории модификаций.
`\GlossaryPrologue{`*text*`}`
Текст, помещаемый в начало истории модификаций взамен принятого по умолчанию.
Параметры, относящиеся к набору и верстке
`\*`
Символ, используемый в элементах указателя для ссылок на более высокий уровень (по умолчанию используется '~').
`\@idxitem`
Макрокоманда, определяющая формат элементов указателя на печати (по умолчанию это абзацы с «отрицательным» абзацным отступом, равным 30 пунктам).
`\AltMacroFont`
Шрифт, используемый для набора программного кода модуля DOCSTRIP (по умолчанию при работе с NFSS используется `\small\sl\tt`).
`\DocstyleParms`
Макрокоманда, определяющая формат TEX'овского кода.
`\MacroFont`
Шрифт, которым будет напечатана основная часть программного кода (по умолчанию `\small\tt`).
`\MacroIndent`
Величина отступа для любой строки кода.
`\MacroTopsep`
Вертикальный отступ перед окружением `macro` и после него.

окончание на следующей странице

Таблица 14.1.

окончание (см. предыдущую страницу)
`\MacrocodeTopsep`
Вертикальный отступ перед окружением `macrocode` и после него.
`\MakePrivateLetters`
Макрокоманда, указывающая символы, которые будут считаться буквами (по умолчанию — только @).
`\Module`
Макрокоманда с одним аргументом, определяющая формат модульных директив DOCSTRIP.
`\PrintDescribeEnv`
Макрокоманда с одним аргументом, определяющая формат `\DescribeEnv`.
`\PrintDescribeMacro`
Макрокоманда с одним аргументом, определяющая формат `\DescribeMacro`.
`\PrintMacroName`
Макрокоманда, аналогичная `\PrintDescribeMacro`, но предназначенная для аргумента окружений `macro`.
`\ps@titlepage`
Макрокоманда, задающая стиль для титульных страниц статей, объединенных в журнал (по умолчанию `\ps@plain`).
`StandardModuleDepth`
Счетчик, в котором хранится номер последнего уровня вложенности директив DOCSTRIP, для которого используется шрифт `\MacroFont`. Директивы более глубоких уровней вложенности печатаются шрифтом `\AltMacroFont`.
`\theCodelineNo`
Стиль для номеров строк (по умолчанию арабские цифры размера script).

Таблица 14.1.

14.2.6 Управляющий файл

Для того чтобы при помощи пакета doc сгенерировать документацию к некоторому набору макрокоманд, необходимо создать управляющий файл следующего содержания:

```
\documentclass[options]{document-class}
\usepackage{doc}

    preamble
\begin{document}
    input commands
\end{document}
```

В качестве класса (*document-class*) можно использовать любой из LaTeX'овских классов, например, article. В преамбулу (*preamble*) должны быть

включены декларации, определяющие режим использования системы doc, такие, скажем, как `\DisableCrossrefs`, `\OnlyDescription`, `\CodelineIndex` и т. д.

Наконец, *input commands* означает одну или несколько команд вида `\DocInput{`*имя файла*`}` и/или `\IndexInput{`*имя файла*`}`. Команда `\DocInput` применяется к файлам, специально созданным для работы с системой doc, а `\IndexInput` — к любым файлам макрокоманд. Действие команды `\IndexInput` состоит в том, что она создает листинг файла, и по ходу дела составляет полный указатель всех команд. Такой указатель бывает полезен, когда приходится разбираться в работе макрокоманд в отсутствие достаточно полной документации. Если применить указанную команду к файлу `latex.tex`, содержащему базовые определения команд LaTeX'а, получится листинг из примерно 8800 строк кода с комментариями и указатель на 15 страницах.

Для распечатки результатов можно воспользоваться командами `\PrintIndex` и `\PrintChanges` (последняя работает при условии, что история модификаций была записана при помощи команды `\RecordChanges`). Впрочем, эти команды могут также быть заранее включены в обрабатываемый файл.

14.2.7 Простой пример файла, документированного при помощи пакета doc

Проиллюстрируем использование команд, рассмотренных в предыдущем разделе, на примере управляющего файла `docexam.drv` (рис. 14.1), при помощи которого генерируется документация к файлу `docexam.doc` (рис. 14.2). Для того чтобы получить документацию, показанную на рис. 14.3, нужно один раз обработать управляющий файл LaTeX'ом, затем, используя *MakeIndex*, сгенерировать указатель и глоссарий и, наконец, снова обработать LaTeX'ом управляющий файл.

```
latex docexam.drv
makeindex -s gind.ist docexam
makeindex -s gglo.ist -o docexam.gls docexam.glo
latex docexam.drv
```

```
\documentclass{article}
\usepackage{doc}                    % Подключить пакет doc
\EnableCrossrefs                    % Создать полный указатель
\CodelineIndex                      % по номерам строк
\RecordChanges                      % Записать историю модификаций
\setlength{\parindent}{0pt}         % Установить нулевой абзацный отступ
\begin{document}
   \DocInput{docexam.doc}   \PrintIndex   \PrintChanges
\end{document}
```

Рис. 14.1. Пример управляющего файла пакета doc

```
\def\fileversion{2.1}
\def\filedate{1 Apr 93}
\def\docdate {15 Aug 93}
%
% \changes{v1.0}{1 Apr 93}{First release}
% \changes{v2.0}{3 Apr 93}{Documentation added}
% \changes{v2.1}{15 Aug 93}{Minor corrections and additions}
%
% \MakeShortVerb{\|}
%
% \title{Illustrating \texttt{doc} with \texttt{docexam}}
% \author{All of us}
% \maketitle
% \begin{abstract}
% This is an example of a package documented
% using \LaTeX's \texttt{doc} system~\cite{Companion}.
% \end{abstract}
%
% \section{Introduction}
% This package does nothing useful, but serves as a simple
% example of how to document your packages using \LaTeX's
% \texttt{doc} system.
%
% \section{The user interface}
%
% This section defines everything an average user should
% know. Note the use of the
% \DeleteShortVerb{\|}
% ``\texttt{|}'' character as shorthand for
% \MakeShortVerb{\|}
% |\verb|, and how we switch back and forth between its
% two possible meanings.
%
% \DescribeMacro{\docsamplecmd}
% The command |\docsamplecmd| will print the text specified
% as argument in a small caps typeface, prepended by the
% string ``\texttt{doc:}''.
%
% \DescribeEnv{docsampleenv}
% The environment |docsampleenv| will print the text bracketed
% inside the environment as quoted and italicized text,
% preceded by a line containing \textbf{Nice, you are using doc!}.
%
%\StopEventually{
% \begin{thebibliography}{9}
%   \bibitem{Companion} M.~Goossens, F.~Mittelbach, and A.~Samarin
%   \emph{The \LaTeX{} Companion}, 1994, Addison-Wesley.
% \end{thebibliography}
% }
%
% \section{The code}
%
% We begin by identifying the version of this file on the
% terminal and in the transcript file.
%    \begin{macrocode}
\typeout{Package `docexam' \fileversion\space<\filedate>}
\typeout{English documentation\space\space\space<\docdate>}
%    \end{macrocode}
%
% \begin{macro}{\docsamplecmd}
% We prepend the text \texttt{doc} and switch to a small caps font.
%    \begin{macrocode}
\newcommand{\docsamplecmd}[1]{\texttt{doc:}\textsc{#1}}
%    \end{macrocode}
% \end{macro}
%
% \begin{environment}{docsampleenv}
% We print a line in bold, then specify a \texttt{quote}
% environment.
%    \begin{macrocode}
\newenvironment{docsampleenv}{\par\textbf{Nice, you are using doc!}
                              \begin{quote}\itshape}{\end{quote}}
%    \end{macrocode}
% \end{environment}
%
% \Finale
\endinput
```

Рис. 14.2. Пример документирования файла при помощи системы doc

Illustrating doc with docexam

All of us

December 10, 1994

Abstract

This is an example of a package documented using LaTeX's doc system [1].

1 Introduction

This package does nothing useful, but serves as a simple example of how to document your packages using LaTeX's doc system.

2 The user interface

This section defines everything an average user should know. Note the use of the "|" character as shorthand for \verb, and how we switch back and forth between its two possible meanings.

\docsamplecmd The command \docsamplecmd will print the text specified as argument in a small caps typeface, prepended by the string "doc:".

docsampleenv The environment docsampleenv will print the text bracketed inside the environment as quoted and italicized text, preceded by a line containing **Nice, you are using doc.**

3 The code

We begin by identifying the version of this file on the terminal and in the transcript file.

```
1 \typeout{Package 'docexam' \fileversion\space<\filedate>}
2 \typeout{English documentation\space\space\space<\docdate>}
```

\docsamplecmd We prepend the text doc and switch to a small caps font.

```
3 \newcommand{\docsamplecmd}[1]{\texttt{doc}\textsc{#1}}
```

docsampleenv We print a line in bold, then specify a quote environment.

```
4 \newenvironment{docsampleenv}{\par\textbf{Nice, you are using doc!}
5                               \begin{quote}\itshape}{\end{quote}}
```


References

[1] M. Goossens, F. Mittelbach, and A. Samarin *The LaTeX Companion*, 1994, Addison-Wesley.

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.

B
\begin 5

D
\docdate 2
\docsamplecmd . *1*, 3, 3
docsampleenv (environment) *1*

E
\end 5
environments:
docsampleenv .. *1*

F
\filedate 1
\fileversion 1

I
\itshape 5

N
\newcommand 3
\newenvironment .. 4

O
docsampleenv 4

P
\par 4

S
\space 1, 2

T
\textbf 4
\textsc 3
\texttt 3
\typeout 1, 2

Change History

v1.0
"General": First release 1
v2.0
"General": Documentation added . 1
v2.1
"General": Minor corrections and additions 1

Рис. 14.3. Документация, сгенерированная системой doc для примера на рис. 14.2

14.3 Утилита docstrip

Основная функция утилиты DOCSTRIP заключается в том, что она удаляет почти все строки, начинающиеся со знака процента, и поддерживает «условную» обработку кода, когда некоторые части файла обрабатываются только при соблюдении определенного условия. Кроме того, она позволяет создавать из нескольких файлов один LaTeX'овский макрокомандный файл либо, наоборот, расщеплять обрабатываемый файл на несколько файлов меньшего размера. Утилиту DOCSTRIP можно запускать в интерактивном режиме, подавая на вход LaTeX'а файл `docstrip.tex`:

```
latex docstrip.tex
```

При этом LaTeX задаст несколько вопросов о том, как следует обрабатывать данный файл. После того как пользователь ответит на эти вопросы, DOCSTRIP выполнит свою работу — «очистит» исходный файл от комментариев.

Удобнее, однако, заготовить пакетный файл (batch file), содержащий все необходимые инструкции для прогона DOCSTRIP. По умолчанию этому файлу присваивается имя `docstrip.cmd`. Если в текущем каталоге есть файл с таким именем, DOCSTRIP приступит к его обработке; в противном случае DOCSTRIP стартует в интерактивном режиме.

Путь, обеспечивающий еще более «дружественный» интерфейс, состоит в том, чтобы подготовить управляющий файл, который будет обрабатываться непосредственно LaTeX'ом. В этом случае пользователю достаточно написать:

```
latex batch-file
```

в результате чего из исходных текстов будут автоматически сгенерированы все «исполняемые» файлы.

Именно в такой форме распространяются так называемые Mainz distributions — дистрибутивы из немецкого города Майнца, или, другими словами, макропакеты и файлы, поддерживаемые Франком Миттельбахом и Райнером Шопфом, в числе которых сама система doc, макрорасширения array, ftnright, multicol, theorem, а также система LaTeX2_ε. Пример инсталляционной процедуры, использующей простой управляющий файл, приведен в разд. 14.4.

В следующем подразделе рассматриваются команды, которые можно использовать в вышеупомянутом пакетном файле для DOCSTRIP. Там же объясняется, как можно в исходных файлах системы doc создать фрагменты кода, допускающие «условную» обработку.

14.3.1 Команды, используемые в пакетных файлах

Макрокоду, получаемому на выходе программы DOCSTRIP, можно предпослать некоторый фиксированный вводный текст, например, упоминание об авторских правах или стандартное заявление об отказе от них. Этот текст должен быть по-

мещен между командами \preamble и \endpreamble. Аналогично, заключительный текст, добавляемый в самом конце, должен быть помещен между командами \postamble и \endpostamble. Весь материал, который DOCSTRIP обнаруживает в вводной и заключительной частях, записывается в результирующий файл с двумя символами процента в начале каждой строки. Вообще говоря, это должен быть обычный текст, а воспроизводимые в нем имена команд должны иметь вид \string\foo. Если в одной из строк, содержащих этот текст, встретится символ ^^J, вся идущая за ним часть строки будет в результирующем файле напечатана с новой строки, причем без символов процента в начале. Это обстоятельство можно использовать для того, чтобы добавлять в файл, «очищаемый» при помощи DOCSTRIP, команды \typeout или \message.

\generateFile{*output*}{*ask*}{\from{*input*}{*optionlist*}...}

Макрокоманда \generateFile сообщает утилите DOCSTRIP, что та должна делать. Аргументы *output* и *input* представляют собой имена файлов в том формате, какой принят на используемой компьютерной платформе. Параметр *optionlist* — это список опций (т. е. набор параметров, отделенных друг от друга запятой), определяющий, какие из необязательных фрагментов кода, содержащихся в файле *input*, должны войти в файл *output*. От аргумента *ask* TEX получает одно из двух распоряжений: либо молча записать выходной файл поверх уже существующего (значение f), либо выдать предупреждение о существовании файла с тем же именем и спросить пользователя, надо ли заменять существующий файл (значение t). Команд \from (а значит, входных файлов) может быть несколько, каждая — со своим собственным параметром *optionlist*.

Чтобы создать для DOCSTRIP управляющий файл, который будет обрабатываться непосредственно LATEX'ом, нужно поместить в начало файла следующие две строчки:

```
\def\batchfile{<batch-file-name>}
\input docstrip.tex
```

Команды пакетных файлов (batch file commands), к коим относится команда \generateFile, можно распределить по нескольким пакетным файлам, которые будут вызываться из главного пакетного файла. Это полезно, например, в том случае, когда дистрибутив состоит из нескольких различных частей, записанных в отдельные файлы. В таком случае пакетные файлы, относящиеся к отдельным частям, вызываются из главного файла при помощи команды \batchinput. Команда \input здесь неприменима: в управляющем файле DOCSTRIP данную команду можно использовать только для ввода самой утилиты DOCSTRIP.

Диалог с пользователем пакета поддерживается командами \Msg и \Ask. При этом команда \Msg выводит свой аргумент на терминал. Если в инсталляционный файл вписана команда \keepsilent, то бóльшая часть сообщений, генерируемых DOCSTRIP в процессе работы, будет подавляться; вновь включить режим

их показа можно командой \showprogress. О других командах, используемых в пакетных файлах, можно узнать из документации к DOCSTRIP.

В принципе создание инсталляционных файлов, в которых поддерживался бы диалог с пользователем и решались бы задачи любой степени сложности, всегда возможно. Однако из-за способа, которым DOCSTRIP читает пакетные файлы, сложность самих файлов при этом может оказаться совершенно невообразимой: если вы любопытны, взгляните на инсталляционные файлы больших пакетов.

14.3.2 Условное включение кода

Фрагменты кода, предназначенного для условного включения, помечаются в исходном файле специальными «тегами». В простейшем варианте используется парные теги <*tag> и </tag>. Между ними должен быть заключен фрагмент, который подлежит включению в *output* при условии, что в списке опций *optionlist* команды \generateFile присутствует опция tag. Теги должны располагаться в начале строки сразу вслед за открывающим ее символом процента. Вот как это выглядит:

```
%<*style>
     text or code
%</style>
```

Когда фрагмент кода не включается, теги, встречающиеся внутри этого фрагмента, не считываются. В общем случае тег может представлять собой произвольную комбинацию опций либо булеву функцию от них. Символ | используется для обозначения логического ИЛИ, & — логического И, а ! — для обозначения отрицания. Опция считается принимающей значение ИСТИНА тогда и только тогда, когда она есть в списке опций *optionlist*.

14.4 Пример инсталляционной процедуры

Среди больших макропакетов, описанных в этой книге, значительная часть распространяется в форме так называемого «инсталляционного комплекта» (installation kit), состоящего из инсталляционного пакетного файла и одного или нескольких .dtx- или .doc-файлов, содержащих весь программный код и всю документацию.[1]

Подобный инсталляционный комплект дает возможность запускать пакетный файл под управлением LaTeX'а. При этом система автоматически распаковывает все файлы, не требуя вмешательства пользователя. Не составляет большого

[1] В соответствии с общепринятым соглашением, документационные файлы, которые можно обрабатывать непосредственно LaTeX'ом, имеют расширение .dtx, а файлы, которым требуется управляющий файл, — расширение .doc.

```
\def\batchfile{fileerr.ins}
\input docstrip.tex
\preamble
   File to exit 'missing file name' loop in TeX.
\endpreamble
\generateFile{x.tex}       {t}{\from{fileerr.dtx}{exit}}
\generateFile{e.tex}       {t}{\from{fileerr.dtx}{edit}}
\generateFile{h.tex}       {t}{\from{fileerr.dtx}{help}}
\generateFile{s.tex}       {t}{\from{fileerr.dtx}{scroll}}
\Msg{************************************************}
\Msg{* I'm now trying to generate a file called '.tex'}
\Msg{* This may fail on some operating systems}
\Msg{************************************************}
\generateFile{.tex}        {t}{\from{fileerr.dtx}{return}}
```

Рис. 14.4. Пакетный файл для системы «file-error»

труда создать инсталляционные процедуры, генерирующие как необходимые макрокомандные файлы, так и документацию.

Принципы работы этого механизма демонстрируются на простом примере, так называемой системе «file-error». Система состоит из пакетного файла `fileerr.ins` и `.dtx`-файла `fileerr.dtx`. С помощью LaTeX'а и DOCSTRIP файл `.dtx` считывается, и на его основе генерируется несколько маленьких файлов со специальными короткими именами. Эти файлы бывают полезны в следующей ситуации. Когда TeX жалуется, что не может найти некий файл, типичная реакция пользователя — нажатие одной из обычно используемых букв — не дает результата. TeX воспринимает напечатанную букву как имя альтернативного файла, снова жалуется, пользователь не знает, как выйти из этого цикла, и т. д. Наличие в системе файлов с однобуквенными именами, соответствующими различным вариантам пользовательской реакции, помогает выйти из этого затруднения. По этому поводу см. также текст на рис. 14.7.

На рис. 14.4 приведен пакетный файл `fileerr.ins`, который сперва считывает файл `docstrip.tex`, а затем, исходя из файла `fileerr.dtx`, генерирует необходимые файлы при помощи команды `\generateFile`. Единственное, что требуется от пользователя, — запустить обработку вышеуказанного файла LaTeX'ом, после чего можно спокойно наблюдать, как LaTeX читает `.dtx`-файл `fileerr.dtx` и извлекает из него все нужные макрокомандные файлы. Такой ход событий показан на рис. 14.5, а сам исходный файл `fileerr.dtx` — на рис. 14.6. Наконец, чтобы сгенерировать документацию, показанную на рис. 14.7, достаточно обработать LaTeX'ом файл `fileerr.dtx`.

```
This is TeX, Version 3.141 (C version d)
(fileerr.ins
LaTeX2e <1994/03/06> PRELIMINARY TEST RELEASE
(/usr2/users/latex3/distrib/inputs/docstrip.tex
Utility: 'docstrip' 2.2f <1994/02/26>
English documentation     <1994/02/26>

************************************************************
* This program converts documented macro-files into fast *
* loadable files by stripping off (nearly) all comments! *
************************************************************


Generating file ./x.tex

. . . .
Processing File fileerr.dtx (exit) -> x.tex

% % % %
```

```
% \def\fileversion{v1.0d} \def\filedate{1994/12/01}
% \CheckSum{15}
% \iffalse    This is a METACOMMENT
% Doc-Source file to use with LaTeX2e
% Copyright (C) 1993-94 Frank Mittelbach, all rights reserved.
% \fi
% \title{File not found error\thanks{This file has version
% \fileversion\ last revised \filedate}}
% \author{Frank Mittelbach}
% \maketitle
%
% \section{Introduction}
%    When \LaTeXe{} is unable to find a file it will ask for an
%    alternative file name. However, sometimes the problem is
%    only noticed by \TeX{}, and in that case \TeX{} insists on
%    getting a valid file name; any other attempt to leave this
%    error loop will fail.\footnote{On some systems, \TeX{}
%    accepts a special character denoting the end of file to
%    return from this loop, e.g.\ Control-D on UNIX or Control-Z
%    on DOS.} Many users try to respond in the same way as to
%    normal error messages, e.g.\ by typing \meta{return}, or |s|
%    or |x|, but \TeX{} will interpret this as a file name and
%    will ask again.
%    \par To provide a graceful exit out of this loop, we define
%    a number of files which emulate the normal behavior of
%    \TeX{} in the error loop as far as possible.
%    \par After installing these files the user can respond with
%    |h|, |s|, |e|, |x|, and on some systems also with
%    \meta{return} to \TeX's missing file name question.
% \StopEventually{}
%
% \section{The documentation driver}
%    This code will generate the documentation. Since it is the
%    first piece of code in the file, the documentation can be
%    obtained by simply processing this file with \LaTeXe.
%    \begin{macrocode}
%<*driver>
\documentclass{ltxdoc}
\begin{document}  \DocInput{fileerr.dtx}  \end{document}
%</driver>
%    \end{macrocode}
% \section{The files}
%
% \subsubsection{Asking for help with {\tt h}}
%    When the user types |h| in the file error loop \TeX{} will
%    look for the file |h.tex|. In this file we put a message
%    informing the user about the situation (we use |^^J| to
%    start new lines in the message) and then finish with a
%    normal |\errmessage| command thereby bringing up \TeX's
%    normal error mechanism.
%    \begin{macrocode}
%<*help>
\newlinechar=`\^^J
\message{!The file name provided could not be found.^^J%
Use `<enter>' to continue processing,^^J%
`S' to scroll further errors^^J%
or `X' to terminate TeX}
\errmessage{}
%</help>
%    \end{macrocode}
%
% \subsubsection{Scrolling this and further errors with {\tt s}}
%    For the response |s| we put a message into the file |s.tex|
%    and start |\scrollmode| to scroll further error messages in
%    this run.  On systems that allow |.tex| as a file name we
%    can also trap a single \meta{return} from the user.
%    \begin{macrocode}
%<+scroll|return> \message{File ignored}
%<+scroll>        \scrollmode
%    \end{macrocode}
%
% \subsubsection{Exiting the run with {\tt x} or {\tt e}}
%    If the user enters |x| or |e| to stop \TeX{}, we need to
%    put something into the corresponding file which will force
%    \TeX{} to give up.  We achieve this by turning off terminal
%    output and then asking \TeX{} to stop: first by using the
%    internal \LaTeX{} name |\@@end|, and if that doesn't work
%    because something other than \LaTeX{} is used, by trying the
%    \TeX{} primitive |\end|.
%    \begin{macrocode}
%<+edit|exit>  \batchmode \csname @@end\endcsname \end
%    \end{macrocode}
%    We end every file with an explicit |\endinput| which prevents
%    the docstrip program from putting the character table into
%    the generated files.
%    \begin{macrocode}
\endinput
%    \end{macrocode}
%% \CharacterTable
%%  {Upper-case    \A\B\C\D\E\F\G\H\I\J\K\L\M\N\O\P\Q\R\S\T\U\V\W\X\Y\Z
%%   Lower-case    \a\b\c\d\e\f\g\h\i\j\k\l\m\n\o\p\q\r\s\t\u\v\w\x\y\z
%%   Digits        \0\1\2\3\4\5\6\7\8\9
%%   Exclamation   \!     Double quote  \"     Hash (number) \#
%%   Dollar        \$     Percent       \%     Ampersand     \&
%%   Acute accent  \'     Left paren    \(     Right paren   \)
%%   Asterisk      \*     Plus          \+     Comma         \,
%%   Minus         \-     Point         \.     Solidus       \/
%%   Colon         \:     Semicolon     \;     Less than     \<
%%   Equals        \=     Greater than  \>     Question mark \?
%%   Commercial at \@     Left bracket  \[     Backslash     \\
%%   Right bracket \]     Circumflex    \^     Underscore    \_
%%   Grave accent  \`     Left brace    \{     Vertical bar  \|
%%   Right brace   \}     Tilde         \~}
%%
% \Finale
```

Рис. 14.6. Исходный файл пакета doc для системы «file-error»

File not found error*

Frank Mittelbach

December 11, 1994

1 Introduction

When LaTeX 2ε is unable to find a file it will ask for an alternative file name. However, sometimes the problem is only noticed by TeX, and in that case TeX insists on getting a valid file name; any other attempt to leave this error loop will fail.[1] Many users try to respond in the same way as to normal error messages, e.g. by typing ⟨*return*⟩, or `s` or `x`, but TeX will interpret this as a file name and will ask again.

To provide a graceful exit out of this loop, we define a number of files which emulate the normal behavior of TeX in the error loop as far as possible.

After installing these files the user can respond with `h`, `s`, `e`, `x`, and on some systems also with ⟨*return*⟩ to TeX's missing file name question.

2 The documentation driver

This code will generate the documentation. Since it is the first piece of code in the file, the documentation can be obtained by simply processing this file with LaTeX 2ε.

```
1 ⟨*driver⟩
2 \documentclass{ltxdoc}
3 \begin{document}  \DocInput{fileerr.dtx}  \end{document}
4 ⟨/driver⟩
```

3 The files

3.1 Asking for help with `h`

When the user types `h` in the file error loop TeX will look for the file `h.tex`. In this file we put a message informing the user about the situation (we use `^^J` to start new lines in the message) and then finish with a normal `\errmessage` command thereby bringing up TeX's normal error mechanism.

```
5 ⟨*help⟩
6 \newlinechar=`\^^J
7 \message{!The file name provided could not be found.^^J%
8 Use `<enter>' to continue processing,^^J%
9 `S' to scroll further errors^^J%
10 or `X' to terminate TeX}
11 \errmessage{}
12 ⟨/help⟩
```

3.2 Scrolling this and further errors with `s`

For the response `s` we put a message into the file `s.tex` and start `\scrollmode` to scroll further error messages in this run. On systems that allow `.tex` as a file name we can also trap a single ⟨*return*⟩ from the user.

```
13 ⟨+scroll | return⟩ \message{File ignored}
14 ⟨+scroll⟩          \scrollmode
```

3.3 Exiting the run with `x` or `e`

If the user enters `x` or `e` to stop TeX, we need to put something into the corresponding file which will force TeX to give up. We achieve this by turning off terminal output and then asking TeX to stop: first by using the internal LaTeX name `\@@end`, and if that doesn't work because something other than LaTeX is used, by trying the TeX primitive `\end`.

```
15 ⟨+edit | exit⟩  \batchmode \csname @@end\endcsname \end
```

We end every file with an explicit `\endinput` which prevents the docstrip program from putting the character table into the generated files.

```
16 \endinput
```

*This file has version v1.0d last revised 1994/12/01

[1] On some systems, TeX accepts a special character denoting the end of file to return from this loop, e.g. Control-D on UNIX or Control-Z on DOS.

Рис. 14.7. Вывод, полученный в результате обработки LaTeX'ом файла `fileerr.dtx`

Приложение А

LaTeX: основы программирования

В этом приложении рассматриваются базовые понятия и приемы, лежащие в основе программирования для LaTeX'а. В первом разделе объясняется, как следует определять новые команды и окружения, в том числе содержащие необязательные аргументы. Другие темы первого раздела — счетчики в LaTeX'е, а также техника работы с вертикальными и горизонтальными промежутками. Фундаментальное значение имеет тематика второго раздела — боксы в (La)TeX'е и их использование. Овладение соответствующим кругом понятий совершенно необходимо для полного усвоения материала книги в целом. Далее подробно описан интерфейс макропакета LaTeX 2_ε, позволяющий пользователю вводить нестандартные опции в макропакеты и классы. Вычисления и реализация логических схем управления в LaTeX'е упрощаются при использовании макропакетов calc и ifthen. Эти пакеты, описанные в последних двух разделах, в действительности использовались в многочисленных примерах LaTeX'овских программ на протяжении всей книги.

А.1 Разметка и форматирование

Этот раздел посвящен главным образом синтаксису LaTeX'а. Очень важно, чтобы при определении пользователем новых команд и окружений применялись не низкоуровневые TeX'овские команды, а исключительно LaTeX'овские конструкции, описанные ниже. В таком случае, помимо тех преимуществ, которые дает LaTeX'овский механизм контроля совместимости и непротиворечивости, появляется гарантия, что создаваемые макрорасширения и документы будут совместимы (возможно, полностью) с будущими версиями LaTeX'а.

A.1.1 Определение новых команд

Потребность в определении новых команд возникает весьма часто. Целью может быть, например, уменьшение объема набираемого текста благодаря использованию сокращений вместо часто встречающихся комбинаций символов и/или команд. Для определения новых команд используется команда `\newcommand` [Л 55,173], [Л 210,218], имеющая один необязательный аргумент — число аргументов у новой команды.

`\newcommand{\name}[narg]{command definition}`

Возможные значения числа аргументов — $0 \leq narg \leq 9$. Если новая команда не имеет аргументов, значение [0] можно не указывать. В «замещающем тексте»[1] определения — *command definition* — аргументы обозначаются через #1, #2, ..., #*narg*.

PostScript и его разновидность Encapsulated PostScript часто используются для вставки графических изображений в LaTeX'овские документы ...

```
\newcommand{\Ps}{Post\-Script}
\newcommand{\EPs}{Encapsulated \Ps}
\Ps{} и его разновидность \EPs{} часто
используются для вставки графических
изображений в \LaTeX'овские
документы \ldots
```

Предыдущий пример нетрудно обобщить, добившись того, чтобы одновременно с определением команды соответствующая информация сразу же вносилась в предметный указатель.

```
\newcommand{\PsI}{\Ps\index{\Ps}}
\newcommand{\EPsI}{Encapsulated \Ps\index{Encapsulated \Ps}%
                   \index{\Ps!Encapsulated}}\index{EPS}}
```

Нередко требуется, чтобы определяемая команда работала как в математическом, так и в текстовом режимах. Такая особенность команды должна быть специально предусмотрена в ее определении. Можно, например, использовать следующую конструкцию:

Последовательность $x_1, \ldots, x_n$ или $x_1, \ldots, x_n + G_{x_1, \ldots, x_n}$

```
\newcommand{\xvec}{\mbox{$x_1,\ldots,x_n$}}
Последовательность \xvec\ или $\xvec+G_{\xvec}$
```

[1] В терминологии, принятой, например, в книге С. М. Львовского [45, с. 210], два обязательных аргумента команды `\newcommand` — это имя определяемой макрокоманды и «замещающий текст», которым будет замещаться имя макрокоманды в процессе обработки документа LaTeX'ом.— *Прим. перев.*

Однако гораздо более правильными подобные определения получаются в LaTeX 2_ε, где для этого есть специальная команда:

`\ensuremath{`*math code*`}`

Как следует из самого имени команды `\ensuremath`, ее аргумент в любом случае будет входить в документ в математическом режиме (для этого, если потребуется, он будет окружен символами $). При помощи данной команды предыдущее определение можно модифицировать следующим образом:

Последовательность $x_1, \ldots, x_n$ или $x_1, \ldots, x_n + G_{x_1,\ldots,x_n}$

```
\newcommand{\xvec}{\ensuremath{x_1,\ldots,x_n}}
Последовательность \xvec\ или $\xvec+G_{\xvec}$
```

Очевидно, последний вариант определения имеет то преимущество, что размер символов, входящих в замещающий текст, автоматически уменьшается, когда команда используется в верхнем или нижнем индексах. Конструкция, основанная на `\mbox`, этим свойством не обладает.

Другое возможное обобщение команды `\xvec` состоит в том, чтобы буква, используемая для обозначения переменных, не фиксировалась в определении, а задавалась бы в качестве *аргумента*:

Последовательность $y_1, \ldots, y_n$ или $z_1, \ldots, z_n \ldots$

```
\newcommand{\avec}[1]{%   arg1: имя вектора
                \ensuremath{#1_1,\ldots,#1_n}}
Последовательность \avec{y} или $\avec{z}$
\ldots
```

В примере ниже определения команд дополнительно содержат указания относительно полиграфического стиля: сообщается, что сокращения, используемые для названий элементарных частиц, должны быть набраны прямым шрифтом.

Массы частиц W^-, W^+ и Z^0 равны, соответственно, m_{W} и m_{Z^0}.
Обычно $m_{\mathrm{W}} > m_{\mathrm{Z}^0}$.

```
\newcommand{\PWm}{\ensuremath{\mathrm{W}^-}}
\newcommand{\PWp}{\ensuremath{\mathrm{W}^+}}
\newcommand{\PZz}{\ensuremath{\mathrm{Z}^0}}
\newcommand{\PMW}{\ensuremath{m_{\mathrm{W}}}}
\newcommand{\PMZ}{\ensuremath{m_{\mathrm{Z}^0}}}
Массы частиц \PWm, \PWp\ и \PZz\
равны, соответственно, \PMW\ и \PMZ.
\par Обычно \(\PMW > \PMZ\).
```

Ранее определенные команды могут быть *переопределены* при помощи команды `\renewcommand`. [$\mathcal{L}$ **57,173**], [$\mathcal{A}$ **235,238**]. Заметим, что переопределенная команда не обязана иметь столько же аргументов, сколько имела исходная. В частности, команду `\PMW` можно переопределить, введя в определение один аргумент:

Обычно $m_{\mathrm{W}+} = m_{\mathrm{W}-}$.

```
\renewcommand{\PMW}[1]{%
                \ensuremath{m_{\mathrm{W}^{#1}}}}
Обычно \(\PMW{+} = \PMW{-}\).
```

При переопределении команды (или окружения, см. ниже) следует, разумеется, проявлять осторожность, поскольку может оказаться, что переопределяемая команда используется основным классом или одним из подгруженных макропакетов (читатель может в качестве эксперимента попробовать переопределить команду `\uppercase` в процессе работы с документом, формат которого основан на классе book).

В LaTeX2ε имеется также возможность определять команды с необязательным первым аргументом. Такое определение имеет следующий вид:

`\newcommand{\`*mycom*`}[`*narg*`][`*default*`]{`*command definition*`}`

Вот два примера:

```
\newcommand{\LB}[1][3]{\linebreak[#1]}
\newcommand{\PK}[1][0]{\ensuremath{\mathrm{K}^{#1}}}
```

Значение необязательного аргумента, используемое по умолчанию, указывается внутри второй пары квадратных скобок (в первом случае это 3, во втором — 0). Внутри замещающего текста необязательный аргумент обозначается через `#1`, а обязательные (если таковые имеются) — через `#2`, ... , *`#narg`*. С учетом сказанного, команда `\LB` — это сокращение от `\linebreak[3]`, а команда `\LB[2]` содержит фактически используемое значение, т. е. производит то же действие, что и команда `\linebreak[2]`. В приведенном ниже примере используется вторая из определенных нами команд — `\PK`.

Три K-частицы K^0, K^+ и K^-. `Три K-частицы \PK, \PK[+] и \PK[-].`

Вообще говоря, определяя макрокоманды-сокращения, целесообразно наиболее часто встречающиеся комбинации символов заменять короткими командами без параметров, а встречающиеся относительно реже — более длинными командами с необязательным аргументом.

Чуть более сложно устроена команда `\Rule`, порождающая вертикальную линейку и имеющая один необязательный аргумент — ширину линейки (по умолчанию равную 0.4 pt) и один обязательный — ее высоту. Читателю рекомендуется обратить внимание на приводимые примеры использования этой команды с явно заданным необязательным аргументом. В последнем примере команда создает «невидимую линейку» (линейку нулевой ширины), которая позволяет раздвинуть по вертикали рамку, порождаемую командой `\fbox`.

текст текст | текст ▮ текст [xxx] и [xxx].

```
\newcommand{\Rule}[2][.4pt]{\rule{#1}{#2}}
текст текст \Rule{4mm} текст \Rule[1mm]{4mm}
текст \fbox{xxx} и \fbox{\Rule[0mm]{4mm}xxx}.
```

В LaTeX2$_\varepsilon$ для определения макрокоманд имеется еще одна возможность:

\providecommand{*mycom*}[*narg*][*default*]{*command definition*}

Эта команда работает в точности так же, как и \newcommand, с единственным отличием, что если определяемая ею макрокоманда раньше уже была определена, то вновь вводимое определение игнорируется. Команда \providecommand бывает полезна при наборе текстов, которые предполагается затем многократно вставлять в различные документы, например, при подготовке библиографической базы данных для последующего генерирования списков литературы. В частности, используя \providecommand вместо \newcommand в BibTeX'овской команде @PREAMBLE для набора логотипов и других специфических вещей, используемых в BibTeX'овских записях, можно избавиться от сообщений об ошибке в тех случаях, когда вводимые макрокоманды уже встречались в данном документе раньше.

A.1.2 Определение новых окружений

Определение и переопределение окружений осуществляются соответственно с помощью команд \newenvironment и \renewenvironment [*Л* **57,173**], [*Л* **235,238**]. В каждом отдельном случае требуется точное описание действий, которые должны выполняться при «входе» в окружение и «выходе» из него. Для окружения с условным именем myenv вход и выход осуществляют команды \begin{myenv} и \end{myenv}, помещенные в соответствующие места документа.

\newenvironment{*name*}[*narg*]{*begdef*}{*enddef*}
\renewenvironment{*name*}[*narg*]{*begdef*}{*enddef*}

Как и в случае команды \newcommand, число аргументов, или параметров, у этих команд лежит в диапазоне $0 \le narg \le 9$, причем в случае, когда параметров вообще нет, [0] можно не писать. В обозначенном через *begdef* описании процедуры открытия окружения, параметры обозначаются через #1, ... , #*narg*. Если параметры у окружения имеются, они должны быть заданы при *входе в окружение* посредством их явного указания в команде \begin{myenv}:

\begin{myenv}{*arg_1*}...{*arg_k*}

Выход из окружения производит команда \end{myenv}, не имеющая собственных параметров. Более того, в процедуре закрытия окружения *myenv* (обозначенной через *enddef*) не могут фигурировать и параметры команды \begin{myenv}, открывшей это окружение. Это означает, что об информации, необходимой в момент закрытия окружения, нужно позаботиться отдельно (см. в связи с этим приведенное ниже определение окружения Citation).

В качестве первого примера рассмотрим окружение Abstract, используемое для оформления краткого резюме (аннотации) содержания статьи или книги.

Данное окружение открывается заголовком «Abstract», центрированным и напечатанным полужирным шрифтом, за которым следует текст аннотации, заключенный в окружение quote.

Abstract

This abstract explains the approach used to develop the necessary tools to solve the problems at hand.

```
\newenvironment{Abstract}
  {\begin{center}\textbf{Abstract}%
   \end{center} \begin{quote}}
  {\end{quote}}
\begin{Abstract}
  This abstract explains the approach used
  to develop the necessary tools to solve
  the problems at hand.
\end{Abstract}
```

Окружение Citation, рассматриваемое в следующем, более сложном примере, удобно использовать для оформления цитат из высказываний известных людей.

```
\newcounter{Citctr}\newsavebox{\Citname}
\newenvironment{Citation}[1]
   {\stepcounter{Citctr}%
    \sbox{\Citname}{\textit{#1}}
    \begin{description}\item[Citation \arabic{Citctr}]}
   {\hspace*{\fill}\nolinebreak[1]\hspace*{\fill}%
    \usebox{\Citname}\end{description}}
```

В приведенном определении вводится счетчик Citctr, нумерующий цитаты, и бокс \Citname[2], в который записывается имя (фамилия) автора цитируемого высказывания для последующего создания подписи под цитатой. Необходимость иметь специальный бокс для запоминания значения аргумента окружения (каковым является имя автора) связана с тем, что подпись создается командой \end{Citation}, когда, как говорилось выше, непосредственно использовать значения аргументов команды \begin{Citation} уже нельзя. Итак, при входе в окружение Citation текущее значение счетчика цитат увеличивается на единицу, а значение аргумента окружения, аттрибутированное курсивом, запоминается в боксе \Citname. После этого открывается окружение description, в котором команда \item используется только один раз, причем ее аргументом служит слово «Citation» с номером, равным текущему значению счетчика цитат. При выходе из окружения Citation вначале в документ вносятся два растяжимых горизонтальных промежутка с указанием на то, что переход на новую строку после первого из них нежелателен, хотя и не запрещен. Затем в документ вносится содержимое бокса \Citname, после чего окружение description закрывается. В результате имя автора окажется прижатым вправо и напечатанным либо в последней строке цитаты, либо, если там для этого не хватает места, в следующей строке (см. приведенный ниже пример). Во втором случае последняя строка цитаты будет прижата влево, а не выровнена и по левому, и по правому краям (причем, возможно,

[2] В терминологии, используемой в книге С. М. Львовского [45, с. 264], это называется блоковой переменной.— *Прим. перев.*

с большими пустотами между словами), как получилось бы при использовании более примитивной конструкции[3].

Citation 1 Necessity is the plea for every infringement of human freedom.
William Pitt

This is some regular text in between two Citation environments.

Citation 2 Man is the measure of all things. *Protagoras*

More regular text ...

Citation 3 On mourra seul.
Blaise Pascal

```
\begin{Citation}{William Pitt}
  Necessity is the plea for every infringement
  of human freedom.
\end{Citation}
This is some regular text in between two
Citation environments.
\begin{Citation}{Protagoras}
  Man is the measure of all things.
\end{Citation}
More regular text \ldots
\begin{Citation}{Blaise Pascal}
   On mourra seul.
\end{Citation}
```

Подробнее о командах создания счетчиков и боксов, использованных в рассмотренном только что примере, написано в разд. А.1.3 и разд. А.2. Так же как и при определении макрокоманд с помощью команды `\newcommand`, в LaTeX 2_ε имеется возможность сделать первый аргумент создаваемого окружения необязательным. В общем случае команда, создающая новое окружение, имеет следующий вид:

`\newenvironment{`*myenv*`}[`*narg*`][`*default*`]{`*begdef*`}{`*enddef*`}`

Значение необязательного аргумента, принимаемое по умолчанию, указывается внутри второй пары квадратных скобок (`[`*default*`]`).

Внутри описания процедуры открытия окружения (*begdef*) необязательный аргумент обозначается через `#1`, а обязательные (если таковые имеются) — через `#2`, ..., `#`*narg*.

В случае, когда окружение *myenv* используется без необязательного параметра, `#1` совпадает с той последовательностью символов, которая была задана как `[`*default*`]`.

В качестве иллюстрации приведем определение окружения `deflist` [14], которое является разновидностью окружения `Ventry`, рассмотренного в разд. 3.2.2. Окружение `deflist` ведет себя как обычное окружение `description`, если необязательный аргумент окружения не задан. Если же необязательный аргумент задан, то его длина принимается в качестве расстояния, зарезервированного для заголовков элементов перечня. Поэтому если в качестве значения необязательного аргумента взять заголовок элемента перечня, имеющий наибольшую длину, то можно быть уверенным, что описания всех элементов будут выровнены надлежащим образом.

---

[3] В примере приводятся следующие высказывания:
Уильям Питт (William Pitt). Поводом для ущемления человеческой свободы всегда служит необходимость.
Протагор (Protagoras). Человек есть мера всех вещей.
Блэз Паскаль (Blaise Pascal). Каждый умирает в одиночку.
— *Прим. перев.*

Первая часть следующего примера демонстрирует поведение окружения `deflist` в случае, когда значение необязательного аргумента не задано, вторая часть показывает, чего удается добиться, используя необязательный аргумент.

Исходный текст

```
\newenvironment{deflist}[1][\quad]{\begin{list}{}{%
   \renewcommand{\makelabel}[1]{\textbf{##1}\hfil}%
   \settowidth{\labelwidth}{\textbf{#1}}\setlength{\leftmargin}%
      {\labelwidth+\labelsep}}}{\end{list}}
\begin{deflist}
\item[Метка] Элемент с короткой меткой.
\item[Длинная метка] Элемент с длинной меткой.
\item[Очень длинная метка] Элемент с очень длинной меткой.
\end{deflist}
\begin{deflist}[Очень длинная метка]
\item[Метка] Элемент с короткой меткой.
\item[Длинная метка] Элемент с длинной меткой.
\item[Очень длинная метка] Элемент с очень длинной меткой.
\end{deflist}
```

Метка Элемент с короткой меткой.

Длинная метка Элемент с длинной меткой.

Очень длинная метка Элемент с очень длинной меткой.

Метка Элемент с короткой меткой.

Длинная метка Элемент с длинной меткой.

Очень длинная метка Элемент с очень длинной меткой.

Текст на выводе

A.1.3 Создание счетчиков и изменение их текущих значений

Всякое число, генерируемое LaTeX'ом, в действительности порождается соответствующим *счетчиком* [$\mathcal{L}$ **91**], [$\mathcal{A}$ **219**]. Как правило, имя счетчика с точностью до символа \ совпадает с именем использующего его окружения или команды. Вот перечень всех счетчиков, используемых в стандартных LaTeX'овских классах:

`part`	`paragraph`	`figure`	`enumi`
`chapter`	`subparagraph`	`table`	`enumii`
`section`	`page`	`footnote`	`enumiii`
`subsection`	`equation`	`mpfootnote`	`enumiv`
`subsubsection`			

Для любого окружения, создаваемого при помощи команды \newtheorem, также автоматически создается счетчик с именем данного окружения, за исключением тех случаев, когда необязательный аргумент указывает, что нумерация должна осуществляться в соответствии с текущим значением счетчика некоторого другого окружения.

Значениями счетчика могут быть только целые числа, каждый счетчик может иметь лишь одно текущее значение. Номер (раздела, уравнения, теоремы и т. п.) может быть комбинацией значений нескольких счетчиков; как правило, именно так нумеруются подразделы. Например, в классах book и report номер 7.4.5 присваивается пятому подразделу четвертого раздела седьмой главы.

Управление LaTeX'овскими счетчиками

Ниже перечислены все основные LaTeX'овские команды, предназначенные для создания счетчиков и модификации их значений [Л 92,175], [Л 200-3]. Возможности этих команд значительно возрастают, если они используются совместно с пакетом calc, описанным в разд. А.4.

\newcounter{*newctr*}[*oldctr*]

Эта команда создает новый счетчик *newctr* и присваивает ему нулевое значение. Данное определение — глобальное. Если счетчик с именем *newctr* уже был определен ранее, на дисплей выводится сообщение об ошибке. В качестве необязательного аргумента может быть задано имя некоторого другого, уже существующего счетчика — *oldctr*; в этом случае значение нового счетчика *newctr* «обнуляется» всякий раз, когда значение счетчика *oldctr* увеличивается при помощи одной из команд \stepcounter или \refstepcounter. Кроме того, данная команда определяет \the*newctr* как \arabic{*newctr*}.

\setcounter{*ctr*}{*val*}

Этой командой счетчику *ctr* глобально присваивается значение *val*.

\addtocounter{*ctr*}{*val*}

Эта команда глобально увеличивает значение счетчика *ctr* на величину *val*.

\value{*ctr*}

Этой командой считывается текущее значение счетчика *ctr*, чаще всего для того, чтобы использовать его в качестве аргумента *val* команд \setcounter и \addtocounter.

\stepcounter{*ctr*}

Эта глобальная команда увеличивает значение счетчика *ctr* и одновременно «обнуляет» все подчиненные счетчики, т. е. те, которые вводились командой

\newcounter, где в качестве необязательного аргумента *oldctr* использовалось имя *ctr*, либо фигурировали в качестве первого аргумента команды \@addtoreset при значении второго аргумента, равном *ctr* (см. разд. 2.3.1).

\refstepcounter{*ctr*}

Эта команда работает так же, как \stepcounter, но дополнительно определяет соответствующее значение \ref как текст, генерируемый командой \the*ctr*.

Приведенные ниже команды позволяют управлять тем, как содержимое счетчика будет выглядеть на печати.

\arabic{*ctr*} Значение счетчика *ctr* печатается арабскими цифрами.

\roman{*ctr*} Значение счетчика *ctr* печатается римскими цифрами с использованием строчных латинских букв.

\Roman{*ctr*} Значение счетчика *ctr* печатается римскими цифрами с использованием прописных латинских букв.

\alph{*ctr*} Значение счетчика *ctr* отображается строчными латинскими буквами: a, b, ... , z. При этом значение *ctr* не должно превосходить 26.

\Alph{*ctr*} Значение счетчика *ctr* отображается прописными латинскими буквами: A, B, ... , Z. При этом значение *ctr* не должно превосходить 26.

\fnsymbol{*ctr*} Значение счетчика *ctr* отображается одним из символов, используемых для маркировки сносок: *, †, ... Эту команду можно использовать только в математическом режиме, причем значение *ctr* не должно превосходить 9.

\the*ctr* Команда выдачи на печать значения счетчика *ctr* (в форме, заданной при помощи одной из перечисленных выше команд).

В качестве иллюстрации к сказанному приведем определения счетчиков, используемые в стандартных LaTeX'овских классах.

Для нумерации разделов используются определения, эквивалентные следующим:

```
\newcounter{part}
\newcounter{section}
\newcounter{subsection}[section]
\newcounter{subsubsection}[subsection]
\renewcommand{\thepart}          {\Roman{part}}
\renewcommand{\thesection}       {\arabic{section}}
\renewcommand{\thesubsection}    {\thesection.\arabic{subsection}}
\renewcommand{\thesubsubsection}{\thesubsection.\arabic{subsubsection}}
```

Здесь ясно видно, как значения счетчиков нижних уровней (подчиненных) обнуляются при изменении значений счетчиков более высоких уровней (подчиняющих), а также и то, как представления значений счетчиков (команды типа `\the...`) конструируются на основе значений текущего счетчика и счетчика более высокого уровня. Видно, между прочим, что счетчик самого верхнего уровня `part` не оказывает никакого влияния на счетчики более низких уровней.

В табл. 3.2 показана структура счетчиков, используемых в перечнях `enumeration`. В действительности эти счетчики определены в файле `latex.tex`, содержащем базовые определения LaTeX'а. На уровне же LaTeX'овских классов определяются только визуальное представление счетчиков и тип нумерации элементов перечня:

```
\renewcommand{\theenumi}{\arabic{enumi}}
  \newcommand{\labelenumi}{\theenumi.}
\renewcommand{\theenumii}{\alph{enumii}}
  \newcommand{\labelenumii}{(\theenumii)}
\renewcommand{\theenumiii}{\roman{enumiii}}
  \newcommand{\labelenumi}{\theenumiii.}
\renewcommand{\theenumiv}{\Alph{enumiv}}.
  \newcommand{\labelenumi}{\theenumiv.}
```

A.1.4 Управление параметрами расстояния

Как и в TeX'е, в LaTeX'е есть два основных типа параметров расстояния (длин): «жесткие» (т. е. фиксированные) длины (в книге Д. Кнута [30] для них используется обозначение `<dimen>`) и «растяжимые» длины (обозначаемые в [30] через `<skip>`), характеризуемые «нормальной» длиной и некоторыми «пределами эластичности» при растяжении и сжатии. В LaTeX'е новая длина всегда изначально относится к более общему типу `<skip>`, поэтому пользователь имеет возможность задать ее как жесткую либо как растяжимую. В последнем случае требуется лишь указать допустимые пределы растяжения и сжатия (параметры с префиксами `plus` и `minus` соответственно). Все стандартные длины в LaTeX'е, напротив, относятся к типу жестких, за исключением нескольких растяжимых длин, перечисленных в приложении C книги LaTeX book [$\mathcal{L}$ **149–205**]. В LaTeX'е имеются следующие команды для задания и изменения длин [$\mathcal{L}$ **95,193**], [$\mathcal{M}$ **122,259**].

`\fill`

Данная команда задает длину с нормальным значением нуль, обладающую неограниченной растяжимостью (т. е. растяжимую до любого положительного значения). Эту команду не следует переопределять ни при каких обстоятельствах!

`\stretch{`*dec_num*`}`

Растяжимая длина, определяемая этой командой, является более универсальной и, пожалуй, более полезной. Команда `\fill` эквивалентна команде

\stretch{1}. В общем случае \stretch{*dec_num*} имеет растяжимость, эквивалентную *dec_num*-кратной команде \fill. Используется для создания тонких эффектов выравнивания текста по горизонтали или вертикали.

\newlength{*cmd*}

Эта команда определяет новую переменную длины и присваивает ей имя *cmd*. Если команда *cmd* уже существует, выводится соответствующее сообщение об ошибке. Созданная длина, которой присваивается значение 0 pt, является растяжимой. В следующих ниже примерах текущее значение введенной переменной выводится на печать при помощи команды \the.

```
\newlength{\Mylen} % Декларируется новая длина \Mylen
Mylen = \the\Mylen % Текущее значение длины выводится
                   % на печать
```

Mylen = 3.08331pt plus 1.54166pt minus 1.02777pt

\setlength{*cmd*}{*len*}

Переменной длины *cmd* присваивается значение *len*.

```
\setlength{\Mylen}{10mm} % Присваивается значение 10мм
Mylen = \the\Mylen       % Используется растяжимая длина
\setlength{\Mylen}{5mm plus 1mm minus .5mm}
\par Mylen = \the\Mylen
```

Mylen = 28.45274pt
Mylen = 14.22636pt plus 2.84526pt minus 1.42262pt

Единицы измерения, которые можно использовать для задания длин, приведены в табл. А.1. Следует помнить о разнице между типографским пунктом (pt), обычно используемом в TEX'e, и так называемом большом пункте (big point, сокращенно — bp), который используется, например, в PostScript'e. В частности, отводя в документе место для PostScript'овской картинки, следует указывать размер ограничивающего прямоугольника изображения (bounding box) в bp — только в этом случае результат будет действительно точным.

\addtolength{*cmd*}{*len*}

Эта команда добавляет к текущему значению переменной длины *cmd* величину *len*.

```
\setlength{\Mylen}{1em}
Mylen = \the\Mylen
\addtolength{\Mylen}{1pc}  % Добавляется 1 пика
\par Mylen = \the\Mylen
```

Mylen = 9.24994pt
Mylen = 21.24994pt

\settowidth{*cmd*}{*text*}

Переменной *cmd* присваивается значение ширины текста *text* (т. е. величины того промежутка, который занимает на бумаге напечатанный текст *text*). С

sp	Приведенный пункт (scaled point) (65536 sp = 1 pt) Наименьшая TeX'овская единица длины	
pt	Пункт (point) = $\frac{1}{72.27}$ in = 0.351 mm	▮
bp	Большой пункт (big point) (72 bp = 1 in), известен и как PostScript'овский пункт	▮
dd	Пункт Дидо (Didôt point) = $\frac{1}{72}$ французского дюйма, = 0.376 mm	▮
mm	Миллиметр (millimeter) = 2.845 pt	␣
pc	Пика (pica) = 12 pt = 4.218 mm	␣
cc	Цицеро (cicero) = 12 dd = 4.531 mm	␣
cm	Сантиметр (centimeter) = 10 mm = 2.371 pc	␣
in	Дюйм (inch) = 25.4 mm = 72.27 pt = 6.022 pc	␣
ex	Высота строчной буквы "x" в текущем шрифте	␣
em	Ширина прописной буквы "M" в текущем шрифте	␣
mu	Математическая единица (18 mu = 1 em), используемая для измерения расстояний в математическом режиме	▮

Таблица А.1. (LA)TeX'овские единицы длины

помощью этой команды можно задавать длины, меняющиеся в зависимости от количества текста и от используемого кегля.

```
% \Mylen --- это ширина текста ABCD в текущем шрифте
\settowidth{\Mylen}{ABCD}
Mylen = \the\Mylen
% Укрупняем шрифт и пересчитываем ширину
\settowidth{\Mylen}{\large ABCD}
\par Mylen = \the\Mylen
```

```
Mylen = 27.23311pt
Mylen = 34.58855pt
```

`\settoheight{`*cmd*`}{`*text*`}` `\settodepth{`*cmd*`}{`*text*`}`

LaTeX 2ε дополнительно содержит еще две команды, которые работают так же, как `\settowidth`, но измеряют не ширину, а соответственно высоту и глубину текста *text*.

Горизонтальные промежутки

Пустые горизонтальные промежутки создаются командой `\hspace` [Л **95,96,193**], [И **91,92,123**]. Команда `\hspace*` отличается от `\hspace` тем, что создаваемый ею промежуток не игнорируется даже в том случае, когда попадает в начало или в конец строки. Имеющиеся в LaTeX'е команды для создания горизонтальных промежутков перечислены в табл. А.2.

\enspace	создает промежуток, равный 0.5 \quad
\quad	создает промежуток, равный 1em в текущем шрифте
\qquad	удвоенный \quad
\hfill	горизонтальный промежуток, растяжимый в пределах от 0 до ∞
\hrulefill	аналогично \hfill, но промежуток заполняется горизонтальной прямой
\dotfill	аналогично \hfill, но промежуток заполняется последовательностью точек

Таблица А.2. Стандартные горизонтальные промежутки

Как показывает следующий пример, пробелы перед командой \hspace и после нее LaTeX'ом не игнорируются.

Вот пустой промежуток шириной 0.5 дюйма.
Вот пустой промежуток шириной 0.5 дюйма.
Вот пустой промежуток шириной 0.5 дюйма.

```
\par Вот пустой промежуток
     шириной\hspace{0.5in}0.5~дюйма.
\par Вот пустой промежуток
     шириной \hspace{0.5in}0.5~дюйма.
\par Вот пустой промежуток
     шириной \hspace{0.5in} 0.5~дюйма.
```

Следующий пример показывает, как при помощи «растяжимых длин» можно гибко и с высокой точностью управлять размещением текста в пределах строки. Заметим, что \hfill в действительности есть не что иное, как сокращенная запись команды \hspace{\fill}. Используемая ниже команда \HS имеет необязательный числовой параметр, в отсутствие которого она ведет себя просто как \hfill, а при задании параметра обеспечивает бóльшую либо меньшую растяжимость создаваемого пустого промежутка, чем команда \hfill (последняя соответствует единичному значению параметра).

слева справа
слева $\boxed{\frac{1}{3}}$ справа
слева в центре справа
слева ___ в центре ___ справа
слева ..... справа
слева .... справа
слева .... справа
слева .... справа

```
\newcommand{\HS}[1][1.]{\hspace{\stretch{#1}}}
\begin{center}
слева \hfill                                  справа\\
слева \HS[.5]\fbox{$\frac{1}{3}$}\hfill       справа\\
слева \HS          в центре \hfill            справа\\
слева \hrulefill\ в центре \hrulefill\        справа\\
слева \dotfill\                               справа\\
слева \dotfill\ \HS[.5] \dotfill\             справа\\
слева \dotfill\ \HS     \dotfill\             справа\\
слева \dotfill\ \HS[2.] \dotfill\             справа
\end{center}
```

Вертикальные промежутки

Пустые вертикальные промежутки создаются командой \vspace [$\mathcal{L}$ **95,96,193**], [$\mathcal{M}$ **121–2**], аналогичной команде \hspace. В частности, команда \vspace* создает промежуток, который не игнорируется даже в том случае, когда попадает в на-

`\smallskip`	Вертикальный промежуток, равный `\smallskipamount` (по умолчанию это приблизительно четверть величины `\baselineskip`).
`\medskip`	Вертикальный промежуток, равный `\medskipamount` (по умолчанию это приблизительно половина величины `\baselineskip`).
`\bigskip`	Вертикальный промежуток, равный `\bigskipamount` (по умолчанию это приблизительно `\baselineskip`).
`\vfill`	Вертикальный промежуток, растяжимый в пределах от 0 до ∞.

Таблица А.3. Стандартные вертикальные промежутки

чало или в конец страницы (промежуток, создаваемый командой `\vspace` в этом случае удаляется). Имеющиеся в LaTeX'е команды для создания вертикальных промежутков (общие для всех стандартных классов) перечислены в табл. А.3.

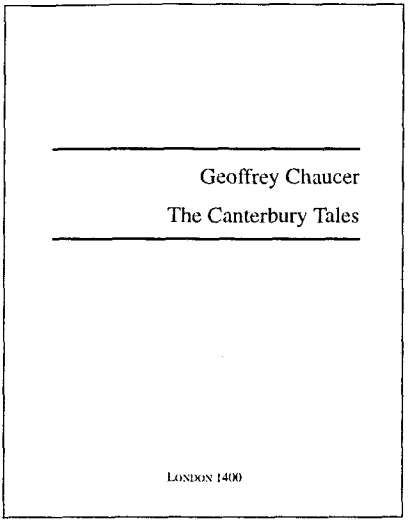
Поведение команды `\vspace` нередко приводит в замешательство пользователей LaTeX'а. Если команда используется внутри абзаца, то вертикальный промежуток создается почему-то только после завершения содержащей ее строки, тогда как будучи помещенной между абзацами, команда `\vspace` вроде бы ведет себя вполне предсказуемо.

При использовании внутри абзаца команда `\vspace` ведет себя довольно странно. Чтобы избавиться от нежелательного вертикального пробела между строками, можно попробовать задать отрицательное значение параметра (отрицательную величину промежутка).

Впрочем, более осмысленным является размещение команды между абзацами, когда появление пустого вертикального промежутка оказывается вполне предсказуемым и контролируемым.

```
При \vspace{5mm}использовании внутри абзаца
команда \verb!\vspace! ведет себя довольно
странно. Чтобы избавиться от нежелательного
вертикального пробела между строками, можно
попробовать задать отрицательное значение
параметра (отрицательную величину
промежутка).

\vspace{\baselineskip}

Впрочем, более осмысленным является
размещение команды между абзацами, когда
появление пустого вертикального промежутка
оказывается вполне предсказуемым и
контролируемым.
```

Растяжимые промежутки оказываются полезными и для управления вертикальным размещением материала в документе. Команда `\vfill` — это в действительности сокращенная запись команды `\vspace{\fill}`. Более общей конструкцией, позволяющей эффективно управлять верстской страниц, является комбинация команд `\vspace` и `\stretch`. Например, с ее помощью сгенерирована титульная страница, показанная на рис. А.1, где имя автора и название книги помещены «со спуском» от верхнего края, равным приблизительно одной трети высоты страницы, а место и год публикации опущены к нижнему краю.




```
\documentclass{article}
\usepackage{times}
\thispagestyle{empty}
\newcommand{\HRule}{\rule{\linewidth}{1mm}}
\setlength{\parindent}{0mm}
\setlength{\parskip}{0mm}
\begin{document}
  \vspace*{\stretch{1}}
  \HRule
  \begin{flushright}
    \Huge Geoffrey Chaucer\\[5mm]
          The Canterbury Tales
  \end{flushright}
  \HRule
  \vspace*{\stretch{2}}
  \begin{center}
    \Large\textsc{London 1400}
  \end{center}
\end{document}
```

Рис. А.1. Пример титульной страницы (Джеффри Чосер, *Кентерберийские рассказы*, Лондон, 1400)

A.2 Разметка страниц: различные типы боксов

Модель создания страниц из боксов без преувеличения может считаться центральной идеей TEX'а, на ней же основаны и многие LATEX'овские конструкции.

Бокс — это объект, рассматриваемый TEX'ом как один символ. Бокс не может быть расщеплен на части, его нельзя поделить между несколькими строками или страницами. Боксы можно сдвигать вверх, вниз, влево и вправо. В LATEX'е есть три типа боксов:

LR[4] (LR-бокс, или бокс-строка) [$\mathcal{L}$ **97,98**], [$\mathcal{A}$ **243–4**] Содержимое такого бокса печатается слева направо.

Par[5] (бокс-абзац) [$\mathcal{L}$ **98–100**], [$\mathcal{A}$ **245–7**] Боксы этого типа могут содержать по несколько строк, которые будут напечатаны как обычный текст в виде одного абзаца. Боксы-абзацы помещаются друг под другом. Пользователь может управлять их шириной.

Rule (бокс-линейка) Линия (тонкая или толстая) [$\mathcal{L}$ **100**], [$\mathcal{A}$ **137**], обычно используемая для того, чтобы отделить друг от друга логически различные элементы страницы, например, поделить таблицу на столбцы и строки или же отделить колонтитул от основного текста.

[4] От английского left–right.— *Прим. перев.*
[5] От английского paragraph.— *Прим. перев.*

A.2.1 LR-боксы

`\mbox{`*text*`}`	`\fbox{`*text*`}`
`\makebox[`*width*`][`*pos*`]{`*text*`}`	`\framebox[`*width*`][`*pos*`]{`*text*`}`

Команды, приведенные в первой строке, создают бокс и помещают в него текст *text*, стоящий в фигурных скобках, в виде одной строки [*Л* 97,194], [*Л* 243,247]. Размер бокса определяется шириной текста. Различие между командами в том, что `\fbox` дополнительно окаймляет бокс рамкой: из `\fbox{пара слов}` на печати получается пара слов. Команды, записанные во второй строке, являются обобщением первых двух. Они дают возможность пользователю задавать ширину бокса и управлять расположением текста внутри бокса.

пара слов
пара слов

```
\makebox[5cm]{пара слов}      \par
\framebox[5cm][r]{пара слов}
```

Помимо центрирования, которое обеспечивается указанием позиционирующего параметра `[c]` и принято по умолчанию, можно сдвинуть текст в крайнее левое положение (при параметре `[l]`) или в крайнее правое (при параметре `[r]`). Кроме того, LaTeX 2ε позволяет указать в качестве значения параметра `[s]`[6] — в этом случае текст *text* растянется от левой до правой границы бокса (конечно, при условии, что в тексте содержатся растяжимые промежутки, скажем, типа `\hspace` или же какие-нибудь из стандартных промежутков, перечисленных в табл. А.2). Обычные пробелы между словами тоже являются растяжимыми (и до определенного предела сжимаемыми), как говорилось на с. 234.

В LaTeX 2ε указанные команды, генерирующие боксы, размер которых задается в качестве аргумента, допускают использование четырех специальных параметров длины: `\width`, `\height`, `\depth` и `\totalheight`. Значениями этих параметров являются естественные размеры текста *text* — соответственно ширина, высота, глубина, а также сумма высоты и глубины.

Несколько полезных советов
Несколько полезных советов
Несколько полезных советов

```
\framebox{Несколько полезных советов}\par
\framebox[\width + 4mm][s]{Несколько
                           полезных советов}
\par  \framebox[1.5\width]{Несколько
                           полезных советов}
```

Боксы нулевой ширины бывают весьма полезны, когда требуется поместить на страницу маркер (например, чтобы пометить место, куда нужно поместить рисунок) или, скажем, когда возникает необходимость делать надписи на полях. Принцип действия соответствующих команд показан ниже. Из приведенных примеров видно, что текст, помещенный в боксы нулевой ширины, никак не влияет

[6] От английского слова stretch.— *Прим. перев.*

на центрирование остального текста. Важно помнить, что при значении позиционирующего параметра [l] приписанный к боксу текст оказывается смещенным вправо, а при значении [r] — влево.

Центрированный текст.[123]
Еще немного центрированного текста.
[321]Центрированный текст.

```
\begin{center}
Центрированный текст.\makebox[0cm][l]{$^{123}$}\\
Еще немного центрированного текста.          \\
\makebox[0cm][r]{$^{321}$}Центрированный текст.\\
\end{center}
```

⇔Маркер, стоящий на поле в этой строке, получен при помощи бокса нулевой ширины. Говоря точнее, набор текущего абзаца начинается с команд

```
\noindent\makebox[0cm][r]{\(\Leftrightarrow\)}%
Маркер, стоящий на полях ...
```

Другое полезное применение боксы нулевой ширины находят при работе с окружением tabular, когда требуется «экранировать» часть текста внутри окружения, например, чтобы иметь возможность записать его со сдвигом относительно строк таблицы (см. в связи с этим сказанное ниже о команде \raisebox).

Следующие два стилевых параметра [$\mathcal{L}$ 195–6] влияют на вид боксов, окаймленных рамкой:

\fboxrule Ширина линий, образующих рамку, создаваемую командами \fbox и \framebox. Значение, принятое по умолчанию в стандартных классах — 0.4 pt.

\fboxsep Расстояние между краями бокса, созданного командами \fbox и \framebox, и размещенным в нем текстом. По умолчанию равно 3 pt во всех стандартных классах.

Текст в рамке | Текст в рамке

```
\fbox{Текст в рамке}
\setlength{\fboxrule}{2pt}
\setlength{\fboxsep}{2mm}\fbox{Текст в рамке}
```

Очень важно, что есть возможность поднимать и опускать боксы. Для этого имеется весьма мощная команда \raisebox [$\mathcal{L}$ 100,195], [$\mathcal{A}$ 248], имеющая два обязательных и два необязательных параметра. Команда имеет следующий вид:

\raisebox{*lift*}[*depth*][*height*]{*contents*}

x111x ^{вверх} x222x $_{\text{вниз}}$ x333x

```
x111x \raisebox{1ex}{вверх} x222x
  \raisebox{-1ex}{вниз} x333x
```

В данном случае LaTeX при определении расстояния между строками учитывает дополнительные параметры — глубину *depth* и высоту *height*. Меняя значения этих параметров, можно управлять расположением соответствующего текста.

Как показывает вторая пара строк в левой колонке приведенного ниже примера, LaTeX не обращает внимания на то, что текст был сдвинут вверх или вниз, и продолжает верстать строки так, как если бы весь текст оставался на базовой линии.

x111x вниз x222x
x333x вверх x444x

x111x вниз x222x
x333x вверх x444x

```
\begin{flushleft}
x111x \raisebox{-1ex}{вниз} x222x      \\
x333x \raisebox{1ex}{вверх} x444x        \\[4mm]
x111x \raisebox{-1ex}[0cm][0cm]{вниз} x222x\\
x333x \raisebox{1ex}[0cm]{вверх} x444x
\end{flushleft}
```

Более важные применения рассмотренных команд были описаны в гл. 5 при изучении техники построения таблиц.

Подобно `\makebox` и `\framebox`, команда `\raisebox` реализована в LaTeX 2_ε таким образом, что в первых трех аргументах могут использоваться величины `\height`, `\depth`, `\totalheight` и `\width`. Поэтому для того, например, чтобы «сделать вид», что высота бокса составляет лишь 90% его действительной высоты, достаточно написать

```
\raisebox{0pt}[0.9\height]{text}
```

а для того, чтобы повернуть бокс относительно его нижнего левого угла (а не обычной точки отсчета, лежащей на базовой линии), можно сперва поднять его на величину глубины `\depth`:

x1 Плохо x2 Хорошо x3 Плохо x4

```
\newcommand{\DoT}[1]%
          {\begin{turn}{-45}#1\end{turn}}
x1 \DoT{\fbox{Плохо}} x2
\DoT{\raisebox{\depth}{\fbox{Хорошо}}}  x3
\DoT{\raisebox{-\height}{\fbox{Плохо}}} x4
```

A.2.2 Боксы-абзацы

Боксы-абзацы создаются при помощи команды `\parbox` или окружения `minipage` [Л 98,99,195], [Л 245–7]. Параметр *width* определяет ширину создаваемого бокса. В бокс помещается текст *text* в виде абзаца. Расположение бокса по вертикали относительно базовой линии определяется необязательным позиционирующим параметром *pos*, который может принимать одно из трех однобуквенных значений: `[c]`, `[t]` или `[b]`.

`\parbox[`*pos*`]{`*width*`}{`*text*`}`	`\begin{minipage}[`*pos*`]{`*width*`}`
	text
	`\end{minipage}`

Как видно из приведенного ниже примера, по умолчанию позиционирующий параметр равен `[c]` и бокс центрируется по вертикали. Кроме того, нетрудно

видеть, что при малой ширине бокса пробелы между словами в строках создаваемого абзаца могут оказаться очень большими.

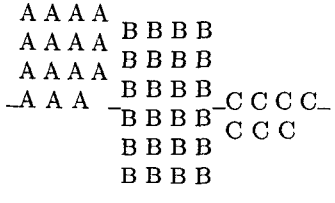
Исходный текст

```
\parbox{.3\linewidth}{Это содержимое самого левого бокса-абзаца.}
\hfill ТЕКУЩАЯ СТРОКА \hfill
\parbox{.3\linewidth}{Это самый правый бокс-абзац. Он наглядно
 демонстрирует тот факт, что когда бокс узкий, \LaTeX\ зачастую не
 может добиться надлежащей заполненности строк, и абзац получается
 разреженным.}
```

Текст на выводе

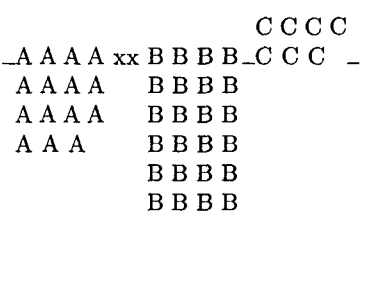
Это содержимое самого левого бокса-абзаца. ТЕКУЩАЯ СТРОКА Это самый правый бокс-абзац. Он наглядно демонстрирует тот факт, что когда бокс узкий, LaTeX зачастую не может добиться надлежащей заполненности строк, и абзац получается разреженным.

Чрезвычайно полезным инструментом макетирования страниц является окружение `minipage`. Оно полностью воссоздает миниатюрный вариант страницы, которая может содержать свои собственные сноски, абзацы, а также окружения `array`, `tabular` и `multicols`. Правда, в окружении `minipage` не допускаются плавающие объекты и команда `\marginpar`, но зато само это окружение можно использовать внутри плавающих объектов, создаваемых окружениями `figure` и `table`, добиваясь весьма интересных эффектов. Ниже приведен совсем простой пример того, на что способно окружение `minipage`. Черточками, создаваемыми командой `\HR`, помечена базовая линия. Данный пример показывает, как работает окружение при трех различных значениях позиционирующего параметра *pos*: `[c]`, `[t]` и `[b]`.



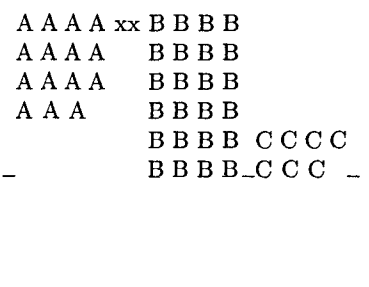

```
\HR
\begin{minipage}[b]{12mm}
                    A A A A A A A A A A A A A A A
\end{minipage}\HR
\begin{minipage}[c]{12mm}
  B B B B B B B B B B B B B B B B B B B B B B B B
\end{minipage}\HR
\begin{minipage}[t]{12mm}  C C C C C C C
\end{minipage}\HR
```

Более сложные варианты выравнивания можно реализовать при вложении различных окружений `minipage` друг в друга. Рассмотрим несколько примеров. Первый пример представляет собой неудачную попытку использовать вложенные окружения `minipage` для того, чтобы выровнять два левых блока по верхнему краю, а потом выровнять по нижнему краю третий блок и блок, составленный из первых двух.



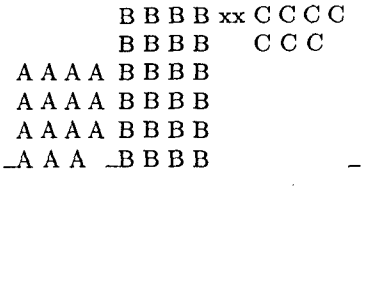

```
\HR
\begin{minipage}[b]{30mm}
 \begin{minipage}[t]{12mm}
                    A A A A A A A A A A A A A A
 \end{minipage} xx
 \begin{minipage}[t]{12mm}
  B B B B B B B B B B B B B B B B B B B B B B B B
 \end{minipage}
\end{minipage}\HR
\begin{minipage}[b]{12mm}  C C C C C C C
\end{minipage}\HR
```

Причина того, что желаемый результат не был получен, состоит в следующем. Два выровненных по верхнему краю блока-окружения `minipage` внутри объемлющего окружения `minipage`, в котором выравнивание производится по нижнему краю, образуют *однострочный* абзац (внутренние блоки-окружения `minipage` рассматриваются LaTeX'ом просто как большие символы в строке, содержащей буквы `xx`). Поэтому нижним (равно как и верхним) краем внешнего окружения `minipage` оказывается все та же строка, содержащая `xx`. Для преодоления этой трудности достаточно вставить после завершения абзаца невидимый вертикальный промежуток.



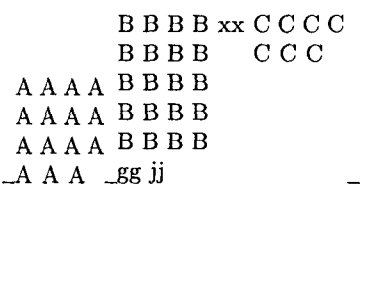

```
\HR
\begin{minipage}[b]{30mm}
 \begin{minipage}[t]{12mm}
                    A A A A A A A A A A A A A A
 \end{minipage} xx
 \begin{minipage}[t]{12mm}
  B B B B B B B B B B B B B B B B B B B B B B B B
 \end{minipage}
 \par\vspace*{0mm}
\end{minipage}\HR
\begin{minipage}[b]{12mm}  C C C C C C C
\end{minipage}\HR
```

В следующем примере два правых блока-окружения выровнены по общему верхнему краю внутри объемлющего окружения, которое выравнивается по нижнему краю вместе с левым блоком-окружением. Сравнение этого примера с предыдущим показывает, что хотя последовательность значений параметров выравнивания в обоих случаях одна и та же, получающиеся результаты совершенно различны из-за того, что по-разному организовано вложение окружений.



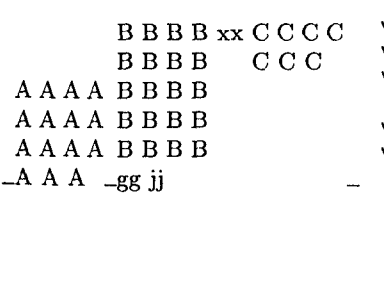
```
\HR
\begin{minipage}[b]{12mm}
                    A A A A A A A A A A A A A A A
\end{minipage}\HR
\begin{minipage}[b]{30mm}
 \begin{minipage}[t]{12mm}
  B B B B B B B B B B B B B B B B B B B B B B B B B
 \end{minipage} xx
 \begin{minipage}[t]{12mm}  C C C C C C C
 \end{minipage}
  \par\vspace*{0mm}
\end{minipage}\HR
```

Здесь для получения требуемого выравнивания снова пришлось добавлять нулевой вертикальный промежуток. Заметим, однако, что этот прием не всегда сразу приводит к нужному результату. Если в последней строке вложенного блока есть буквы с выносным нижним элементом, как в следующем примере, то выравнивание оказывается неточным.



```
\HR
\begin{minipage}[b]{12mm}
     A A A A A A A A A A A A A A A
\end{minipage}\HR
\begin{minipage}[b]{30mm}
  \begin{minipage}[t]{12mm}
    B B B B B B B B B B B B B B B B B B B B B gg jj
  \end{minipage} xx
  \begin{minipage}[t]{12mm}  C C C C C C C
  \end{minipage}
  \par\vspace*{0mm}
\end{minipage}\HR
```

Чтобы добиться точного выравнивания в этой ситуации, нужно вместо нулевого использовать отрицательный вертикальный промежуток, компенсирующий глубину букв последней строки. В следующем примере это достигается путем явного измерения глубины букв при помощи имеющейся в LaTeX 2_ε команды `\settodepth`.



```
\settodepth{\Mylen}{gj}
\HR
\begin{minipage}[b]{12mm}
     A A A A A A A A A A A A A A A
\end{minipage}\HR
\begin{minipage}[b]{30mm}
  \begin{minipage}[t]{12mm}
    B B B B B B B B B B B B B B B B B B B B B gg jj
  \end{minipage} xx
  \begin{minipage}[t]{12mm}  C C C C C C C
  \end{minipage}
  \par\vspace*{-\Mylen}
\end{minipage}\HR
```

Иногда может потребоваться создание боксов-абзацев заданной высоты. Для этих случаев LaTeX 2ε предусматривает использование окружения `minipage` и команды `\parbox` с дополнительными необязательными аргументами.

```
\parbox[pos][height][inner-pos]{width}{text}
\begin{minipage}[pos][height][inner-pos]{width} text \end{minipage}
```

Аргумент *inner-pos* задает положение текста *text* внутри бокса. Он может принимать значения `t`, `c`, `b` или `s`. Если этот аргумент не задан явно, то по умолчанию ему присваивается то значение, какое имеет аргумент *pos*. Можно достаточно точно охарактеризовать действие параметров *height* и *inner-pos*, сказав, что они являются «вертикальными» аналогами «горизонтальных» параметров *width* и *pos* команды `\makebox`. Если в качестве *inner-pos* задано `s`, то текст *text* будет растянут по вертикали на всю заданную высоту бокса *height*. В этом случае, чтобы избежать сообщений о слабой заполненности, от пользователя может потребоваться ввести в бокс растяжимые вертикальные промежутки, используя, например, команду `\vspace`.

Как и в случае других команд, создающих блоки, значение необязательного аргумента *height* можно формировать, используя естественные размеры текста *text* (величины `\height`, `\totalheight` и т. д.).

xx Текст в верхней части бокса.

И несколько строк в нижней его части. Вот несколько строк в верхней части бокса и только одна строка

внизу. xx

```
xx \fbox{\parbox[b]%
      [\height+\baselineskip][s]
      {20mm}{Текст в верхней части бокса.
            \par\vfill
            И несколько строк в нижней
            его части.}}
   \fbox{\parbox[b]%
      [\height+\baselineskip][s]
      {20mm}{Вот несколько строк
            в верхней части бокса и
            только одна строка
            \par\vfill внизу.}} xx
```

A.2.3 Боксы-линейки

Боксы-линейки создаются при помощи команды `\rule` [Л 100,195], [Л 137], имеющей следующий вид:

```
\rule[lift]{width}{total_height}
```

Написав `\rule{2cm}{2mm}`, мы получим линейку ▬ шириной 2cm и толщиной 2mm. Команду `\rule` можно использовать и для создания боксов-линеек нулевой ширины, или невидимых линеек, называемых еще «распорками» (*strut*). Невидимые линейки помогают регулировать высоту или ширину боксов. Например, с их помощью можно увеличивать высоту боксов с рамками, поро-

ждаемых командами \fbox и \framebox, а также подправлять расстояние между некоторыми строками в таблице. Как это делается, видно из следующего примера:

Исходный текст

```
x111x
 \fbox{некий текст}
x222x
 \fbox{\rule[-5mm]{0cm}{15mm}другой текст}
x333x
```

Текст на выводе

x111x некий текст x222x другой текст x333x

А.2.4 Работа с боксами

Набранный фрагмент документа можно сохранить в специально созданном боксе, присвоив этому боксу некоторое имя. Сохраненный фрагмент можно будет в дальнейшем считать, когда это потребуется [$\mathcal{L}$ **101,194**], [$\mathcal{A}$ **265–7**].

\newsavebox{*cmd*}	декларация бокса
\sbox{*cmd*}{*text*}	заполнение бокса
\savebox{*cmd*}[*width*][*pos*]{*text*}	заполнение бокса
\usebox{*cmd*}	считывание содержимого бокса

Команды \sbox и \savebox аналогичны командам \mbox и \makebox. Глобальная команда \newsavebox вводит команду *cmd* с именем бокса, скажем, \boxname. Созданный бокс с указанным именем является накопителем, в котором будет сохранен текст *txt* для того, чтобы потом его считывать всякий раз, когда возникнет такая необходимость. Не следует думать, однако, что команду \boxname можно использовать саму по себе. Увы, эта команда всего лишь выдает TEX'овский номер данного бокса. Другими словами, будучи непосредственно вставлена в документ, команда \boxname просто напечатает в текущем шрифте символ, стоящий в таблице кодов символов шрифта на позиции, соответствующей номеру бокса. Поэтому при работе с боксами нужно пользоваться исключительно командами, рекомендуемыми в настоящем разделе. Команда \usebox позволяет использовать содержимое бокса \boxname, причем это содержимое остается в боксе в целости и сохранности. К одному и тому же накопителю \boxname можно многократно обращаться в пределах текущего окружения или группы, ограниченной фигур-

ными скобками — каждый раз из него будет переноситься в документ то, что было в него помещено при последнем запоминании.

Исходный текст

```
\newsavebox{\myboxa}\newsavebox{\myboxb}
\sbox{\myboxa}{текст из бокса a}
\savebox{\myboxb}[3cm][l]{текст из бокса b}
  x1x \usebox{\myboxa}  x2x \usebox{\myboxb} x3x
\savebox{\myboxb}[3cm][r]{текст из бокса b}  \par
  x1x \usebox{\myboxa}  x2x \usebox{\myboxb} x3x
```

Текст на выводе

x1x текст из бокса a x2x текст из бокса b x3x
x1x текст из бокса a x2x текст из бокса b x3x

Помимо рассмотренных команд в LaTeX 2_ε имеется окружение `lrbox`, синтаксис которого таков:

```
\begin{lrbox}{cmd}  text  \end{lrbox}
```

Здесь *cmd* означает боксовый регистр, ранее декларированный с помощью команды `\newsavebox`. Окружение `lrbox` сохраняет в этом боксе текст *text* с возможностью его последующего использования при помощи команды `\usebox`. При этом пробелы в начале и в конце текста игнорируются. По существу, окружение `lrbox` является аналогом команды `\sbox`. Данное окружение удобно использовать, когда требуется сохранить содержимое некоторого другого окружения в боксе для последующей обработки. Рассмотрим, например, следующее определение окружения `fminipage`, которое работает так же, как `minipage`, но окаймляет помещенный в него текст рамкой.

```
\newsavebox{\fminibox}
\newlength{\fminilength}
\newenvironment{fminipage}[1][\linewidth]
  {\setlength{\fminilength}{#1-2\fboxsep-2\fboxrule}%
   \begin{lrbox}{\fminibox}\begin{minipage}{\fminilength}}
  {\end{minipage}\end{lrbox}\noindent\fbox{\usebox{\fminibox}}}
```

Приведенное определение интересно в нескольких отношениях. Введенное окружение имеет один необязательный аргумент — ширину определяемого им бокса (по умолчанию она равна `\linewidth`). В первой строке описания процедуры открытия вычисляется (при помощи пакета calc) внутренняя длина строки — параметр, который надлежит передать окружению `minipage`. При этом приходится вычитать из `\linewidth` суммарную длину промежутков, которые затем будут добавлены командой `\fbox` на левой и правой гра-

ницах. Затем открываются окружения lrbox и minipage, внутри которых текст, предназначенный для записи в окружение fminipage, заверстывается в бокс \fminibox в виде абзаца с длиной строки \fminilength. По завершении записи текста оба указанных окружения закрываются. Наконец, содержимое \fminibox переносится в бокс, создаваемый командой \fbox. Стоящая перед ней команда \noindent подавляет нежелательный абзацный отступ, которые может появиться в случае, если текст, помещаемый в окружение fminipage, является началом некоторого абзаца или целым абзацем.

Внутри этого окружения допускается буквальное воспроизведение текста (в стиле verbatim), например, вот это: \fminibox.

```
\begin{fminipage}
  Внутри этого окружения допускается
  буквальное воспроизведение
  текста (в стиле verbatim), например,
  вот это: \verb=\fminibox=.
\end{fminipage}
```

A.3 Структура пакетов и классов в LaTeX 2ε

Как было отмечено в разд. 2.1, при создании LaTeX 2ε структура файлов, представляющих пакеты и классы, а также техника работы с ними подверглись значительному усовершенствованию. В настоящем разделе рассматриваются команды, позволяющие разработчикам пакетов и классов использовать новые возможности, появившиеся в LaTeX 2ε.

Материал раздела будет полезен и тем, кто пока не планирует создавать свои собственные пакеты, поскольку позволяет лучше разобраться в структуре и содержании пакетов и классов, таких, например, как book или varioref, а следовательно, дает возможность научиться более эффективному их использованию.

Общая структура файлов, представляющих классы и пакеты, одинакова и состоит из следующих элементов:

⟨*идентификация*⟩
⟨*начальный командный код*⟩
⟨*декларация опций*⟩
⟨*выполнение опций*⟩
⟨*загрузка пакетов*⟩
⟨*основной командный код*⟩

Любой из перечисленных элементов является необязательным. Ниже рассматриваются команды, относящиеся к каждому элементу в отдельности. Краткая сводка команд содержится в табл. А.4.

Идентификация
`\NeedsTeXFormat{`*format*`}[`*release*`]` Сообщается, что для пакета нужен формат *format* (`LaTeX2e`) версии *release* или более поздней
`\ProvidesClass{`*name*`}[`*release info*`]` `\ProvidesPackage{`*name*`}[`*release info*`]` Сообщается имя (*name*) пакета или класса и приводятся сведения *release info*
`\ProvidesFile{`*name*`}[`*release info*`]` Сообщается имя *name* (с расширением) дополнительно используемого файла и приводятся *release info*
Декларация опций
`\DeclareOption{`*option*`}{`*code*`}` Декларируется командный код *code*, который должен быть исполнен, если выбрана данная опция *option*
`\PassOptionsToPackage{`*option-list*`}{`*package-name*`}` Список опций *option-list* передается пакету *package-name*
`\DeclareOption*{`*code*`}` Декларируется командный код *code*, который должен быть исполнен, если указана любая неизвестная опция *option*
`\CurrentOption` Предлагается в `\DeclareOption*` использовать текущую опцию
Выполнение опций
`\ExecuteOptions{`*option-list*`}` Для каждой опции, перечисленной в *option-list*, исполняется ее командный код
`\ProcessOptions` `\ProcessOptions*` Обрабатываются указанные опции для текущего класса или пакета; команда со звездочкой производит эту обработку в порядке перечисления опций
Загрузка пакетов
`\RequirePackage[`*option-list*`]{`*package*`}[`*release*`]` Загружается пакет *package* с заданным списком опций *option-list* и датой выпуска не ранее *release*
Команды, специально предназначенные для пакетов и классов
`\AtEndOfPackage{`*code*`}` `\AtEndOfClass{`*code*`}` Задается командный код *code*, который предписывается исполнить при завершении работы текущего пакета или класса
`\AtBeginDocument{`*code*`}` `\AtEndDocument{`*code*`}` Задается командный код *code*, исполнение которого инициируется соответственно командами `\begin{document}` и `\end{document}`
`\IfFileExists{`*file*`}{`*then-code*`}{`*else-code*`}` В случае существования файла *file* исполняется командный код *then-code*, в противном случае — код *else-code*
окончание см. на след. стр.

Таблица А.4. Команды, описывающие структуру пакетов и классов

окончание (см. предыдущую страницу)
`\InputIfFileExists{`*file*`}{`*then-code*`}{`*else-code*`}` В случае существования файла *file* исполняется командный код *then-code* и файл подгружается, в противном случае исполняется код *else-code*
Команды, специально предназначенные для классов
`\LoadClass[`*option-list*`]{`*class*`}[`*release*`]` Команда работает так же, как `\RequirePackage` для классов, но не воспринимает глобальные опции, если они не указываются ей явным образом
`\PassOptionsToClass{`*option-list*`}{`*class*`}` Список опций *option-list* передается классу *class*
`\OptionNotUsed` Для использования при необходимости в команде `\DeclareOption*`

Таблица А.4.

А.3.1 Команды идентификации

Эта часть файла, представляющего класс или пакет, указывает, во-первых, его собственные выходные данные и, во-вторых, ту версию макропакета LaTeX 2ε, для которого был написан файл.

`\ProvidesClass{`*name*`}[`*release information*`]`

Идентификацию файла-класса производит команда `\ProvidesClass`. Ее аргумент *name* соответствует имени класса в том виде, как оно указывается в качестве обязательного аргумента команды `\documentclass` (т. е. в виде имени файла без расширения). Необязательный аргумент *release information* должен начинаться с даты в формате YYYY/MM/DD, за которой должен следовать текст с указанием версии. Например, класс report обязательно содержит строчку наподобие вот такой:

```
\ProvidesClass{report}[1994/01/01 LaTeX2e standard class]
```

Информацию *release information* можно использовать в документе, указав соответствующую дату в качестве второго необязательного аргумента команды `\documentclass`:

```
\documentclass[twocolumn]{report}[1994/06/01]
```

Это позволяет LaTeX'у следить за тем, чтобы используемая версия класса report была не старше версии от 1994/06/01. Если файл, представляющий данный класс, оказывается более старым, генерируется соответствующее предупреждение. Таким образом, если при создании документа использовалась относительно новая версия класса, а потом документ был передан в организацию, где все еще используется более старая версия, там поймут, что их дистрибутив LaTeX'а устарел.

`\ProvidesPackage{`*name*`}[`*release information*`]`

Эта команда идентифицирует пакеты. Она имеет ту же структуру, что и команда `\ProvidesClass`. Дата в аргументе *release information* точно так же может быть использована в качестве второго необязательного аргумента команды `\usepackage` для того, чтобы предупредить пользователя, если тот использует устаревшую версию пакета. Соответствующая команда может выглядеть, например, вот так:

```
\usepackage[german]{varioref}[1994/01/01]
```

`\ProvidesFile{`*filename*`}[`*release information*`]`

Эта команды служит для идентификации любых типов файлов. По этой причине аргумент *filename* должен содержать полное имя файла, включая расширение.

Помимо одной из перечисленных выше команд, пакеты и классы обычно содержат идентифицирующую команду `\NeedsTeXFormat`, уже упоминавшуюся в разд. 2.1.1, которая может выглядеть, например, так: `\NeedsTeXFormat{LaTeX2e}[1993/11/11]`.

Все четыре декларации являются необязательными. Тем не менее, использование их в дистрибутивах классов и пакетов упрощает поддержку LaTeX'а в целом.

A.3.2 Начальный командный код

В качестве ⟨*начального командного кода*⟩ можно указать любой набор корректно написанных LaTeX'овских команд. Здесь, например, может содержаться загрузка тех или иных макропакетов при помощи команды `\RequirePackage` (см. разд. A.3.5), если к этим макропакетам происходит обращение при последующих декларациях опций. В частности, можно именно здесь загрузить пакет calc, чтобы иметь возможность использовать его в последующих фрагментах файла. Тем не менее, обычно эта часть файла является пустой.

A.3.3 Декларация опций

Все опции, известные пакету или классу, декларируются здесь при помощи команды `\DeclareOption`. В этой части файла запрещено загружать макропакеты.

`\DeclareOption{`*option*`}{`*code*`}`

Аргумент *option* — это имя декларируемой опции, а аргумент *code* — это командный код, который должен быть исполнен, когда данная опция оказывается востребованной. Например, стилевая опция a4paper, задающая размер бумаги (лист формата A4), обычно определяется следующим образом:

```
\DeclareOption{a4paper}{\setlength{\paperheight}{297mm}%
                        \setlength{\paperwidth}{210mm}}
```

В принципе, в аргументе *code* команды \DeclareOption можно задавать любые действия, от простой установки флажка до весьма сложных инструкций по набору текста.

Важным частным случаем кода, используемого в команде \DeclareOption, является команда \PassOptionsToPackage. Она обеспечивает передачу одной или нескольких опций другому пакету, который будет загружен позже.

\PassOptionsToPackage{*option-list*}{*package-name*}

Аргумент *option-list* представляет собой список опций, отделенных друг от друга запятой, которые должны быть переданы пакету с именем *package-name*, когда тот будет загружен (команда загрузки находится в разделе ⟨*package loading*⟩ нашего файла.[7] Предположим, например, что требуется определить класс, использующий два макропакета A и B, причем в обоих макропакетах поддерживается опция infoshow. Чтобы обеспечить поддержку данной опции также и в определяемом классе, нужно написать следующее:

```
\DeclareOption{infoshow}{%
    \PassOptionsToPackage{infoshow}{A}%
    \PassOptionsToPackage{infoshow}{B}%
    ⟨code to support infoshow in the class⟩}
```

Если макропакет или класс загружается с опцией, которую он не распознает, то либо генерируется предупреждение (в случае макропакета), либо опция молчаливо игнорируется (в случае класса). При этом предполагается, что данная опция является глобальной и предназначена для передачи другим пакетам, которые будут загружены командой \usepackage в дальнейшем. Такая реакция, однако, не является «зашитой намертво» и может быть изменена при помощи декларации \DeclareOption*.

\DeclareOption*{*code*}

Аргумент *code* данной команды определяет те действия, которые должны быть выполнены, если задана неизвестная опция. Внутри аргумента запись \CurrentOption имеет смысл имени той самой неизвестной опции. Так, например, чтобы создать макропакет с не меньшими функциональными возможностями, чем у некоторого другого макропакета, достаточно написать:

```
\DeclareOption*{\PassOptionsToPackage{\CurrentOption}{A}}
```

При этом все опции, не декларированные в имеющемся макропакете, будут передаваться макропакету A. Если декларация \DeclareOption* не используется,

[7] О том, чтобы загрузка пакетов на самом деле имела место, должен позаботиться автор пакета. LaTeX не проверяет, действительно ли пакеты, которым с помощью команды \PassOptionsToPackage передаются опции, впоследствии загружаются.

будут выполняться действия, заданные по умолчанию, о которых было сказано выше.

Комбинируя \DeclareOption* с \InputIfFileExists (см. ниже), можно даже реализовать условную обработку опций. Так, например, следующая команда будет пытаться найти файлы, имена которых образованы из имени опции по указанному в команде правилу.

```
\DeclareOption*{\InputIfFileExists{g-\CurrentOption.xyz}{}{}}
```

Если файл g-*option*.xyz существует, он будет загружен; в противном случае опция будет проигнорирована.

A.3.4 Исполнение опций

После декларирования всех опций производятся действия двух типов. Прежде всего, некоторые опции (например, размер бумажного листа) можно выбрать для использования по умолчанию. После этого нужно будет исполнить командные коды всех опций, оказавшихся в списке выбранных.

\ExecuteOptions{*option-list*}

Команда \ExecuteOptions исполняет командные коды всех опций, перечисленных в списке *option-list*, причем делает это в порядке перечисления. Установка умолчаний путем исполнения кода, указанного ранее в виде аргумента команды \DeclareOption, представляет собой удобный и наглядный способ организации этого процесса, аналогичный нажатию на нужные клавиши. Так, например, чтобы установить умолчания для стандартного класса book, достаточно написать

```
\ExecuteOptions{letterpaper,twoside,10pt}
```

или что-то в этом роде. Команду \ExecuteOptions можно использовать и при декларировании опций. С ее помощью можно, например, создавать опции, представляющие собой комбинации некоторых других опций. Следует помнить, что команда \ExecuteOptions обязательно должна исполняться раньше, чем команда \ProcessOptions. Дело в том, что последняя на одном из завершающих этапов своей работы «забирает» всю память, отведенную для хранения командного кода декларированных опций.

\ProcessOptions

Команда \ProcessOptions просматривает список опций, заданных классом или макропакетом, и исполняет соответствующий командный код. Говоря более точно, при работе с макропакетом просматриваются глобальные опции (заданные командой \documentclass) и опции, заданные непосредственно (как необязательный аргумент одной из команд \usepackage или \RequirePackage). Для каждой

опции, декларированной макропакетом, исполняется соответствующий командный код, причем опции обрабатываются в той последовательности, в какой они были декларированы командами \DeclareOption, а не в той, в какой они были перечислены в команде \usepackage. Нераспознанные глобальные опции игнорируются. Для всех остальных нераспознанных опций исполняется код, заданный декларацией \DeclareOption*, либо, если такой декларации нет, генерируется сообщение об ошибке.

В случае класса команда \ProcessOptions выполняет те же действия в отношении глобальных опций.

\ProcessOptions*

Для некоторых пакетов предпочтительной является обработка опций в том порядке, в котором они перечислены в команде \usepackage, а не в порядке следования команд \DeclareOption. Примером может служить пакет babel, где последняя из языковых опций должна определять основной язык документа. Для обработки опций в заданном порядке следует вместо \ProcessOptions использовать команду \ProcessOptions*.

A.3.5 Загрузка макропакетов

Когда с обработкой опций, наконец, покончено, наступает время загрузки дополнительных макропакетов, в том числе тех, которым при помощи команды \PassOptionsToPackage были переданы некоторые опции.

\RequirePackage[*option-list*]{*package*}[*release*]

Эта команда, предназначенная для использования в макропакетах и классах, является аналогом команды \usepackage, используемой в теле документа. Если макропакет *package* не был загружен раньше, он будет загружен теперь, причем с опциями, указанными в аргументе *option-list*, с глобальными опциями, указанными в команде \documentclass, а также со всеми теми опциями, которые передаются загружаемому макропакету при помощи команд \PassOptionsToPackage.

LaTeX 2ε загружает любой макропакет только один раз, поскольку во многих случаях повторное исполнение командного кода макропакета небезопасно. В связи с этим могут возникать определенные сложности. Допустим, что макропакет A требует загрузки некоторого другого макропакета B с определенным набором опций, а макропакет B ранее уже был загружен с другим набором опций. В этом случае пользователь макропакета A будет проинформирован LaTeX'ом о возникшей проблеме, причем ему будет предложено загрузить макропакет B с нужными опциями, пользуясь командой \usepackage.

Необязательный аргумент *release* позволяет следить за тем, чтобы не использовались устаревшие версии загружаемых макропакетов. Чтобы этот механизм работал, загружаемый макропакет должен содержать декларацию \ProvidesPackage с указанием даты выпуска.

A.3.6 Основной командный код

Команды, содержащиеся в этой, заключительной части файла определяют основные свойства данного класса или макропакета и реализуют его функциональные возможности. При этом допускается использование любых LaTeX'овских конструкций. Обычно здесь вводятся новые переменные и новые команды. Считается хорошим стилем вводить определения в стандартной LaTeX'овской форме, используя команды `\newlength`, `\newcommand` и т. п. (см. первый раздел настоящего приложения), а не прибегать к помощи TeX'овских примитивов, поскольку последние не обеспечивают контроля за тем, чтобы различные макропакеты не конфликтовали друг с другом.

A.3.7 Команды, специально предназначенные для макропакетов и классов

`\AtEndOfPackage{`*code*`}` `\AtEndOfClass{`*code*`}`

Иногда бывает нужно отложить исполнение некоторого командного кода до завершения чтения и обработки текущего файла, представляющего макропакет или класс. Приведенные выше декларации сохраняют свой аргумент *code* и инициируют его исполнение по достижении конца файла. Если в файле имеется несколько таких деклараций, то фрагменты кода, заданные в различных декларациях, запоминаются и впоследствии исполняются в той же последовательности, в какой в файле записаны декларации.

`\AtBeginDocument{`*code*`}` `\AtEndDocument{`*code*`}`

Другими важными моментами, на которые иногда требуется перенести исполнение командного кода, являются начало и конец чтения документа, точнее, моменты обработки команд `\begin{document}` и `\end{document}`. Приведенные выше команды позволяют макропакету добавлять свой командный код к окружению `document`, не вступая в конфликт с другими макропакетами, пытающимися сделать то же самое.

`\IfFileExists{`*file*`}{`*then-code*`}{`*else-code*`}`
`\InputIfFileExists{`*file*`}{`*then-code*`}{`*else-code*`}`

Когда данный макропакет или класс пытается при помощи команды `\input` загрузить несуществующий файл, пользователю выдается циклически повторяющееся[8] сообщение об ошибке в имени файла. Единственной возможностью[9] выйти

[8] Цикл возникает, если реакцией пользователя является нажатие клавиши <Enter>, что дает указание (LA)TeX'у проигнорировать допущенную ошибку. — *Прим. перев.*

[9] В данной ситуации можно прервать цикл, а вместе с ним и весь процесс компиляции, например, нажав <CTRL>-Z. — *Прим. перев.*

из цикла является указание имени какого-нибудь действительно существующего файла. Использование в макропакете или классе команды `\IfFileExists` позволяет избежать возникновения подобных ситуаций. Аргумент *file* содержит имя файла, существование которого должно проверяться. Если LaTeX обнаруживает файл *file*, исполняются команды, указанные в аргументе *then-code*; в противном случае исполняются команды аргумента *else-code*. Команда `\InputIfFileExists` не только проверяет, существует ли файл *file*, но еще и загружает его сразу же после исполнения кода *then-code*. Кроме того, имя файла *file* добавляется в список используемых файлов, который генерирует команда `\listfiles`.

A.3.8 Команды, специально предназначенные для классов

Новые классы во многих случаях создаются путем модификации существующих стандартных классов. Для реализации этого подхода предусмотрены две специальные команды.

`\LoadClass[`*option-list*`]{`*class*`}[`*release*`]`

Команда `\LoadClass` работает так же, как `\RequirePackage`, с тремя следующими отличиями:

- Данная команда применима только к классам.
- В любом классе не должно быть больше одной команды `\LoadClass`.
- Загружаемый класс *class* не воспринимает глобальные опции иначе как путем явной передачи их ему командой `\PassOptionsToClass` или посредством указания их в списке *option-list*.

`\PassOptionsToClass{`*option-list*`}{`*class*`}`

Команда `\PassOptionsToClass` используется для передачи опций загружаемому классу. Пример нового класса, созданного путем расширения уже существующего, приведен на рис. А.2. Новый класс `myart` определяется путем введения в класс `article` двух дополнительных опций, cropmarks и bind, и одного дополнительного окружения `Notes`. Опция cropmarks предназначена для создания на странице ограничивающих меток, используемых при обрезке, а опция bind — для небольшого увеличения внутреннего поля за счет дополнительного сдвига полосы к внешнему краю в том случае, когда предполагается сшивание (или склейка) страниц.

Для реализации опции cropmarks вводится булев переключатель и, когда он принимает значение `true`, используемый стиль страницы (`\pagestyle`) соответствующим образом переопределяется. Опция bind модифицирует величины `\oddsidemargin` и `\evensidemargin`. Поскольку к моменту, когда в файле появляется опция bind, указанным величинам еще не присвоены надлежащие значения

```
% ------------------------- идентификация --------------------------
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{myart}[1994/01/01]
% -------------------------  начальный код --------------------------
\RequirePackage{ifthen}
\newboolean{cropmarks}
% ------------------------  декларация опций -------
\DeclareOption{cropmarks}{\setboolean{cropmarks}{true}}
\DeclareOption{bind}
    {\AtEndOfClass{\addtolength\oddsidemargin{.5in}%
                   \addtolength\evensidemargin{-.5in}}%
                  }
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{article}}
% ------------------------  исполнение опций ------------------------
\ProcessOptions
% ------------------------ загрузка пакетов ------------------------
\LoadClass{article}                    % стандартный класс
% -------------------------- основной код --------------------------
\newenvironment{Notes}{...}{...}       % новое окружение
\ifthenelse{\boolean{cropmarks}}       % поддержка ограничивающих меток
   {%
    \renewcommand{\ps@plain}{....}%
    ....
   }{}
```

Рис. А.2. Пример файла-класса, являющегося расширением класса article

(это происходит позже, при загрузке класса article командой \LoadClass), их модификация откладывается при помощи команды \AtEndOfClass на конец исполнения файла-класса myart.

\OptionNotUsed

В принципе, командный код *code*, заданный как аргумент команды \DeclareOption* в файле-классе, может быть устроен более сложным образом, чем в приведенных выше примерах. Например, он может часть опций обрабатывать, а остальные отвергать. В этом случае необходимо, используя команду \OptionNotUsed, поставить в известность LaTeX 2_ε, что некая опция не принята. Если этого не сделать, LaTeX 2_ε будет считать, что опция принята, и не станет генерировать предупреждение, если в дальнейшем окажется, что опция не воспринимается загружаемыми макропакетами.

A.4 calc — макропакет для арифметических вычислений

Макропакет calc, разработанный Крестеном Торапом и Франком Йенсеном, содержит расширенный набор макрокоманд для арифметических вычислений в LaTeX'е. Обычная TeX'овская арифметика основана на простых низкоуровневых командах типа `\advance` и `\multiply`. Данный пакет позволяет производить в LaTeX'е арифметические операции, используя инфиксную запись.[10] Для этого в нем переопределяются команды `\setcounter`, `\addtocounter`, `\setlength` и `\addtolength` [*Л* **175,193**] [*Л* **220–1,234–5**], причем так, что в новой версии эти команды воспринимают не только обычные целые числа и длины, но и составленные из них выражения.

Целочисленное выражение может содержать целые числа, TeX'овские целочисленные регистры, LaTeX'овские счетчики (например, вида `\value{ctr}`), скобки и символы бинарных операций `-`, `+`, `*`, `/`. Вот, например, как можно увеличить значение счетчика:

Значение стало равно 3.

```
\newcounter{local}\setcounter{local}{2}
\setcounter{local}{\value{local}+1}
Значение стало равно \thelocal.
```

А вот команда, печатающая показания часов в обычном формате (напомним, что TeX'овский регистр `\time` содержит число минут, прошедших после полуночи):

Московское время — 12 час. 2 мин.

```
\newcounter{hours}\newcounter{minutes}
\newcommand{\printtime}{%
 \setcounter{hours}{\time/60}%
 \setcounter{minutes}%
     {\time-\value{hours}*60}%
 \thehours~час. \theminutes~мин.}
Московское время "--- \printtime
```

При работе с длинами складываемые или вычитаемые выражения должны быть однотипными, хотя и не обязаны иметь одну и ту же размерность. Например, нельзя написать «`2cm+4`», но выражение «`2cm+4pt`» допустимо, поскольку оба слагаемых — расстояния (оба имеют размерность длины). Делить или умножать можно только на целые числа, поэтому писать «`2cm*4`» можно, а «`2cm*4pt`» — нельзя. Кроме того, в выражениях подобного типа величины, имеющие размерность, должны стоять на первом месте, так что запись «`4*2cm`» не допускается.

При помощи перечисленных выше команд можно вычислять ширину отдельной колонки при наборе в n-колонок. Искомую ширину определяет следующее

[10] Под инфиксной записью или системой обозначений понимается наиболее привычная форма записи операций, когда символ бинарной операции располагается между операндами. Наряду с инфиксной существуют префиксная и постфиксная записи, где знаки операций выносятся соответственно в начало или конец записи. В языках программирования выбор формы записи обычно связан с тем, как организованы вычисления.— *Прим. перев.*

выражение (предполагается, что значение n является первым аргументом некоторого LaTeX'овского макро):

```
\setlength{\linewidth}{(\textwidth-\columnsep*(#1-1))/#1}
```

Если вычисления производятся над расстояниями (т. е. над величинами, имеющими размерность длины), то умножать и делить можно не только на целые, но и на любые действительные числа. При этом можно использовать одну из следующих двух форм представления действительных чисел:

`\real{`*decimal constant*`}` `\ratio{`*length expression*`}{`*length expression*`}`

В первом случае действительное число просто записывается в десятичной форме, а команда преобразует текстовую запись в числовое значение. Во втором случае безразмерное действительное число вычисляется как отношение двух величин, имеющих линейную размерность.

Пусть, например, требуется промасштабировать рисунок так, чтобы его ширина оказалась равной ширине страницы, т. е. величине `\textwidth`. Если исходные размеры рисунка равны значениям переменных `\Xsize` и `\Ysize`, то высота промасштабированного рисунка определится следующим образом:

```
\setlength{\newYsize}{\Ysize*\ratio{\textwidth}{\Xsize}}
```

В этой книге пакет calc используется во многих примерах. В принципе можно было бы и не применять данный пакет, переписав соответствующие фрагменты кода в терминах (LA)TeX'овских низкоуровневых команд. Так, например, при определении величины `\fminilength` на с. 517 следовало бы написать не

```
\setlength{\fminilength}{#1-2\fboxsep-2\fboxrule}%
```

а

```
\setlength{\fminilength}{#1}%
\addtolength{\fminilength}{-2\fboxsep}%
\addtolength{\fminilength}{-2\fboxrule}
```

Однако помимо того, что командный код в инфиксной записи, используемый макропакетом calc, легче читается и гораздо легче модифицируется, чем обычные LaTeX'овские команды, некоторые конструкции, используемые для деления и умножения в этой записи, вообще нельзя реализовать при помощи стандартных LaTeX'овских средств. Например, для того чтобы реализовать вычисление величины `\fminilength`, определенной на с. 517, требуется следующая последовательность команд:

```
\setlength{\topmargin}{297mm}
\addtolength{\topmargin}{-\textheight}
\divide\topmargin by 3                  % Это  TeX'овская команда!
\addtolength{\topmargin}{-1in}
\addtolength{\topmargin}{-\headheight}
\addtolength{\topmargin}{-\headsep}
```

A.5 ifthen — усовершенствованный условный переход

Иногда требуется, чтобы в документ включался тот или иной фрагмент в зависимости от значения некоторого логического выражения. Необходимые для этого управляющие конструкции LaTeX'а реализованы в стандартном макропакете ifthen, разработанном Лесли Лэмпортом и модифицированном для LaTeX 2ε Дэвидом Карлайлом.

`\ifthenelse{`*test*`}{`*then-code*`}{`*else-code*`}`

Если условие *test* истинно, выполняется командный код *then-code*, в противном случае — набор команд *else-code*.

Простейшим типом условия является сравнение двух целых чисел. Рассмотрим следующий пример. Допустим, что требуется сгенерировать английские порядковые числительные, соответствующие значениям счетчика глав `chapter` в пределах от 1 до 20. Эта задача решается так:

This is the 1st appendix.

```
\newcommand{\toEng}[1]{%
  \the\value{#1}%
  \ifthenelse{\value{#1} = 1}{$^{\hbox{st}}$}{}%
  \ifthenelse{\value{#1} = 2}{$^{\hbox{nd}}$}{}%
  \ifthenelse{\value{#1} = 3}{$^{\hbox{rd}}$}{}%
  \ifthenelse{\value{#1} > 3}{$^{\hbox{th}}$}{}}

This is the \toEng{chapter} appendix.
```

В качестве следующего примера определим команду для записи текущих показаний часов в краткой форме. В данном случае операторы условного перехода используют величины, вычисляемые при помощи пакета calc.

На часах — 12:02.

```
\newcommand{\Printtime}{\setcounter{hours}{\time/60}%
      \setcounter{minutes}{\time-\value{hours}*60}%
      \ifthenelse{\value{hours}<10}{0}{}\thehours:%
      \ifthenelse{\value{minutes}<10}{0}{}\theminutes}
На часах "--- \Printtime.
```

`\equal{`*string1*`}{`*string2*`}`

Команда `\equal` сравнивает цепочки литер *string1* и *string2* после их полного раскрытия (развертывания) и выдает значение *true* (*истина*), если цепочки оказываются тождественными. Хрупкие команды, используемые в цепочках, обязательно должны быть защищены командой `\protect`.

False.
True.
True.

```
\newcommand{\BB}{\CC}\newcommand{\CC}{\DD}
\newcommand{\DD}{AA}\newcommand{\EE}{EE}
\ifthenelse{\equal{\BB}{\EE}}{True}{False}.\\
\ifthenelse{\equal{\BB}{\CC}}{True}{False}.\\
\ifthenelse{\equal{\DD}{\BB}}{True}{False}.
```

При помощи операторов условного перехода можно определить команду, которая вписывает в основной текст документа заданный термин и одновременно

вносит его в предметный указатель. При этом ссылка в указателе на определение данного термина оформляется полужирным шрифтом, а другие ссылки на этот термин — обычным светлым шрифтом. В качестве значения по умолчанию для необязательного аргумента команды выбран чаще встречающийся случай простой ссылки, а не более редкий случай ссылки на определение.

Условимся говорить, что имеет место **AAAA** ... Если имеет место **AAAA**

```
\newcommand{\IX}[2][R]{\texttt{#2}%
  \ifthenelse{\equal{#1}{D}}%
     {\index{#2|textbf}}{\index{#2}}}
 Условимся говорить, что имеет место \IX[D]{AAAA}
 \ldots{}     Если имеет место \IX{AAAA}
```

В результате исполнения приведенных выше команд в .idx-файле появляются записи, обеспечивающие требуемое шрифтовое оформление ссылок:

```
\indexentry{AAAA|textbf}{530}    \indexentry{AAAA}{530}
```

Более сложным является следующий пример, в котором определена обобщенная команда создания элементов указателя .idx, решающая вопрос о том, вносить или не вносить данный фрагмент в предметный указатель, а также в основной текст.

```
\newcommand{\IXE}[2][!*!,!]{%
 \ifthenelse{\equal{#1}{!*!,!}}{%
      \ifthenelse{\equal{#2}{}}{}{\textbf{#2}\index{#2}}}%
   {\ifthenelse{\equal{#1}{}}{}{\index{#1}}%
    \ifthenelse{\equal{#2}{}}{}{\textbf{#2}}}}
```

Необязательный аргумент команды \IXE по умолчанию определен как последовательность символов «!*!,!», которая, по всей вероятности, не встречается в документе. При использовании команды только с одним (обязательным) аргументом и в указатель, и в основной текст будет вноситься одно и то же. Если же необязательный аргумент задан, появляется возможность вносить в указатель нечто отличное от того, что стоит в соответствующем месте основного текста. В приведенном ниже текстовом фрагменте команда использована во всех возможных вариантах. Короткие тире внесены в текст, чтобы показать отсутствие лишних пробелов.

В указателе и в тексте — одно и то же –**AAAA!both**–
В указателе — одно, в тексте — другое –**textentry**–
Только в указателе —
Только в тексте –**textonly**–
Ни в указателе, ни в тексте —.

```
\par В указателе и в тексте
     "--- одно и то же
     --\IXE{AAAA!both}--
\par В указателе "--- одно,
     в тексте "--- другое
     --\IXE[AAAA!indexentry]{textentry}--
\par Только в указателе
     --\IXE[AAAA!indexonly]{}--
\par Только в тексте --\IXE[]{textonly}--
\par Ни в указателе, ни в тексте
     --\IXE[]{}--.
```

В соответствующем .idx-файле будут содержаться только три элемента, так как при пустом необязательном аргументе ([]) элемент указателя не генерируется.

```
\indexentry{AAAA!both}{530}
\indexentry{AAAA!indexentry}{530}
\indexentry{AAAA!indexonly}{530}
```

Заглянув в предметный указатель данной книги, читатель обнаружит в нем элементы, сгенерированные двумя последними примерами.

Пакет ifthen, распространяемый в составе LaTeX 2ε, включает в себя еще несколько операторов сравнения, предложенных Александром Самариным.

\boolean{*string*} \newboolean{*string*} \setboolean{*string*}{*value*}

В базовой версии TeX'а имеется несколько переключателей, принимающих значения `true` («истина») или `false` («ложь»)[11]. Новый переключатель можно определить, пользуясь командой `\newboolean`, где в качестве имени переключателя *string* должна быть указана какая-либо последовательность букв. Исходное состояние переключателя — `false`. Изменение его состояния производится командой `\setboolean`, где аргумент *value* может принимать значение одной из двух цепочек литер `true` или `false`. Установить, каково текущее состояние переключателя, можно при помощи команды `\ifthenelse`, используя команду `\boolean` с именем переключателя в качестве первого аргумента. Тем же способом, кстати, можно проверять текущее состояние любых LaTeX'овских внутренних переключателей (значения наиболее известных из них даны в табл. А.5). В частности, нетрудно написать команду, проверяющую, какой режим набора текста установлен в данный момент — в одну колонку или в две.

Two-sided printing. `\ifthenelse{\boolean{@twoside}}{Two-sided}{One-sided} printing.`

\lengthtest{*test*}

Команда `\lengthtest` служит для сравнения величин, имеющих линейную размерность. В ее аргументе *test* может быть задано сравнение двух длин (либо заданных явно, например, `20cm`, либо указанных посредством имен, ранее определенных при помощи команды `\newlength`) с использованием одного из операторов сравнения <, = или >.

Рассмотрим в качестве примера ситуацию, когда рисунок, имеющий линейные размеры `\Xsize` и `\Ysize`, должен быть заверстан в прямоугольник со сторонами `\Xarea` и `\Yarea` без искажения пропорций (т. е. с сохранением отношения сторон). Приведенные ниже команды вычисляют новые линейные размеры рисунка `\newX` и `\newY`. Основная идея состоит в том, что вначале вычисляются и сравниваются между собой отношения сторон прямоугольника, с одной стороны, и

[11] При определении этих переключателей используется команда `\newif`.

TeX'овские переключатели	
`hmode`	true, если набираемый текст записывается по горизонтали, например, внутри абзаца или LR-бокса.
`vmode`	true, если происходит верстка по вертикали, например, при переходе от одного абзаца к другому.
`mmode`	true, если происходит набор формулы.
LaTeX'овские переключатели	
`@twoside`	true, если LaTeX верстает страницы, предназначенные для двусторонней печати.
`@twocolumn`	true, если LaTeX находится в режиме стандартного двухколонного набора (и false внутри окружений `multicols`).
`@firstcolumn`	true, если переключатель `@twocolumn` находится в состоянии true и при этом LaTeX производит верстку первого столбца.
`@newlist`	true в начале окружения типа перечня (переход в состояние false произойдет, когда встретится текст, *после* первой команды `\item`).
`@inlabel`	true после очередной команды `\item`, до тех пор, пока не встретится следующий за ней текст.
`@noskipsec`	true после заголовка, идущего в подбор, до тех пор, пока не встретится следующий за ней текст.

Таблица А.5. Важнейшие внутренние переключатели типа `\boolean`

рисунка, с другой, а затем с учетом результата сравнения вычисляется искомый коэффициент масштабирования.

```
\newlength{\sizetmp}\newlength{\areatmp}
\setlength{\sizetmp}{1pt*\ratio{\Xsize}{\Ysize}}
\setlength{\areatmp}{1pt*\ratio{\Xarea}{\Yarea}}
\ifthenelse{\lengthtest{\sizetmp > \areatmp}}%
   {\setlength{\newX}{\Xarea}
    \setlength{\newY}{\newX*\ratio{\Ysize}{\Xsize}}}
   {\setlength{\newY}{\Yarea}
    \setlength{\newX}{\newY*\ratio{\Xsize}{\Ysize}}}
```

`\isodd{`*number*`}`

Команда `\isodd` проверяет, является ли данное целое число *number* нечетным. Если, например, последовательность символов, генерируемая командой `\pageref`, представляет собой натуральное число (как оно обычно и бывает), то при помощи данной команды можно получить следующее:

Это — страница с четным номером.

```
Это\label{testref} "--- страница с
\ifthenelse{\isodd{\pageref{testref}}}{нечетным}{четным}
номером.
```

Команда \isodd в состоянии выполнять указанную функцию даже при первом прогоне LaTeX'a, когда значение \pageref еще не определено.

\whiledo{*test*}{*do-clause*}

Команду \whiledo удобно использовать для выполнения повторяющихся последовательностей команд. Следующий простой пример (в котором используется счетчик, введенный в предыдущих примерах) показывает, как работает данная команда:

```
\setcounter{hours}{1}
\whiledo{\value{hours}<5}{%
      Через \thehours~ч.\\%
      \stepcounter{hours}}
```

Через 1 ч.
Через 2 ч.
Через 3 ч.
Через 4 ч.

\and	\or	\not	\(	\)

Набор из нескольких условий можно объединить в одно логическое условие при помощи логических операторов (\or, \and и \not), используя управляющие символы \(и \) в качестве скобок. Простой пример этого подхода приведен ниже.

```
\newcommand{\QU}[2]{%
 \ifthenelse{%
   \(\equal{#1}{ENG} \and \equal{#2}{yes}\)
         \or
   \(\equal{#1}{FRE} \and \equal{#2}{oui}\)}%
 {``OK''}{``not OK''}%
}
You agree \QU{ENG}{yes} or don't \QU{ENG}{no}.
\par\bigskip
D'accord \QU{FRE}{oui} ou pas \QU{FRE}{non}?
```

You agree "OK" or don't "not OK".

D'accord "OK" ou pas "not OK"?

Приложение B

TEX'ническое обеспечение и группы пользователей

Если, прочтя эту книгу, вы все еще не можете найти нужный вам файл, следует обратиться к файлу `TeX-index` [23], поддерживаемому Дэвидом Джоунзом (MIT). В этом каталоге содержатся макро для TEX'а, LATEX'а и др. Перечислены все макро, доступные через анонимный ftp или mail-сервер на всех упомянутых ниже сайтах.

Сам файл `TeX-index`, подобно всем описанным в этой книге файлам и пакетам, доступен через анонимный ftp в Internet'е или через mail-сервер.

B.1 Главные сайты TEX'а в Internet'е

В настоящее время различные сообщества пользователей TEX'а координируют свои усилия с целью облегчить как можно большему количеству пользователей TEX'а доступ к TEX'ническим материалам. Самой новой разработкой в этом направлении является Всеобъемлющий сетевой TEX-архив (Comprehensive TEX Archive Network (CTAN)), который функционирует с 1993 г. [20]. Сейчас он состоит из сайтов Государственного университета Сэма Хьюстона в Эстоне (Великобритания), и группы пользователей Dante (Германия). Эти сайты поддерживают одно и то же TEX'ническое программное обеспечения, являясь зеркалами друг друга. Содержание архивов CTAN также доступно через CD-ROM [12].

Учитывайте, пожалуйста, географическое расположение сайта, которым вы хотите воспользоваться через ftp. Выбирайте как можно более близкий к вам сайт. Кроме того, скорость передачи данных может стать решающим моментом, если вы пересылаете большие файлы, требующие больше обычного (локального) рабочего времени. Ниже приводится краткий список основных TEX-архивов

в Европе и США. Директория, где можно найти TeX-index Джоунза, указана в первой строке каждого адреса (сдвинутый вправо текст). Архивы CTAN помечены звездочкой (*).

Америка

`ftp.shsu.edu*` `/tex-archive/help/TeX-index`
Сайт CTAN.

`ftp.math.utah.edu` `/pub/tex/tex-index`
Сервер, поддерживаемый Нельсоном Биби, содержит множество стилей, относящихся к BIBTEX'у и базам данных, особенно в области текстовых процессоров и математики. Имеется, например, база `tugboat.bib`, в которой перечислены все статьи, опубликованные в журнале *TUGBoat*.

`labrea.stanford.edu`
Официальное хранилище для TEX'а, METAFONT'а, dvips и родственных им файлов.

Европа

`ftp.dante.de*` `/pub/tex/help/TeX-index`
Сайт CTAN. Является также первоисточником для программ emTeX, publicTeX, publicMF и пакетов «из Майнца» Франка Миттельбаха и Райнера Шопфа (multicol, verbatim, theorem, NFSS, ftnright и array).

`ftp.tex.ac.uk*` `/pub/archive/help/TeX-index`
Сайт CTAN. Обиталище электронного журнала группы UKTeX.

Чтобы локализовать один или более хорошо известных пакетов, нужно искать archie в каком-нибудь из архивных сайтов, поддерживающих этот файл. Можно также найти массу полезной информации, используя услуги gopher, www или wais. Дополнительная информация относительно (LA)TEX'а содержится также в файле `tex.faq` часто задаваемых вопросов, поддерживаемом Бобби Боденхеймером (Caltech, USA) (доступном во всех упомянутых выше архивах из сайта `pit-manager.mit.edu` через `/pub/usenet/news.answers/tex-faq`) и в файле «Supplementary TeX Information» (Дополнительная TeX-информация), поддерживаемом Гуоином Чженем (NYU, USA) (сайт `cs.nyu.edu` в директории `ftp/pub/tex`).

Получение файлов из архива

На рис. B.1 и B.2 представлен сеанс ftp эстоновского архива CTAN. Первая команда `cd ctan:` помещает нас в верхнюю директорию дерева CTAN. Затем мы даем команду `quote site index xspace.sty`, которая локализует файл `xspace.sty` и дает его полное имя в файле иерархий CTAN'а. По команде `cd macros/latex/contrib/misc` мы идем в директорию, где этот файл помещается, и получаем интересующий нас файл посредством `get xspace.sty`.

Эту процедуру мы повторяем для файла Джоунза TeX-index, направляясь на его поиски в соответствующую директорию. В этом случае мы даем команду get TeX-index.gz, которая на лету сжимает файл, используя утилиту gzip (она выбирается благодаря указанному расширению gz). Эти сжатые файлы должны быть переданы в бинарном режиме, поэтому команда binary предшествует команде get. Мы можем в любой момент дать команду dir или ls, чтобы увидеть, что доступно, или команду pwd, чтобы узнать, в какой директории мы находимся. Как видно из примера, архивы CTAN поддерживают динамическую процедуру сжатия массивов данных и извлечения этих массивов из сжатого вида, так что большие файлы удобно перекачивать (например, в случае файла TeX-index сжатый файл оказался в четыре раза меньше — 86 168 вместо 367 136 байтов, так что время передачи резко сократилось). Если на вашей машине нет средств распаковки, сжатый файл можно распаковать на лету. Как эта процедура работает, подробно изложено в электронной документации на сайтах CTAN'a. Можно также получить листинг содержания архива в форматах tar, zip, zoo, добавив к имени соответствующего архива суффикс -lst, как показано в приведенной ниже команде get morebin.zip-lst. Команда !more показывает содержание архива morebin в формате zip на вашем локальном узле.

```
$ ftp ftp.tex.ac.uk
Connected to ftp.tex.ac.uk.
220 ftp.tex.ac.uk FTP server (Version 2.0WU(11) Wed Apr 28 22:25:23 GMT 1993) ready.
Name (ftp.tex.ac.uk:goossens): anonymous
331 Guest login ok, send e-mail address as password.
Password: goossens@mysystem.cern.ch (use your email address)
230-Welcome, archive user!  This is an FTP server for the UK TeX Archive.
          .... several lines of information not shown ....
230 Guest login ok, access restrictions apply.
ftp> cd ctan:
250 CWD command successful.
ftp> quote site index xspace.sty
200-index xspace.sty
200-NOTE. This index shows at most 20 lines. for a full list of files,
200-retrieve /pub/archive/FILES.byname
200-1992/09/21 |       5494 | macros/latex/contrib/misc/xspace.sty
200  (end of 'index xspace.sty')
ftp> cd macros/latex/contrib/misc
200 CWD command successful.
ftp> get xspace.sty
200 PORT command successful.
150 Opening ASCII mode data connection for xspace.sty (5494 bytes).
226 Transfer complete.
5645 bytes received in 3.67 seconds (1.50 Kbytes/s)
```

Рис. В.1. Пример сеанса связи ftp с TEX-архивом CTAN в Эстоне (часть 1)

```
ftp> quote site index TeX-index
200-index tex-styles
200-NOTE. This index shows at most 20 lines. for a full list of files,
200-retrieve /pub/archive/FILES.byname
200-1993/01/04 |     367136 | help/TeX-index
200  (end of 'index tex-styles')
ftp> cd ctan:
250 CWD command successful.
ftp> cd help
250 CWD command successful.
ftp> binary
200 Type set to I.
ftp> get TeX-index.gz
200 PORT command successful.
150 Opening BINARY mode data connection for /bin/gzip.
226 Transfer complete.
86168 bytes received in 17.61 seconds (4.78 Kbytes/s)
ftp> cd ctan:
250 CWD command successful.
ftp> cd systems/msdos/emtex-contrib/bonus
250 CWD command successful.
ftp> get morebin.zip-lst
200 PORT command successful.
150 Opening BINARY mode data connection for /bin/LIST.
226 Transfer complete.
1595 bytes received in 0.68 seconds (2.28 Kbytes/s)
ftp> !more morebin.zip-lst
Listing of ./morebin.zip
Length  Method   Size  Ratio   Date    Time   CRC-32     Name ("^" ==> case
------  ------   ----  -----   ----    ----   ------     ----   conversion)
 36000  Implode  17648  51%  01-27-92  16:34  7a371924  ^emtex/wp2latex.exe
 23535  Implode  14583  38%  11-25-91  17:31  8dff2c94  ^emtex/dvi2tty.exe
 50422  Implode  27970  45%  12-12-91  21:04  7e2ac940  ^emtex/dviselec.exe
 26897  Implode  12942  52%  12-12-91  09:54  6f3a375e  ^emtex/dvidvi.exe
 48608  Implode  27219  44%  12-12-91  21:29  fd19e62a  ^emtex/dviconca.exe
          .... several lines of information not shown ....
------          ------  ---                              -------
407402          203175  50%                              16
ftp> quit
221 Goodbye.
```

Рис. B.2. Пример сеанса связи ftp с TEX-архивом CTAN в Эстоне (часть 2)

B.2 Mail-серверы

Тот, кто не может выйти на Internet, но имеет доступ к электронной почте, также может получать файлы и информацию, используя mail-серверы, имеющиеся на многих сайтах, поддерживающих TEX'ническую информацию. Обратите внимание, однако, что каждый сервер стремится иметь свой собственный синтаксис, который в каждом случае должен быть скрупулезно соблюден. Более подробную информацию можно найти в списке новостей Supplementary TEX information, поддерживаемом Гуоином Чженем. На рис. B.3 приведены примеры mail-серверов и команд и показано, как с ними контактировать.

адрес:	`ftpmail@dante.de`
сообщение:	`help`
адрес:	`fileserv@shsu.edu`
сообщение:	`help`
адрес:	`mail-server@cs.ruu.nl`
сообщение:	`begin`
	`path user@machine.site` (**ваш** электронный адрес)
	`send help`
	`end`
адрес:	`listserv@hearn.bitnet`
сообщение:	`get tex filelist` чтобы получить список доступных файлов
адрес:	`listserv@vm.urz.uni-heidelberg.de`
сообщение:	`help`
	`get readme first tex`
	`get tex filelist`

Рис. B.3. Пример TEX'нических mail-серверов

B.3 Группы пользователей TEX'а

В разных странах пользователи TEX'а организуют свои группы пользователей TEX'а, в основном по языковому принципу. Если вы нуждаетесь в помощи, то сначала лучше всего обратиться в вашу локальную группу пользователей, потому что там вы можете получить ответ, который наиболее подходит для вашего программного окружения, ориентированного на родной язык. Ниже мы даем известные нам (на конец 1994 г.) адреса групп[1]. Как правило, они могут помочь вам с TEX'ническими материалами на дискетах, CD-ROM'ах или посредством пересылки файлов по e-mail, если у вас нет доступа к Internet.

TEX Users Group (Международная)
Mimi Jett, President
TEX Users Group
1466 NW Front Avenue, Suit 3141
Portland, OR 97209 USA
Tel: 503-223-9994
Fax: 503-223-3960
Email: `office@tug.org`

[1] При переводе внесены коррективы по состоянию на начало 1999 г. — *Прим. ред.*

AsTEX (Франкоговорящие)
Michel Lavaud, President
Association pour la diffusion de logiciels scientifiques liés TeX
Association AsTEX
BP 6532
45066 ORLEANS cedex 2
FRANCE
Тел.: 33 2 38 64 09 94
Email: `astex-admin@univ-orleans.fr`
discussion list: `astex@univ-orleans.fr`

Servan TEX Grupo de Usuarios de TEX Hispanohablantes
(Испаноговорящие)
Public mail list `spanish-tex@eunet.es`
José Ra Portillo Fernández Departamento de Matemática Aplicada I
Escuela Técnica Superior de Arquitecturas
Avenida de la Reina Mercedes, 2
E-41012 Sevilla, Spain
Email: `josera@gordo.us.es`

CSTUG (Чехия и Словакия)
Petr Sojka, President
Československé sdružení uživatelů TEXu
CSTUG, c/o FI MU, Botanicka 68a
CZ-602 00 Brno, Czech Republic
`ftp://ftp.cstug.cz/pub/tex`
Email: `cstug@cstug.cz`

CyrTUG (Россия)
Панкратьев Евгений Васильевич, Президент
Маховая Ирина Анатольевна, Исполнительный директор
Ассоциация пользователей кириллического TEX'а
Издательство Мир
Россия, 129820, Москва
Первый Рижский переулок, д. 2
Тел.: 095 286-0622
Fax: 095 288-9522
Email: `cyrtug@mir.msk.su`
www-page: www.cemi.rssi.ru/cyrtug

DANTE e.V. (Немецкоговорящие)
Tomas Koch, President
Deutschsprachige Anwendervereinigung TEX e.V.
Postfach 101840
D-69008 Heidelberg, Germany
Tel: 06221/29766
Fax: 06221/167906
Email: `dante@dante.de`

DK-TUG (Дания)
Thomas Widman, President DK-TUG
Department of Mathematical Sciences
University of Aarhus
Ny Munkegade Building 530 DK-8000
Aarhus DENMARK
Email: `dk-tug-bestyrelse@sunsite.auc.dk`

Estonian User Group (Эстония)
Enn Saar, Tartu
Astrophysical Observatory, Tõravere
EE2444 Estonia
Email: `saar@aai.ee`

ΕΦΤΕΧ — The Greek TEXFriends Group (Греция)
Apostolos Syropoulos, President
366, 28th October Str.
GR-671 00 Xanthi GREECE
Тел.: +30 541 28704
Email: `apostolo@obelix.ee.duth.gr`

Grupo de Utilizadores de TEX (Португалия, неформальная)
For information, contact Pedro Quaresma de Almeida Email: `pedro@mat.uc.pt`

GUST (Польша)
Tomasz Plata Przechlewski, President
Polska Grupa Użytkowników Systemu TEX
Instytut Matematiki Uniwersytetu Gdanskiego
ul. Wita Stwosza 57
80-952 Gdansk, Poland
Email: `ecot@univ.gda.pl`

GUTenberg (Франкоговорящие)
Goossens Michel, President
Groupe francophone des Utilisateurs de TEX
GUTenberg, c/o IRISA
Campus de Beaulieu
F-35042 Rennes cedex, France
Email: `gut@irisa.fr`

Hungarian TUG (Венгрия)
Email: `ludens@math.klte.hu`

ITALIC
Irish TEX And LATEX Interest Community (Ирландия, неформальная)
Discussions on mailing list `ITALIC-L` hosted on the list server
`listserv@irlearn.ucd.ie`
(открыт для подписки).

Japan TUG (Япония)
Nobuo Saitoh, Chairman
Japan TEX Users' Group
Faculty of Environmental Information
Keio University
5322 Endo, Fujisawa-shi
JP-252 Japan
Tel: +81 466 47 5111
Email: `ns@keio.ac.jp`

Lietovos TEX'o Vartotojø Grupë (Lithuanian TeX Users Group) (Литва)
Vytas Statulevicius, Chair
Akademijos 4 LT-2600 Vilnius, Lithuania
Тел.: +370 2 359 609
Fax: +370 2 359 804
Email: `vytass@ktl.mii.lt`

Nordic TEX Group (Скандинавия)
Dag Langmyhr, Chair
Department of Informatics
University of Oslo
Email: `dag@ifi.uio.no`

NTG (Голландскоговорящие)
Erik Frambach, Chair
Nederlandstalige TEX Gebruikersgroep
Postbus 394
NL-1740 AJ Schagen
The Netherlands
Email: `ntg@nic.surfnet.nl`

TEXCeH (Slovenian TeX User Group) (Словения)
Vladimir Batagelj
Jadranska 19 SI-61111 Ljubljana, Slovenia
Email: `Tex.Ceh@fmf.uni-lj.si`

Tirant lo TEX (Catalan TeX Users Group) (Каталания)
Gabriel Valiente Feruglio
Technical University of Catalonia
Department of Software Mòdul C6,
Campus Nord Jordi Girona Salgado, 1-3
E-08034 Barcelona, Catalonia
Email: `valiente@lsi.upc.es` discussion list: Email: `catala-tex@aliga.cesca.es` (send subscription requests to Email: `listserv@cesca.es`).

TUG India (Indian TeX Users Group) (Индия)
Prof. (Dr.) K. S. S. Nambooripad, Chairman
Kripa, TC 24/548, Sastha Gardens
Thycaud, Trivandrum 695014, India
Тел.: +91 471 324341
Fax.: +91 471 333186
Email: `tugindia@mailexcite.com`

TUG-Philippines (Philippines TeX Users Group) (Филиппины)
Dr. Felix P. Muga II, President
Mathematics Department Ateneo de Manila University
Loyola Heights, Quezon City Тел.: (63-2) 426 6001 local 2515
Fax: (63-2) 426 6008
Email: `fpmuga@admu.edu.ph`
`fpmuga@philonline.com`

UK TUG (Великобритания)
Philip Taylor, Chairman
UK TEX Users' Group
c/o Peter Abbott
1 Eymore Close
Selly Oak
Birmingham B29 4LB, United Kingdom
Email: `uktug-enquiries@tex.ac.uk`

Список литературы[1]

[1] Андре, Гримо (Jacques André and Jeanine Grimault). Emploi des capitales (première partie). *Cahiers GUTenberg*, 6:42–50, July 1990.

Использование капители (прописных букв) во французском языке.

[2] Андре, Лоран (Jacques André and Philippe Louarn). Notes en bas de pages: comment les faire en LaTeX? *Cahiers GUTenberg*, 12:57–70, December 1991.

Обсуждается несколько специальных случаев использования подстрочных примечаний в LaTeX'е; например, как получить ссылку внутри tabular или окружения minipage, либо как сослаться на одно и то же подстрочное примечание более одного раза.

[3] Аурбах (Richard Aurbach). IdxTeX and GloTeX— indexes and glossaries. *TUGboat*, 7(3):187, October 1986.

Краткое описание двух специальных процессоров для указателей и глоссариев LaTeX'а на VAX/VMS.

[4] Аурбах (Richard Aurbach). Automated index generation for LaTeX. *TUGboat*, 8(2):201, July 1987.

После краткого введения в проект IdxTeX, Ричард Аурбах описывает, каким образом в программе IdxTeX на VAX/VMS обеспечивается полный спектр услуг для получения указателя. Представлен входной синтаксис и дан обзор структуры внутренних данных и различных частей этой программы.

[5] Берри (Karl Berry). Filenames for fonts. *TUGboat*, 11(4):517–520, November 1990.

Предлагается последовательная разумная схема наименований для названий файлов шрифта. Каждое имя состоит из не более восьми символов, которые отождествляют каждый файл шрифта уникальным способом.

[6] Биби (Nelson H. F. Beebe). Bibliography Prettyprinting and Syntax Checking. *TUGboat*, 14(4):395–419, December 1993.

Описывается три программных средства для поддержания BibTeX'а. Первое, `bibclean`, предоставляет возможность печати, проверки синтаксиса и лексического анализа файлов BibTeX'а. Второе, `biblex`, это — возможность лексического анализа элементов в файле BibTeX'а. Третье, `bibparse`, это — средство грамматического разбора, анализирующее

[1] Работы, добавленные при переводе, отмечены знаком *.— *Прим. ред.*

лексический поток элементов, поступающий из `bibclean` и `biblex`. `bibclean` использует постоянно порождаемые файлы инициализации и образцы, определяемые во время работы программы, для проверки цепочек символов BibTeX'а. value strings.

[7] Биллавала (Neenie Billawala). Metamarks: Preliminary studies for a Pandora's Box of shapes. Technical Report STAN-CS-89-1256, Stanford University, Department of Computer Science, May 1989.

Небольшой буклет, описывающий элементы начертания шрифтов, такие как засечки, дуги, впадины и т. д. Изучение этих элементов в конечном счете привело к созданию шрифта, названного Pandora.

[8] Биллавала (Neenie Billawala). Opening Pandora's Box. *TUGboat*, 10(4):481–489, December 1989.

При создании семейства Pandora автор пытался использовать METAFONT как средство дизайна, а не как средство генерации. В общем виде были разработаны зрительные пропорции между фрагментами литер, между взаимным расположением литер в шрифте и между шрифтами в набранном тексте, так что просто меняя установочные параметры в одной и той же среде, можно получать разнообразные шрифты. Это позволяет дизайнеру быстро освоить различные возможности.

[9] Боде (Hans-Hermann Bode). Neue BibTeX-Style-Files: Die *adaptable family*. *Die TEXnische Komödie*, 4(2):31–41, August 1992.

Вводится новое семейство стилевых файлов BibTeX'а, в котором пользователь может изменить макет (шрифты используются для особых полей) и отдельные слова или фразы, используемые этим стилем (такие, как названия месяцев, глав, статей и союзов). Эти цепочки литер параметризуются внутри стилей BibTeX'а и получают нужное значение, когда обработанный TeX'ом документ прочитывается внешним независимым от языка файлом, который определяет эти термины для того или иного языка.

[10] Борсо (Francis Borceux). User's guide for the *Diagram* Macros. Electronic document distributed with the package, February 1993.

Пакет построения коммутативных диаграмм. Особенно полезен для диаграмм, принятых в теории категорий. Использует режим рисования LaTeX'а как основной механизм.

[11] Брамс (Johannes Braams). Babel, a multilingual style-option system for use with LaTeX's standard document styles. *TUGboat*, 12(2):291–301, June 1991.

Описывается, как приспособить LaTeX к более чем одному языку с помощью специфических языковых стилевых опций. Последняя версия опубликована в *TUGboat*, 14(1):60–62, апрель 1993.

[12] Браун (Vicki Brown, editor). *Prime Time TEXcetera*, volume 1. Prime Time Freeware, Sunnyvale, CA, 1994.

CD-ROM, содержащий TeX'ническое программное обеспечение, имеющееся в архивах CTAN, дополненный буклетом.

[13] Вичура (Michael J. Wichura). PiCTeX: Macros for drawing PiCtures. *TUGboat*, 9(2):193–197, August 1988.

PiCTeX представляет собой набор макрокоманд TeX'а, при помощи которых облегчается работа с рисунками, преимущественно со сложными графиками. Руководство по пакету PiCTeX можно получить через TeX Users Group.

[14] Воннеберг (Reinhard Wonneberger). *LaTeX Kompaktführer*. Addison-Wesley Verlag, Bonn, Germany, third edition, April 1993.

Полезный обзор (на немецком языке) всех средств LaTeX'а, в которых рядовой пользователь нуждается повседневно.

[15] Воннебергер, Миттельбах (Reinhard Wonneberger and Frank Mittelbach). BibTeX reconsidered. *TUGboat*, 12(1):111–124, March 1991.
После общего обсуждения BibTeX'а дается несколько предложений относительно улучшений и изменений.

[16] Вулис (Michael Vulis). VTeX enhancements to the TeX language. *TUGboat*, 11(3):429–434, September 1990.
Автор описывает свой коммерческий продукт VTeX, который поддерживает собственные шрифты в особом масштабируемом формате. Более подробно с этой системой можно ознакомиться по книге того же автора: Michael Vulis, *Modern TeX and Its Application*, CRC Press, Ann Arbor, 1993

[17]* Гиленсон П. *Справочник технического редактора.* — М.: Книга, 1979.
Книга для профессиональных издательских работников, содержащая полный перечень правил типографского набора русскоязычных текстов.

[18] Голль (Bernard Gaulle). Notice d'utilisation du style french (Version 3,25). Electronic document distributed with the package, November 1993.
Руководство пользователя по пакету french (на французском языке).

[19] Гретцер (George Grätzer). *Math into TeX: A Simple Introduction to AMS-LaTeX.* Birkhäuser, Boston, 1993.
В первой части книги, предназначенной для новичков, описывается AMS-LaTeX на элементарном уровне; описание основывается на многих примерах, вереницах формул, образцах файлов и шаблонов. Вторая часть содержит методичное обсуждение во всех деталях примеров и правил AMS-LaTeX, а в последней части даны более сложные приемы.

[20] Гринуэйд (George D. Greenwade). The Comprehensive TeX Archive Network (CTAN). *TUGboat*, 14(3):342–351, October 1993.
В общих чертах описывается принцип устройства, разработка и использование архива CTAN, который дает возможность получить из сети все имеющие отношение к TeX'у файлы. CTAN обеспечивает тесную взаимосвязь основных TeX-архивов на разных континентах, благодаря чему они являются зеркалами друг друга в синхронном режиме. На CTAN'е предусмотрена возможность постоянной идентификации файлов и их поиска, благодаря чему снижается общая загруженность сети и увеличивается скорость поиска.

[21] Грэхем, Кнут, Паташник (Ronald L. Graham, Donald E. Knuth, and Oren Patashnik). *Concrete Mathematics: A Foundation for Computer Science.* Addison-Wesley, Reading, 1989. Second edition 1993. [Имеется перевод 2-го издания: Грэхем Р., Кнут Д., Паташник О. Конкретная математика. Основание информатики. — М.: Мир, 1998.]
Прекрасно оформленный учебник по математике, подготовленный в системе TeX с использованием шрифта Concrete Roman в качестве основного; см. также [36].

[22] Дерст (Lincoln Durst). Some tools for making indexes: Part I. *TUGboat*, 12(2):248–252, June 1991.
Обзор основных приемов TeX'а для создания указателей. Можете также заглянуть в статью David Salomon'а *Macros for indexing and table-of-contents* in: *TUGboat*, 10(3):394–400, November 1989.

[23] Джоунз (David M. Jones). A TEX macro index. *TUGboat*, 13(2):188–189, July 1992.

Данная статья представляет собой официальное сообщение о каталоге TEX'овских макропакетов, созданном и поддерживаемом автором. Доступ к каталогу возможен либо по анонимному ftp, либо через почтовый сервер. Адреса архивов, содержащих данный каталог, были опубликованы в *TEX and TUG NEWS*, 2(2):12–13, April 1993.

[24] Доуэрти, О'Райли (Dale Dougherty and Tim O'Reilly). *UNIX Text Processing*. Howard W. Sams & Company, Hayden Books, Indianapolis, 1988.

В этой книге дано полное описание средств обработки текста в системе UNIX. В частности, в гл. 10, «Drawing Pictures», объясняется синтаксис языка процессора `pic` и приводится много примеров. Информация об этом языке содержится также в UNIX-руководстве по процессорам pic и gpic (GNU).

[25] Зокки (Maurizio Zocchi). LATEX's Index Processing. *TUGboat*, 8(1):62, April 1987.

Описание INDTEX — процессора LATEX'а для создания указателя на платформах VAX/VMS и MS-DOS.

[26] Кларк А. (Adrian F. Clark.) Practical halftoning with TEX. *TUGboat*, 12(1):157–165, March 1991.

Обзор, в котором перечислены практические вопросы использования TEX'а для получения полутоновых иллюстраций и проводится сравнительный анализ других методов включения графического материала. Описываются достоинства и недостатки разных подходов и обсуждаются попытки получения цветных иллюстраций.

[27] Кларк М. (Malcolm Clark). Portable graphics in TEX. *TUGboat*, 13(3):253–260, October 1992, and Chapter 17 of *A Plain TEX Primer*, by Malcolm Clark, Oxford University Press, 1992.

Обсуждаются три основных способа включения графики в TEX'нические документы: особые шифты, команда `\special` и промежуточные решения. Имеются другие обзоры: *TEX and Graphics: The State of the Problem*, by Nelson Beebe, в: *Cahiers GUTenberg*, 2:13–53, May 1989, и *Including Pictures in TEX*, by Alois Heinz, в: *TEX applications, uses, methods*, Malcolm Clark (ed.), Ellis Horwood Publishers, Chichester, England, pp. 141–151, 1990.

[28] Кнут (Donald E. Knuth). *TEX and Metafont, New Directions in Typesetting*. The American Mathematical Society and Digital Press, Bedford, MA, 1979.

Первая часть данной книги содержит перепечатку статьи Дональда Кнута, озаглавленной «Mathematical Typography», в которой автор объясняет, что побудило его взяться за разработку TEX'а, и рассказывает о начальном периоде развития компьютерной полиграфии. Описания TEX'а и METAFONT'а, содержащиеся во второй и третьей частях книги, относятся к ранним (к настоящему времени устаревшим) версиям этих систем; тем не менее для лиц, интересующихся историей TEX'а, эти разделы представляются весьма полезными.

[29] Кнут (Donald E. Knuth). Remarks to Celebrate the Publication of Computers & Typesetting. *TUGboat*, 7(2):95–98, June 1986.

21 мая 1986 года Компьютерный музей Бостона устроил праздник по случаю завершения работы над TEX'ом, системой компьютерного полиграфического набора, созданной Дональдом Кнутом. Данная статья воспроизводит речь Д. Кнута на празднике, в которой рассказано о девяти годах работы над системами TEX и METAFONT.

[30] Кнут (Donald E. Knuth). *The TEXbook*, volume A of *Computers and Typesetting*. Addison-Wesley, Reading, 1986.
[Имеется перевод: Кнут Д. Все про TEX. — Протвино: АО RDTEX, 1993.] Исчерпывающее руководство пользователя и полный справочник по системе TEX.

[31] Кнут (Donald E. Knuth). *TEX: The Program*, volume B of *Computers and Typesetting*. Addison-Wesley, Reading, 1986.
Хорошо оформленный полный листинг исходного программного кода для системы TEX. Программа TEX написана на языке WEB; данная книга представляет собой слегка отредактированную версию программного кода, полученного в результате прогона программы WEAVE и обработки полученного текста TEX'ом.

[32] Кнут (Donald E. Knuth). *The METAFONTbook*, volume C of *Computers and Typesetting*. Addison-Wesley, Reading, 1986.
Руководство пользователя и справочник по системе METAFONT— парной к TEX'у программе, предназначенной для создания шрифтов.

[33] Кнут (Donald E. Knuth). *METAFONT: The Program*, volume D of *Computers and Typesetting*. Addison-Wesley, Reading, 1986.
Полный листинг исходного текста программы METAFONT.

[34] Кнут (Donald E. Knuth). *Computer Modern Typefaces*, volume E of *Computers and Typesetting*. Addison-Wesley, Reading, 1986.
Книга содержит графические изображения более чем пятисот букв греческого и латинского алфавитов, а также знаков препинания и большого числа математических символов. Приведены «METAFONT'овские описания» всех глифов (изображений символов), т. е. тексты на входном языке программы METAFONT, позволяющие сгенерировать соответствующие графические изображения. Показано, каким образом, управляя параметрами METAFONT'овских описаний, можно получить все семейство гарнитур Computer Modern.

[35] Кнут (Donald E. Knuth). Fonts for digital halftones. *TUGboat*, 8(2):135–160, July 1987.
Описаны эксперименты по использованию программы METAFONT для создания полутоновых изображений, предназначенных для печати на лазерном принтере.

[36] Кнут (Donald E. Knuth). Typesetting *Concrete Mathematics*. *TUGboat*, 10(1):31–36, April 1989.
Статья Д. Кнута об особенностях набора в TEX'е книги *Concrete Mathematics*.

[37] Кнут (Donald E. Knuth). Virtual Fonts: More Fun for Grand Wizards. *TUGboat*, 11(1):13–23, April 1990.
Тема статьи — что такое виртуальные шрифты и зачем они нужны. Приведены технические подробности и описаны способы реализации основных идей.

[38] Кнут (Donald E. Knuth). The Future of TEX and Metafont. *TUGboat*, 11(4):489, November 1990.
«Моя работа по развитию TEX'а, METAFONT'а и шрифтов Computer Modern закончена. Дальнейшие изменения этих продуктов будут производиться лишь в случае, если в них обнаружатся серьезные ошибки.» Этими словами Д. Кнут начинает статью, в которой он сообщает о своем намерении законсервировать работу над системой TEX, остановившись на текущих версиях программ.

[39] Кнут, Маккей (Donald Knuth and Pierre MacKay). Mixing right-to-left texts with left-to-right texts. *TUGboat*, 8(1):14, April 1987.

В этой статье освещаются некоторые средства, необходимые в документах со смешанными направлениями набора, такие как переупорядочение литер внутри слов и переупорядочение слов внутри строк плюс перетекание материала на следующую строку вправо, если текущая строка заполнена. Обсуждаются изменения, расширяющие возможности TEX'а в плане форматирования в двух направлениях. Эта техника реализована в приложениях TEX-XET и TEX--XET.

[40]* Котельников И. А., Чеботаев П. З. *Издательская система LaTeX2ε*. Новосибирск: Сибирский хронограф, 1998.

Наиболее полное руководство по системе LaTeX2ε на русском языке. Дано подробное описание команд, работа которых иллюстрируется большим количеством примеров.

[41] Куок (Conrad Kwok). EEPIC: Extensions to EPIC and LaTeX picture environment. Unpublished machine-readable document, 1988.

Руководство пользователя, распространяемое вместе с макропакетом eepic, который расширяет графические возможности как LaTeX'а, так и пакета epic.

[42] Лаваньино, Вуястик (John Lavagnino and Dominik Wujastyk). An Overview of EDMAC: A plain TEX format for critical editions. *TUGboat*, 11(4):623–643, November 1990.

Пакет EDMAC предназначен для набора в TEX'е литературоведческих работ в традиционном оформлении (в стиле Oxford Classical Texts, Шекспировской серии и др.). Пакет позволяет проставлять номера строк на полях и создавать серии подстрочных и выносных примечаний, привязанных к номерам строк. Приведение формата в соответствие с запросами пользователя осуществляется достаточно просто.

[43] Левин (Micheal J. S. Levine). A LaTeX Graphics Routine for drawing Feynman Diagrams. *Computer Physics Communications*, 58:181–198, 1990.

Описание пакета Feynman, позволяющего рисовать диаграммы Фейнмана. Более подробное описание пакета имеется в диссертации автора, которая распространяется вместе с пакетом в виде LaTeX'овского файла.

[44] Лианг (Franklin Mark Liang). Word Hy-phen-a-tion by Com-pu-ter. Ph.D. thesis, Stanford University, Stanford, CA 94305, June 1983. Also available as Technical Report No. STAN-CS-83-977, Stanford University, Department of Computer Science.

Подробное описание алгоритма переноса слов, используемого в TEX'е.

[45]* Львовский С. М. *Набор и верстка в пакете LaTeX*, 2-е издание. М.: Космосинформ, 1995.

Одно из наиболее полных и доступных изданий на русском языке, посвященное работе в пакете LaTeX. Второе издание книги дополнено материалом о LaTeX2ε.

[46] Лэмпорт (Leslie Lamport). *LaTeX—A Document Preparation System—User's Guide and Reference Manual*. Addison-Wesley, Reading, 1985.

Основополагающее руководство по LaTeX'у версии 2.09, написанное его автором. [Новое издание книги, охватывающее LaTeX2ε, вышло в том же издательстве в 1994 году.— *Ред.*] Книга может служить хорошим дополнением к настоящей работе.

[47] Лэмпорт (Leslie Lamport). *MakeIndex*, An Index Processor For LaTeX. Electronic document coming with the *MakeIndex* distribution, 1987.

Текст, содержащий описание синтаксиса LaTeX'овской команды `\index`, используемой для создания указателя с помощью макропакета *MakeIndex*. Содержит также полный перечень возможных сообщений об ошибках.

[48] Миттельбах (Frank Mittelbach). An extension of the LaTeX theorem environment. *TUGboat*, 10(3):416–426, November 1989.

В различных математических журналах по-разному оформляются теоремы. С помощью пакета theorem и предлагаемых им «стилей» оформление теорем можно привести в соответствие с требованиями конкретного журнала. В статье описан пользовательский интерфейс пакета и его реализация.

[49] Миттельбах (Frank Mittelbach). E-TeX: Guidelines for future TeX extensions. *TUGboat*, 11(3):337–345, September 1990.

TeX создавался Д. Кнутом для компьютерного набора его собственных книг. В настоящее время им пользуются десятки тысяч людей. Отвечает ли TeXкак система компьютерного набора требованиям девяностых годов? Для ответа на этот вопрос документы, напечатанные при помощи TeX'а, сравниваются с результатами ручного набора. Приводятся примеры областей деятельности, где возможности TeX'а оказываются недостаточными. Выясняется, что многие важные аспекты высококачественной полиграфии не охватываются TeX'ом. Делается вывод, что должны быть проведены дальнейшие исследования с целью создания «преемника» системы TeX.

[50] Миттельбах (Frank Mittelbach). Comments on "Filenames for Fonts" (*TUGboat* 11#4). *TUGboat*, 13(1):51–53, April 1992.

Обсуждаются трудности, возникающие в случае, когда имена шрифтов строятся по схеме, предложенной К. Берри; особое внимание уделяется возможности независимого определения некоторых шрифтовых свойств, а также использованию указанной схемы в сочетании с NFSS.

[51] Миттельбах, Роули (Frank Mittelbach and Chris Rowley). LaTeX 2.09 ↪ LaTeX3. *TUGboat*, 13(1):96–101, April 1992.

Краткое введение в проект LaTeX3, включающее историю его возникновения и развития, а также описание структуры разрабатываемой системы. Переработанная версия этой заметки опубликована в *TUGboat*, 13(3):390–391, October 1992. В *TUGboat*, 13(4):510–515, December 1992, было объявлено о приглашении добровольцев, желающих участвовать в реализации проекта, и был приведен список актуальных задач. В статье, помимо прочего, объясняется, как, пользуясь электронной почтой или ftp, можно получить текущую версию списка задач и документы, подготовленные рабочей группой, и как подписаться на новости по проекту LaTeX3.

[52] Миттельбах, Шопф (Frank Mittelbach and Rainer Schöpf). A new font selection scheme for TeX macro packages — the basic macros. *TUGboat*, 10(2):222–238, July 1989.

Описание основных макрокоманд, реализующих первую версию New Font Selection Scheme для LaTeX'а.

[53] Миттельбах, Шопф (Frank Mittelbach and Rainer Schöpf). With LaTeX into the Nineties. *TUGboat*, 10(4):681–690, December 1989.

Поскольку в настоящее время LaTeX применяется в самых различных областях человеческой деятельности, возникает опасность, что разного рода локальные изменения и улучшения разрушат единые стандарты и сделают невозможным обмен LaTeX'овскими документами. В статье предлагается провести полную ревизию LaTeX'а, цель которой —

учесть растущие потребности различных пользовательских сообществ, сохранив основные черты существующего пользовательского интерфейса. Приводятся и некоторые соображения о путях дальнейшего развития LaTeX'а.

[54] Миттельбах, Шопф (Frank Mittelbach and Rainer Schöpf). Reprint: The new font family selection — User interface to standard LaTeX. *TUGboat*, 11(2):297–305, June 1990.

Полное описание пользовательского интерфейса первой версии New Font Selection Scheme для LaTeX'а.

[55] Миттельбах, Шопф (Frank Mittelbach and Rainer Schöpf). Towards LaTeX3.0. *TUGboat*, 12(1):74–79, March 1991.

Описаны основные цели проекта LaTeX3. Рассмотрены улучшения, которые необходимо внести в пользовательский интерфейс LaTeX'а и в механизм стилевых файлов для того, чтобы привести LaTeX в соответствие с современными подходами, в первую очередь, с концепциями метаязыка SGML. Перечислены несколько внутренних конструкций LaTeX'а, подлежащих изменению.

[56] Николь, Андре, Голль (Olivier Nicole, Jacques André, and Bernard Gaulle). Notes en bas de pages: commentaires. *Cahiers GUTenberg*, 15:46–52, April 1993.

Комментарии, пояснения и добавления к статье [2].

[57] Партль (Hubert Partl). German TeX. *TUGboat*, 9(1):70–72, April 1988.

Одна из первых попыток использовать LaTeX применительно к языку, отличному от английского, в данном случае — немецкому. Описаны возникшие при этом проблемы.

[58] Паташник (Oren Patashnik). BibTeXing. Documentation for general BibTeX users, 8 February 1988. Electronic document coming with the BibTeX distribution.

В статье, тесно связанной с приложением B основного руководства Л. Лэмпорта, описан пользовательский интерфейс программы BibTeX. Статья вносит поправки в раздел B.2 упомянутого руководства и содержит большое число полезных советов, позволяющих более эффективно управлять поведением BibTeX'а.

[59] Паташник (Oren Patashnik). Designing BibTeX styles. The part of BibTeX's documentation that's not meant for general users, 8 February 1988. Electronic document coming with the BibTeX distribution.

Подробное описание языка с постфикс-стековой структурой, используемого внутри стилевых файлов BibTeX'а. Предназначается разработчикам стилей для BibTeX'а. Помимо общего описания языка, приведены все команды и встроенные функции. Подробно рассмотрен BibTeX'овский механизм форматирования имен.

[60]* Пикок (John Peacock) *Book production*, second edition. Blueprint, 1995.

[Имеется перевод: Пикок Дж. Издательское дело. — М.: Издательство ЭКОМ, 1998.] Книга посвящена книгопроизводству во всех его аспектах. Русский перевод дополнен действующими в России стандартами.

[61] Подар (Sunil Podar). Enhancements to the picture environment of LaTeX. (Version 1.2) Technical Report 86-17, Department of Computer Science, S.U.N.Y, 1986.

Описаны новые команды, предназначенные для LaTeX'овского окружения picture. Это, главным образом, команды высокого уровня, создающие более мощный и более дружественный пользовательский интерфейс и расширяющие графические возможности

LaTeX'a. С помощью этих команд можно создавать более сложные иллюстрации и затрачивать меньшие усилия, чем при использовании одних лишь базовых средств LaTeX'a.

[62] Райт (Haviland Wright). SGML frees information. *Byte*, 17(6):279–286, June 1992.

Описывается, как SGML помогает вам избежать переполнения данными, посредством выделения определений структур доступа для вашей информации.

[63] Ратц (Sebastian Rahtz). A survey of TeX and graphics. Technical Report CSTR 89-7, University of Southampton, Department of Electronics and Computer Science, Southampton SO9 5NH, England, October 1989.

Подробно рассмотрены различные способы включения графического материала в LaTeX'овские документы.

[64] Ратц, Баррока (Sebastian Rahtz and Leonor Barroca). A style option for rotated objects in LaTeX. *TUGboat*, 13(2):156–180, July 1992.

Подробное описание пользовательского интерфейса и внутренних механизмов пакета rotating, позволяющего осуществлять разнообразные вращения графических объектов, в том числе фигур целиком, с помощью всевозможных PostScript'овских драйверов.

[65] Рейд Г. (Glenn C. Reid). *PostScript Language Program Design.* Addison-Wesley, Reading, 1988.

Так называемая «зеленая книга» — учебник по основам программирования на языке PostScript, по которому можно научиться создавать эффективные программы на PostScript'е, легкие для понимания и удобные в работе. Среди рассматриваемых задач — набор текста, работа с графикой, обработка ошибок и техника отладки программ.

[66] Рейд Г. (Glenn C. Reid). *Thinking in PostScript.* Addison-Wesley, Reading, 1990.

Цель книги — помочь читателю повысить уровень его мастерства в написании программ на языке PostScript путем овладения некоторыми полезными, простыми и элегантными приемами программирования. В числе рассматриваемых вопросов (не освещенных практически ни в одном другом руководстве) — оптимизация циклов, условных переходов, процедур ввода-вывода, а также техника работы с файлами.

[67] Рейд Т. (Thomas J. Reid). Floating figures at the right — and — Some random text for testing. *TUGboat*, 8(3):315, November 1987.

Описание приемов, используемых для размещения иллюстраций.

[68] Рокицки (Tomas Rokicki). DVIPS: A TeX Driver. Electronic document coming with the dvips distribution, January 1993.

Руководство пользователя программы dvips и сопутствующих программ и пакетов, в том числе программы afm2tfm для создания tfm-файла на основе имеющегося afm-файла (т. е. файла, содержащего Adobe font metrics — метрические характеристики PostScript'овского шрифта типа Adobe Type 1), а также программы colordvi предназначенной для цветной печати.

[69] Рот (Stephen E. Roth, editor). *Real World PostScript. Techniques from PostScript Professionals.* Addison-Wesley, Reading, 1988.

Так называемая «оранжевая книга» — сборник статей, написанных профессионалами в области разработки или использования приложений на языке PostScript. Среди вопросов, рассмотренных в книге, — техника работы со шрифтами и словарями, работа с цветом, полутоновые изображения, высокоточный кернинг, трекинг и межсимвольные пробелы.

[70] Роуз (Kristoffer H. Rose). Typesetting Diagrams with XY-pic: User's Manual. In *Proceedings of the 7th European TEX Conference, Prague*, pp. 273–292, September 1992.

Подробное описание возможностей пакета XY-pic, предназначенного для вычерчивания графиков и диаграмм в TEX'е.

[71] Рубинштейн (Richard Rubinstein). *Digital Typography—An Introduction to Type and Composition for Computer System Design.* Addison-Wesley, Reading, November 1988. Reprinted with corrections.

Тема книги — технология компьютерной полиграфии. Описаны возможности использования компьютеров для проектирования, создания и размещения графических элементов, применяемых при компьютерной подготовке документов.

[72] Смит (Ross Smith). *Learning PostScript — A Visual Approach.* Peachpit Press, 1085 Keith Avenue, Berkeley, CA 94708, 1990.

«Наглядное» (и неторопливое) введение в язык PostScript. Основные конструкции языка вводятся с помощью примеров. «Наглядность» достигается за счет того, что на четных страницах (левых страницах разворота книги) располагаются тексты программного кода примеров и краткие пояснения, а на нечетных (правых) страницах приводятся результаты, полученные на печати.

[73] Сова (Friedhelm Sowa). Bitmaps and halftones with BM2FONT. *TUGboat*, 12(4):534–538, November 1991.

Программа BM2FONT дает возможность преобразовывать различные виды растровых изображений в TEX'овские шрифты и создавать входной файл, позволяющий включать эти изображения в документы.

[74]* Спивак (Michael Spivak) *The Joy of TEX. A gourmet guide to typesetting with the AMS-TEX macro package.* American Mathematical Society, Providence, RI, 1990.

[Имеется перевод: М. Спивак. Восхитительный TEX: руководство по комфортному изготовлению научных публикаций в пакете AMS-TEX. — М.: Мир, 1993.] Первое TEXническое издание, переведенное на русский язык. Книга посвящена работе в пакете AMS-TEX для подготовки математических публикаций; в методическом отношении до сих пор считается непревзойденной среди учебников по работе в TEX'е и родственным ему пакетам.

[75] Тейлор П. (Paul Taylor). Commutative Diagrams in TEX (version 4). Electronic document distributed with the package, January 1993.

Макропакет для вычерчивания коммутативных диаграмм. Совместим с большинством TEX'овских форматов. Для создания диагональных стрелок используются либо LATEX'овские шрифты, содержащие линейные элементы, либо команда `special` с PostScript'овскими вставками, генерирующими стрелки и производящими их поворот.

[76] Тейлор Ф. (Philip Taylor). The future of TEX. In *Proceedings of the 7th European TEX Conference, Prague*, pp. 235–254, September 1992.

Обсуждается следующий вопрос: как добиться сохранения и поддержания TEX'овской философии в компьютерной полиграфии после того, как Д. Кнут принял решение законсервировать текущую реализацию TEX'а.

[77] Тимблби (Harold Thimbleby). "See also" indexing with Makeindex. *TUGboat*, 12(2):290, June 1991.

Заметка о технике генерирования ссылок вида «see also» при создании указателей.

[78] Тимблби (Harold Thimbleby). Erratum: "See also" indexing with Makeindex, *TUGboat* 12, no. 2, p. 290. *TUGboat*, 13(1):95, April 1992.
Исправление и замечания к статье [77].

[79] Фолленвайдер (Peter Vollenweider). *Encapsulated PostScript: Applications for the MacIntosh and PC.* Prentice-Hall and Carl Hanser, Hertfordshire, 1990.
В книге внимание концентрируется на том, каким образом собирать воедино текст, графический материал и образы на уровне PostScript'а с использованием формата инкапсулированного PostScript'а в качестве стандарта. Особое внимание уделяется тому, как импортировать графику в ваш документ средствами настольных издательских систем на базе персональных компьютеров, рабочих станций UNIX или крупных вычислительных систем.

[80] Хараламбус (Yannis Haralambous). Typesetting old german: Fraktur, Schwabacher, Gotisch and initials. *TUGboat*, 12(1):129–138, March 1991.
Показано, как можно использовать METAFONT для точного воспроизведения шрифтов из древних рукописей для последующего применения с TEX'ом. Приведены правила набора с этими шрифтами и примеры страниц, так набранных.

[81] Хендерсон (Doug Henderson). Outline fonts with METAFONT. *TUGboat*, 10(1):36–38, April 1989.
Описание кода METAFONT'а, используемого для генерации контуров из существующих описаний литер.

[82] Хёниг (Alan Hoenig). When TEX and Metafont Work Together. In *Proceedings of the 7th European TEX Conference, Prague*, pp. 1–19, September 1992.
Описывается, как TEX и METAFONT могут передавать данные, чтобы позволить вам готовить диаграммы и рисунки при помощи METAFONT'а и делать надписи средствами TEX'а. И наоборот, METAFONT может генерировать особые шрифты, которые можно использовать, например, для набора текста вдоль искривленных базовых линий.

[83] Хервейнен, ван (Eric van Herwijnen). *Practical SGML.* Wolters-Kluwer Academic Publishers, Boston, second edition, 1994.
Введение в язык SGML и его приложения. Содержится много примеров областей, где успешно применяется SGML, и обсуждаются различные его применения.

[84] Хос, О'Кейн (Roswitha T. Haas and Kevin C. O'Kane). Typesetting Chemical Structure Formulas with the Text Formatter TEX/LATEX. *Computers and Chemistry*, 11(4):251–271, 1987.
В этой статье дается обзор возможностей пакета ChemTEX. Более подробная информация содержится в отдельных главах докторской диссертации Хос, которая распространяется вместе с пакетом. Другие средства набора химических формул описываются в *Electronic Publishing and Chemical Text Processing*, by A.C. Norris and A.L. Oakley, в: *TEX applications, uses, methods*, Malcolm Clark (ed.), Ellis Horwood Publishers, Chichester, England, pp. 207–225, 1990, и в *Chemical Structure Formulae and x/y Diagrams with TEX*, by Michael Ramek, там же, pp. 227–258.

[85] Хофманн (Thomas Hofmann). A LATEX addition for formatting indexes. *TUGboat*, 7(3):186, October 1986.
Описывается `latexindex` — скрипт Bourne Shell для обработки указателей в LATEX'е.

[86] Чжень, Харрисон (Pehong Chen and Michael A. Harrison). Index preparation and processing. *Software—Practice and Experience*, 19(9):897–915, September 1988. The LaTeX text of this paper is included in the `makeindex` software distribution.

В этой статье показано, как можно автоматизировать нудную и длительную процедуру подготовки указателя, в значительной мере независимо от использованной системы набора и формата. Эта идея иллюстрируется на примере системы *MakeIndex*, которая расставляет строки элементов указателя в алфавитном порядке. Дается сравение этой системы с другими средствами создания указателя.

[87] Шрод (Joachim Schrod). The Components of TeX. *TeXline*, 14:7–11, February 1992.

Краткое описание вспомогательных программ и файлов, образующих, в сочетании с TeX'ом, полную систему для полиграфического набора и авторской подготовки публикаций.

[88] Adobe Systems Incorporated. *PostScript Language Tutorial and Cookbook.* Addison-Wesley, Reading, 1985.

Так называемая "синяя книга" представляет собой самый полный учебник по языку PostScript. Состоит из двух частей: доступное для восприятия и усвоения введение в язык и его примитивы и рецептурный раздел, состоящий из 21 программы, показывающей, как PostScript используется в приложениях. Эти программы покрывают большинство общих приложений, с которыми когда-либо вам придется столкнуться, и их можно без изменений включать в ваши пакеты. Основным недостатком книги является то, что в ней описывается только PostScript уровня 1. Более новым и очень хорошим введением в PostScript уровня 2 для пользователей всех уровней является книга *PostScript by Example*, by Henry McGilton and Mary Campione, Addison Wesley, 1992. Здесь изложение начинается тоже с нижнего уровня, но в последних главах излагаются средства уровня 2, такие, как композитные (смешанные) шрифты (composite fonts), растровые структуры (patterns), формы (forms), цвет (color) и полутона. Приводится свыше 500 фрагментов программ в PostScript'е и 750 иллюстраций.

[89] Adobe Systems Incorporated. *PostScript Language Reference Manual.* Addison-Wesley, Reading, second edition, 1990.

Так называемая "красная книга" содержит полное описание языка PostScript. Приводятся все операторы PostScript'а уровня 1 и уровня 2 равно как и все расширения Display PostScript. В приложениях определяются *Document Structuring Conventions* и *Encapsulated PostScript File Format.*

[90] Adobe Systems Incorporated. *Adobe Type 1 Font Format.* Addison-Wesley, Reading, 1990.

Так называемая "черная книга" содержит спецификации для шрифтов Adobe формата Type 1. Включена информация по структуре шрифтовых программ; объясняется, как определяется компьютерное вычерчивание; приводится содержание различных шрифтовых библиотек. Кроме того, демонстрируются подпрограммы, хинты, сообщения о совместимости при использовании Adobe Type Manager.

[91] *Code typographique.* Syndicat national des cadres et maîtrise du livre, de la presse et des industries graphiques, 13e édition, 1954.

Типографские правила набора франкоязычных текстов, используемые во Франции.

[92]* CyrTUG-97. Труды ежегодной международной конференции пользователей кириллического TEX'а. Петергоф (Санкт-Петербург), 8–11 сентября 1997 г.

Содержит работы отечественных авторов по русификации LATEX'а.

[93]* CyrTUG-98. Труды ежегодной международной конференции пользователей кириллического TEX'а. Юдино (Казань), 16–18 сентября 1998 г.

Содержит работы отечественных авторов по русификации LATEX'а.

[94] *Guide du typographe romand.* Association suisse des compositeurs à la machine, 4e édition, 1982.

Типографские правила набора франкоязычных текстов, применяемые швейцарскими наборщиками.

[95] IBM, International Business Machines Corporation. *Font Object Content Architecture: Reference*, New York, December 1988.

В этом руководстве описываются концепции шрифтового обеспечения IBM и его приложения. Дается общая информация о шрифтовой памяти, доступе, ссылках и объясняется понятие информации о начертании литеры. Дается полный перечень параметров, используемых для шрифтов IBM, включая параметры описания шрифта, параметры метрик, параметры начертания литер и схема кодировки.

[96] IBM, International Business Machines Corporation. *Document Composition Facility — General Information Release 4.0 for DCF (GH20-9158-09)*, Boulder, 1990.

Это руководство содержит общую информацию о средствах IBM подготовки документов, которые позволяют форматировать текст: SCRIPT/VS. Эта программа обрабатывает документы, размечая их при помощи собственных командных слов, равно как и при помощи GML — Generalized Markup Language — предшественника SGML. Эта система имеет также (факультативное) средство форматирования математических формул (Script Mathematical Formula Formatter — SMFF).

[97] *Lexique des règles typographiques en usage à l'Imprimerie nationale.* Imprimerie nationale, 3e édition, 1990.

Полиграфические правила Франции, принятые к исполнению в парижской типографии «Imprimerie nationale», в которой печатаются вся официальная правительственная документация и все книги, издаваемые французскими государственными учреждениями.

[98] International Organization for Standardization. 8-bit single-byte coded graphics character sets, parts 1 to 10. ISO 8859, ISO Geneva, 1986–92.

Описание литер из различных алфавитов. Части 1–4, 9 и 10 соответствуют множеству литер, необходимых для кодирования разных групп языков, использующих латинский алфавит, часть 5 отведена под кириллицу, часть 6 соответствует арабским языкам, часть 7 — еврейской группе языков.

[99] International Organization for Standardization. Standard Generalised Markup Language (SGML). ISO 8879, ISO Geneva, 1986.

Это (не очень легкий для чтения) стандарт ISO, описывающий язык SGML во всех технических подробностях. Гораздо проще для восприятия обзорная статья, описывающая роль SGML как первого элемента в цепи стандартов обработки текста в сравнении с DSSSL (форматирование документа), SPDL (представление документа) и

Unicode/ISO 10646 (таблицы кодировок): *Scientific Text Processing*, by Michel Goossens and Eric van Herwijnen, в *International Journal of Modern Physics C*, 3(3):479–546, June 1992.

[100] International Organization for Standardization. Universal Coded Character Set. ISO/IEC 10646, ISO Geneva, 1993.

Этот стандарт утверждает структуру и определения таблиц Universal Multiple-Octet Coded Character Set (UCS). Это стандарт 32-битовой кодировки литер, который можно использовать для кодирования всех существующих в мире систем письменности. В настоящее время были определены только 16-битовые схемы кодировки, и это идентично стандарту Unicode; см. [102]. Краткий обзор этого стандарта и небольшой исторический очерк содержатся в *Untying tongues*, by Michael Y. Ksar, в *iso bulletin*, 24(6):2–8, June 1993.

[101] TEXplorators Corporation, 1572 West Gray, #377, Houston, TX 77019-4948 *Mathtime, PostScript fonts for typesetting mathematics with TEX*, 1993.

Руководство пользователя, объясняющее, как следует устанавливать и использовать PostScript'овские шрифты семейства *Mathtime* (распространяется вместе со шрифтами).

[102] The Unicode Consortium. Unicode 1.0 — Draft Standard. Technical report, The Unicode Consortium, 1990.

Описание стандартной 16-разрядной кодировки, принятой большинством производителей компьютеров. Версия 1.1 кодировки Unicode тождественна базовой 16-разрядной таблице кодировки ISO-10646, см. [100].

[103] *The Chicago Manual of Style.* University of Chicago Press, thirteenth edition, 1982.

Начиная с 1906 года справочник *Manual of Style*, издаваемый в University of Chicago Press, является стандартным справочным пособием для авторов, редакторов и корректоров в США и других странах. Благодаря четкому изложению, многочисленным примерам, подробному указателю и исчерпывающему объяснению разнообразных терминов, *Chicago Manual* предоставляет легкие для понимания и прямые способы подготовки публикаций для большинства американских издательств.

[104]* TUG'96. 17th Annual Meeting of the TEXUsers Group. Dubna, Russia, The Joint Institute for Nuclear Research. July 28–August 2, 1996.

Труды международной конференции пользователей TEX'а TUG'96, впервые проводимой не в англоговорящей стране. Представлено много работ, посвященных кириллице и многоязыковой поддержке.

[105] Y&Y, 106 Indian Hill, Carlisle, MA 01741. *LucidaBright + LucidaNewMath*, 1992.

Руководство пользователя, распространяемое вместе со шрифтами, в котором объясняется, как инсталлировать и использовать шрифты *LucidaBright + LucidaNewMath*.

Именной указатель

Арсено, Дональд (Arseneau, Donald) 67, 72, 153, 174, 419

Баррока, Леонор (Barroca, Leonor) 361
Беллантони, Стефен (Bellantoni, Stephen) 100
Бердников А. С. 9
Берри, Карл (Berry, Karl) 371, 372, 374, 376
Биби, Нельсон (Beebe, Nelson) 422, 442, 443, 535
Бигелоу, Чарльз (Bigelow, Charles) 380
Билавала, Назин (Billawala, Nazeen N.) 211
Блезер, Йоахим (Bleser, Joachim) 324
Боде, Ганс-Герман (Bode, Hans-Hermann) 469
Боденхеймер, Бобби (Bodenheimer, Bobby) 535
Борсо, Фрэнсис (Borceaux, Francis) 271
Брайтенлохнер, Петер (Breitenlohner, Peter) 301
Брамс, Йоханнес (Braams, Johannes) 107, 138, 302, 365, 472

Ван Зандт, Тимоти (Van Zandt, Timothy) 318, 372
Ван Острюм, Пит (Van Oostrum, Piet) 114
Ванруз, Питер (Vanroose, Peter) 322
Васильев К. А. 7
Виноградов М. М. 8
Волович В. В. 7
Волчко, Марио (Wolczko, Mario) 107, 317
Воронин А. В. 8
Вуястик, Доминик (Wujastyk, Dominik) 92

Гамильтон Келли, Брайан (Hamilton Kelly, Brian) 321
Ганелин П. 8
Гилди, Стефен (Gildea, Stephen) 104
Глонти Н. 8
Голдфарб, Чарльз (Goldfarb, Charles) 26
Голль, Бернар (Gaulle, Bernard) 298, 312
Грин, Йен (Green, Ian) 419

Дагган, Ангус (Duggan, Angus) 85, 316
Даррелл, Тревор (Darrell, Trevor) 357
Даунз, Майкл (Downes, Michael) 253
Дейл, Ван (Dale, Van) 466
Дейли, Пэтрик (Daly, Patrick W.) 469
Джеффри, Алан (Jeffrey, Alan) 380
Джонстоун, Адриан (Johnstone, Adrian) 333
Джоунз, Дэвид (Jones, David) 179, 410, 534
Джурафски, Дан (Jurafsky, Dan) 55
Друкбер, Жан-Пьер (Drucbert, Jean-Pierre) 55, 64
Дэйли, Пэтрик (Daly, Patrick W.) 420
Дюшье, Дени (Duchier, Denys) 472

Знаменская Л. Н. 8
Знаменский С. В. 8

Исодзака, Хидеки (Isozaka, Hideki) 344, 348

Йенсен, Франк (Jensen, Frank) 211, 527

Карлайл, Дэвид (Carlisle, David) 46, 64, 68, 76, 109, 124, 133, 136, 137, 142, 147, 151, 168, 173, 529
Кемпсон, Нил (Kempson, Niel) 431
Кларк, Джеймс (Clark, James) 357
Кларк, Малколм (Clark, Malcolm) 315
Кнут, Дональд (Knuth, Donald) 19, 20, 184, 210, 217, 222, 298, 472
Кокрен, Стефен (Cochran, Steven) 176
Колодин М. Ю. 9
Котц, Дэвид (Kotz, David) 440
Кульман, Фолкер (Kuhlmann, Volker) 107
Куок, Конрад (Kwok, Conrad) 340

Лаваньино, Джон (Lavagnino, John) 93
Лами, Жан-Франсуа (Lamy, Jean-François) 107

Лапко О. Г. 9, 10
Левин, Майкл (Levine, Michael) 333
Лейхтер, Джерри (Leichter, Jerry) 156
Линьно, Ансельм (Lingnau, Anselm) 169
Лонг, Ф. У. (Long, F. W.) 410
Львовский С. М. 8, 10, 11, 493, 497
Лэмпорт, Лесли (Lamport, Leslie) 5, 6, 10, 11, 20, 21, 84, 104, 117, 138, 320, 393, 529
Лэнг, Эдмунд (Lang, Edmund) 324

Мак-Коули, Джеймс Даррелл (McCauley, James Darrell) 176
Маклэйн-кросс, И. Л. (Maclaine-cross, I. L.) 327
Макферсон, Кент (McPherson, Kent) 103
Малышев В. К. 6, 301
Маттес, Эберхард (Mattes, Eberhard) 327, 357
Маховая И. А. 10
Маховая О. А. 10
Миттельбах, Франк (Mittelbach, Frank) 7, 21, 60, 94, 98, 124, 166, 182, 253, 290, 472, 485, 535
Мэлло, Джером (Maillot, Jérôme) 370

О'Кейн, Кевин (O'Kane, Kevin) 333
Орландини, Мауро (Orlandini, Mauro) 317

Партль, Губер (Partl, Hubert) 109
Паташник, Орен (Patashnik, Oren) 24, 417, 443
Пейдж, Стефен (Page, Stephen) 71
Петилл, Марк (Purtill, Mark) 182
Подар, Сунил (Podar, Sunil) 333
Попелье, Нико (Poppelier, Nico) 107

Райхль, Бернд (Raichle, Bernd) 303
Ратц, Себастьян (Rahtz, Sebastian) 182, 315, 357, 361, 372
Рейд, Брайен (Reid, Brian) 26
Рейд, Томас (Reid, Thomas J.) 173
Рид, Дэвид (Rhead, David) 466
Роженко А. И. 9
Рокицки, Томаш (Rokicki, Tomas) 109, 354, 357
Роуз, Кристофер (Rose, Kristoffer) 271, 315
Роули, Крис (Rowley, Chris) 7, 21, 92

Самарин, Александр (Samarin, Alexander) 531
Сова, Фридгельм (Sowa, Friedhelm) 71
Спивак, Майкл (Spivak, Michael) 380

Тейлор, Пол (Taylor, Paul) 271
Тейлор, Филипп (Taylor, Philip) 66
Торап, Крестен (Thorup, Kresten) 372, 527
Треворров, Эндрю (Trevorrow, Andrew) 357
Третьяков Н. В. 10
Тюменцев Ю. В. 10

Уинтон, Нил (Winton, Neil) 365
Уорд, Мартин (Ward, Martin) 123
Уорд, Найгел (Ward, Nigel) 55

Файн, Майкл (Fine, Michael) 365
Фернандес, Хосе Альберто (Fernandez, Jose Alberto) 434

Хайткёттер, Йорг (Heitkoetter, Jörg) 468
Хараламбус, Яннис (Haralambous, Yannis) 211, 212, 217, 300
Харрисон, Майкл (Harrison, Michael) 23
Хёниг, Алан (Hoenig, Alan) 71
Ходулёв А. Б. 9
Холмс, Крис (Holmes, Chris) 380
Хос, Росвита (Haas, Roswitha) 333
Хэфнер, Джеймс (Hafner, James) 371

Цапф, Герман (Zapf, Hermann) 210, 212

Чжень, Гуоин (Chen, Guoying) 535, 538
Чжень, Пехон (Chen, Pehong) 23
Чистяков В. В. 10

Шварц, Норберт (Schwarz, Norbert) 210
Шень А. 8
Шопф, Райнер (Schöpf, Rainer) 7, 21, 84, 85, 105, 182, 253, 485, 535
Шпит, Веренфрид (Spit, Werenfried) 466
Шрод, Йоахим (Schrod, Joachim) 21, 90, 438

Эйлер, Леонард (Euler, Leonhard) 212

Юринс, Тео (Jurriens, Theo) 138

Янишевский А. В. 9

Предметный указатель

На страницах, выделенных полужирным шрифтом, содержится важная информация, например, точное определение команды или подробное объяснение; обычные ссылки на страницы набраны светлым шрифтом.

\' **122**
\(**533**
\) **533**
* **480**
\, 265, **280**
\- **122**, 304
  в ulem **67**
\/ **193**
\: **280**
\; **280**
\= **122**
\> **122**
\! **280**
\@ **68**
\@addtoreset **33**, **40**, 90, 279, 501
\@cite 418
\@dotsep **52**
\@dottedtocline **51**, 52
\@evenfoot **112**
\@evenhead **112**
\@idxitem **480**
\@makecaption **178**
\@makefigcaption **178**
\@makefnmark **89**, **90**
\@makefntext **89**, **91**
\@mkboth **113**, **114**
\@oddfoot **112**
\@oddhead **112**
\@pnumwidth, жесткая длина **51**, **52**
\@ptsize **107**
\@rightskip, растяжимая длина **69**
\@seccntformat **41**, **42**
\@startsection **42**, **43**, 44–46, 167
\@starttoc **54**
\@thefnmark **90**
\@tocrmarg **51**, **52**
@(((272
@))) 272
@<<< 272
@>>> 272
@AAA **272**
@VVV **272**
@ литера (character)
  в именах команд (in command names) 33
  делать активной (made active) 18
\‘ **122**
\{ **257**
\\ 46, 69, 123, 127, 130, 134, 138, 139, 271, 277, 278
  внутри колонтитула (within header or footer) 116
* 277
\| **257**
\} **257**
11pt, опция 30
7-битный ввод (7-bit input) 299
8-битный ввод (8-bit input) 299

\a’ **122**
a0paper, опция **104**
A4 *см.* Формат бумаги (paper size)
a4, пакет стилевой **107**, 438
a4dutch, пакет стилевой **107**, **108**
a4paper, опция 30, **104**, 520
a4wide, пакет стилевой **107**
A5 *см.* Формат бумаги (paper size)
a5, пакет стилевой **107**
a5comb, пакет стилевой **107**
a5paper, опция **104**
\a= **122**
\a‘ **122**
AAAA 530, **530**
  both 530
  indexentry 530
  indexonly 530
abbrev, BibTeX’овский стиль **423**, 447
abbrv, BibTeX’овский стиль **423**, **425**, **429**, 459, **469**
\abovedisplayshortskip, растяжимая длина **297**

\abovedisplayskip, растяжимая длина **296**
abstract, BibTeX'овский стиль **423**, 454, 466
abstract, окружение 48, 49
\abstractname **49**, **307**
\accent 235, 236
\accentedsymbol **261**, 282
acm, BibTeX'овский стиль **423**, **425**, **430**
\acro **195**
\actualchar **478**
\acute **255**
\addcontentsline 47, 48, **53**, **54**, 55
\adddialect **305**
\addlanguage **305**
\addto **306**
\addtocontents **53**, **54**
\addtocounter 39, 279, **500**, **527**
\addtolength 80, 91, 105, 129, 275, 296, **503**, **527**
\advance 527
.adx-файл 413
\AE 199
.afm-файл **355**
afm2tfm, программа **355**, 380
\afterpage **168**, 173
afterpage, пакет стилевой **168**, **173**
agsm, BibTeX'овский стиль **423**
\aleph **256**
algorithm, окружение 171
align, окружение 205, **272**, **273**, **274**, 275, 283, 285, 288, 289, 382
align*, окружение **272**, 285, 288, 289
alignat, окружение **272**, **273**, 274, 289, 290
alignat*, окружение **272**, 290
aligned, окружение **276**
alignedat, окружение **276**
\allinethickness **342**, 343
\allowdisplaybreaks **277**
alltt, окружение **84**, **85**, 87
alltt, пакет стилевой **84**
\Alph 40, 47, 75, 76, **501**
\alph 75, 76, **501**
Alph, стиль счетчиков полос (page number style) **111**
alph, стиль счетчиков полос (page number style) **111**
alpha, BibTeX'овский стиль **423**, **425**, **429**, 449, 457, 459, 461, 465, **469**
\alpha 203, **255**
\alsoname **307**
\AltMacroFont **480**
\amalg **255**
american, опция **307**
AMS-символы (AMS symbols) 257–259
amsalpha, BibTeX'овский стиль **423**
amsart, класс документа **283**
amsbook, класс документа **283**
amsbsy, пакет стилевой **207**, 255, **282**
amscd, пакет стилевой 271, 272, **282**
amsfonts, пакет стилевой **213**, 254, 255, **282**
amsintx, пакет стилевой **282**
amsmath, пакет стилевой 252, **253**, 254, 255, 260–266, 268, 271–273, 275, 277–282, **282**, 283, 284, 296
amsopn, пакет стилевой **282**
amsplain, BibTeX'овский стиль **423**
amssymb, пакет стилевой **213**, 254–259, **282**
amstex, пакет стилевой 13, 15, **18**
amstext, пакет стилевой 207, 255, 264, **282**
amsthm, пакет стилевой **282**
amsxtra, пакет стилевой **261**, **262**, **282**
\and **533**
.and-файл 413
\angle **256**, **259**
annotate, BibTeX'овский стиль **423**, 465
annotation, BibTeX'овский стиль **423**, 465
annote, BibTeX'овский стиль **456**
apa, BibTeX'овский стиль **423**
apalike, BibTeX'овский стиль 420, **423–426**, **430**
apalike, пакет стилевой **423**, **426**
apalike2, BibTeX'овский стиль **423**
\Appendix **47**
\appendix **47**, 48
\appendixname **49**, **307**
\approx **256**
\approxeq **258**
\arabic 40, 41, 75, 76, 500, **501**
arabic, стиль счетчиков полос (page number style) **111**
\arc **330**, **331**, 332, **342**, **343**
\arccos **257**
archie, программа **535**
\arcsin **257**
\arctan **257**
\arg **257**
array, окружение 14, **69**, 120, 121, **124**, 126, 127, 129, 131, 132, 134, 137, 147, 153, 156, 269, 273, 296, 511
стилевые параметры (style parameters) 129
array, пакет стилевой 120, **124–132**, **137**, 158, 485, **535**
\arraybackslash **134**
\arraycolsep 273
\arraycolsep, жесткая длина 126, **129**, **296**
\arrayrulewidth 273
\arrayrulewidth, жесткая длина **130**, 152

\arraystretch 124, **130**, 273
\Arrowvert **257**
\arrowvert **257**
art11.clo 31
article, класс документа 16, **23**, 31, 33, 36–38, 40, 49, 90, 105, **115**, 283, 302, 409, 438, 481, 526
\Ask **486**
\ast 77, **255**
AsTeX **539**
astron, BibTeX'овский стиль 420, **423**
\asymp **256**
\AtBeginDocument **518**, **524**
\AtEndDocument **518**, **524**
\AtEndOfClass **518**, **524**, 526
\AtEndOfPackage **518**, **524**
austrian, опция **307**
authordate1, BibTeX'овский стиль 422, **423**, **466**
authordate1-4, пакет стилевой **423**
authordate2, BibTeX'овский стиль 422, **423**
authordate3, BibTeX'овский стиль 422, **423**
authordate4, BibTeX'овский стиль 422, **423**, **466**
.aux-файл **22**, 23, 24, 34, 142, 143, 410, 417–419, 421, 424, 431, 433, 440, 442
aux2bib, программа 422, **440**
avant, пакет стилевой **374**
awk, программа **440**, **442**

b5paper, опция **104**
babel, пакет стилевой 6, 9, 10, 15, 30, 300, **302**, 303–308, 310, 312, 397, **405**, 523
\backepsilon **258**
\backprime **259**
\backsim **258**
\backsimeq **258**
\backslash **257**
\bar **255**, **324**
bar, пакет стилевой **324**, 325
barenv, окружение **324–327**
  стилевые параметры (style parameters) 324–326
\barwedge **259**
\baselineskip, растяжимая длина 70, 71, 90, **105**, **106**, 118, 124, 211, 506
\baselinestretch **70**, **71**
basker, пакет стилевой **374**
\batchfile **488**
\batchinput **486**
\Bbb **242**
\Bbbk **259**
.bbl-файл **22**, 24, 421, 422, 424, 431, 451, 453, 460, 462, 463
bbs, BibTeX'овский стиль **423**
Bcenter, окружение **319**
Bdescription, окружение **319**
\because **258**
\belowdisplayshortskip, растяжимая длина **297**
\belowdisplayskip, растяжимая длина **297**
bembo, пакет стилевой **374**
Benumerate, окружение **319**
Beqnarray, окружение **319**
Beqnarray*, окружение **319**
\beta 203, **255**
\beth **257**
beton, пакет стилевой **211**
\between **258**
\bezier **320**
bezier, пакет стилевой **320**, 330, **333**
\bf 181, **202**
  использование в математических формулах **204**, 215
\bfdefault **200**, **201**
Bflushleft, окружение **319**
Bflushright, окружение **319**
\bfseries 43, 67, 77, 124, **192**, 196, 198, **200–203**, 212, 214, 294
  использование в математических формулах **203**, **205**
.bib-файл **22**, 24, 421, 422, 440, 442, 443, 445, 450, 453, 454, 457, 460
bibclean, программа **442**
bibextract, программа **442**
\bibhang, жесткая длина **420**
\bibindent, жесткая длина **420**
\bibitem **417**, **418**, 420, 457, 460
bibkey, программа **440**
\bibliography **425**, 431, 434, 438, 451
\bibliographystyle 424, **425**, 431, 433, 434, 438
\bibliographyunit **434**
biblist, пакет стилевой **438**, **439**
bibmods, пакет стилевой **443**, **444**
\bibname **49**, **307**
\bibpunct **420**
bibsort.sh, программа **442**
BibTeX 24, 416–471
  аббревиатуры (abbreviation) 450
  акцентированные символы (accented characters) 449
  база данных (data base) 421, 422
  встроенные функции (built-in function) 458
  гарвардская схема, ключевое слово (Harvard system, keyword) 422
  запись в базе данных (entry) 443

имена переменных (variable name) 458
имена функций (function name) 458
имя, содержащие дефис (hyphenated name) 447
ключевое слово (keyword) 421
комментарии (comments) 457
публикации нескольких авторов (multiple authors) 448
семейство адаптирующихся стилей (adaptable style family) 469
система Delphi (Delphi system) 468
специальный символ (special character) 449
стилевой файл (style file) 421, 422
типы переменных 458
формат имени (name format) 447
цепочка литер постоянная (string constant) 458
чувствительность к регистру (case sensitivity) 458, 459
BibTeX'овский стиль 423–424
abbrev **423**, 447
abbrv **423**, **425**, **429**, 459, **469**
abstract **423**, 454, 466
acm **423**, **425**, **430**
agsm **423**
alpha **423**, **425**, **429**, 449, 457, 459, 461, 465, **469**
amsalpha **423**
amsplain **423**
annotate **423**, 465
annotation **423**, 465
annote **456**
apalike2 **423**
apalike 420, **423–426**, **430**
apa **423**
astron 420, **423**
authordate1 422, **423**, **466**
authordate2 422, **423**
authordate3 422, **423**
authordate4 422, **423**, **466**
bbs **423**
cbe **423**
cell **423**
chicago 420, **466**
dcu **423**
fabbrv **469**
falpha **469**
fplain **469**
funsrt **469**
gabbrv **469**
galpha **469**
gplain **469**
gunsrt **469**
harvard 420, 466
humanbio **423**
humannat **423**
ieeetr **423**
is-abbrv **423**, 466
is-alpha **423**, 466
is-plain **423**, 466
is-unsrt **423**, 466
jmb 422, **423**
jtb **423**
kluwer **423**
myunsrt 464
named 420, 422, **423**
namunsrt **423**
nar **423**
natbib **420**, **424**
nature **424**
nederlands **466**, 469
newapa 420, **424**
phaip **424**
phcpc **424**
phiaea **424**
phjcp **424**
phnflet **424**
phnf **424**
phpf **424**
phppcf **424**
phreport **424**
phrmp **424**
plainyr **424**
plain **423–425**, **428**, 459, 461, 465, **469**
siam **424**
unsrt **423–425**, **428**, 459, 464, **469**
bibunits, пакет стилевой **431**, **434–438**
bibunit, окружение **434**, 435
bibview, программа **443**
\Big **268**
\big **268**
\bigcap **256**
\bigcirc **255**
\bigcircle **330**
\bigcup **256**
\Bigg **268**
\bigg **268**
\biggl 284, 286
\Biggr **268**
\biggr 284, 286
\bigl **268**
\Bigm **268**
\bigodot **256**
\bigoplus **256**, 281
\bigotimes **256**, 281
\bigskip **506**
\bigskipamount, растяжимая длина 145, **506**
\bigsqcup **256**
\bigstar **259**
\bigstrutjot, жесткая длина **156**
\bigtriangledown **255**

\bigtriangleup **255**
\biguplus **256**
\bigvee **256**
\bigwedge **256**
bind, опция 525
\binom **266**, 267, 382
bipartite, окружение **344**
Bitemize, окружение **319**
bk11.clo 31
\blacklozenge **259**
\blacksquare **259**
\blacktriangle **259**
\blacktriangledown **259**
\blacktriangleleft **258**
\blacktriangleright **258**
.blg-файл **22**, 24, 421
BM2FONT, программа 8, **316**
bmatrix, окружение **269**
\bmod **266**
\boldmath **207**, 254
\boldsymbol 207, 253, **255**, 282
book, класс документа **23**, 31, 38, 40, 43, 55, 90, 105, 112, **115**, 169, 283, 302, 409, 495, 500, 517, 522
bookman, пакет стилевой **374**
\boolean 526, **531**
\bot **256**
\botfigrule **165**
\bottomcaption **139**
\bottomfraction **163**, 166, 167
bottomnumber, счетчик **163**
\bowtie **256**
\Box **213**, **256**
\boxdot **259**
\boxed **262**
boxedminipage, окружение **317**, 318
boxedminipage, пакет стилевой **317**
boxedverbatim, окружение **86**
boxitpara, окружение **370**
\boxminus **259**
\boxplus **259**
\boxtimes **259**
bp, большой пункт (big point) **504**
\bracevert **257**
\branch **322**
\branchlabels **322**
brazil, опция **307**
\breve **255**
\brush **347**
\bslash **477**
.bst-файл **22**, 24, 421, 422, 469
\bullet 77, **255**
\Bumpeq **258**
\bumpeq **258**
bundle, окружение **348**, 349
\bye **217**
Cahiers Gutenberg 443
calc, пакет стилевой 15, 76, 81, 106, 130, 492, **500**, 516, 520, **527**, 528, 529
\Cap **259**
\cap **255**
\caption 16, 50, 54, 59, 69, 142, **144–146**, 171, **178**, 179, 365
\caption* **144–146**
\captions*lang* **305**
\captionwidth, жесткая длина **179**
cases, окружение **269**
catalan, опция **307**
\catcode **197**
\cbdelete **368**
cbe, BibTeX'овский стиль **423**
\cbend **368**, 369, 370
\cbstart **368**
cc, цицеро (cicero) **504**
\ccname **307**
CD, окружение **272**, 282
\cdot 77, 150, **255**
\cdots **256**
cell, BibTeX'овский стиль **423**
center, окружение 59, **69**, **80**, 319, 359
\centerdot **259**
\centering 45, **69**, 134, 157
centertags, опция 276, **281**, 284
\cfoot **115**
\cfrac 268
changebar, окружение **368**
changebar, пакет стилевой 6, **356**, **357**, **365**, 365–370
changebargrey, счетчик **369**
\changebarsep **369**
\changebarwidth 368, **369**
\changes **476**, **480**
\chapter 36, 37, **38**, 39, 46, 48, 55, 58, 59, 110, 113, 115, 116, 434
chapter, счетчик **39**, 112, **499**, 529
нумерация внутри частей 40
\chapter* 113
chapterbib, пакет стилевой **431–435**, **438**
\chaptermark **112**, 113, 118
\chaptername **49**, **307**
\CharacterTable **477**
\chead **115**
\check **255**
\CheckModules **477**
\CheckSum **477**
\chi **255**
chicago, BibTeX'овский стиль 420, **466**
chicago, пакет стилевой **419**
\chunk **348**
\circ **255**
\circeq **258**
\circle 320, 321, **341**, 342
\circle* 320, 321, 333, **341**, 342

\circlearrowleft **258**
\circlearrowright **258**
\circledast **259**
\circledcirc **259**
\circleddash **259**
\circledS **259**
Citation, окружение 496, 497
\cite **417**, **418–420**, **421**, **425**, 431, 445, 452, 453, 456, 462
cite, пакет стилевой **419**
\citeA **419**
citefind, программа **442**
\citeN **419**
\citen **419**
\citeNP **419**
citesort, пакет стилевой **419**
citetags, программа **442**
\citeyear **419**
\cleardoublepage **111**
\clearemptydoublepage **111**
\clearpage 109, **111**, **117**, 142, 163, 168, 173
\cline 130
.clo-файл **22**, 30
\closecurve **330**
.cls-файл 21, **22**, 30
\clubsuit **256**
cm, сантиметр (centimeter) **504**
\CodelineFont **223**, **224**
\CodelineIndex **475**, **477**, 482
\CodelineNumbered **477**
collectmore, счетчик 97, **98**
color, пакет стилевой **371**
colordvi, пакет стилевой **371**
columnbadness, счетчик **97**
\columnsep, жесткая длина 96, **97**, **102**, **105**, 175
\columnseprule, жесткая длина **96**, **97**, **102**, **105**
\columnwidth, жесткая длина **91**, **102**
comment, окружение **84**, **100**
\complement **259**
concrete, пакет стилевой 207, **211**
\cong **256**
\contentsline **50**, **51**, 52, 53
\contentsname **49**, **307**
\coprod **256**, 281
\cornersize **318**
\cornersize* **318**
Cor, окружение 293
\cos **257**
\cosh **257**
\cot **257**
\coth **257**
croatian, опция **307**
cropmarks, опция 525
\csc **257**
\csname 41
$\mathcal{C}_S$TUG **539**
CTAN (Всеобъемлющий сетевой TEX-архив (Comprehensive TEX Archive Network)) 534
\Cup **259**
\cup **255**
\curlyeqprec **258**
\curlyeqsucc **258**
\curlyvee **259**
\curlywedge **259**
\CurrentOption **518**, **521**, **522**, 526
\curve **330**, **331**, 333
\curvearrowleft **258**
\curvearrowright **258**
\curvedashes 333
curves, пакет стилевой **327**
curvesls, пакет стилевой **327**
\curvesymbol **332**, 333
CyrTUG (*Cyrillic TEX Users Group*) 10, **539**
czech, опция **307**

\dagger **255**
\daleth **257**
danish, опция **307**
DANTE e.V. 21, 303, **539**
dashjoin, окружение **339**, 341, 344, 346
\dashleftarrow **258**
\dashline **337**, **338**, 339, 341, 345
\dashlinestretch **338**, **339**
\dashrightarrow **258**
\dashv **256**
\date*lang* **305**
\dbinom **266**, 267, 382
\dblfigrule **165**
\dblfloatpagefraction **164**
\dblfloatsep, растяжимая длина **164**
\dbltextfloatsep, растяжимая длина **164**
\dbltopfraction **164**
dbltopnumber, счетчик **163**
dcolumn, пакет стилевой 121, **147**
dcu, BIBTEX'овский стиль **423**
dd, пункт Дидо (Didôt point) **504**
\ddagger **255**
\ddddot **262**
\dddot **262**
\ddot **255**, **262**
\ddots **256**
debugshow, опция **216**
\DeclareErrorFont **246**, **250**
\DeclareFixedFont **223**
\DeclareFontEncoding **222**, **236**, 239, 243, **245**, **247**
\DeclareFontFamily **226**, **232**, **233**, **235**, 237, **243**, **247**

\DeclareFontShape **226–229**, **232**, **233**, **235**, 237, **243**, **247–249**, 251, 379
пробелы в **227**
\DeclareFontSubstitution **236**, **250**
\DeclareMathAlphabet **205**, **206**, **208**, **242**, 244, 246, **249**
\DeclareMathOperator **265**, 272, 282
\DeclareMathOperator* **265**
\DeclareMathSizes 221, **238**
\DeclareMathSymbol 205, **240–242**, 244, **246**, 250, 254, 259
\DeclareMathVersion **242**, 244, 250
\DeclareOption **518**, **520**, **521**, 522, **523**, 526
\DeclareOption* **518**, **519**, **521–523**, **526**
\DeclareSymbolFont **239–241**, 242, 244, 250
\DeclareSymbolFontAlphabet **242**, 244, 246, **249**
Def, окружение 294
\defaulthyphenchar **233**
deflist, окружение **498**, 499
\deg **257**
delarray, пакет стилевой 121, **137**
\deletebarwidth **368**, **369**
\DeleteShortVerb **476**, **477**
Delphi (многоязычная BibTeX'овская система(BibTeX multi-language system)) 468
\Delta **255**
\delta **255**
\depth, жесткая длина **508**, **510**
\DescribeEnv **474**, **475**, **477**
\DescribeMacro **474**, **475**, **477**
description, окружение 65, 74, **78**, 319, 497, 498
\descriptionlabel **78**
\det **257**, 281
\dfrac **266**, 267
\diagdown **259**
\diagup **259**
\Diamond **213**, **256**
\diamond **255**
\diamondpar **73**
\diamondsuit **256**
\digamma **257**
\dim **257**
\ding **375**
dingautolist, окружение 76, **376**, 377
\dingfill **376**, 377
\dingline **376**, 377
dinglist, окружение **376**, 377
\DisableCrossrefs **475**, **477**, 482
\displaybreak **277**
\displaystyle 238, 267, **294**, 296
\div **255**
\divide 528
\divideontimes **259**
DK-TUG **539**
doc, пакет стилевой 15, **223**, **472–485**
стилевые параметры (style parameters) 480
.doc-файл **487**
\docdate **480**
\DocInput **477**, **482**
DOCSTRIP 469, 485–487
docstrip.cmd 485
docstrip.tex 485, 488
\DocstyleParms **480**
document, окружение 31, 32, 109, 244, 394, 518, **524**
исполняемый код **524**
\documentclass **13**, 27, **30–32**, 35, 98, 209, 210, 211, 215, 237, 281, 283, 302, 356, **519**, 522, 523
\documentstyle **13**, **31**, **215**
\dominitoc **58**
\DoNotIndex **478**
\DontCheckModules **477**
\dot **255**, **262**
\doteq **256**
\doteqdot **258**
\dotfill 406, **505**
\dotplus **259**
\dots **261**
\dotsb **261**
\dotsc **261**
\dotsi **261**
\dotsm **261**
dottedjoin, окружение **339**, 341, 344, 346
\dottedline **337**, 339, 341, 345
\doublebarwedge **259**
\doublebox **318**
\doublerulesep 273
\doublerulesep, жесткая длина **129**, **130**
doublespace, пакет стилевой **70**, **71**
\Downarrow **256**, **257**
\downarrow **256**, **257**
\downdownarrows **258**
\downharpoonleft **258**
\downharpoonright **258**
\dq **304**
draft, опция **359**
draftcopy, пакет стилевой **371**
drawjoin, окружение **339**, 341
\drawline **338**, **339**, 341–343, 345
\drawlinestretch **338**, **339**
\drawwith **348**, **349**
\driver **357**, **367**
.dtx-файл **22**, 482, **487**, **488**, 489, 490, **491**
dutch, опция **307**
.dvi-файл **22**, 23, 24, 113, 142, 152, 180, 315, 404, 440

dvi-драйвер (dvi driver) 23, 152, 316
поле (margin) 104, 106
dvicopy, программа **301**
dvips, программа 8, 109, 301, 340, 350, **354–357**, **365**, **371**, **372**, **380**, 535, 549, 590
dvips, опция **357**
dvitops, программа **355**, **357**
dvitops, опция **357**

\ecaption **54**
eclbip, пакет стилевой **344**
ecltree, пакет стилевой **348**
\EdgeLabelSep **349**
eepic, пакет стилевой 12, 15, 316, 317, **340–344**, 345, 346
eepicemu, пакет стилевой **343**
εφτ (The Greek TEX Friends Group) **539**
egrep, программа **440**
\ell **256**
\ellipse **342**
\ellipse* **342**
\em **195**, 198
выделение посредством подчеркивания (producing underline) **67**
em, em-промежуток (em-space) **504**
\emergencystretch, жесткая длина **69**, **70**
\emph 67, 181, **195**, **198**
выделение посредством подчеркивания (producing underline) **67**
empty *см.* Размерная функция (size function)
empty, стиль полосы (pagestyle) **110**
\emptyset **256**
emTeX, программа 8, 9, **357**, **535**
emtex, опция **356**, **357**
\EnableCrossrefs **475**, **477**
\encapchar **479**
\enclname **307**
\encodingdefault **200**, **201**, 223, **224**
\endcsname 41
\endfirsthead **144**, **145**
endfloat, пакет стилевой **176**, **177**
\endfoot **144**, **145**
\endhead **144**, **145**
\endlastfoot **144**, **145**
\endnote **93**
endnotes, пакет стилевой **93**
\endpostamble **486**
\endpreamble **486**, 488
english, опция **307**
\enlargethispage **118**
\enlargethispage* 118, **119**
\enotesize **93**
\enspace **505**
\ensuremath **494**
.ent-файл **93**
entry, окружение 80, **81**, 82, 83
\entrylabel **81**, 82, 83
enumerate, окружение 16, 59, 65, **74–77**, 319
стилевые параметры (style parameters) 75
enumerate, пакет стилевой **76**
enumeration, окружение 502
enumi, счетчик **75**, 499
enumii, счетчик **75**, 499
enumiii, счетчик **75**, 499
enumiv, счетчик **75**, 499
environment, окружение **474**, 478
epic, пакет стилевой 12, 15, 316, 317, **333–340**, 340–348
\epsfig **357**, **358**, 359, 364
epsfig, пакет стилевой **356**, **357**, 357–359
\epsilon **255**
\eqcirc **258**
eqnarray, окружение **272**, **273**, 277, 280, 296, 319
eqnarray*, окружение **273**, 296, 319
\eqref **279**
\eqslantgtr **258**
\eqslantless **258**
\equal **529**
equation, окружение 59, **272**, **273**, 275, 278, 280, 283, 284, 296, 382
equation, счетчик **279**, **499**
equation*, окружение **272**, 280, 283
\equiv **256**
errorshow, опция **216**
esperanto, опция **307**
Estonian TEX Users Group **539**
\eta **255**
\eth **259**
eucal, пакет стилевой **282**
\EuFrak **213**
eufrak, пакет стилевой **213**, **282**
euler, пакет стилевой **212**, **213**, 240
\EuScript **213**
euscript, пакет стилевой **213**
\evensidemargin, жесткая длина **102**, **104**, **105**, **108**, 525
ex *см.* x-высота (x-height)
Exa, окружение 293
\excludeversion **100**
\ExecuteOptions **518**, **522**
executivepaper, опция **104**
\exists **256**
\exp **257**
exscale, пакет стилевой **214**
\externaldocument **64**
\extracolsep 132, 146, **154**
\extra*lang* **305**

\extrarowheight, жесткая длина 124, **130**
\extratabsurround, жесткая длина **158**

fabbrv, BibTeX'овский стиль **469**
\faketableofcontents **58**
\fallingdotseq **258**
falpha, BibTeX'овский стиль **469**
\familydefault **200**, **201**, 223
fancy, стиль полосы (pagestyle) **114**, **115**, 116
fancybox, пакет стилевой **318**, **319**
fancyheadings, пакет стилевой **114**, **115**, **117**
  стилевые параметры (style parameters) 116
\fancyplain **116**, 118
fancyplain, стиль полосы (pagestyle) **116**
\fbox 262, 317, 318, 370, **508**, 509, 515–517
\fboxrule, жесткая длина 317, 318, **509**
\fboxsep, жесткая длина 317, 318, 370, **509**
.fd-файл **22**, 23, 235, **237**, **238**, 239, 251
.fff-файл **177**
figure, окружение 54, 59, 71, 94, 146, 162, 170, 173, 178, 365, 511
  метки (labels) в 59
  стилевые параметры (style parameters) 163–165
figure, счетчик **499**
\figurename **307**
\figureplace **177**
figwindow, окружение **71**, **72**
filecontents, окружение **35**
\filedate **480**
\filename **480**
\fileversion **480**
\fill, растяжимая длина 145, 146, 497, **502**, **503**, **505**, **506**
\filltype **342**
finalcolumnbadness, счетчик **97**
final, опция **359**
\Finale **475**, **476**, **478**
finnish, опция **307**
\Finv **259**
\firsthline **158**
fixed *см.* Размерная функция (size function)
flafter, пакет стилевой 61, **166**
flalign, окружение **272**, 274, 290
flalign*, окружение **272**
\flat **256**
fleqn, опция 275, **281**, **296**, **297**
float, пакет стилевой **168**, **169**, 171, **172**, **173**
floatfig, пакет стилевой **173**, **174**
floatingfigure, окружение **173**, 174
\floatname **170**, 171
\floatpagefraction **163**, **164**, 165
\floatplacement **170**
\floatsep, растяжимая длина **164**
\floatstyle **169**, **170**, 171
\flq **303**
\flqq **303**
\flushbottom **119**
\flushcolumns **96**
flushleft, окружение **69**, **80**, 319
flushright, окружение **69**, **80**, 319
\fminilength, жесткая длина 517, 528
fminipage, окружение **516**, 517
.fmt-файл **22**, 23
fnpara, пакет стилевой **92**
\fnsymbol **501**
fontdef.ltx 237, 238
\fontdimen 45, 52, **234**, **235**, 243
\fontencoding **218**, **222**, **236**, **247**
\fontfamily **218**, **219**, **225**
\fontseries 193, **218–220**
\fontshape **220**
\fontsize **218**, 223, 245
\footnote 59, 69, **89**, **90**, 91
  стилевые параметры (style parameters) 88–92
footnote, счетчик **89**, **90**, **499**
\footnotemark **89**, **90**, 91, 153
\footnoterule **89**, **91**, 164
\footnotesep, жесткая длина **89**, **90**
\footnotesize **90**, **196**
\footnotetext **90**, 91
footnpag, пакет стилевой **90**
\footrule **115**
\footrulewidth, жесткая длина **115**
\footskip, растяжимая длина **102**, **105**, 116
\forall **256**
.fot-файл **90**
fplain, BibTeX'овский стиль **469**
\frac **257**, **266**, 267
\fracwithdelims 382
\frakfamily **211**
\framebox 321, 336, **508**, 509, 510, 515
FrameVerb, окружение **319**
francais, опция **307**
french, опция 30, **304**, **307**, **310**
french, пакет стилевой 18, **298**, **312–314**
\from **486**, 488
\frown **256**
\frq **303**
\frqq **303**
ftnright, пакет стилевой 94, **98**, 99, 485, **535**
ftp (file transfer protocol) 535
\fullref 60, **64**
funsrt, BibTeX'овский стиль **469**
\fussy **70**

gabbrv, BibTeX'овский стиль **469**
galician, опция **307**
galpha, BibTeX'овский стиль **469**
\Game **259**
\Gamma **255**
\gamma **255**
\GapDepth **349**
\GapWidth **349**
garamond, пакет стилевой **374**
gather, окружение **272**, **273**, 275, 280, 287–289, 382
gather*, окружение **272**
gathered, окружение **276**
\gcd **257**
gen *см.* Размерная функция (size function)
\generateFile **486**, **487**, 488
\genfrac **267**
\geq **256**
\geqq **258**
\geqslant **258**
german, опция 30, 32, **302–304**, **307**, **308**, 397, 405, 438
german, пакет стилевой **303**, 397
germanb, опция **302**, **307**
\gg **256**
\ggg **258**
ghostscript, программа **351**
ghostview, программа **351**, 355
\gimel **257**
\glossary **408**, 476
\glossaryentry **408**
\GlossaryMin, жесткая длина **480**
\GlossaryParms **480**
\GlossaryPrologue **480**
\glq **303**
\glqq **303**
\gnapprox **259**
\gneq **259**
\gneqq **259**
\gnsim **259**
gopher, программа **535**
\gothfamily **211**
gpic, программа **340**, **341**
gplain, BibTeX'овский стиль **469**
graphics, пакет стилевой **356**
\grave **255**
\grid **336**
\grq **303**
\grqq **303**
Grupo de Utilizadores de TeX **539**
\gtrapprox **258**
\gtrdot **258**
\gtreqless **258**
\gtreqqless **258**
\gtrless **258**
\gtrsim **258**
gunsrt, BibTeX'овский стиль **469**
GUST **539**
GUTenberg **539**
\gvertneqq **259**
gzip, программа **536**

hackalloc, пакет стилевой **17**
hangcaption, пакет стилевой **179**
harvard, BibTeX'овский стиль 420, 466
harvard, пакет стилевой **423**
\hat **255**
\hbar **256**, **259**
\hdotsfor **270**
\headheight, жесткая длина **102**, **105**, **106**, 116
headings, стиль полосы (pagestyle) **110**, 113
\headrule **115**
\headrulewidth, жесткая длина **115**
\headsep, жесткая длина **102**, **105**, **106**
\headtoname **307**
\headwidth, жесткая длина **116**, 118
\heartpar **73**
\heartsuit **256**
\height, жесткая длина **508**, **510**, 514
\help **217**
helvet, пакет стилевой **374**
here, пакет стилевой **173**
\hfill **505**
\hhline **151**, 152
hhline, пакет стилевой 121, **151**
\hline 130, 139, 151, 152, 158
\hlineon **324**
\hoffset, жесткая длина **104**, **108**
\hom **257**
\hookleftarrow **256**
\hookrightarrow **256**
\hrule 45, 152
\hrulefill **505**
\hslash **259**
\hspace 83, 92, 152, **504**, **505**, 508
\hspace* **504**
\Huge **196**
\huge **196**
humanbio, BibTeX'овский стиль **423**
humannat, BibTeX'овский стиль **423**
Hungarian TeX Users Group **539**

\idotsint 260
.idx-файл **22**, 23, 386–388, 391, 392, 395, 398, 399, 408, 410, 412, 530, 531, 590
ieeetr, BibTeX'овский стиль **423**
\IfFileExists **518**, **524**, **525**
\iflanguage **303**
ifthen, пакет стилевой 15, 492, **529**, 531
\ifthenelse 418, 526, **529**, 531

\ignorespaces 80
\iiiint **260**, 382
\iiint **260**, 382
\iint **260**
.ilg-файл **22**, 24, 395
\Im **256**
\imath **256**
in, дюйм (inch) **504**
\in **256**
\include **34**, 412, 431, 438
\includeonly **34**, 412
\includeversion **100**
.ind-файл **22**, 23, 386, 395, 398, 410
indentfirst, пакет стилевой **46**
\index 385–388, 390–394, 396, 398, 399, 406, 408, **410**, 412
index, пакет стилевой **410–415**
\indexentry 387, **401**, 408
\IndexInput **477**, **482**
\IndexMin, жесткая длина **479**
\indexname **49**, **307**
\IndexParms **479**
\IndexPrologue **479**
\indexproofstyle 412
INDEXSTYLE системная переменная (system variable) 398
\inf **257**, 281
infoshow, опция **216**, 521
\infty **256**
\init **217**
\initfloatingfigs 174
\input 237, 524
в DOCSTRIP-файле 486
inputenc, пакет стилевой 9
\InputIfFileExists **519**, 522, **524**, **525**
\int **256**
\intercal **259**
Internet 534
\intertext **277**
\intextsep, растяжимая длина **164**, 175
intlimits, опция **281**
\iota **255**
is-abbrv, BibTeX'овский стиль **423**, 466
is-alpha, BibTeX'овский стиль **423**, 466
is-plain, BibTeX'овский стиль **423**, 466
is-unsrt, BibTeX'овский стиль **423**, 466
\isodd **532**, **533**
isolatin1, пакет стилевой **300**
.ist-файл **22**, 23
\isucaption **179**
\it **202**, 215
использование в математических формулах **204**, 215
italian, опция **302**, **307**
ITALIC **540**
\itdefault **200**
\item 77–81, 409, 497, **532**
в theindex **409**
\itemindent, жесткая длина **79**
itemize, окружение 65, 74, **77**, **78**, 319, 376
стилевые параметры (style parameters) 77
\itemsep, растяжимая длина **79**
\itshape 67, **193**, **195**, 198, **200**, 212, 215, 294
использование в математических формулах **203**, **205**

Japan TeX Users Group **540**
\jmath **256**
jmb, BibTeX'овский стиль 422, **423**
jmb, пакет стилевой **423**
\Join **213**, **256**
\jot, жесткая длина 156, **296**
\jput **339**, 344, 346
jtb, BibTeX'овский стиль **423**

\kappa **255**
\keepsilent **486**
\ker **257**
\kill **145**
kluwer, BibTeX'овский стиль **423**

\l@chapter **52**
\l@example **55**
\l@figure **52**
\l@paragraph **52**
\l@part **52**
\l@section **52**
\l@subparagraph **52**
\l@subsection **52**
\l@subsubsection **52**
\l@table **52**
\label 41, **58–60**, 64, 77, 91, 146, 176, 278, 279
\labelenumi **75**
\labelenumii **75**
\labelenumiii **75**
\labelenumiv **75**
\labelitemi **77**, **78**
\labelitemii **77**
\labelitemiii **77**
\labelitemiv **77**
\labelsep, жесткая длина 78, **79**, 122
\labelwidth, жесткая длина **79**, 81–83
\Lambda **255**
\lambda **255**
\landscape **109**
landscape, окружение **109**
\langle **257**, 265
\LARGE **196**
\Large 43, 70, **196**

\large **196**
\lasthline **158**
LaTeX-символы 255–257
latex.tex-файл 482, 502
latexsym, пакет стилевой **213**, 255, 256
Latin-1 *см.* Стандарт ISO-8859
\layout **103**
layout, пакет стилевой **103**
\lceil **257**
\ldots **256**, 314
\le **256**
\leadsto **213**
\leaf **323**
\left 268, 284, 285, 288
\Leftarrow **256**
\leftarrow **256**
\leftarrowtail **258**
\leftharpoondown **256**
\leftharpoonup **256**
\leftleftarrows **258**
\leftmargin, жесткая длина **79**, 81
\leftmark **113**, 117, 118
\leftnode **344**, **347**
\Leftrightarrow **256**
\leftrightarrow **256**
\leftrightarrows **258**
\leftrightharpoons **258**
\leftrightsquigarrow **258**
\leftroot **262**
\leftthreetimes **259**
legalpaper, опция **104**
\legend **324**
Lem, окружение 294
\lengthtest **531**
Lentry, окружение **82**
\Lentrylabel **82**
\leq **256**
leqno, опция 278, **281**, **296**
\leqq **258**
\leqslant **258**
\lessapprox **258**
\lessdot **240**, **258**
\lesseqgtr **258**
\lesseqqgtr **258**
\lessgtr **258**
\lesssim **258**
letter, класс документа **23**, 37, 302
letterpaper, опция **104**, **105**
\letterspace **66**, **67**
letterspace, пакет стилевой **66**
\levelchar **479**
\lfloor **257**
\lfoot **115**
\lg **257**
\lgroup **257**
\lhd **213**, **255**
\lhead **115**, 116, 118
Lietovos TEX'o Vartotojø Grupë **540**
\lim **257**, 265, 281, 282
\liminf **257**
\limsup **257**
\line 320–322, 332, 334, 335, **341**, 343
\linebreak 495
\linethickness 320–322, 332, 337, 339, 343, 346
\linewidth, жесткая длина 52, 96, **102**, 130, 516
.lis-файл **22**, 23
.list-файл **22**
list, окружение 65, **69**, **78**, **80**, 83, 91, 175
стилевые параметры (style parameters) 79
\listfigurename **49**, **307**
\listfiles **35**, 525
listing, окружение **87**
listing*, окружение **87**
listingcont, окружение **87**, 88
listingcont*, окружение **87**
\listinginput **88**
\listof 55, **170**, **171**
\listofexamples **54**, **55**
\listoffigures 36, **50**, 54, 113, 171, 173
\listoftables 36, **50**, 54, 113, 142, 171
\listparindent, жесткая длина **79**
\listtablename **49**, **307**
\ll **256**
\llcorner **257**
\Lleftarrow **258**
\lll **258**
\lmoustache **257**
\ln **257**
ln, опция **357**
\lnapprox **259**
\lneq **259**
\lneqq **259**
\lnsim **259**
\LoadClass **519**, **525**, **526**
loading, опция **216**
.lof-файл **22**, 23, 50, 53
\log 243, **257**, 265
.log-файл **22**, 23, 142, 397, 404
\Longleftarrow **256**
\longleftarrow **256**
\Longleftrightarrow **256**
\longleftrightarrow **256**
\longmapsto **256**
\longpage **118**
\Longrightarrow **256**
\longrightarrow **256**
longtable, окружение 14, 120, 122, **142**, 143, **144**, 145–147, 149, 153, 168
стилевые параметры (style parameters) **145**

longtable, пакет стилевой **138**, **142**, **143**
looktex, программа **440**
\looparrowleft **258**
\looparrowright **258**
.lot-файл **22**, 23, 50, 53
\lowercase **194**
\lozenge **259**
lrbox, окружение **516**, 517
\lrcorner **257**
lscape, пакет стилевой **109**
\Lsh **258**
\LTcapwidth, жесткая длина 144, **145**
\LTchunksize 144, **145**
\ltimes **259**
\LTleft, растяжимая длина **145**, 146
\LTpost, растяжимая длина **145**
\LTpre, растяжимая длина **145**
\LTright, растяжимая длина **145**, 146
.ltx-файл **22**
lucid, пакет стилевой **374**
LucidaBright 380
LucidaNewmath 380
lucidbrb, пакет стилевой 192, **374**, **380**
lucidbry, пакет стилевой **374**, **380**
lucmath, пакет стилевой **374**
\lVert 265
\lvert 265
\lvertneqq **259**

\m@th **77**
macro, окружение **474**, 475, 478, 480, 481
macrocode, окружение **473**, 474–476, 478, 481
macrocode*, окружение 474, 475, 478
\MacrocodeTopsep, растяжимая длина **481**
\MacroFont **480**
\MacroIndent, жесткая длина **480**
\MacroTopsep, растяжимая длина **480**
magyar, опция **307**
Mail-сервер, почтовый сервер (mail server) 538
\main **479**
\makeatletter **33**, **34**, 40
\makeatother **33**, **34**, 40
makebib, программа **440**
\makebox 320, 321, 336, **508**, 510, 514, 515
makebst, программа 420, 422, 469, 470
\makeglossary **408**
makeidx, пакет стилевой **391**, **394**, 396
MakeIndex 4, 12, 15, 22–24, 385–392, 394–402, 404–406, 408, 410, 412, 413, 475, 476, 482, 547, 552, 590
\makeindex 394, **410**
\makelabel **81**
\MakePrivateLetters **481**
\MakeShortVerb **476**, **478**
\maketitle 36, 110
\mapsto **256**
\marginlabel **92**
\marginpar 70, **92**, 511
в multicols 98
стилевые параметры (style parameters) 93
\marginparpush, жесткая длина **93**, **102**, **105**
\marginparsep, жесткая длина **93**, **102**, **105**, 116, 118
\marginparwidth, жесткая длина **93**, **102**, **105**, 116, 118
\markboth **113**, 115
\markright **113**, 115
\match **344**, **347**
\mathalpha **240**
\mathbb 254, **255**, 282
\mathbf **204**, 207, 253
\mathbin **240**
\mathcal **204**, 254, 282
\mathclose **240**
\mathfrak 254, **255**, 282
\mathindent, жесткая длина **296**
\mathit **204**
\mathnormal **204**, **205**
\mathop **240**, 254
\mathopen **240**
\mathord **240**
\mathpunct **240**
\mathrel **240**
\mathrm **204**, **205**, 254
\mathsf **204**, 206, **208**
\mathsfsl **205**, **208**
\mathsurround, жесткая длина **77**
\mathtt **204**
\mathversion **207**, **208**, 249
\matrix 252
matrix, окружение **269**
\matrixput **336**
\max 243, **257**, 281
MaxMatrixCols, счетчик **269**, **270**
\maxovaldiam **341**
\mbox 67, 92, 264, 494, **508**, 515
\mddefault **200**
\mdseries **192**, 198, **200**
\measuredangle **259**
\medskip 45, **506**
\medskipamount, растяжимая длина **506**
\medspace **280**
\Mentry **83**
\Mentrylabel **83**
\message 486
\meta **478**
METAFONT 24
.mf-файл **22**, 24
\mho **213**, **256**, **259**
\mid **256**

MIME 299
\min **257**
minipage, окружение **69**, **89**, **90**, 94, 119, 153, 317, 359, **510**, **511**, 512, **514**, 516, 517
\minitoc **55**, 56, **58**
minitoc, пакет стилевой 55, 56
minitocdepth, счетчик **55**, 56, **58**
minitocoff, пакет стилевой **55**
mm, миллиметр (millimeter) **504**
\mod **266**
\models **256**
\Module **481**
moreverb, пакет стилевой **84**, **85**
\mp **255**
mpfootnote, счетчик **89**, **499**
\Msg **486**, 488
\mspace **280**
.mtc<N>-файл **55**
\mtcfont 56, **58**
\mtcindent, жесткая длина 56, **58**
mtimes, пакет стилевой **374**
mu, математическая единица (math unit) 52, 280, **504**
\mu **255**
multibox, пакет стилевой **321**
multicol, пакет стилевой 30, **94**, 98, 409, 438, 485, **535**
multicols, окружение **94**, **95**, 96–98, 409, 511, 532
  стилевые параметры (style parameters) 96–98
\multicolsep, растяжимая длина 96, **97**
\multicolumn 135, 144, 153
\multiframe **321**, 322
\multimake **321**, 322
\multimap **258**
multind, пакет стилевой **410**–**411**
\multiply 527
\multiput 322, 332, 334, 335, **336**, 346
\multiputlist **335**, **336**, 344, 346
\multirow **156**, 157, 160
multirow, пакет стилевой **156**, 160
\multirowsetup **156**
multline, окружение **272**, 275, 286
multline*, окружение **272**, 286, 287
\multlinegap, жесткая длина **275**, 286, 287
myheadings, стиль полосы (pagestyle) **110**, 113, 114
myunsrt, BibTeX'овский стиль 464

\nabla **256**
named, BibTeX'овский стиль 420, 422, **423**
named, пакет стилевой **423**
namelimits, опция **281**
\names 194
namunsrt, BibTeX'овский стиль **423**
nar, BibTeX'овский стиль **423**
nar, пакет стилевой **423**
natbib, BibTeX'овский стиль **420**, **424**
natbib, пакет стилевой **420**, **424**
\natural **256**
\naturalwidth, жесткая длина **66**
nature, BibTeX'овский стиль **424**
nature, пакет стилевой **424**
\ncong **259**
.ndx-файл 413
\nearrow **256**
nederlands, BibTeX'овский стиль **466**, 469
\NeedsTeXFormat **33**, **518**, **520**
\neg **256**
\negmedspace **280**
\negthickspace **280**
\negthinspace **280**
\neq **256**
newapa, BibTeX'овский стиль 420, **424**
newapa, пакет стилевой **424**
\newblock **420**, **460**
\newboolean 526, **531**
newcent, пакет стилевой **374**
\newcolumntype **131**, **132**, 134, 150
\newcommand **47**, 132, 261, 267, **493**, **495**, **496**, 498, 524
\newcounter **500**, **501**
\newenvironment 474, **496**, **498**
\newfloat **169**, 170, 171
  использование H в 169
\newfont 181
\newif 531
\newindex **412**
\newlength **503**, 524, 531
newlfont, пакет стилевой **215**
\newpage 45, 96, **117**, **144**
  в longtable **145**
\newsavebox **515**, **516**
\newtheorem 279, 282, **291**, **292**, 293, 500
  стилевые параметры (style parameters) 293
\newX 531
\newY 531
\nexists **259**
NFSS 180–251
  выявление источников ошибок (detecting problems in) 216
  ошибка совместимости версий (release 1 error) 249
  создание и поддержка (customizing and maintaining) 215–217
  сообщения об ошибках (error messages) 244–251
  трассировка (tracing font changes) 216

\nfssfont 241
nfssfont.tex 199
\ngeq **259**
\ngeqq **259**
\ngeqslant **259**
\ngtr **259**
\ni **256**
\nintt 215
\nLeftarrow **258**
\nleftarrow **258**
\nLeftrightarrow **258**
\nleftrightarrow **258**
\nleq **259**
\nleqq **259**
\nleqslant **259**
\nless **259**
\nmid **259**
.nnd-файл 413
\nobreak **118**
\nochangebars **368**
\nocite **418**, 438, 452, 453
\nocorr **197**
\nocorrlist **197**
\noextras*lang* **306**
\nofiglist **176**
\nofiles 142
\nohyphens **46**, 48
\noindent 517
nointlimits, опция **281**
\nolimits **263**
\nomarkersintext **177**
nonamelimits, опция **281**
\nonfrenchspacing 234
\nonumber **281**
\nopagebreak **117**
Nordic TEX Group **540**
\normalem **68**
\normalfont 44, 45, **191**, 198, 293, 294
\normalmarginpar **93**
\normalsize 106, **196**
norsk, опция **307**
nosumlimits, опция **281**
\not 254, **533**
\notablist **176**
\notag **278**, 281, 287
Notes, окружение **83**, **84**
notes, счетчик **84**
\nparallel **259**
\nprec **259**
\npreceq **259**
\nRightarrow **258**
\nrightarrow **258**
\nshortmid **259**
\nshortparallel **259**
\nsim **259**
\nsubseteq **259**
\nsucc **259**
\nsucceq **259**
\nsupseteq **259**
\nsupseteqq **259**
NTG **540**
\ntriangleleft **259**
\ntrianglelefteq **259**
\ntriangleright **259**
\ntrianglerighteq **259**
\nu **255**
\numberline 48, **50**, 51, 53, 54
\numberwithin **279**
\nVDash **259**
\nvDash **259**
\nvdash **259**
\nwarrow **256**
nynorsk, опция **307**

\oddsidemargin, жесткая длина **102**, **104**, **105**, **108**, 525
\odot **255**
\oint **256**
oldgerm, пакет стилевой **211**, **212**
oldlfont, пакет стилевой **204**, **214**, **215**
\oldstylenums **199**
\Omega **255**
\omega **255**
\ominus **255**
\onecolumn 94
\OnlyDescription **476**, **477**, 482
openbib, пакет стилевой **420**
\openin 237
\operatorname 287
\oplus **255**
\OptionNotUsed **519**, **526**
\or **533**
order, окружение **313**
\oslash **255**
\otimes **255**
\outerbarstrue **369**
\oval **341**, 343
\Ovalbox **318**
\ovalbox **318**
\overbrace **257**
overcite, пакет стилевой **419**
\overleftarrow **257**, **260**
\overline **257**
\overrightarrow **257**, **260**
\overset **263**, 284, 285
OzTeX, программа **357**
oztex, опция **357**

\p@enumi 74, **75**
\p@enumii 74, **75**
\p@enumiii **75**
\p@enumiv **75**
page, счетчик 110, **499**
\pagebreak 98, **117**, 277

\PageIndex **475**, **477**
\pagename **307**
\pagenumbering 60, **111**
\pageref **58**, **59**, 60, 62, 64, **532**, **533**
\pagestyle **110**, **112**, 114, 525
palatino, пакет стилевой **374**
pandora, пакет стилевой **211**
\paperheight, жесткая длина **102**, **104**, **109**
\paperwidth, жесткая длина **102**, **104**, **109**
\paragraph 28, **38**
paragraph, счетчик **39**, **499**
\parallel **256**
\parbox 69, 91, 125, 134, **510**, **514**
parentequation, счетчик **279**
\parindent, жесткая длина 126
\parsep, растяжимая длина **79**
\parskip, растяжимая длина **43**
\part 36, **38**, 40, 43, 46, **47**, 110
part, счетчик **39**, 40, **499**, 502
\part* **47**
\partial **256**
\partname **49**, **307**
\partopsep, растяжимая длина **79**, 296
\PassOptionsToClass **519**, **525**, 526
\PassOptionsToPackage **518**, **521**, **523**
\path **342**, 343, 345
pausing, опция **216**
pbmtopk, программа **316**
pc, пика (pica) **504**
perl, программа **440**
\perp **256**
phaip, BibTeX'овский стиль **424**
\phantom 284, 332
phcpc, BibTeX'овский стиль **424**
\Phi **255**
\phi **255**
phiaea, BibTeX'овский стиль **424**
phjcp, BibTeX'овский стиль **424**
phnf, BibTeX'овский стиль **424**
phnflet, BibTeX'овский стиль **424**
phpf, BibTeX'овский стиль **424**
phppcf, BibTeX'овский стиль **424**
phreport, BibTeX'овский стиль **424**
phrmp, BibTeX'овский стиль **424**
\Pi **255**
\pi **255**
Pi-шрифт (Pi font) 375
Piautolist, окружение **377**
picinpar, пакет стилевой **71**
\picsquare **337**, **339**, **340**
PiCTEX 315
picture, окружение 12, 15, 161, 271, 315, 316, 320–333, 335, 340, 345, 346
\Pifill **377**
\Pifont **376**, 377
pifont, пакет стилевой 76, **374–376**
\Piline **377**
Pilist, окружение **377**
\Pisymbol **377**
\pitchfork **258**
.pk-файл **22**, 24, 180, 209, 316, 372, 379
plain, BibTeX'овский стиль **423–425**, **428**, 459, 461, 465, **469**
plain, стиль полосы (pagestyle) **110**, 112, 116, 409
\plainfootrulewidth, жесткая длина **116**
\plainheadrulewidth, жесткая длина **116**
plainyr, BibTeX'овский стиль **424**
\pm **255**
pmatrix, окружение **269**
\pmb 253, 254, **255**, 282
\pmod **266**
\pod **266**
.pol-файл **22**
polish, опция **307**
.poo-файл **22**
.pool-файл **22**
portland, пакет стилевой **109**
\portrait **109**
portrait, окружение **109**
portuges, опция **307**
\postamble **486**
PosTeX, программа **301**
\postmulticols, жесткая длина 96, **97**
PostScript 12, 15, 23, 24, 121, 180–182, 201, 210, 214, 221, 225, 227, 316, 320, 340, **350–355**, 355–380, 458, 503
составной шрифт (composite font) 351
шрифт Type 0 351
шрифт Type 1 351
шрифт Type 3 351
PostScript'овский пункт (PostScript point) *см.* bp
PostScript-шрифт ZapfDingbats (ZapfDingbats PostScript font) 375
\Pr **257**
\preamble **486**, 488
\prec **256**
\precapprox **258**
\preccurlyeq **258**
\preceq **256**
\precnapprox **259**
\precnsim **259**
\precsim **258**
\prefacename **307**
\premulticols, жесткая длина 96, **97**
\PreserveBackslash 69, **127**, 130, 134
\prime **256**
printbib, программа **440**, 441
\PrintChanges **476**, **478**, **482**

\PrintDescribeEnv **481**
\PrintDescribeMacro **481**
\PrintIndex 475, **478**, **482**
\printindex 394, **410**, 412, 413
\PrintMacroName **481**
\ProcessOptions **518**, **522**, **523**, 526
\ProcessOptions* **518**, **523**
\prod **256**, 281
program, окружение **123**
program, пакет стилевой 17, **123**
programbox, окружение **123**
proof, окружение 282
\proofmodetrue **412**, 413
\propto **256**
\protect 53, 280, 392, 529
\providecommand **496**
\ProvidesClass **518–520**, 526
\ProvidesFile **518**, **520**
\ProvidesPackage **518**, **520**, 523
\ps@... **112**
\ps@plain 526
\ps@titlepage **481**
psamsfonts, опция **283**
\psboxit **370**
psboxit, пакет стилевой **370**
\PScommands **370**
\psdraft **359**
\psfigdriver **357**, **358**
psfrag, программа **316**
\psfull **359**
\Psi **255**
\psi **255**
PSNFSS 372–380
pstimesm, пакет стилевой **380**
pstricks, программа **316**
pt, пункт (point) 188, **504**
publicMF, программа **535**
publicTeX, программа **535**
\put 320–322, 332, 333, 336, **339**, **340**
\putbib **434**, 435
\putfile **339**, **340**

\qbezier **320**, **321**
\qquad **280**, **505**
Quad *см.* em
\quad **280**, **505**
quotation, окружение **80**, 465
Quote, окружение **80**
quote, окружение **80**, 497
\quotechar **479**

\raggedcolumns **96**
\raggedleft 45, **69**, 134
\raggedright 43, 45, **69**, 127, 134, 156
\raisebox 155, 156, **509**, **510**
\rangle **257**, 265
\ratio **528**
\rceil **257**
\Re **256**
\real **528**
\RecordChanges **476**, **477**, 482
\ref 41, **58**, **59**, **60**, 63, 64, 76, 77, 91, 278, 279, 282, 501
\refname **49**, **307**
\refstepcounter 47, 48, **500**, **501**
\reftextafter **62**, 63
\reftextbefore **62**, 63
\reftextcurrent **60**, **61**, 63
\reftextfaceafter **62**, 63
\reftextfacebefore **62**, 63
\reftextfaraway **62**, 63
\reftextvario **62**, 63
Rem, окружение 293
\renewcommand **45**, 51, 52, 62, 130, 156, 163, 164, **177**, **200**, 338, **494**
\renewenvironment **496**
\renewindex **412**
report, класс документа **23**, 38–40, 43, 44, 52, 55, 90, 105, **115**, 169, 302, 409, 500, 519
reqno, опция **281**, **296**
\RequirePackage **518**, 520, **522**, **523**, 525, 526
\restylefloat **170**, 173
resume, окружение **313**
\reversemarginpar **93**
\rfloor **257**
\rfoot **115**
\rgroup **257**
\rhd **213**, **255**
\rhead **115**, 118
\rho **255**
\right 268, 284, 285, 288
\Rightarrow **256**
\rightarrow **256**
\rightarrowtail **258**
\rightharpoondown **256**
\rightharpoonup **256**
\rightleftarrows **258**
\rightleftharpoons **258**
\rightmargin, жесткая длина **79**
\rightmark **113**, 117, 118
\rightnode **347**
\rightrightarrows **258**
\rightskip, растяжимая длина **69**
\rightsquigarrow **258**
\rightthreetimes **259**
\risingdotseq **258**
\rm **202**, **204**
использование в математических формулах **204**, 215
\rmdefault **200**, **201**

\rmfamily **192**, 198, **200**, 206, 292
использование в математических формулах **203**, **205**
\rmoustache **257**
\Roman 76, **501**
\roman 75, 76, **501**
Roman, стиль счетчиков полос (page number style) **111**
roman, стиль счетчиков полос (page number style) **111**
romanian, опция **307**
\root **322**
rotate, окружение **361**, 364
rotate, пакет стилевой **109**
rotating, пакет стилевой 6, **357**, **361**, 361–365
\rotcaption **365**
\rotdriver **357**, **361**
\Rsh **258**
\rtimes **259**
\rule 126, **514**
russian, опция 10, **307**
\rVert 265
\rvert 265

s *см.* Размерная функция (size function)
\samepage **117**, **118**
\sAppendix 47, 48
\savebox **515**
Sbox, окружение **318**, **319**
\sbox 318, 497, **515**, **516**
\sboxrule, жесткая длина **317**
\sboxsep, жесткая длина **317**
\sc **202**
использование в математических формулах 215
\scaleput **331**, 332
\scdefault **200**
\scriptscriptstyle 238, 267, **294**
\scriptsize **196**
\scriptstyle 238, 267, **294**
\scshape **193**, **194**, 198, **200**
использование в математических формулах **203**, **205**
\sdim, жесткая длина **317**
\searrow **256**
\sec **257**
\secdef 42, **46**, 47
secnumdepth, счетчик **38**, **39**, **42**, **45**, 48
\section 36, **38**, 39, 42, 44, 47, 48, 59, 113, 115, 190, 291, 434
section, счетчик **39**, 47, 112, **499**
\section* **38**
\sectionmark 48, **112**, 118
sed, программа **440**, **442**
\see **391**
\seename **307**
\selectfont **218**, 223
\selectlanguage **303**, 304, 306
seminar, пакет стилевой 318
\seriesdefault **200**, **201**, 223
Servan TEX **539**
\setboolean 526, **531**
\setcounter 76, 144, 163, 279, 369, **500**, **527**
\setdepth **324**
\sethspace **324**
\setlength 80, 91, 105, 116, 126, 129, 146, 164, 275, 292, 296, 368, 369, **503**, **527**
\setlinestyle **324**
\setlongtables 142, 143, **144**, **145**
\setmargins **107**
\setmarginsrb **107**
\SetMathAlphabet **208**, 244, 246, 248
\setminus **255**
\setnumberpos **324**
\setpapersize **107**
\setprecision **325**
\setstretch **325**
\setstyle **324**, **325**
\SetSymbolFont **239**, **241**, 244, 249
\settodepth **504**, 513
\settoheight **504**
\settowidth 160, **503**, **504**
\setwidth **325**
\setxaxis **325**
\setxname **325**
\setxvaluetyp **325**
\setyaxis **325**
\setyname **326**
\sf 181, **202**, 215
использование в математических формулах **204**, 215
\sfdefault **200**
\sffamily **192**, 194, 196, 198, **200**, 215
использование в математических формулах **203**, **205**
sfixed *см.* Размерная функция (size function)
sgen *см.* Размерная функция (size function)
SGML 26
\shabox **317**
shadow, пакет стилевой **317**
\shadowbox **318**
\shadowsize, жесткая длина **318**
\shapedefault **200**, **201**, 223
\shapepar **72**, **73**
shapepar, пакет стилевой **72**
\sharp **256**
\shortcite **419**
\shortciteA **419**
\shortciteN **419**

\shortindexingoff **412**
\shortindexingon 412, 413
\shortmid **258**
\shortpage **118**
\shortparallel **258**
\shortstack 72, 322, **334**, 344
\shoveleft **275**
\shoveright **275**
\showcols **132**
showidx, пакет стилевой **393**, **396**, **412**
\showprogress **487**
showtags, пакет стилевой **443**, **444**
siam, BibTeX'овский стиль **424**
\sideset **263**, **264**
sideways, окружение **361**, 364, 367
sidewaysfigure, окружение 364, **365**
sidewaystable, окружение **364**, 365, 366
\Sigma **255**
\sigma **255**
\sim **256**
\simeq **256**
\sin **257**, 265, 282
\sinh **257**
\skip\footins, растяжимая длина **89**, **91**
\sl **202**
  использование в математических формулах 215
\sldefault **200**
slides, класс документа 13, **23**, **214**
SliTeX 13, 214
\sloppy **70**
slovak, опция **307**
slovene, опция **307**
\slshape **193**, **195**, 198, **200**, 293
  использование в математических формулах **203**, **205**
\small 70, **196**
\smallfrown **258**
smallmatrix, окружение **270**
\smallsetminus **259**
\smallskip **506**
\smallskipamount, растяжимая длина **506**
\smallsmile **258**
\smash **264**
\smile **256**
sort, программа **442**
\SortIndex **479**
\SortNoop 450, **452**
\sout **68**
sp, приведенный пункт (scaled point) **504**
\space
  использование в .fd file 238
  использование в \DeclareFontEncoding 236
  использование в \DeclareFontShape 227
spacing, окружение **71**
\spadesuit **256**
spanish, опция **307**
\spbox **370**
\spbreve **262**
\spcheck **262**
\spdddot **262**
\spddot **262**
\spdot **262**
\special 109, 315, 316, 327, 340, 341, 350, 355, 358, 359, 361, 367, 371
\SpecialEnvIndex **479**
\SpecialEscapechar **478**
\SpecialIndex **479**
\SpecialMainIndex **479**
\SpecialUsageIndex **479**
\sphat **262**
\sphericalangle **259**
\spline **342**, 343
split, окружение **272**, 275, 276, 278, 281, 283–285, 288, 289, 382
\sptilde **262**
\sqcap **255**
\sqcup **255**
\sqrt **257**
\sqsubset **213**, **256**, **258**
\sqsubseteq **256**
\sqsupset **213**, **256**, **258**
\sqsupseteq **256**
\square **259**
\squarepar **73**
\ss 199
ssub *см.* Размерная функция (size function)
ssubf *см.* Размерная функция (size function)
\stackrel **263**, 272
StandardModuleDepth, счетчик **481**
\star **255**
\stepcounter **500**, **501**
\stop **217**
\StopEventually **475**, **476**, **478**
\stretch **502**, **503**, 506
\string 486
\strut 156
.sty-файл 21, **22**, 30, 31, 33, 63
sub *см.* Размерная функция (size function)
subarray, окружение **271**
subequations, окружение **279**
subf *см.* Размерная функция (size function)
\subfigcapskip, растяжимая длина **176**
\subfigtopskip, растяжимая длина **176**
\subfigure **177**
subfigure, пакет стилевой **176**
subfigure, счетчик **176**
\subitem **409**

\subparagraph **38**, 39, 40
subparagraph, счетчик **39**, **499**
\subsection 36, **38**, 39, 115
subsection, счетчик **39**, 40, **499**
\Subset **258**
\subset **256**
\subseteq **256**
\subseteqq **258**
\subsetneq **259**
\subsetneqq **259**
\substack **271**
\subsubitem **409**
\subsubsection **38**
subsubsection, счетчик **39**, **499**
\succ **256**
\succapprox **258**
\succcurlyeq **258**
\succeq **256**
\succnapprox **259**
\succnsim **259**
\succsim **258**
\sum **256**
sumlimits, опция **281**
\suntaxonly **215**
\sup **257**
supertab, пакет стилевой **138**, 142
supertabular, окружение 14, 120, 122, **138**, **139**, 140, 142–144, 147, 149
supertabular*, окружение **140**, 141
\suppressfloats **166**
\Supset **258**
\supset **256**
\supseteq **256**
\supseteqq **258**
\supsetneq **259**
\supsetneqq **259**
\surd **256**
\swabfamily **211**
\swarrow **256**
swedish, опция **307**
\symbol **199**
syntonly, пакет стилевой **215**

t1enc, пакет стилевой **210**
tabbing, окружение 120, **121**, **122**, 123
\tabbingsep, жесткая длина **122**
\tabcolsep, жесткая длина 126, **129**, 155, 364
\table **217**
table, окружение 59, **121**, **122**, 142, 146, 154, 162, 168, 170, 173, 178, 511
метки (labels) в 59
стилевые параметры (style parameters) 163–165
table, счетчик 142, **499**
\tablecaption **139**
\tablefirsthead **139**, 144
\tablehead **139**, 144
\tablelasttail **139**, 144
\tablename **307**
\tableofcontents 36, **50**, 54, 58, 113
\tableplace **177**
\tabletail **139**, 144
tabular, окружение 14, 16, 27, **69**, 70, 120, **121**, **122**, **124**, 126, 127, 129, 131, 132, 134, 135, 138–140, 142, 147, 153, 157, 160, 161, 176, 364, 366, 509, 511
стилевые параметры (style parameters) 129
tabular*, окружение **124**, 133, 135, 140, 155
TabularC, окружение **130**, 135, 136
tabularx, окружение 131, **133**, 134–136, 153, 160
tabularx, пакет стилевой 120, **133–137**, 160
\tabularxcolumn **134**
tabwindow, окружение **71**, **72**
\tag **278**, 281
\tag* **278**, 286, 287
\tagcurve **330**, **331**
\tan **257**
\tanh **257**
\tau **255**
\tbinom **266**, 267, 382
\tbranch **322**, **323**
tbtags, опция 275, **281**
testpage.tex 104
.tex-файл 21, **22**, 147, 150
TeX-index-файл **534**, 536
TEXCeH **540**
.texlog-файл **22**
Die TEXnische Komödie 443
\text **207**, 252, **255**, **264**, 269, 282, 290
\textbf **192**, **193**, **198**, **200**, 207
использование в математических формулах **206**
\textfloatsep, растяжимая длина **164**, **165**, 166
\textfraction **163**, 167
\textfrak **212**
\textgoth **212**
\textheight, жесткая длина **102**, **104**, **105**, **107**, **108**, 118, 119, 139, 167, 364
\textit **193**, **198**, **200**, 390
использование в математических формулах **206**
\textmd **192**, **193**, **198**, **200**
\textnormal **191**, **198**
\textrm **192**, **198**, **200**, 206
использование в математических формулах **206**

\textsc **193**, **198**, **200**
использование в математических формулах **206**
\textsf **192**, **198**, **200**, **225**
использование в математических формулах **206**
\textsl **193**, **198**, **200**
использование в математических формулах **206**
\textstyle 238, 267, **294**
\textswab **212**
\texttt **192**, **198**, **200**, 394
использование в математических формулах **206**
\textup **193**, **198**, **200**
TEXTURES program **357**
textures, опция **357**
\textwidth, жесткая длина **102**, **105**, **108**, 114, 116, 141, 528
TEX--XET 299
.tfm-файл **22**, 23, 180, 193, 196, 209, 218, 225, 227, 229, 233, 235, 355, 379
\tfrac **266**, 267
\the 501, 503
thebibliography, окружение 36, 113, **417**, **418**, 420, 422, 431, 463
\thechapter **40**
\theCodelineNo **481**
\theendnotes **93**
\theenumi 74, **75**
\theenumii 74, **75**
\theenumiii **75**
\theenumiv **75**
\theequation 279, 280
\thefootnote **89**, **90**, 153
theglossary, окружение **409**
theindex, окружение 36, 113, 393, **409**, 410
\thempfootnote **89**
theorem, окружение 290
theorem, пакет стилевой **290**, 485, **535**
\theorembodyfont **292**, 293, 294
\theoremheaderfont **292**, 294
\theorempostskipamount, растяжимая длина **292**, **293**
\theorempreskipamount, растяжимая длина **292**, **293**
\theoremstyle **291**, **292**
\thepage **111**, 116–118
\therefore **258**
\TheSbox **318**
\thesection 40, 41
\thesubfigure **176**
\thesubsection 40
\Theta **255**
\theta **255**
\thickapprox **258**
\Thicklines **342**, 343
\thicklines 318, 332, 337–339, 342–344, 346
\thicksim **258**
\thickspace **280**
\thinlines 318, 320, 337–339, 344–346
\thinspace **280**
\thispagestyle 47, **110**, 116, 171
threeparttable, окружение **154**
threeparttable, пакет стилевой **153**
\tilde **255**
\time **527**
\times **255**
times, пакет стилевой 171, **374**, **379**, **380**
\tiny **196**
Tirant lo TEX **540**
.toc-файл **22**, 23, 38, 47, 49, 53–55, 409, 410
tocdepth, счетчик 42, 51, **53**, 55, 56
\today **304**, 305
\tolerance **69**, **70**
\top **256**
\topcaption **139**
\topfigrule **164**, **165**
\topfraction **163**, **164**, 166
\topmargin, жесткая длина **102**, **104–106**
topnumber, счетчик **163**
\topsep, растяжимая длина **79**, 296, 297
\topskip, растяжимая длина **106**, 108
\totalheight, жесткая длина **508**, **510**, 514
totalnumber, счетчик **163**
tracefnt, пакет стилевой **216**, **217**, 246
\tracingfonts **246**
tracingmulticols, счетчик 97, **98**
\tracingtabularx **135**
trees, пакет стилевой **322**
\triangle **256**
\triangledown **259**
\triangleleft **255**
\trianglelefteq **258**
\triangleq **258**
\triangleright **255**
\trianglerighteq **258**
troff, программа **340**
\tt **202**
использование в математических формулах **204**, 215
\ttdefault 192, **200**
\ttfamily **192**, 198, **200**
использование в математических формулах **203**, **205**
.ttt-файл **177**
TUG (TEX Users Group) **538**
TUG India 540
TUG-Philippines **540**
TUGboat 443

turkish, опция **307**
turn, окружение 161, **361**, 367
TWGMLC 300
\twlrm 215
\twocolumn 94
twocolumn, опция 30, **94**, 98, 168, 174, 175, 438
\twoheadleftarrow **258**
\twoheadrightarrow **258**
twoside, опция **108**, 369, 438
\typeout 237, 238, 486

UKTEX Users Group **540**
\ulcorner **257**
ulem, пакет стилевой **67**
\ULforem **68**
\uline **68**
unbalance, счетчик **97**
\unboldmath **207**
\underbrace **257**
\underleftarrow **260**
\underleftrightarrow **260**
\underline **257**
\underrightarrow **260**
\underset **263**
Unicode 299
\unitlength, жесткая длина 330, 337, 342, 345, 346
\unlhd **213**, **255**
\unrhd **213**, **255**
\unskip 80
unsrt, BIBTEX'овский стиль **423–425**, **428**, 459, 464, **469**
\Uparrow **256**, **257**
\uparrow **256**, **257**
\updefault **200**
\Updownarrow **256**, **257**
\updownarrow **256**, **257**
\upharpoonleft **258**
\upharpoonright **258**
\uplus **255**
upref, пакет стилевой **282**
\uproot **262**
\upshape **193**, **195**, 198, **200**
\Upsilon **255**
\upsilon **255**
\upuparrows **258**
\urcorner **257**
US executive *см.* Формат бумаги (paper size)
US legal *см.* Формат бумаги (paper size)
US letter *см.* Формат бумаги (paper size)
\usage **479**
\usebox 497, **515**, **516**
\usecounter **83**
\usefont **223**
\usepackage 18, 27, **30–33**, 55, 56, 62, 171, 174, 253, 281, 302, 304, 356, 394, 426, **520**, 521–523
\uwave **68**

\vadjust 72
\value 279, **500**, 527
\varepsilon **255**
\varinjlim **266**
varioref, пакет стилевой 6, **60**, 63, 517
\varkappa **257**
\varliminf **266**
\varlimsup **266**
\varnothing **259**
\varphi **255**
\varpi **255**
\varprojlim **266**
\varpropto **258**
\varrho **255**
\varsigma **255**
\varsubsetneq **259**
\varsubsetneqq **259**
\varsupsetneq **259**
\varsupsetneqq **259**
\vartheta **255**
\vartriangle **259**
\vartriangleleft **258**
\vartriangleright **258**
\Vdash **258**
\vDash **258**
\vdash **256**
\vdots **256**
\vec **255**
\vector 332, 346
\vee **255**
\veebar **259**
Ventry, окружение **81**, 498
\verb **84**, **135**, 280, 476
\verb* **135**
verbatim, окружение **84**, **85**, 87, 476, 478
verbatim, пакет стилевой **84**, **85**, 100, **535**
verbatim*, окружение **84**, 478
\verbatimchar **479**
verbatimcmd, окружение **87**
verbatimtab, окружение **86**
\verbatimtabinput **86**
verbatimwrite, окружение **85**, 88
version, пакет стилевой **100**
.vf-файл **355**
\vfill **506**
\vline 130, 151, 152
vmargin, пакет стилевой **107**
Vmatrix, окружение **269**
vmatrix, окружение **269**
\voffset, жесткая длина **104**
\vpageref **60**, **61**, 62
\vref **60**, 62, 63

\vrule **126**
\vspace 72, 116, **505**, **506**, 514
\vspace* **505**
\Vvdash **258**

wais, программа **535**
warningshow, опция **216**
\wedge **255**
\whiledo **533**
\widehat **257**, 262
\WideMargins **108**
\widetilde **257**, 262, 284, 285
\width, жесткая длина **508**, **510**
window, окружение **71**, **72**
\wlog **237**, **238**
\wp **256**
\wr **255**
wrapfig, окружение **174**
wrapfig, пакет стилевой **174**
wrapfigure, окружение 71, **174**, 175, 176
www, программа **535**

x-высота (x-height) 186, 188, **234**, 504
\Xarea 531
\Xi **255**
\xi **255**
\xleftarrow **263**
.xmp-файл **54**, **55**
\xout **68**
xr, пакет стилевой **64**
\xrightarrow **263**
\Xsize 531
\xspace **68**
xspace, пакет стилевой **68**
XY-pic 315

\Yarea 531
\Ysize 531

\zeta **255**

Аббревиатура (abbreviation) 312
  при работе с BibTeX'ом 450
  пробел (space) после 68
Аббревиатура (acronym)
  определение (definition) для 194
Абзац (paragraph)
  стилевые параметры (style parameters) 69–70
  узкий (narrow) 152
Автор (author)
  публикации нескольких (BibTeX) 448
Адаптирующееся семейство (adaptable family)
  многоязычная BibTeX'овская система (BibTeX multi-language system) 469
Ажурный алфавит (blackboard bold) 255
Аксонометрическая проекция (axonometric projection) 332
Активные литеры (active character) 18, 314
Акцент (accent)
  в tabbing 122
  в BibTeX 449
Акценты над прописными буквами (capitalization rules) 313
Аппроксимации Безье (bezier approximations) 320–321

База данных (data base) (BibTeX) 421, 422
Библиографические ссылки (citation) 58, 416–471
Большой пункт (big point) *см.* bp

Вертикальная ориентация (portrait orientation) 109
Верхний колонтитул (header of page) 101
Виртуальный шрифт (virtual font) 301, 355, 372
Включаемый материал (including material) 100
Внешнее поле (outer margin) 101
Внутреннее поле (inner margin) 101, 108
Всеобъемлющий сетевой TeX-архив (Comprehensive TeX Archive Network (CTAN)) 534
Выделение (emphasis) *см.* команды \em и \emph, 195
Выключные формулы (displayed equations)
  центрирование (centering) 281
Выносные примечания (endnote)
  сноски (footnotes) как 93

Гарвардская схема (Harvard system)
  ключевое слово (keyword) (BibTeX) 422
Гистограмма (bar chart) 324–327
Глоссарий (glossary) 408
Горизонтальная ориентация (landscape orientation) 109
Готические шрифты (Old German fonts) **211–212**
  классификация в NFSS **212**
Граф (graph)
  двудольный (bipartite) 344
Графика (graphics)
  переносимая (portable) 315–349
Группы пользователей TeX'а (TeX users groups) 538–540

Двусторонняя печать (recto-verso printing, two-sided printing) 101
Демонстрационные слайды (overhead slide)
получение (producing) **214**
Дерево (tree)
бинарное (binary) 322
рисование 348
тернарное (ternary) 322
Диакритический знак (diacritic) 303
Диалект языка (dialect) 304, 305
Диапазон размеров (size range) **227**
Диапазоны страниц (page range)
в указателе (index) 389
Документация (documentation) 472–491
Документы ранее созданные (old documents)
обработка в LaTeX 2ε (processing with LaTeX 2ε) **214–215**
Дюйм (inch) *см.* in

Единицы измерения (measure) 504

Жесткая длина (rigid length) **502**
\@pnumwidth **51**, **52**
\arraycolsep 126, **129**, **296**
\arrayrulewidth **130**, 152
\bibhang **420**
\bibindent **420**
\bigstrutjot **156**
\captionwidth **179**
\columnsep 96, **97**, **102**, **105**, 175
\columnseprule **96**, **97**, **102**, **105**
\columnwidth **91**, **102**
\depth **508**, **510**
\doublerulesep **129**, **130**
\emergencystretch **69**, **70**
\evensidemargin **102**, **104**, **105**, **108**, 525
\extrarowheight 124, **130**
\extratabsurround **158**
\fboxrule 317, 318, **509**
\fboxsep 317, 318, 370, **509**
\fminilength 517, 528
\footnotesep **89**, **90**
\footrulewidth **115**
\GlossaryMin **480**
\headheight **102**, **105**, **106**, 116
\headrulewidth **115**
\headsep **102**, **105**, **106**
\headwidth **116**, 118
\height **508**, **510**, 514
\hoffset **104**, **108**
\IndexMin **479**
\itemindent **79**
\jot 156, **296**
\labelsep 78, **79**, 122
\labelwidth **79**, 81–83
\leftmargin **79**, 81
\linewidth 52, 96, **102**, 130, 516
\listparindent **79**
\LTcapwidth 144, **145**
\MacroIndent **480**
\marginparpush **93**, **102**, **105**
\marginparsep **93**, **102**, **105**, 116, 118
\marginparwidth **93**, **102**, **105**, 116, 118
\mathindent **296**
\mathsurround **77**
\mtcindent 56, **58**
\multlinegap **275**, 286, 287
\naturalwidth **66**
\oddsidemargin **102**, **104**, **105**, **108**, 525
\paperheight **102**, **104**, **109**
\paperwidth **102**, **104**, **109**
\parindent 126
\plainfootrulewidth **116**
\plainheadrulewidth **116**
\postmulticols 96, **97**
\premulticols 96, **97**
\rightmargin **79**
\sboxrule **317**
\sboxsep **317**
\sdim **317**
\shadowsize **318**
\tabbingsep **122**
\tabcolsep 126, **129**, 155, 364
\textheight **102**, **104**, **105**, **107**, **108**, 118, 119, 139, 167, 364
\textwidth **102**, **105**, **108**, 114, 116, 141, 528
\topmargin **102**, **104–106**
\totalheight **508**, **510**, 514
\unitlength 330, 337, 342, 345, 346
\voffset **104**
\width **508**, **510**
Жирность шрифтов (weight of fonts) 187

Заголовок (heading)
«в подбор» (run-in) 42, 44, 45, 532
«в разрез» (display) 42–44
стилевые параметры (style parameters) 42–46
Заголовок «в подбор» (run-in heading) *см.* Заголовок (heading)
Заголовок «в разрез» (display heading) *см.* Заголовок (heading)
Знак цитирования (quotation character) 303

Имени формат (name format) (BibTeX) 447

Инкапсулированный (encapsulated) PostScript 17, 161
Интегралы (integrals)
расстановка пределов (placement of limits) 281
Интервал (space)
межбуквенный (inter-letter) (трэкинг (tracking)) 66
межсловный (inter-word) 66, **232**, 234, 235, 508
изменение (changing) 234
межстрочный (inter-line) (интерлиньяж (leading)) 70
неудачный после переключения шрифта (wrong after font change) *см.* Поправка на курсив (italic correction)
после аббревиатуры (abbreviation) 68
Интерлиньяж (leading) *см.* Интервал (space), Межстрочный интервал (baseline skip)
Информация о размерах (size info) **227**
структура (structure) 227–228
Информация об именах шрифтов (font info) **227**
Исключаемый материал (excluding material) 100

Кегль (font size) *см.* Шрифта размер (font size)
Класс (class) *см.* Класс документа (document class)
Класс документа (document class) **23**, **30**
идентификация (identification of) **519**
опция (option) 23, **30**
расширение функциональных возможностей (extending functionality) **525–526**
amsart **283**
amsbook **283**
article 16, **23**, 31, 33, 36–38, 40, 49, 90, 105, **115**, 283, 302, 409, 438, 481, 526
book **23**, 31, 38, 40, 43, 55, 90, 105, 112, **115**, 169, 283, 302, 409, 495, 500, 517, 522
letter **23**, 37, 302
report **23**, 38–40, 43, 44, 52, 55, 90, 105, **115**, 169, 302, 409, 500, 519
slides 13, **23**, **214**
Ключевое слово (keyword) (BibTeX) 421
гарвардская схема (Harvard system) 422
Кодировка шрифта (encoding of fonts) *см.* Шрифта кодировка (font encoding)
Колонтитул верхний (page header) 101
Колонтитул нижний (page footer) 101
Команда (command)
аргумент (argument) 494
необязательный аргумент (optional argument) 495
нехрупкая (robust) 280, 391
определение (definition) 493–496
переопределение (redefinition) 494
хрупкая (fragile) 280, 392, 529
Команды для работы со счетчиками (counter commands) 499–502
Команды для стрелок (arrow commands) 256
в AMS-шрифтах (AMS fonts) 258
в коммутативных диаграммах (commutative diagrams) 271
как ограничители (delimiters) 257
над символами 257, 260
растяжимые (extensible) стрелки 263
Команды, задающие длину (length commands) 502–506
жесткие (rigid) **502**
с растяжением (rubber) **502**
Команды, создающие боксы (box commands) 507–517
Комментарии (comments)
в BibTeX'овской базе данных 457
Корковская кодировка (Cork encoding) 199, 201, **209–210**, 227, 233, 235, 300
Кривая (curve)
произвольная (arbitrary) 327

Линейка (rule) 514
невидимая (invisible) 495
Линейка нулевой ширины (strut) 495, **514**
Литера (character)
доступ в шрифте (accessing in fonts) 199
перенос (hyphen) 232
позиция в кодовой таблице (position in encoding) 233
Литера (character) @
в именах команд (in command names) 33
делать активной (made active) 18

Макет полосы набора (page layout) 101
стилевые параметры (style parameters) 101–107
Макета разметка (layout markup) 27

Математическая версия (math version) **207–208**, 211, 239, 242, 244, 247–250
bold 207, 239, 242, 243
euler 207
normal 207, 213, 242, 243
объявление новых (declaring new) **242**
ограничение на шрифты (limit of fonts) **242**
переопределенная пакетом (changed by package) 212
Математическая единица (math unit) *см.* mu
Математический алфавит (math alphabet) **203**
литера (character) **202–203**
по умолчанию (default) 205
Математический шрифт дополнительный (math extension font) **214**
Математических символов шрифт (math symbol font) 202, 234, **239**, 240–244, 247, 249, 250
largesymbols 234, 243
letters 243
operators 243
symbols 234, 243
версия по умолчанию (default version) 239
изменение настроек (changing the setup) 242–243
объявление новых (declaring new) 239–242
по умолчанию для математической версии (default for math version) 239
Математического алфавита идентификатор (math alphabet identifier) 203
в версиях математических формул (in math versions) 208
объявление нового (declaring new) **205–206**
предопределенный (predefined) **204**, 204–205
Математического алфавита литера (math alphabet character) 240
Межбуквенный интервал (inter-letter space) *см.* Интервал (space)
Международный документ (international document) 299
Межсловный интервал (inter-word space) *см.* Интервал (space)
Межстрочный интервал (baseline skip) установка (setting) 220–223
Миллиметр (millimeter) *см.* mm
Многоязычность (multilanguage) 299
Моноширинные шрифты (monospaced fonts) 183

Начертание шрифтов (shape of fonts) *см.* Шрифта начертание (font shape)
Не зависящий от устройства файл (Device independent file) *см.* .dvi-файл
Необязательный аргумент (optional argument)
команды 495
окружения 498
Нехрупкая команда (robust command) 280, 391
Нижние и верхние индексы (subscripts and superscripts) расположение 281
Нижний колонтитул (footer of page) 101
Новая схема выбора шрифтов (New Font Selection Scheme) *см.* NFSS

Общая разметка (generic markup) 26
Оглавление (contents list)
стилевые параметры (style parameters) 50–53, 58
Односторонняя печать (one-sided printing) 101
Окружение (environment)
необязательный аргумент (optional argument) 498
определение (definition) 496–499
переопределение (redefinition) 496
Окружение перечня, списка (list environments) 74–84, 497–498
заключенное в рамку (boxed) 319
Окружения, рисующие рамки (boxed environments) 86, 317–319, 370, 516
Опция (класса или пакета) (option (class or package))
декларация (declaration of) **520–523**
передаваемая другому пакету **521**
11pt 30
a0paper **104**
a4paper 30, **104**, 520
a5paper **104**
american **307**
austrian **307**
b5paper **104**
bind 525
brazil **307**
catalan **307**
centertags 276, **281**, 284
croatian **307**
cropmarks 525

czech **307**
danish **307**
debugshow **216**
draft **359**
dutch **307**
dvips **357**
dvitops **357**
emtex **356**, **357**
english **307**
errorshow **216**
esperanto **307**
executivepaper **104**
final **359**
finnish **307**
fleqn 275, **281**, **296**, **297**
francais **307**
french 30, **304**, **307**, **310**
galician **307**
german 30, 32, **302–304**, **307**, **308**, 397, 405, 438
germanb **302**, **307**
infoshow **216**, 521
intlimits **281**
italian **302**, **307**
legalpaper **104**
leqno 278, **281**, **296**
letterpaper **104**, **105**
ln **357**
loading **216**
magyar **307**
namelimits **281**
nointlimits **281**
nonamelimits **281**
norsk **307**
nosumlimits **281**
nynorsk **307**
oztex **357**
pausing **216**
polish **307**
portuges **307**
psamsfonts **283**
reqno **281**, **296**
romanian **307**
russian 10, **307**
slovak **307**
slovene **307**
spanish **307**
sumlimits **281**
swedish **307**
tbtags 275, **281**
textures **357**
turkish **307**
twocolumn 30, **94**, 98, 168, 174, 175, 438
twoside **108**, 369, 438
warningshow **216**

Ориентация (orientation)
вертикальная (portrait) 109
горизонтальная (landscape) 109
Орнамент (ornament) 317–319
Основной шрифт (normal font)
см. Шрифт документа (document font)
Отрицание символа (negated symbol) 254
Ошибочно выбранный шрифт (wrong font selected) 201, 225

Пакет стилевой (package) 23
идентификация (identification of) **520**
a4 **107**, 438
a4dutch **107**, **108**
a4wide **107**
a5 **107**
a5comb **107**
afterpage **168**, **173**
alltt **84**
amsbsy **207**, 255, **282**
amscd 271, 272, **282**
amsfonts **213**, 254, 255, **282**
amsintx **282**
amsmath 252, **253**, 254, 255, 260–266, 268, 271–273, 275, 277–282, **282**, 283, 284, 296
amsopn **282**
amssymb **213**, 254–259, **282**
amstex 13, 15, **18**
amstext 207, 255, 264, **282**
amsthm **282**
amsxtra **261**, **262**, **282**
apalike **423**, **426**
array 120, **124–132**, **137**, 158, 485, **535**
authordate1-4 **423**
avant **374**
babel 6, 9, 10, 15, 30, 300, **302**, 303–308, 310, 312, 397, **405**, 523
bar **324**, 325
basker **374**
bembo **374**
beton **211**
bezier **320**, 330, **333**
biblist **438**, **439**
bibmods **443**, **444**
bibunits **431**, **434–438**
bookman **374**
boxedminipage **317**
calc 15, 76, 81, 106, 130, 492, **500**, 516, 520, **527**, 528, 529
changebar 6, **356**, **357**, **365**, 365–370
chapterbib **431–435**, **438**
chicago **419**
cite **419**

citesort **419**
color **371**
colordvi **371**
concrete 207, **211**
curves **327**
curvesls **327**
dcolumn 121, **147**
delarray 121, **137**
doc 15, **223**, **472–485**
doublespace **70**, **71**
draftcopy **371**
eclbip **344**
ecltree **348**
eepic 12, 15, 316, 317, **340–344**, 345, 346
eepicemu **343**
endfloat **176**, **177**
endnotes **93**
enumerate **76**
epic 12, 15, 316, 317, **333–340**, 340–348
epsfig **356**, **357**, 357–359
eucal **282**
eufrak **213**, **282**
euler **212**, **213**, 240
euscript **213**
exscale **214**
fancybox **318**, **319**
fancyheadings **114**, **115**, **117**
flafter 61, **166**
float **168**, **169**, 171, **172**, **173**
floatfig **173**, **174**
fnpara **92**
fontenc 9
footnpag **90**
french 18, **298**, **312–314**
ftnright 94, **98**, 99, 485, **535**
garamond **374**
german **303**, 397
graphics **356**
hackalloc **17**
hangcaption **179**
harvard **423**
helvet **374**
here **173**
hhline 121, **151**
ifthen 15, 492, **529**, 531
indentfirst **46**
index **410–415**
inputenc 9
isolatin1 **300**
jmb **423**
latexsym **213**, 255, 256
layout **103**
letterspace **66**
longtable **138**, **142**, **143**
lscape **109**
lucid **374**
lucidbrb 192, **374**, **380**
lucidbry **374**, **380**
lucmath **374**
makeidx **391**, **394**, 396
minitoc 55, 56
minitocoff **55**
moreverb **84**, **85**
mtimes **374**
multibox **321**
multicol 30, **94**, 98, 409, 438, 485, **535**
multind **410–411**
multirow **156**, 160
named **423**
nar **423**
natbib **420**, **424**
nature **424**
newapa **424**
newcent **374**
newlfont **215**
oldgerm **211**, **212**
oldlfont **204**, **214**, **215**
openbib **420**
overcite **419**
palatino **374**
pandora **211**
picinpar **71**
pifont 76, **374–376**
portland **109**
program 17, **123**
psboxit **370**
pstimesm **380**
rotate **109**
rotating 6, **357**, **361**, 361–365
seminar 318
shadow **317**
shapepar **72**
showidx **393**, **396**, **412**
showtags **443**, **444**
subfigure **176**
supertab **138**, 142
syntonly **215**
t1enc **210**
tabularx 120, **133–137**, 160
theorem **290**, 485, **535**
threeparttable **153**
times 171, **374**, **379**, **380**
tracefnt **216**, **217**, 246
trees **322**
ulem **67**
upref **282**
varioref 6, **60**, 63, 517
verbatim **84**, **85**, 100, **535**
version **100**
vmargin **107**
wrapfig **174**

xr **64**
xspace **68**
Параметры макета (layout parameters) *см.* Стилевые параметры (style parameters)
Параметры математических стилей (math style parameters) 296
Перекрестные ссылки (cross-reference) 58–64
.toc-файл 23
в указателе (index) 389
ключ (key) 58
на внешние документы (to external documents) 64
стилевые параметры (style parameters) 63
Переменная (variable)
имена в BibTeX'овских стилевых файлах 458
типы в BibTeX'овском стилевом файле 458
Переносимая графика (portable graphics) 315–349
Переносимый растровый формат (portable bitmap format) 316
Переносы (hyphenation) 23, 83, 127, 152, 299, 302, 305, 306
в примечаниях на полях (in marginal notes) 92
в узких колонках (narrow columns) 152
сложных слов (compounds) 233, 304
Печать (printing)
двусторонняя (recto-verso) 101
двусторонняя (two-sided) 101
односторонняя (one-sided) 101
Пика (pica) *см.* pc
Плавающий объект (float) 162–179
в multicols 98
метки (labels) в 59
стилевые параметры (style parameters) 163–165, 169–170
Поле (margin)
внешнее (outer) 101
внутреннее (inner) 101, 108
Полиграфические шрифты (typographical fonts) 183
Полутоновый шрифт (halftone font) 316
Поправка на курсив (italic correction) **193**
автоматическое добавление (automatical supply of) 197–198
с командой \em 195
Преамбула документа (document preamble) **31**, 105
Пределы (limits) *см.* Нижние и верхние индексы
Предметный указатель (index)
генерируемый командами (generated by commands) 493, 529
Пробел (space)
в формулах 280
в элементах указателя (spaces in index entries) 388, 393
после аббревиатуры (abbreviation) 68
Программа (program)
afm2tfm **355**, 380
archie **535**
aux2bib 422, **440**
awk **440**, **442**
bibclean **442**
bibextract **442**
bibkey **440**
bibsort.sh **442**
bibview **443**
BM2FONT 8, **316**
citefind **442**
citetags **442**
dvicopy **301**
dvips 8, 109, 301, 340, 350, **354–357**, **365**, **371**, **372**, **380**, 535, 549, 590
dvitops **355**, **357**
egrep **440**
emTeX 8, 9, **357**, **535**
ghostscript **351**
ghostview **351**, 355
gopher **535**
gpic **340**, **341**
gzip **536**
looktex **440**
makebib **440**
makebst 420, 422, 469, 470
OzTeX **357**
pbmtopk **316**
perl **440**
PosTeX **301**
printbib **440**, 441
psfrag **316**
pstricks **316**
publicMF **535**
publicTeX **535**
sed **440**, **442**
sort **442**
troff **340**
wais **535**
www **535**
Пропорциональные шрифты (proportionally spaced fonts) 183
Простой размер (simple size) **227**
Публикации нескольких авторов (multiple authors) (BibTeX) 448

Пункт (point) *см.* pt
Пункт Дидо (Didôt point) *см.* dd
Пунктуация (punctuation) 312

Размер полосы набора (paper size) 107
Размер шрифтов (size of fonts)
см. Шрифта размер (font size)
Размерная функция (size function) **227**, 228–232
empty **228–230**, 230, 232
fixed 232–247
gen 230–231
s 230
sfixed 232
sgen 231
ssub 232, 249
ssubf 232
sub 224, **231**, 248, 249
subf 232, 247
Разметка (markup)
макета (layout) 27
общая (generic) 26
Разрыв страницы (page break) 27
в формуле (equation) 277
Разрыв строки (line break) 27
Рамка (frame) 318
Рамки (boxes)
двойные (double) 318
декоративные (fancy) 318
несколько (multiple) 321
овальные (oval) 318
с тенью (shadow) 317, 318
стилевые параметры (style parameters) 317, 509
Растр (bitmap)
переносимый (portable) формат 316
шрифты *см.* .pk-файл
Растяжимая длина (rubber length) **502**
\@rightskip 69
\abovedisplayshortskip 297
\abovedisplayskip 296
\baselineskip 70, 71, 90, **105**, **106**, 118, 124, 211, 506
\belowdisplayshortskip 297
\belowdisplayskip 297
\bigskipamount 145, **506**
\dblfloatsep 164
\dbltextfloatsep 164
\fill 145, 146, 497, **502**, **503**, **505**, **506**
\floatsep 164
\footskip 102, **105**, 116
\intextsep 164, 175
\itemsep 79
\LTleft 145, 146
\LTpost 145
\LTpre 145
\LTright 145, 146
\MacrocodeTopsep 481
\MacroTopsep 480
\medskipamount 506
\multicolsep 96, **97**
\parsep 79
\parskip 43
\partopsep 79, 296
\rightskip 69
\skip\footins 89, **91**
\smallskipamount 506
\subfigcapskip 176
\subfigtopskip 176
\textfloatsep 164, **165**, 166
\theorempostskipamount 292, **293**
\theorempreskipamount 292, **293**
\topsep 79, 296, 297
\topskip 106, 108
Рисунок (figure)
в multicols 98
метки (labels) в 59
плавающий объект (floating object) 162–179
Романские шрифты (roman fonts) 184
Рубленые шрифты (sans serif fonts) 184

Сантиметр (centimeter) *см.* cm
Семейство шрифтов (family of fonts)
см. Шрифтов семейство (font family)
Сжатие файлов (compression of files) 536
Символ (character)
доступ в шрифте (accessing in fonts) 199
перенос (hyphen) 232
позиция в кодовой таблице (position in encoding) 233
Символ (symbol)
AMS 257–259
LaTeX 213, 255–257
отрицание 254
Символьные шрифты (symbol fonts)
см. Математических символов шрифт (math symbol font)
Синтаксиса проверка (syntax check) 215
Скобки в элементах указателя (braces in index entries) 392
Слайд (slide) *см.* Демонстрационные слайды (overhead slide)
Сложное слово (compound word)
переносы (hyphenating) 233, 304
Сноска (footnote)
в longtable 144
в minipage 89–90
в multicols 98
в tabular 153–154
в двухколонном формате (two-column format) 98

Сообщение об ошибке (error message)
в файле протокола (in transcript file) 23
Сортировка в указателе (sorting the index) 23, **395**
Составной шрифт (composite font)
см. Виртуальный шрифт (virtual font)
Специальный символ (special character) (BibTeX) 449
Список литературы (bibliography) 58, 416–471
`.bbl`-файл 24
стилевые параметры (style parameters) 418, 420
стили для 24, 423–426
Ссылки (reference)
см. Библиографические ссылки (citation), *см.* Перекрестные ссылки (cross-reference), *см.* Указатель (index)
библиографические (bibliographic) 416
на сноски (to footnote) 91
Стандарт ISO-10646 299, 301
Стандарт ISO-8859 299–301
Стандарт ISO-8879 26
Стилевой файл (style file)
BibTeX421, 422
*MakeIndex*400
Стилевые параметры (style parameters)
`array` и `tabular` 129
`barenv` 324–326
doc 480
`enumerate` 75
fancyheadings 116
`figure` & `table` 163–165
fonts 200
`\footnote` 88–92
`itemize` 77
`list` 79
`longtable` 145
`\marginpar` 93
`multicols` 96–98
абзац (paragraph) 69–70
в формулах 294–297
для набора математики 294–297
для теорем 293
заголовки (headings) 42–46
оглавления (contents lists) 50–53, 58
отчеркиваний (change bars) 369
перекрестные ссылки (cross-references) 63
полоса набора (page) 101–107
рамки (boxes) 317, 509
список литературы (bibliography) 418, 420
указатель (index) 401, 403
Стиль display для набора математики (display style in math) 294
Стиль script для набора математики (script style in math) 294
Стиль scriptscript для набора математики (scriptscript style in math) 294
Стиль text для набора математики (text style in math) 294
Стиль нумерации страниц (page number style)
`Alph` **111**
`alph` **111**
`arabic` **111**
`Roman` **111**
`roman` **111**
Стиль полосы (page style)
fancy 116
headings 113
myheadings 113, 114
plain 112, 116, 409
Стиль страницы (page style)
empty **110**
fancy **114**, **115**
fancyplain **116**
headings **110**
myheadings **110**
plain **110**
Схема упорядочения (collating sequence) 299

Таблица (table)
в `multicols` 98
метки (labels) в 59
плавающий объект (floating object) 162–179
Таблица кодировки (codepage)
см. Шрифта кодировка (font encoding)
Таблица кодировки (encoding table)
программа генерации (program for generating) 217
Таблица литер (character table)
программа генерации (program for generating) 217
Таблица раскладки шрифта (font encoding table)
программа генерации (program for generating) 217
Тело текста (body of page) 101
Тестирование шрифтов (testing fonts) 217

Типографские единицы измерения (typographic measure) 504
Трэкинг (tracking) *см.* Интервал (space)

Указатель (index) 58, 385–415
  `.idx`-файл 23
  `.ind`-файл 23
  `INDEXSTYLE` системная переменная (system variable) 398
  диапазоны страниц (page range) 389
  использование скобок в элементах 392
  ошибки (errors)
    этап записи (writing phase) 399
    этап чтения (reading phase) 398
  перекрестные ссылки (cross-references) 389
  пробелы в элементах (spaces in entries) 388, 393
  символы (symbols) 392
    ! (уровень (level)) 389, 391, 398, 399, **401**
    " (кавычка (quote)) 391, **401**
    @ (ключ сортировки (sortkey)) 390, 391, 398, 399, **401**
    | (encap) 389, 391, 398, 399, **401**
    |((начало диапазона (range start)) 389–391, 399, 400, **401**
    |) (конец диапазона (range end)) 389, 391, 399, 400, **401**
  сортировка номеров страниц (page number sorting) 392
  стилевые параметры (style parameters) 401, 403
  уровень (level) 389
  числа (numbers) 392
  чувствительность к регистру (case sensitivity) 388, 392, 397
  элемент (entry) 389
    see 389
    верхнего уровня (main) 389
    второй подчиненный (subsub) 389
    первый подчиненный (sub) 388, 389
    простой (simple) 387
Умляут (umlaut) 303
Управление версиями (version control) 100
Уровень в указателе (level in index) *см.* Указатель, уровень (index, level)

Файл метрик шрифта (font metric file) *см.* `.tfm`-файл
Файл определения шрифта (font definition file) *см.* `.fd`-файл
Файл формы литеры (packed character file) *см.* `.pk`-файл
Файл формы символа (packed character file) *см.* `.pk`-файл
Файлов сжатие (file compression) 536
Файлы, созданные LaTeX'ом (files, produced by LaTeX) 23
Фейнмановкая диаграмма (Feynman diagram) 316
Фейнмановская диаграмма (Feynman diagram) 333, 334
Формат (format) (TeX)
  идентификация (identification of) 33, **520**
  file 23
Формат архива tar (tar archive format) 536
Формат архива zip (zip archive format) 536
Формат архива zoo (zoo archive format) 536
Формат бумаги (paper size) 105, 107
Формула (equation)
  вертикальный пробел в 277
  выровненные по вертикали 274, 285, 287–290
  заключенная в рамку (boxed) 319
  нумерация 278–280
    подчиненная (subordinate) 279
    расположение слева или справа 281
  разбитая на части 275, 283–287
Французские кавычки (guillemet) 303, 312
Функция (function)
  встроенные (built-in) (BibTeX) 458
  имена в BibTeX'овских стилевых файлах 458

Химическая формула (chemical formula) 316, 333, 335
Хрупкая команда (fragile command) 280, 392, 529

Цвет (color)
  стандарт CIE (CIE standard) 352
  шаблон CMYK (CMYK Model) 352
  шаблон HSB (HSB Model) 352
  шаблон RGB (RGB Model) 352
Цветной стандарт CIE (CIE color standard) 352
Цветной шаблон CMYK (CMYK color model) 352
Цветной шаблон HSB (HSB color model) 352
Цветной шаблон RGB (RGB color model) 352

Цепочка литер постоянная (string constant) в BibTeX'овском стилевом файле 458
Цифры (digit)
невыровненные (non-aligning) 199
Цифры «под старину» (old style numerals) 199
Цицеро (cicero) *см.* cc

Чувствительность к регистру (case sensitivity)
в указателе (index) 388, 392, 397
при работе с BibTeX'ом 458, 459

Ширина шрифтов (width of fonts) 187
Шрифт (font) *см. также* Литера, Символ
PostScript, Type 0 351
PostScript, Type 1 351
PostScript, Type 3 351
без засечек (sans serif) 184
виртуальный (virtual) *см.* Виртуальный шрифт (virtual font)
жирность (weight) 187
использование в формулах (use in math) 202–208, 238–243
команды переключения шрифта текста в формулах (text commands in math) 203, 205
моноширинный (monospaced) 183
объявление кодировки (declaring encodings) 238
объявление новых (declaring new) 225–244
ошибочно выбранный (wrong selected) 201
полиграфический (typographical) 183
пропорциональный (proportionally spaced) 183
прямой (романский (roman)) 184
рубленый (sans serif) 184
с засечками (serifed) 184
составной (composite) 351
специальный (special) 315
стилевые параметры (style parameters) 200
тестирование (testing) 217
ширина (width) 187
Шрифт документа (document font) **191**, 192, 201
изменение (changing) **201**
Шрифта x-высота (font x-height) *см.* x-высота (x-height)
Шрифта атрибуты (font attribute) 183–190
комбинирование (combining) 194, 196–197
установка (setting) **218–219**
Шрифта загрузка (font loading)
динамическая (dynamical) 237
опции (options) 232–235
возникающие проблемы (problems) 234–235
Шрифта кодировка (font encoding) 189–190, 199–201, 206, 208–213, 217, **222–223**, 224–227, 231, 233, 235–238, 240, 241, 243–245, 247, 251, 299
классификация (classification) **222**
объявление новых (declaring new) **235–236**
по умолчанию (default) **201**, 210
смешивание (mixing) **236**
соглашение об обозначениях (naming conventions) **236**
устаревшая (obsolete) 222
Шрифта насыщенность (font series) 192–194, 196, 197, 201, 206, 208, 212, 217, **219–220**, 224–226, 231, 236, 245
классификация (classification) **220**
кодировка по умолчанию (default in encoding) **224**
Шрифта начертание (font shape) **184–187**, **193–195**, 197, 201, 206, 208, 212, 217, **220**, 224–226, 231, 232, 235, 236, 245, 248
oblique *см.* наклонное
sloped *см.* наклонное
капитель (small capitals) **186**, 193–195, 198, 200, 221
классификация (classification) **221**
кодировка по умолчанию (default in encoding) **224**
курсивное (italic) **185**, 186, 193, 195, 198, 200, 221
наклонное (slanted) **185**, 193, 195, 198, 200, 206, 221
прямое (upright) **185**, 186, **193**, 195, 198, 200, 205, 221, 224
Шрифта подстановка (font substitution) **224**
по умолчанию (defaults) 194, 196, 200, 217, **224**, **231**, **236**, 244, 245, 250
Шрифта размер (font size) 186, **188–189**, **195–196**, 217, **220–222**, 224–226, 229, 230, 232, 247, 248
для набора математики (in math) 221, **238–239**, **294–296**

команды изменения (commands for changing) **196**
Шрифтов группа начертаний (font shape group) 203, 205, **226**, 231–235, 237, 239, 244, 245, 247, 248
модификация (modifying) **235**
объявление новой (declaring new) **226–235**
Шрифтов классификация (font classification)
Стандартные PostScript-шрифты (Common PostScript fonts) **373**
готические шрифты (Old German fonts) **212**
шрифты Computer Modern **209**
шрифты Concrete Roman **210**
шрифты Pandora **211**
эйлеровы шрифты (Euler fonts) **213**
Шрифтов переключение (font selection) *см.* NFSS
Шрифтов семейство (font family) 184–187, 189, 190, **192**, 193, 194, 196, 197, 201, 206, 208, 217, **219**, 224–226, 229, 231–233, 236, 237, 243, 245, 247, 251
кодировка по умолчанию (default in encoding) **224**
модификация (modifying) **235**
объявление новых (declaring new) **226**, 232–233, 237, 243
Шрифтовые команды (font commands)
с аргументами (with arguments) 197–198
как окружения (as environments) 191
низкоуровневые (low-level) 217–225
установка по умолчанию (default settings) 200–201
Шрифты Almost Computer Modern 180
Шрифты AMS (AMS font package) 214, 239
Шрифты Computer Modern 180–181, 185, 193
классификация в NFSS **209**
Шрифты Concrete Roman 185, **210–211**
класстификация в NFSS **210**
Шрифты DC (DC fonts) **209**, 300
Шрифты EC (EC fonts) **209**, 300
Шрифты Mathtime (Mathtime fonts) 380, 383
Шрифты Pandora 206, **211**
классификация в NFSS **211**
Шрифты без засечек (sans serif fonts) 184

Эйлеровы шрифты (Euler fonts) **212–213**
классификация в NFSS **213**

Язык (language)
многоязычный (multi-language) 302
переключение (changing) 223
поддержка в BibTeX'e (support in BibTeX) 468, 469
поддержка в *MakeIndex*(support in *MakeIndex*) 397
Язык описания страницы (page description language) 350

Список таблиц

1 Пакеты, поддерживающие русификацию 7

2.1 Стандартные команды секционирования LaTeX'a 38
2.2 Синтаксис команд секционирования 38
2.3 Команды для стандартных заголовков разделов 49
2.4 Перечень параметров minitoc 58

3.1 Эффективные значения \baselinestretch для различных размеров шрифтов 71
3.2 Команды, управляющие окружением enumerate 75
3.3 Команды, управляющие перечнем itemize 77
3.4 Параметры длины, используемые в multicols 97
3.5 Счетчики, используемые в multicols 97

4.1 Опции стандартных листов бумаги в LaTeX 2ε 104
4.2 Значения по умолчанию параметров макета полосы набора формата letterpaper 105
4.3 Стиль полосы, определяемый командами LaTeX'a 115

5.1 Опции, помещаемые в преамбулу пакета array 125
5.2 Перечень команд окружения longtable 145
5.3 Сравнение определений таблиц для окружений longtable и supertabular . 149
5.4 Example of a threeparttable environment 154

6.1 Спецификаторы размещения плавающих объектов пакета float 172

7.1 Стандартные команды изменения размеров шрифтов 196
7.2 Команды и декларации переключения шрифтов 198
7.3 Параметры управления атрибутами текстовых шрифтов 200
7.4 Набор идентификаторов математических алфавитов, встроенных в NFSS . 204
7.5 Классификация шрифтов Computer Modern, принятая в NFSS 209
7.6 Семейство шрифтов Concrete 210
7.7 Семейство шрифтов Pandora 211
7.8 Семейства готических текстовых шрифтов 212
7.9 Эйлеровы семейства математических шрифтов 213
7.10 Классификация ширины очка литеры и жирности шрифтов 220
7.11 Классификация начертаний шрифтов 221

7.12 Стандартные схемы кодирования, используемые в NFSS 222
7.13 Типы математических символов 240

8.1 Идентификаторы математических алфавитов, подключаемые пакетом amsmath 255
8.2 Акценты в математическом режиме 255
8.3 Греческие буквы 255
8.4 Символы бинарных операций 255
8.5 Символы отношений 256
8.6 Стрелки 256
8.7 Разнородные символы 256
8.8 Символы больших операторов 256
8.9 Символы математических функций 257
8.10 Ограничители 257
8.11 Большие ограничители 257
8.12 Некоторые математические конструкции LaTeX'a 257
8.13 Буквы греческого и еврейского алфавитов 257
8.14 Символы типа ограничителей 257
8.15 Стрелки семейства AMS 258
8.16 Отрицания стрелок семейства AMS 258
8.17 Бинарные отношения семейства AMS 258
8.18 Отрицания бинарных отношений семейства AMS 259
8.19 Бинарные операторы семейства AMS 259
8.20 Разнородные символы семейства AMS 259
8.21 Команды, отвечающие за расстановку пробелов в формулах 280
8.22 Список существующих стилей теорем 293

9.1 Раскладка расширенного TeX'овского шрифта, принятая в г. Корк (Ирландия) в 1990 г. 300
9.2 Примеры перевода названий LaTeX'овских разделов документа в системе babel 307
9.3 Опции, поддерживаемые системой babel 307
9.4 Сравнение французского и английского печатных текстов 314

11.1 Основные опции программы dvips 356
11.2 Как поддерживаются dvi-драйверы различными пакетами 357
11.3 Поворот информации в окружении tabular 365
11.4 Поворот информации в окружении tabular при помощи окружения sidewaystable 366
11.5 Схема классификации имен файлов, отвечающих за шрифты, согласно Карлу Берри 372
11.6 Классификация базовых PostScript-шрифтов согласно NFSS (в скобках — имена файлов по Карлу Берри) 373
11.7 Шрифты, используемые пакетами системы PSNFSS 374
11.8 Символы PostScript-шрифта ZapfDingbats 375
11.9 Символы PostScript-шрифта Symbol 378
11.10 Соответствие латинских и греческих символов в PostScript-шрифте Symbol 378
11.11 Исходная раскладка шрифта Helvetica компании Adobe 381
11.12 TeX'овская раскладка DC-шрифта в применении к шрифту Helvetica . . . 381

12.1 Входные стилевые параметры для программы *MakeIndex* 401
12.2 Выходные стилевые параметры для программы *MakeIndex* 402

13.1 Подборка BibTeX'овских стилевых файлов 423
13.2 Список типов публикаций, принятых в большинстве BibTeX'овских стилей 455
13.3 Список стандартных полей в BibTeX'овском файле 456
13.4 Список команд BibTeX'овского стилевого файла 461
13.5 Список встроенных функций BibTeX'овского стилевого файла 462
13.6 BibTeX'овские стилевые файлы системы Delphi 469

14.1 Перечень команд пакета doc 477

A.1 (LA)TeX'овские единицы длины 504
A.2 Стандартные горизонтальные промежутки 505
A.3 Стандартные вертикальные промежутки 506
A.4 Команды, описывающие структуру пакетов и классов 518
A.5 Важнейшие внутренние переключатели типа `\boolean` 532

Список иллюстраций

1.1 Перечень основных файлов, необходимых для работы TEX'а и LATEX'а . . . 22

2.1 Пример преамбулы документа . . . 30
2.2 Структурирование документа, подготовленного в LATEX'е . . . 34
2.3 Иерархическая структура простого LATEX'овского документа . . . 36
2.4 Иерархическая структура сложного LATEX'овского документа . . . 37
2.5 Нумерация заголовков раздела . . . 39
2.6 Макет заголовка «в разрез» . . . 43
2.7 Макет заголовка «в подбор» . . . 44
2.8 Изменение стиля заголовка . . . 46
2.9 Соглашения относительно команды `\secdef` . . . 47
2.10 Создание оглавления . . . 51
2.11 Параметры, определяющие макет оглавления . . . 52
2.12 Мини-оглавление — пример ввода . . . 56
2.13 Мини-оглавление — пример вывода . . . 57

3.1 Абзац с увеличенным интерлиньяжем . . . 71
3.2 «Окно» в абзаце . . . 72
3.3 United Kingdom . . . 73
3.4 Абзац с рисунком «в оборку» . . . 73
3.5 Структура перечня в общем виде . . . 79
3.6 Окружение `alltt` . . . 85
3.7 Схематическое построение сносок . . . 89
3.8 Расположение текста и сносок при использовании пакета ftnright . . . 99

4.1 Макет полосы набора книги *The LATEX Companion* . . . 103
4.2 Стили полосы, использованные в *The LATEX Companion* . . . 111
4.3 Схематическое представление работы механизма LATEX'а расстановки маркеров . . . 114
4.4 Параметры макета полосы в пакете fancyheadings . . . 115
4.5 Макет, устанавливаемый по умолчанию в пакете fancyheadings . . . 117
4.6 Установка колонтитулов в книге LATEX book . . . 118

5.1 Сравнение окружений `TabularC` и `tabularx` . . . 136
5.2 Суры Корана (пример применения окружения `supertabular`) . . . 140
5.3 Суры Корана (пример применения окружения `supertabular*`) . . . 141
5.4 Суры Корана (пример применения окружения `longtable`) . . . 143

5.5 Сравнение результатов применения окружений `longtable` и `supertabular` . 148
5.6 Пример вложенных таблиц 159

6.1 Пример определения двух «нестандартных» плавающих объектов — `Program` и `algorithm` 171
6.2 Пример узких плавающих рисунков «в оборку» 174
6.3 Пример окружения `wrapfigure` 175
6.4 Три подрисунка 177

7.1 Основные характеристики шрифта 183
7.2 Сравнение букв с засечками и без засечек 184
7.3 Сравнение прямого и курсивного начертаний 185
7.4 Сравнение прописных букв и капители 186
7.5 Контурное и оттененное начертания 187
7.6 Шрифт 10-го кегля и шрифт, полученный масштабированием из 5-го кегля 188
7.7 Шрифт `yinit` Янниса Хараламбуса для буквиц 212
7.8 Таблица кодировки для шрифта `msbm7`, полученная программой `nfssfont.tex` 241

9.1 Даты и диалекты в системе babel 304
9.2 Пример опции german системы babel 308
9.3 Пример опции french системы babel 310

10.1 Пример гистограммы — плоский вариант 328
10.2 Пример гистограммы — объемный вариант 329
10.3 Пример рисунка, созданного при помощи фейнмановского пакета 334
10.4 Пример химической формулы 335
10.5 Символы элементов электронной схемы, подготовленные при помощи команд окружения `picture` 335
10.6 Команды для вычерчивания линий из пакетов epic и eepic 345
10.7 Графики, нарисованные при помощи пакетов epic и eepic 346

11.1 Примеры возможностей языка PostScript 353
11.2 Одиночный центрированный рисунок 359
11.3 Рисунок в черновом режиме 359
11.4 Европа перед 1990 г. 360
11.5 Центральная Америка 360
11.6 Карта мира 360
11.7 Поворот абзацев 362
11.8 Варианты рисунка в естественном виде, в окружении `turn` и в окружении `sideways` 367
11.9 Пример распечатки страницы с использованием шрифтов Computer Modern 382
11.10 Пример распечатки страницы с использованием шрифтов Mathtime 383
11.11 Пример распечатки страницы с использованием шрифтов Lucida Math . . 384

12.1 Последовательность этапов формирования указателя и вспомогательные файлы, используемые при этом LaTeX'ом и программой *MakeIndex* 386
12.2 Шаги формирования указателя 387
12.3 Пример размещения в тексте документа команд `\index` и вызова пакета showidx. Этот файл обрабатывается LaTeX'ом, затем запускается программа *MakeIndex*, после чего повторяется обработка LaTeX'ом 396
12.4 Указатель, сформированный при обработке текста, показанного на рис. 12.3. Все элементы указателя выведены на поле страницы, что облегчает их проверку, в частности обнаружение дублей 396

12.5 Пример использования специальных символов при обращении к программе *MakeIndex* . . . 405
12.6 Пример модификации выходного формата указателя . . . 407
12.7 Добавление отточий в указатель . . . 407
12.8 Пример входного файла для случая нескольких указателей . . . 411
12.9 Выходной файл с несколькими указателями . . . 411
12.10 Входной файл для случая нескольких указателей . . . 413
12.11 Формирование нескольких указателей — запуск LaTeX'а и пакета *MakeIndex* 414
12.12 Формирование нескольких указателей — пример вывода . . . 415

13.1 Обмен данными при работе BibTeX'а и LaTeX'а . . . 421
13.2 Пример LaTeX'овского файла, предназначенного для использования BibTeX'а . . . 425
13.3 Как LaTeX работает с BibTeX'овской библиографической базой данных . . 426
13.4 Пример BibTeX'овской базы данных . . . 427
13.5 Примеры использования BibTeX'овских стилей plain и unsrt . . . 428
13.6 Примеры использования BibTeX'овских стилей alpha и abbrv . . . 429
13.7 Примеры использования BibTeX'овских стилей acm и apalike . . . 430
13.8 Корневой файл и два подгружаемых файла с самостоятельными списками литературы . . . 432
13.9 Несколько списков литературы в одном файле (готовый документ) . . . 433
13.10 Пример исходного файла для пакета bibunits . . . 435
13.11 Создание нескольких списков литературы при помощи стиля bibunits . . . 436
13.12 Результат, полученный для примера, приведенного на рис. 13.10 . . . 437
13.13 Распечатка записей базы данных `bsample.bib`, полученная при помощи biblist . . . 439
13.14 Распечатка базы данных `bsample.bib`, полученная при помощи printbib . . 441
13.15 Распечатка базы данных `bsample.bib`, полученная при помощи showtags.sty и bibmods . . . 444
13.16 Адаптация BibTeX'овского стиля к голландскому языку . . . 467
13.17 Создание BibTeX'овского стиля для языка, отличного от английского . . . 467
13.18 Устранение влияния артиклей на результат BibTeX'овской сортировки . . 468

14.1 Пример управляющего файла пакета doc . . . 482
14.2 Пример документирования файла при помощи системы doc . . . 483
14.3 Документация, сгенерированная системой doc для примера на рис. 14.2 . . 484
14.4 Пакетный файл для системы «file-error» . . . 488
14.5 Файл протокола инсталляции системы «file-error» . . . 489
14.6 Исходный файл пакета doc для системы «file-error» . . . 490
14.7 Вывод, полученный в результате обработки LaTeX'ом файла `fileerr.dtx` . 491

A.1 Пример титульной страницы (Джеффри Чосер, *Кентерберийские рассказы*, Лондон, 1400) . . . 507
A.2 Пример файла-класса, являющегося расширением класса article . . . 526

B.1 Пример сеанса связи ftp с TeX-архивом CTAN в Эстоне (часть 1) . . . 536
B.2 Пример сеанса связи ftp с TeX-архивом CTAN в Эстоне (часть 2) . . . 537
B.3 Пример TeX'нических mail-серверов . . . 538

Оригинал-макет *The LaTeX Companion*

Книга *Companion* была набрана авторами на рабочих станциях Hewlett-Packard и Sun в CERN'е (Женева) и на станции DEC в Майнце, при этом использовалась одна из ранних версий LaTeX 2_ε. Обмен файлами между этими двумя сайтами происходил через Internet по электронной почте. Файлы были перетранслированы в PostSsript при помощи dvips; корректуры распечатывались на принтерах Apple LaserWriter Pro и Xerox Docutech. Для издательских редакторов и корректоров PS-файлы были переданы через ftp в Ридинг на компьютеры Sun издательства Addison-Wesley, чтобы там была возможность распечатать эти файлы, благодаря чему процесс внесения правки ускорился.

Окончательный вариант книги (от начала до конца) был получен за единый цикл работы большого LaTeX'а. Указатель был сгенерирован из файла `.idx` LaTeX'а с использованием программы *MakeIndex*; библиография была подготовлена при помощи BibTeX'а. Для того чтобы все перекрестные ссылки получились правильными, потребовалось 4 раза обработать файлы LaTeX'ом. Полученный в результате PostScript-файл размером 9.5 Mb отдельными частями был перекопирован через ftp в Ридинг, где оригинал-макет был выведен на пленку в фотоавтомате Varityper 4300P с разрешением 1200 dpi.

Оформление обложки разработал дизайнер Марк Онг (Mark Ong). Франк Миттельбах представил указания дизайнера в виде файла класса LaTeX 2_ε. Основной текст набран шрифтом Lucida Bright 9.5/12pt, разработанным Бигелоу и Холмсом (Bigelow & Holmes), рубленым шрифтом Lucida Sans тех же авторов и моноширинным шрифтом Computer Modern Typewriter 10/12pt Дональда Кнута. Шрифт для заголовков глав — Lucida Bright bold 36/38pt, а для разделов — тот же, но мельче: 14/15pt.

Изготовление оригинал-макета книги было испытанием как для авторов, так и для LaTeX. Когда мы пытались использовать все пакеты одновременно при едином прогоне через LaTeX, в ряде пакетов обнаружились ошибки.

В этой книге мы придерживались политики разделения структуры и формы и предполагали, что явная разметка исключается всюду, где это возможно. Таким образом, после набора этой книги настало, возможно, время спросить: насколько хорошо LaTeX выполняет работу, если она задана в файле класса, отвечающем за дизайн, и в файле документа? Для тех, кто любит цифры, приведем некоторую статистику на примере нашей книги: ниже в таблице указан объем ручного форматирования, который пришлось выполнить на заключительном этапе изготовления оригинал-макета *Companion*.

Чтобы сигнализировать обо всех случаях явного форматирования (тогда их всегда легко найти и удалить, если это нужно), мы никогда не пользовались стандартными командами LaTeX'а. Вместо этого мы определили собственное множество команд, иногда просто задав нечто вроде `\newcommand{\finalpagebreak}{\pagebreak}`.

Приведенные в таблице команды подразделены на три группы. Команды первой группы имеют дело с длиной полосы: `\finallongpage` и `\finalshortpage` оставляют полосу соответственно длиннее или короче на одну строку (`\baselineskip`). Команда `\finalforcedpage` представляет собой приложение команды `\enlargethispage*` и, за ней, следовательно, всегда стоит явный разрыв страницы в исходном файле. Механизм определения таких команд объясняется в разд. 4.4. Во второй группе содержатся команды «коррекции» решений LaTeX'а относительно того, где начинать новую страницу, и последняя группа состоит из единственной команды «тонкой настройки», добавляющей крошечные пробелы по вертикали для улучшения зрительного восприятия.

Среднее число коррекций, сделанное командами первой группы, чуть выше 20%, т. е. одна на пять разворотов, так как эти изменения всегда касались обеих страниц на развороте. Если вы заглянете в главы, процент коррекций в которых достаточно высок, то убедитесь, что там или очень длинные примеры, или большие таблицы, которые нужно разместить в конкретном месте.

Жесткий разрыв страницы вводился в среднем на каждой 10-й странице, часто в паре с командой из первой группы. В большинстве случаев это касалось уменьшения количества строк на странице.

Как правило, `\finalfixedskip` можно классифицировать как «коррекция погрешностей при реализации дизайна». В среднем их порядка 16%, и эта цифра может показаться высокой. Но на самом деле такие микроулучшения почти всегда имеют пару, так что это соответствует примерно одной коррекции на 12 страниц.

В заключение мы должны сказать, что LaTeX успешно справился с таким сложным материалом. Даже без дополнительного ручного форматирования большинство страниц получились неплохо. Только читатель может оценить, нужны ли были эти дополнительные усилия.

Глава	2	3	4	5	6	7	8	9	10	11	12	13	14	A1
Число страниц	36	36	18	40	16	58	44	16	36	36	26	50	18	36
`\finallongpage`	0	3	1	0	3	10	4	2	3	0	4	9	7	4
`\finalshortpage`	0	5	4	4	0	2	10	0	0	8	6	0	0	2
`\finalforcedpage`	1	0	0	2	2	0	1	0	0	1	0	1	0	0
Изменение длины страницы	1	8	5	6	5	12	15	2	3	9	10	10	7	6
В среднем на страницу	.03	.22	.29	.15	.33	.08	.34	.13	.08	.25	.38	.2	.39	.17
`\finalpagebreak`	4	5	2	4	3	7	12	1	0	6	4	5	3	6
`\finalnewpage`	0	1	0	0	0	0	0	0	0	0	0	1	0	0
Изменение разрыва страницы	4	6	2	4	3	7	12	1	0	6	4	6	3	6
В среднем на страницу	.11	.17	.11	.1	.19	.12	.27	.06	0	.17	.15	.12	.17	.17
`\finalfixedskip`	4	3	4	11	0	8	2	2	0	14	6	10	7	3
В среднем на страницу	.11	.08	.22	.28	0	.14	.05	.13	0	.39	.23	.2	.38	.08
Сумма	9	17	11	21	8	27	29	5	3	29	20	26	17	15
В среднем на страницу	.25	.47	.61	.53	.5	.47	.66	.31	.08	.81	.77	.52	.94	.42

Оглавление

Предисловие редактора перевода 5

Предисловие 11

1 Введение 19
1.1 Краткая история TEX'а и LATEX'а . . . 19
1.1.1 Вначале был TEX . . . 19
1.1.2 Потом Лесли Лэмпорт придумал LATEX . . . 20
1.1.3 С LATEX'ом в 2000 год? . . . 21
1.2 LATEX и его составляющие . . . 21
1.2.1 Как работает LATEX? . . . 21
1.2.2 Выходные процессоры (драйверы dvi) . . . 24
1.3 Концепция общей разметки . . . 25
1.3.1 Что такое общая разметка? . . . 25
1.3.2 Преимущества общей разметки . . . 26
1.3.3 Разделение содержания и формы . . . 27
1.4 Необходимость локальной разметки . . . 27
1.4.1 Недостатки локальной разметки . . . 27
1.4.2 Когда использовать локальную разметку . . . 28

2 Структура документа, подготовленного в LATEX'е 29
2.1 Структура исходного файла . . . 29
2.1.1 Обработка опций и пакетов . . . 31
2.1.2 Разделение исходного файла на части . . . 34
2.1.3 Комбинирование нескольких файлов . . . 35
2.2 Логическая структура . . . 35
2.3 Команды секционирования . . . 36
2.3.1 Нумерация заголовков . . . 38
2.3.2 Форматирование заголовков . . . 42
2.3.3 Изменение стандартных заголовков . . . 48
2.4 Структура оглавления . . . 49
2.4.1 Набор оглавления . . . 50
2.4.2 Ввод информации в файлы оглавления . . . 53
2.4.3 Определение нового файла, аналогичного .toc . . . 54
2.4.4 Сложные оглавления . . . 55
2.5 Управление ссылками . . . 58
2.5.1 varioref — более гибкие ссылки . . . 60
2.5.2 Ссылки на внешние документы . . . 64

3 Основные средства форматирования 65
3.1 Фразы и абзацы . . . 66
3.1.1 letterspace — изменение межбуквенных интервалов . . . 66
3.1.2 ulem — выделение посредством подчеркивания . . . 67
3.1.3 xspace — гибкий пробел после макро . . . 68
3.1.4 Выравнивание внутри абзаца . . . 69
3.1.5 doublespace — изменение интерлиньяжа . . . 70
3.1.6 picinpar — набор абзацев с прямоугольными окнами . . . 71
3.1.7 shapepar — набор абзацев необычной формы . . . 72
3.2 Структуры перечня . . . 74
3.2.1 Модификация стандартных перечней . . . 74
3.2.2 Создание собственных перечней . . . 78
3.3 Подражание машинописному шрифту . . . 84
3.3.1 `alltt` — окружение типа `verbatim` . . . 84
3.3.2 verbatim — стиль для литературного текста . . . 84
3.3.3 moreverb — дополнительные окружения типа `verbatim` . . . 85
3.4 Примечания: подстрочные, на полях, выносные . . . 88
3.4.1 Создание сносок . . . 88
3.4.2 Примечания на полях . . . 92
3.4.3 Выносные примечания . . . 93
3.5 Использование многоколонного набора . . . 94
3.5.1 multicol — гибкий способ работы с многоколонным документом 94
3.5.2 Набор текста в колонках . . . 95
3.5.3 Создание окружения `multicols` . . . 96
3.5.4 Плавающие объекты и сноски в multicol . . . 98
3.5.5 ftnright — сноски в правой колонке при двухколонном окружении 98
3.6 Простое управление версиями . . . 100

4 Макет полосы набора 101
4.1 Геометрические параметры макета полосы набора . . . 102
4.2 Изменение макета . . . 105
4.2.1 Пакеты для создания макета полосы набора . . . 107
4.2.2 Горизонтальное расположение полос набора при печати . . . 109
4.3 Стили полосы . . . 110
4.3.1 Написание новых стилей полосы . . . 112
4.3.2 Создание стиля полосы при помощи fancyheadings . . . 114
4.4 Явное форматирование . . . 117

5 Таблицы 120
5.1 Сравнение окружений `tabbing` и `tabular` . . . 121
5.2 Использование окружения `tabbing` . . . 122
5.2.1 Окружение `program` . . . 123
5.3 array — расширение окружений `tabular` . . . 124
5.3.1 Примеры команд преамбулы . . . 124
5.3.2 Стилевые параметры . . . 129
5.3.3 Определение новых спецификаторов колонок . . . 131
5.3.4 Некоторые особенности реализации пакета array . . . 132
5.3.5 tabularx — автоматическое вычисление ширины колонок . . . 133
5.3.6 delarray — определение вида ограничителей для окружения `array` 137
5.4 Многостраничные таблицы . . . 138
5.4.1 supertab — верстка многостраничных таблиц . . . 138
5.4.2 longtable — усложненные многостраничные таблицы . . . 142
5.4.3 Завершающее сравнение окружений `supertabular` и `longtable` 147

5.5 Дополнительные штрихи . . . 147
5.5.1 dcolumn — управление выравниванием в колонках таблицы . . . 147
5.5.2 hhline — комбинирование горизонтальных и вертикальных отрезков . . . 151
5.6 Приложения . . . 152
5.6.1 Переносы в узких колонках . . . 152
5.6.2 Сноски в таблицах . . . 153
5.6.3 Таблицы с широкими графами . . . 154
5.6.4 Колонки, занимающие несколько строк таблицы . . . 155
5.6.5 Таблицы внутри таблиц . . . 157
5.6.6 Еще два примера . . . 160

6 Плавающие объекты 162
6.1 Параметры плавающих объектов . . . 162
6.2 Улучшенное размещение плавающих объектов . . . 166
6.3 float — создание новых видов плавающих объектов . . . 169
6.3.1 Разместить плавающий объект «здесь»! . . . 171
6.4 Другие виды плавающих окружений . . . 173
6.4.1 floatfig — узкие плавающие рисунки «в оборку» . . . 173
6.4.2 wrapfig — неплавающие рисунки «в оборку» . . . 174
6.4.3 subfigure — рисунки внутри рисунков . . . 176
6.4.4 endfloat — размещение рисунков и таблиц в конце документа . . . 176
6.5 Создание своих названий . . . 178

7 Переключение шрифтов 180
7.1 Введение в NFSS . . . 180
7.2 Характеристики шрифтов . . . 182
7.2.1 Моноширинные и пропорциональные шрифты . . . 183
7.2.2 Шрифты с засечками и без засечек . . . 184
7.2.3 Семейства шрифтов и их атрибуты . . . 184
7.2.4 Схемы кодирования . . . 189
7.3 Переключение шрифтов в тексте . . . 190
7.3.1 Стандартные шрифтовые команды NFSS . . . 191
7.3.2 Комбинирование стандартных команд управления шрифтами . . . 196
7.3.3 Сравнение командного и декларативного способов переключения шрифтов . . . 197
7.3.4 Доступ ко всем литерам шрифта . . . 199
7.3.5 Изменение значений по умолчанию для атрибутов текстовых шрифтов . . . 200
7.3.6 Шрифтовые команды LaTeX 2.09 . . . 202
7.4 Переключение шрифтов в формулах . . . 202
7.4.1 Специальные идентификаторы математических алфавитов . . . 203
7.4.2 Текстовые шрифтовые команды при наборе математических формул . . . 206
7.4.3 Версии математических формул . . . 207
7.5 Стандартные пакеты . . . 208
7.5.1 Добавление новых текстовых шрифтов . . . 209
7.5.2 Подключение новых математических шрифтов . . . 212
7.5.3 slides — получение демонстрационных слайдов . . . 214
7.5.4 Обработка ранее созданных документов . . . 214
7.5.5 Специальные пакеты для NFSS . . . 215
7.6 Низкоуровневый интерфейс . . . 217
7.6.1 Установка индивидуальных шрифтовых атрибутов . . . 218

7.6.2 Установка значений для нескольких шрифтовых атрибутов . . 223
7.6.3 Автоматические подстановки шрифтов 224
7.6.4 Использование низкоуровневых команд в документе 225
7.7 Подключение новых шрифтов 225
7.7.1 Общая схема 225
7.7.2 Объявление новых семейств шрифтов и групп начертаний шрифтов 226
7.7.3 Параметры управления загрузкой шрифтов 235
7.7.4 Ввод определений новых схем кодирования 235
7.7.5 Внутренняя организация файла 236
7.7.6 Объявление новых шрифтов для математических формул . . . 238
7.7.7 Порядок записи деклараций 243
7.8 Предупреждения и сообщения об ошибках 244

8 Высшая математика 252
8.1 Создание $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX'а 252
8.2 Шрифты и символы в формулах 253
8.2.1 Команды для математических шрифтов 253
8.2.2 Математические символы 254
8.3 Составные символы, ограничители и операторы 260
8.3.1 Кратные интегралы 260
8.3.2 Стрелки сверху и снизу 260
8.3.3 Многоточия 261
8.3.4 Двойные акценты 261
8.3.5 Акценты как верхние индексы 262
8.3.6 Акценты в виде точек 262
8.3.7 Корни 262
8.3.8 Формулы в рамке 262
8.3.9 Растяжимые стрелки 263
8.3.10 Команды `\overset`, `\underset` и `\sideset` 263
8.3.11 Команда `\smash` 264
8.3.12 Команда `\text` 264
8.3.13 Названия новых операций 265
8.3.14 Команда `\mod` и ее аналоги 266
8.3.15 Дроби и родственные конструкции 266
8.3.16 Непрерывные дроби 268
8.3.17 Ог-г-г-громные ограничители 268
8.4 Окружения типа матрицы и коммутативные диаграммы 269
8.4.1 Окружение `cases` 269
8.4.2 Окружения типа `matrix` 269
8.4.3 Команда `\substack` 271
8.4.4 Коммутативные диаграммы 271
8.5 Выравнивание многострочных формул 272
8.5.1 Несколько формул без выравнивания 273
8.5.2 Несколько формул с выравниванием 274
8.5.3 Разбитые на части формулы без выравнивания 275
8.5.4 Разбитые на части формулы с выравниванием 275
8.5.5 Окружения выравнивания для набора отдельных частей выключных формул 276
8.5.6 Вертикальные пробелы и разрывы страниц при наборе формул 277
8.5.7 Команда `\intertext` 277
8.6 Разное 278
8.6.1 Нумерация формул 278

8.6.2 Установка счетчика формул . . . 279
8.6.3 Подчиненная нумерация формул . . . 279
8.6.4 Тонкая настройка в математическом режиме . . . 280
8.6.5 На что еще обратить внимание . . . 280
8.6.6 Опции к пакету amsmath и отдельные его составляющие . . . 281
8.6.7 Классы документов $\mathcal{AMS}$-LaTeX'а . . . 283
8.7 Примеры многострочных формул . . . 283
8.7.1 Окружение split . . . 283
8.7.2 Окружение multline . . . 286
8.7.3 Окружение gather . . . 287
8.7.4 Окружение align . . . 287
8.7.5 Использование окружений align и split внутри gather . . . 288
8.7.6 Использование окружений alignat . . . 289
8.8 Расширения для окружения theorem . . . 290
8.8.1 Как определять новые окружения типа теоремы . . . 291
8.8.2 Примеры определений и использования теорем . . . 293
8.8.3 Некоторые специальные вопросы . . . 294
8.9 Параметры, задающие математические стили . . . 294
8.9.1 Как управлять размерами символов . . . 294
8.9.2 Параметры математических стилей в LaTeX'е . . . 296

9 LaTeX в многоязычной среде 298
9.1 TeX и языки, отличные от английского . . . 298
9.1.1 Механизм виртуального шрифта . . . 301
9.2 Babel — LaTeX владеет многими языками . . . 302
9.2.1 Среда пользователя . . . 302
9.2.2 Опция german . . . 303
9.2.3 Структура языковых стилевых файлов системы babel . . . 305
9.3 Следование типографским нормам . . . 312
9.3.1 Традиционные французские типографские нормы . . . 312
9.3.2 Команды пакета french . . . 313
9.3.3 Структура пакета french . . . 314

10 Графика в LaTeX'е 315
10.1 Орнаменты . . . 317
10.1.1 Министраницы в рамке . . . 317
10.1.2 Рамки с тенью . . . 317
10.1.3 Декоративные рамки . . . 318
10.2 Окружение picture . . . 320
10.2.1 Аппроксимации Безье . . . 320
10.2.2 Как совмещать несколько рамок . . . 321
10.2.3 Как рисовать бинарные и тернарные деревья . . . 322
10.2.4 Как рисовать гистограммы . . . 324
10.2.5 Примеры окружения barenv . . . 326
10.2.6 Как рисовать произвольные кривые . . . 327
10.2.7 Другие пакеты . . . 333
10.3 epic — усовершенствования к окружению picture . . . 333
10.3.1 Описание команд . . . 335
10.4 Расширение пакета epic . . . 340
10.4.1 Расширения LaTeX'а при помощи eepic . . . 341
10.4.2 Расширения пакета epic при помощи eepic . . . 341
10.4.3 Новые команды в пакете eepic . . . 342

10.4.4 Совместимость . . . 343
10.4.5 Примеры . . . 344
10.5 Пакеты, основанные на epic . . . 344
10.5.1 Как рисовать двудольные графы . . . 344
10.5.2 Как рисовать деревья . . . 348

11 Как использовать PostScript 350
11.1 Язык PostScript . . . 350
11.1.1 Несколько слов о языке . . . 350
11.1.2 Что такое инкапсулированный PostScript? . . . 352
11.2 dvips — преобразование dvi в PostScript . . . 354
11.3 Совмещение текста и графики в формате PostScript . . . 355
11.3.1 Простые рисунки . . . 358
11.3.2 Черновые рисунки . . . 359
11.3.3 Более сложная организация рисунков . . . 359
11.4 Как повернуть материал . . . 361
11.4.1 Как повернуть табличный материал . . . 363
11.4.2 Как повернуть рисунок . . . 364
11.4.3 Как повернуть только подпись к рисунку . . . 365
11.5 Отчеркивания на полях . . . 365
11.5.1 Среда пользователя . . . 367
11.5.2 Параметры команд пакета changebar . . . 369
11.5.3 Недостатки и неточности . . . 369
11.6 Обрамление и затенение . . . 370
11.7 Цветной вывод . . . 371
11.8 Наложение текста на выводимую страницу . . . 371
11.9 Еще одно обращение к NFSS . . . 371
11.9.1 Как называются эти тысячи шрифтов . . . 371
11.9.2 Система PSNFSS . . . 372
11.9.3 Как использовать Pi-шрифты из PostScript'а . . . 375
11.9.4 Общие команды в пакете pifont . . . 376
11.9.5 Шрифт Symbol . . . 377
11.9.6 Как самостоятельно установить новые PostScript-шрифты . . . 378
11.9.7 Как заменить все TEX'овские шрифты PostScript-шрифтами . . . 379
11.10 DCPS — корковская кодировка с PostScript-шрифтами . . . 380

12 Создание указателей 385
12.1 Синтаксис описания элементов указателя . . . 387
12.1.1 Простые описания элементов указателя . . . 387
12.1.2 Формирование подчиненных элементов . . . 388
12.1.3 Диапазоны страниц и перекрестные ссылки . . . 389
12.1.4 Управление формой представления указателя . . . 390
12.1.5 Печать специальных символов . . . 391
12.1.6 Некоторые дополнительные замечания . . . 391
12.1.7 Согласованность элементов указателя . . . 393
12.2 Подготовка указателя . . . 394
12.2.1 Формирование полуфабриката указателя . . . 394
12.2.2 Получение форматированного указателя . . . 394
12.3 Запуск программы *MakeIndex* . . . 395
12.3.1 Опции программы *MakeIndex* . . . 395
12.3.2 Сообщения об ошибках . . . 398
12.4 Изменение вида указателя . . . 400
12.4.1 Пример стилевых файлов указателя . . . 400

12.4.2 Указатель, формируемый отдельно . . . 404
12.4.3 Изменение специальных символов . . . 404
12.4.4 Изменение выходного формата указателя . . . 405
12.4.5 Обработка нетрадиционных номеров страниц . . . 406
12.4.6 Глоссарий . . . 408
12.5 Изменение макета указателя . . . 409
12.5.1 Формирование нескольких указателей . . . 410
12.5.2 Модификация команд формирования указателей . . . 410

13 Создание списка литературы 416
13.1 Создание библиографических ссылок . . . 417
13.1.1 Придание ссылкам требуемого вида . . . 418
13.1.2 Выбор формата меток . . . 420
13.2 Совместное использование BibTeX'а и LaTeX'а . . . 420
13.2.1 Список стилевых файлов для BibTeX'а . . . 422
13.2.2 Примеры BibTeX'овских стилей . . . 424
13.3 Документы с несколькими списками литературы . . . 431
13.3.1 Пакет chapterbib . . . 431
13.3.2 Пакет bibunits . . . 434
13.4 Средства управления библиографическими базами данных . . . 438
13.5 Формат .bib-файлов . . . 443
13.5.1 Общая структура записей в BibTeX'овской базе данных . . . 443
13.5.2 Текстовая часть поля . . . 445
13.5.3 BibTeX и аббревиатуры . . . 450
13.5.4 BibTeX'овская преамбула . . . 451
13.5.5 Перекрестные ссылки . . . 452
13.5.6 Дополнительные замечания . . . 453
13.6 Формат записей базы данных (подробное описание) . . . 453
13.7 Как устроены BibTeX'овские стили . . . 458
13.7.1 Общее представление о BibTeX'овских стилевых файлах . . . 458
13.7.2 Команды BibTeX'овского стилевого файла . . . 459
13.7.3 Встроенные функции . . . 459
13.7.4 Стилевой файл-документация btxbst.doc . . . 459
13.8 Модификация стилевых файлов . . . 463
13.8.1 Добавление нового поля . . . 464
13.8.2 Поддержка языков, отличных от английского . . . 466
13.9 Пакет makebst для модификации библиографических стилей . . . 469
13.9.1 Работа с программой makebst . . . 470

14 Средства документирования макропакетов 472
14.1 Макропакеты с документацией . . . 472
14.2 Пользовательский интерфейс пакета doc . . . 473
14.2.1 Основные понятия . . . 473
14.2.2 Описание новых макрокоманд и окружений . . . 474
14.2.3 Перекрестные ссылки между используемыми макрокомандами 475
14.2.4 Заключительные процедуры создания указателя . . . 475
14.2.5 Дополнительные возможности . . . 475
14.2.6 Управляющий файл . . . 481
14.2.7 Простой пример файла, документированного при помощи пакета doc . . . 482

14.3 Утилита DOCSTRIP 485
14.3.1 Команды, используемые в пакетных файлах 485
14.3.2 Условное включение кода 487
14.4 Пример инсталляционной процедуры 487

A LaTeX: основы программирования **492**
A.1 Разметка и форматирование 492
A.1.1 Определение новых команд 493
A.1.2 Определение новых окружений 496
A.1.3 Создание счетчиков и изменение их текущих значений 499
A.1.4 Управление параметрами расстояния 502
A.2 Разметка страниц: различные типы боксов 507
A.2.1 LR-боксы 508
A.2.2 Боксы-абзацы 510
A.2.3 Боксы-линейки 514
A.2.4 Работа с боксами 515
A.3 Структура пакетов и классов в LaTeX 2ε 517
A.3.1 Команды идентификации 519
A.3.2 Начальный командный код 520
A.3.3 Декларация опций 520
A.3.4 Исполнение опций 522
A.3.5 Загрузка макропакетов 523
A.3.6 Основной командный код 524
A.3.7 Команды, специально предназначенные для макропакетов и классов 524
A.3.8 Команды, специально предназначенные для классов 525
A.4 calc — макропакет для арифметических вычислений 527
A.5 ifthen — усовершенствованный условный переход 529

B TeX'ническое обеспечение и группы пользователей **534**
B.1 Главные сайты TeX'а в Internet'е 534
B.2 Mail-серверы 538
B.3 Группы пользователей TeX'а 538

Список литературы **541**

Именной указатель **555**

Предметный указатель **557**

Список таблиц **591**

Список иллюстраций **594**

Оригинал-макет *The LaTeX Companion* **597**