Справочник по командам LaTeX 2ε*

Описаны средства разметки документа и форматирования текста в LaTeX 2ε, необходимые для подготовки публикаций на русском и английском языках. Дано также детальное описание средств из ряда пакетов, расширяющих стандартный LaTeX.

Средства LaTeX и AMS-LaTeX для печати математики описаны (вместе с многочисленными примерами) во второй части справочника «Набор математических формул в LaTeX 2ε».

Содержание

1 Входной файл **4**
- 1.1 Преамбула . . . 4
- 1.2 Текст документа . . . 4
  - 1.2.1 Включение в документ текста из других файлов . . . 4
- 1.3 Опции стандартных классов . . . 5
- 1.4 Поддержка русского языка . . . 6
- 1.5 Комментарии . . . 6
- 1.6 Включение дополнительных файлов . . . 7

2 Печатный документ **7**
- 2.1 Параметры страницы . . . 7
- 2.2 Титульная страница и аннотация . . . 8
- 2.3 Секционирование документа . . . 9
  - 2.3.1 Части . . . 9
  - 2.3.2 Главы . . . 9
  - 2.3.3 Разделы . . . 10
  - 2.3.4 Параграфы . . . 10
  - 2.3.5 Разделение книги на части . . . 10
  - 2.3.6 Приложения . . . 10
- 2.4 Оглавление, списки рисунков и таблиц . . . 10
- 2.5 Колонтитулы . . . 11
  - 2.5.1 Нестандартные колонтитулы . . . 12
- 2.6 Перекрёстные ссылки . . . 12
  - 2.6.1 Пакет xr . . . 13
- 2.7 Сноски . . . 13
- 2.8 Цитирование литературы . . . 14
- 2.9 Нумерация страниц . . . 15
- 2.10 Печать в две колонки . . . 15
  - 2.10.1 Пакет multicol . . . 15
- 2.11 Алфавитный указатель . . . 16
  - 2.11.1 Дополнительный указатель терминов . . . 18

3 Счётчики **18**

*© 2002 Владимир Сюткин. Замечания приветствуются: syutkin@ns.kinetics.nsc.ru

4 Командные длины **19**

5 Вставка пробелов **20**
5.1 Горизонтальные промежутки . . . 20
5.2 Вертикальные промежутки . . . 21

6 Как управлять форматированием абзацев **21**
6.1 Изменение режима форматирования . . . 21
6.2 Отступ в начале абзаца . . . 22
6.3 Междустрочный интервал . . . 22
6.4 Переносы слов . . . 23
6.5 Разрыв строки . . . 23

7 Как управлять форматированием страниц **24**
7.1 Переключение режима форматирования . . . 24
7.2 Разбиение текста на страницы . . . 24
7.3 Изменение высоты области с текстом . . . 24
7.4 Переход на новую страницу . . . 24

8 Списки **25**
8.1 Маркированные списки . . . 25
8.2 Нумерованные списки . . . 25
8.3 Списки описаний . . . 26
8.4 Настраиваемые списки . . . 26
8.5 Примитивный список . . . 27

9 Специальные абзацы **28**
9.1 Позиционирование текста в строке . . . 28
9.2 Выделение текста . . . 28
9.3 Буквальное воспроизведение текста . . . 29
9.3.1 Пакет shortvrb . . . 29
9.3.2 Пакет alltt . . . 29

10 Боксы **30**
10.1 Строковые боксы . . . 30
10.1.1 Измерение размеров бокса . . . 30
10.1.2 Сдвиг бокса по вертикали . . . 30
10.1.3 Предварительное форматирование бокса . . . 30
10.2 Текстовые боксы . . . 31
10.3 Линейка . . . 31

11 Таблицы **31**
11.1 Пакет array . . . 33
11.2 Таблицы заданной ширины . . . 34
11.3 Размещение таблицы на нескольких страницах . . . 35

12 Плавающие объекты **36**
12.1 Рисунки и таблицы . . . 36
12.1.1 Количество плавающих объектов на странице . . . 37
12.1.2 Доля страницы, отводимая под плавающие объекты . . . 37
12.1.3 Вертикальные пробелы вокруг плавающих объектов . . . 38
12.2 Рисунки и таблицы, обтекаемые текстом . . . 38
12.3 Заметки на полях . . . 39

13 Текстовые шрифты **39**
13.1 Атрибуты шрифтов . . . 39
13.2 Переключение гарнитуры шрифта . . . 39
13.3 Переключение насыщенности шрифта . . . 40
13.4 Переключение начертания шрифта . . . 40
13.5 Переключение размера шрифта . . . 41
13.6 Включение произвольного шрифта . . . 41
13.7 Переключение на основной шрифт документа . . . 42

14 Новые макроопределения **42**
14.1 Команды . . . 42
14.2 Командные скобки . . . 42
14.3 *-форма декларации макроопределений . . . 42

15 Символы **43**
15.1 Служебные символы . . . 43
15.2 Национальные символы европейских алфавитов . . . 43
15.3 Кавычки . . . 43
15.4 Дефис и тире . . . 44
15.5 Дополнительные символы . . . 44
15.6 Пакет textcomp . . . 45

Алфавитный указатель **46**

Список иллюстраций

1 Параметры компоновки страницы . . . 7
2 Параметры компоновки списка . . . 26

Список таблиц

1 Декларации переключения размера шрифтов . . . 41
2 Диакритические знаки (акценты). . . . 43
3 Особые европейские символы. . . . 43
4 Особые европейские символы из кодировки T1. . . . 43
5 Специальные символы. . . . 44
6 Математические символы текстовой моды. . . . 44
7 Математические символы пакета textcomp. . . . 45
8 Научные символы пакета textcomp. . . . 45

1 Входной файл

1.1 Преамбула

Входной файл должен начинаться с преамбулы. Преамбула начинается с декларации

`\documentclass[`*options*`]{`*class*`}[`*release-date*`]`

которая задаёт класс документа: LaTeX читает файл *class*`.cls`, содержащий определения команд, специфических для выбранного типа документа. Необязательный аргумент *options* позволяет изменить значения ряда параметров и некоторые правила форматирования, принятые по умолчанию для этого класса. Опции в списке *options* перечисляются через запятую. Необязательный аргумент *release-date* позволяет указать дату наиболее старой пригодной версии файла *class*`.cls`. Дата задаётся в формате «год/месяц/день». Пример:

```
\documentclass[a4paper,12pt]{article}[2000/05/19]
```

В LaTeX включены так называемые стандартные классы для поддержки наиболее популярных типов документа: `article` (статья), `proc` (доклад), `book` (книга), `report` (отчёт), `letter` (письмо) и `slides` (слайды).

После `\documentclass` для расширения базовой версии LaTeXа используются декларации

`\usepackage[`*options*`]{`*package*`}[`*release-date*`]`

Декларация `\usepackage` стимулирует LaTeX читать файл *package*`.sty`, содержащий переопределения уже имеющихся команд и определения новых команд. Аргументы *options* и *release-date* имеют то же предназначение, что и у `\documentclass`. Количество деклараций `\usepackage` не ограничено. Одной декларацией можно загрузить сразу несколько пакетов, если, конечно, для каждого из них требуются одинаковые опции. Пример:

```
\usepackage[dvips]{graphicx,color}
```

Опции пакетов можно указывать также в аргументе *options* команды `\documentclass`.

Кроме описанных выше деклараций, в преамбуле обычно размещают всё то, что само ничего не печатает. Например, только в преамбуле документа может находиться декларация

`\nofiles`

которая запрещает создавать любые служебные файлы.

1.2 Текст документа

Текст документа размещается за преамбулой в командных скобках

`\begin{document} ... \end{document}`

Всё, что следует после `\end{document}`, LaTeX игнорирует.

1.2.1 Включение в документ текста из других файлов

Любая часть документа может хранится не только во входном файле. Команда

`\input{`*file*`}`

позволяет включить в документ содержимое файла *file*. По умолчанию подразумевается расширение `tex`. LaTeX читает файл *file* от начала до конца или до команды

`\endinput`

Команду `\input` можно использовать и в преамбуле входного файла. В частности, сама преамбула может находиться в файле *file*.

Наряду с \input, имеется команда

\include{*file*}

которая также позволяет включить в документ содержимое файла *file*.tex. Команды \include должны находиться внутри окружения document. Перед и после вставки в документ содержимого файла LaTeX начинает новую страницу (исполняется команда \clearpage). LaTeX читает файл только тогда, когда его имя *file* указано в аргументе декларации

\includeonly{*files*}

которая должна находиться в преамбуле. Имена файлов в списке *files* перечисляются через запятую. Если для какой-нибудь команды \include имени файла нет в этом списке, то LaTeX просто переходит на новую страницу.

Список файлов, которые читает LaTeX при обработке входного файла можно вывести в файл протокола (имеет расширение log), поместив в преамбуле команду

\listfiles

1.3 Опции стандартных классов

Необязательный аргумент *options* у команды \documentclass позволяет изменить значения ряда параметров и некоторые правила форматирования, принятые по умолчанию для разных классов. Опции в списке *options* перечисляются через запятую. Ниже приведён полный набор опций стандартных классов.

10pt|11pt|12pt задают размер базового шрифта документа. По умолчанию используется 10pt. Отсутствуют в классе slides.

a4paper|a5paper|b5paper|letterpaper|legalpaper|executivepaper задают размер страницы документа для печати на листах бумаги разного формата. По умолчанию используется letterpaper. a5paper и b5paper не поддерживаются в классе proc.

landscape устанавливает альбомную ориентацию страницы вместо книжной.

oneside|twoside устанавливают правила форматирования документа для печати на одной или на обеих сторонах листа бумаги. В классе book по умолчанию используется twoside, а в остальных — oneside. Отсутствуют в классе slides.

draft|final устанавливает, помечать или нет проблемные для вёрстки строки. По умолчанию используется final.

titlepage|notitlepage устанавливают печать заголовка и аннотации на отдельной страницы или прямо перед содержанием документа. В классе article по умолчанию используется notitlepage, а в остальных — titlepage. Отсутствуют в классе letter. titlepage не поддерживается в классе proc.

onecolumn|twocolumn задают форматирование документа в одну или в две колонки на странице. По умолчанию используется onecolumn. twocolumn не поддерживается в классах slide и letter, а onecolumn не поддерживается в классе proc.

leqno устанавливает печать номера формулы слева от неё, а не справа.

fleqn устанавливает выравнивание формул по левому краю страниц, а не по центру.

openbib устанавливает печать каждого блока в элементах списка литературы с новой строки. Отсутствует в классах slides и letter.

openright|openany устанавливают печать новой главы с правой страницы или с любой. Определены только в классах report, где по умолчанию используется openany, и book, где по умолчанию используется openright.

1.4 Поддержка русского языка

В декабре 1998 года в рамках проекта LaTeX3 реализована поддержка русского языка в соответствии со стандартом. Все необходимые средства распространяются сейчас вместе с LaTeXом.

Пакет inputenc надо подключать с опцией, соответствующей кодировке символов во входном файле. Так, в среде MS Windows, в которой используется кодовая страница 1251, в преамбулу входного файла надо включить декларацию

```
\usepackage[cp1251]{inputenc}
```

Кроме пакета inputenc, надо подключить пакет babel с опцией `russian`:

```
\usepackage[russian]{babel}
```

Будет установлена кодировка текстовых шрифтов `T2A` с русскими буквами, включены правила переноса русских слов[1], переопределены стандартные заголовки и введены новые команды для набора символов, специфических для русского языка.

Если документ на русском языке содержит целые абзацы английского текста, то перед ними надо ставить любую из двух эквивалентных команд

`\English` `\Eng`

Они включают правила переноса слов английского языка, что даёт более аккуратную вёрстку абзацев. После окончания английского текста надо поставить любую из двух эквивалентных команд

`\Russian` `\Rus`

Они восстановят правила переноса для русского языка. Приведённые выше команды переключения языка становятся доступными после подключения пакета babel с опцией `russian`.

Если документ на английском языке содержит русский текст, то вместо пакета babel можно ограничиться подключением пакета fontenc с опцией `T2A`:

```
\usepackage[T2A]{fontenc}
```

Будет установлена кодировка текстовых шрифтов `T2A` и LaTeX сверстает русский текст, но без переносов в словах, что может ухудшить качество вёрстки. Для небольших фрагментов русского текста качество вёрстки можно улучшить, задавая места переноса в русских словах командой `\-` или в декларации `\hyphenation`.

Если же документ на английском языке содержит большие фрагменты русского текста, то лучше воспользоваться пакетом babel с двумя опциями:

```
\usepackage[russian,english]{babel}
```

Поскольку опция `english` стоит последней, основным языком документа будет английский и все стандартные заголовки будут печататься по английски. Перед абзацами на русском языке надо ставить команду `\Russian` или `\Rus`, а после них — `\English` или `\Eng`. Для вставки коротких фраз на русском языке можно воспользоваться командой

`\textcyrillic{`*text*`}`

1.5 Комментарии

Всё, что следует в строке за символом `%`, LaTeX игнорирует.

Пакет verbatim из коллекции `tools` вводит командные скобки

`\begin{comment}` ... `\end{comment}`

Всё, что находится в них, LaTeX игнорирует.

[1] Образцы допустимых переносов в словах русского языка подключаются на стадии создания файла формата. Если в файл формата не включены правила переноса на русском языке, то LaTeX будет верстать документы без переносов в словах.

1.6 Включение дополнительных файлов

Перед преамбулой входного файла можно использовать любое количество командных скобок

`\begin{filecontents}{`*name.ext*`}`	*contents*	`\end{filecontents}`
`\begin{filecontents*}{`*name.ext*`}`	*contents*	`\end{filecontents*}`

При обработке входного файла для каждого окружения `filecontents` LaTeX создаёт файл *name.ext* (если он отсутствует) и записывает в него сначала комментарий с указанием происхождения и даты создания файла, а затем содержимое *contents* (строки комментария начинаются с %%). В случае $*$-формы комментарий не пишется.

2 Печатный документ

2.1 Параметры страницы

Страница печатного документа состоит из верхнего и нижнего колонтитулов и области, в которой размещается содержание документа: текст и подстрочные примечания. Кроме того, на боковых полях страницы могут размещаться *заметки на полях*, которые печатает команда `\marginpar`. Правила их размещения описаны на стр. 39. Размер и расположение колонтитулов, области с содержанием документа и заметок на полях задаются нерастяжимыми командными длинами, приведёнными на рис. 1. Их значения, установленные по умолчанию, можно изменить в преамбуле документа декларациями `\setlength` и `\addtolength`.

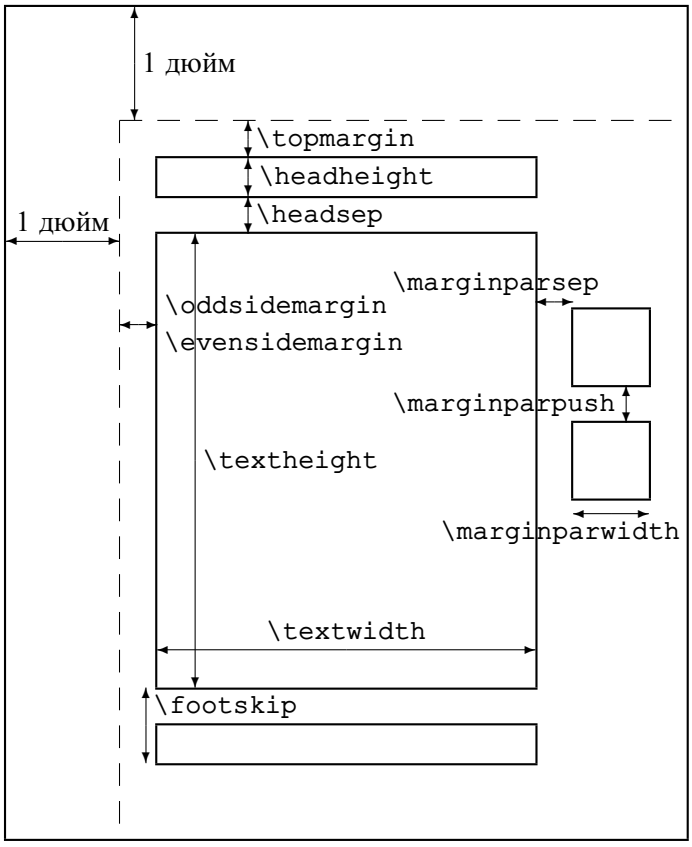

Рис. 1: Макет страницы с командами, задающими размер и расположение верхнего и нижнего колонтитулов, области с содержанием документа и заметок на полях. Команды `\oddsidemargin` и `\evensidemargin` задают левое поле для нечётные (odd) и чётных (even) страниц, соответственно.

Текущие значения параметров компоновки страницы можно узнать с помощью пакета layout из коллекции tools. Команда \layout из этого пакета печатает макет страницы, на которой она находится, с указанием значений всех параметров. Команда различает правые и левые страницы, одно- и двухколоночный режимы печати.

2.2 Титульная страница и аннотация

Стандартный заголовок, состоящий из названия, имён авторов и даты создания документа, печатает команда

```
\maketitle
```

Ей должны предшествовать две команды

```
\title{title}
\author{author(s)}
```

содержащие название документа *title* и имена авторов *author(s)*. Для разбиения длинного названия или списка авторов на строки используется команда \\. Аргументы обеих команд могут быть пустыми.

Перед \maketitle можно с помощью команды

```
\date{date}
```

указать дату создания документа *date*. Если команда \date отсутствует, то печатается текущая дата. Если дата не нужна, то надо использовать команду \date с пустым аргументом {}.

Аргументы команд \title, \author и \date могут содержать команду

```
\thanks{text}
```

которая печатает *text* как подстрочное примечание.

В стандартных классах команда \maketitle печатает заголовок на отдельной странице, если действует опция titlepage. Страница, следующая за титульной, нумеруется как 1-ая. Если действует опция notitlepage, то заголовок печатается с новой страницы прямо перед содержанием документа. В классе article по умолчанию используется notitlepage, а в классах book, report и slides — titlepage.

В аргументе команды \author можно использовать команду

```
\and
```

для разделения *author(s)* на боксы. LaTeX, формируя из этих боксов строку, отделяет их друг от друга большими пробелами. Каждый бокс может сам состоять из нескольких строк.

Если формат стандартного заголовка не соответствует требуемому, то надо использовать командные скобки

```
\begin{titlepage} ... \end{titlepage}
```

для создания титульной страницы. На этой странице печатается содержание окружения titlepage. Страница, следующая за титульной, нумеруется как 1-ая.

В классах article и report определены командные скобки

```
\begin{abstract} ... \end{abstract}
```

для печати аннотации к статье. Она печатается на отдельной странице, если действует опция titlepage. Перед аннотацией LaTeX печатает заголовок **Abstract**. Он хранится в команде

```
\abstractname
```

которую можно переопределить с помощью \renewcommand. Пакет babel с опцией russian переопределяет её как **Аннотация**.

При наличии опции twocolumn аннотация, как и сам документ, печатается в двухколоночном режиме. Пакет abstract позволяет напечатать её в одноколоночном режиме (см. документацию к пакету).

2.3 Секционирование документа

В стандартных классах, за исключением `letter`, определены команды для разделения документа на секции. Все команды имеют по два аргумента, один из которых является обязательным. Обязательный аргумент печатается как название секции и, если отсутствует необязательный аргумент, включается в оглавление и используется при оформлении колонтитулов. Для оглавления и колонтитулов можно задать более компактное название секции в виде необязательного аргумента. Неустойчивые команды в подвижных аргументах должны быть защищены командой `\protect`. Для разбиения длинного названия на строки внутри обязательного аргумента можно использовать команду `\\`.

LaTeX автоматически нумерует секции, уровень которых не превышает число, которое хранится в счётчике

`secnumdepth`

Его значение можно изменить декларацией `\setcounter`. Секциям, созданным командой `\section`, присвоен уровень 1. Счётчики секций более младшего уровня определены как внутренние по отношению к счётчику секций старшего уровня, поэтому в каждой секции подсекции нумеруются независимо друг от друга.

Все команды секционирования имеют $*$-форму. Она имеет только обязательный аргумент, который печатается как название секции. Такие секции не нумеруются и не заносятся в оглавление и колонтитулы.

2.3.1 Части

Команды

`\part[`*toc*`]{`*head*`}` `\part*{`*head*`}`

позволяют разделить документ на части. В классах `book` и `report` команда `\part` печатает заголовок в центре отдельной страницы: сначала слово **Part** и порядковый номер части прописными римскими цифрами, а затем с новой строки название части *head*. В классе `article` заголовок печатается прямо перед содержанием части. При наличии опции, для оглавления и колонтитулов используется не *head*, а *toc*. Команда `\part*` печатает только название части.

Команды `\part` и `\part*` являются необязательными, поэтому они не влияют на порядок нумерации более младших секций.

Слово Part хранится в команде

`\partname`

Её можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет её как **Часть**.

2.3.2 Главы

В классах `book` и `report` определены команды для разделения документа на главы:

`\chapter[`*toc*`]{`*head*`}` `\chapter*{`*head*`}`

Каждая глава печатается с новой страницы: правой, если действует опция `openright`, или любой, если действует опция `openany`. В классе `book` по умолчанию используется `openright`, а в классе `report` — `openany`.

Команда `\chapter` печатает сначала слово **Chapter** и порядковый номер главы арабскими цифрами. Название главы *head* печатается с новой строки. При наличии опции, для оглавления и колонтитулов используется не *head*, а *toc*. Команда `\chapter*` печатает только название главы.

Слово Chapter хранится в команде

`\chaptername`

Её можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет её как **Глава**.

2.3.3 Разделы

Для деления документа на разделы, подразделы и подподразделы определены команды:

`\section[`*toc*`]{`*head*`}`	`\section*{`*head*`}`
`\subsection[`*toc*`]{`*head*`}`	`\subsection*{`*head*`}`
`\subsubsection[`*toc*`]{`*head*`}`	`\subsubsection*{`*head*`}`

Они печатают перед названием секции *head* её порядковый номер. При наличии опции, для оглавления и колонтитулов используется не *head*, а *toc*. *-форма команд печатает только название секции.

2.3.4 Параграфы

Команды

`\paragraph[`*toc*`]{`*head*`}`	`\paragraph*{`*head*`}`
`\subparagraph[`*toc*`]{`*head*`}`	`\subparagraph*{`*head*`}`

печатают в начале первого абзаца секции шрифтом полужирной насыщенности её название *head*. При этом, команда `\paragraph` подавляет абзацный отступ, а `\subparagraph` — нет. Между секциями вставляется дополнительный вертикальный промежуток. При наличии опции, для оглавления и колонтитулов используется не *head*, а *toc*. *-форма команд печатает только название секции.

2.3.5 Разделение книги на части

В классе `book` определены декларации

`\frontmatter`	`\mainmatter`	`\backmatter`

которые открывают вводную, основную и заключительную части книги, соответственно. В вводной части страницы нумеруются римскими цифрами, а в остальных — арабскими (в России принята сквозная нумерация арабскими цифрами). Кроме того, в вводной и заключительной частях команда `\chapter` не печатает номер главы, но заносит её заголовок в оглавление.

2.3.6 Приложения

Если в документе имеются приложения, то перед ними надо поставить декларацию

`\appendix`

После неё новая глава начинается не со слова **Chapter**, а со слова **Appendix**. Кроме того, меняется формат нумерации глав: вместо арабских цифр используются заглавные латинские буквы A, B, C и т. д. В классе `article`, где нет глав, буквами нумеруются разделы, созданные командой `\section`.

Слово Appendix хранится в команде

`\appendixname`

Её можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет её как **Приложение**.

2.4 Оглавление, списки рисунков и таблиц

Оглавление, списки рисунков и таблиц печатают соответственно команды

`\tableofcontents`	`\listoffigures`	`\listoftables`

Каждая из команд считывает данные из соответствующего ей служебного файла, который создаёт LaTeX при обработке входного файла только при наличии в нём этой команды. Служебные файлы

имеют расширения `toc`, `lof` и `lot`, соответственно. Данные в `toc`-файл пишут команды секционирования, а в `lof`- и `lot`-файлы — команда `\caption` из окружений `figure` и `table`. Кроме того, данные в эти файлы можно записать с помощью следующих двух команд.

Команда

`\addcontentsline{`*ext*`}{`*unit*`}{`*text*`}`

добавляет запись *text* в файл с расширением *ext*. *unit* задаёт тип записи, от которого зависит её вид при печати. Для `toc`-файла допустимыми значениями для *unit* являются имена секций, например, `chapter` или `subsection`. Для `lof`- и `lot`-файлов *unit* имеет значение `figure` и `table`, соответственно. Аргумент *text* является подвижным.

Команда

`\addtocontents{`*ext*`}{`*text*`}`

добавляет *text* в файл с расширением *ext*. Аргумент *text* является подвижным.

Перед оглавлением и списками рисунков и таблиц печатаются заголовки: **Contents**, **List of Figures** и **List of Tables**, соответственно. Они хранятся в командах

`\contentsname` `\listfigurename` `\listtablename`

Любую из этих команд можно переопределить декларацией `\renewcommand`. Пакет babel с опцией `russian` переопределяет их как **Содержание** (в классе `article`) или **Оглавление** (в классах `book` и `report`), **Список иллюстраций** и **Список таблиц**, соответственно.

Счётчик

`tocdepth`

хранит уровень секций, которые ещё включаются в оглавление. Его значение можно изменить декларацией `\setcounter`.

2.5 Колонтитулы

Расположение колонтитулов на странице задаётся параметрами, приведёнными на рис. 1.

Содержание верхнего и нижнего колонтитулов зависит от стиля страницы. Изменить его можно декларациями

`\pagestyle{`*style*`}`
`\thispagestyle{`*style*`}`

Декларация `\pagestyle` устанавливает стиль *style*, начиная с текущей страницы. Её область действия подчиняется обычным правилам. `\thispagestyle` устанавливает стиль *style* только текущей страницы. Имеются четыре предопределенных стиля страницы:

`plain` Печатается номер страницы в середине нижнего колонтитула. Верхний колонтитул пуст. Используется по умолчанию в стандартных классах, за исключением `book` и `letter`.

`empty` Верхний и нижний колонтитулы пусты. Используется по умолчанию в классе `letter`.

`headings` Печатается название секций и номер страницы в верхнем колонтитуле. Нижний колонтитул пуст. Используется по умолчанию в классе `book`.

При односторонней печати в классах `book` и `report` печатается название главы из команды `\chapter`, а в классах `article` и `proc` — название раздела из команды `\section`. Такой формат по умолчанию задан описанной ниже декларацией `\markright`.

При двухсторонней печати колонтитул на левых страницах оформляется как при односторонней печати, а на правой странице в классах `book` и `report` печатается название раздела из команды `\section`, а в классах `article` и `proc` — название подраздела из команды `\subsection`. Такой формат по умолчанию для левых страниц задан первым аргументом описанной ниже декларации `\markboth`, а для правых страниц — `\markright`.

`myheadings` В отличие от стиля `headings`, в верхнем колонтитуле печатаются не названия секций, а аргументы деклараций `\markboth` и/или `\markright`. По умолчанию они пусты. Нижний колонтитул пуст.

В стиле `myheadings` декларации

```
\markboth{left}{right}
\markright{right}
```

определяют содержание верхнего колонтитула (кроме номера страницы), а в стиле `headings` они переопределяют содержание верхнего колонтитула (кроме номера страницы): вместо названия секций печатаются аргументы этих деклараций. *left* печатается на левых страницах, причём используется аргумент последней декларации `\markboth` на момент окончания страницы. *right* печатается на правых страницах, причём используется аргумент первой декларации `\markright` или `\markboth` на текущей странице, а если их нет, то последней декларации на момент окончания страницы. В стиле `headings` команды секционирования, которые помещают свой аргумент в колонтитулы, восстанавливают определение деклараций по умолчанию: снова печатаются названия секций. Этого, естественно, не происходит в стиле `myheadings`.

Аргументы *left* и *right* обрабатываются в строковой моде. Неустойчивые команды в них должны быть защищены командой `\protect`.

2.5.1 Нестандартные колонтитулы

Нестандартные колонтитулы можно создать переопределением команд `\@oddhead` и `\@evenhead`, а также `\@oddfoot` и `\@evenfoot`, которые не входят в «LaTeX для пользователей». Ниже приведён простой пример, показывающий, как прямо во входном файле можно задать печать номера нечётных страниц в середине верхнего, а не нижнего колонтитула:

```
\makeatletter
\renewcommand{\@oddhead}{\hfill\thepage\hfill}
\renewcommand{\@oddfoot}{}
\makeatother
```

Нестандартные колонтитулы можно создавать, используя команды из пакета fancyhdr. Они детально описаны в документации к пакету.

2.6 Перекрёстные ссылки

Команда

```
\label{name}
```

ставит метку *name*. Под этим именем в `aux`-файле запоминается значение счётчика *текущего* нумеруемого объекта (раздела, уравнения, таблицы и т. п.) и номер страницы, на которую попадает команда `\label`. Например, если метка стоит внутри командных скобок `equation`, то запоминается порядковый номер уравнения, а если внутри скобок `table` *после* команды `\caption`, то порядковый номер таблицы. Эти объекты сами объявляют свой счётчик текущим командой `\refstepcounter`. Команда

```
\ref{name}
```

печатает номер объекта, внутри которого стоит метка *name*. Номер страницы печатает команда

```
\pageref{name}
```

Команда `\pageref` печатает номер страницы всегда, не исключая ситуаций, когда объект и ссылка на него попадают на одну страницу или оказываются на соседних страницах. Пакет varioref из коллекции `tools` вводит более «умную» команду

```
\vpageref[text1][text2]{name}
```

Она в перечисленных выше ситуациях печатает не номер страницы, а текст, указывающий, где находится объект. Например, «on this page», если объект и ссылка на него попадают на одну страницу, или «on the next page», если объект находится по отношению к ссылке на следующей странице, или «on the facing page», если страницы с объектом и ссылкой находятся на одном развороте при двусторонней печати. Если объект и ссылка на него находятся друг от друга через одну и более страниц, то `\vpageref` печатает слова «on page» и номер страницы командой `\pageref`. Текст, который печатает `\vpageref`, хранится в специальных командах, которые могут быть переопределены. Они описаны в документации к пакету.

Опции команды `\vpageref` позволяют заменить текст, который печатается по умолчанию, на произвольный текст. Если объект и ссылка на него попадут на одну страницу, то будет напечатан *text1*, а в остальных случаях — *text2* или текст по умолчанию в случае одной опции.

Пакет varioref поддерживает русский язык: если его подключить с опцией `russian`, то вместо текста на английском языке будет напечатан соответствующий ему русский текст.

2.6.1 Пакет xr

Пакет xr из коллекции `tools` позволяет ссылаться на нумеруемые объекты из других документов. Имена входных файлов с этими документами (без расширения `tex`) надо объявить в преамбуле текущего документа посредством деклараций

`\externaldocument[`*prefix*`]{`*filename*`}`

Опция *prefix* позволяет исключить совпадение имён меток в разных докуметах. При наличии опции все ссылки в текущем документе на объекты из внешний файлов, помеченные как `\label{`*name*`}`, записываются в виде `\ref{`*prefixname*`}`.

2.7 Сноски

Сноску печатает команда

`\footnote[`*number*`]{`*text*`}`

где *text* — текст сноски, а положительное число *number* — её номер. Маркёр сноски печатается на месте команды, а текст — внизу страницы. Если опция *number* опущена, то сноске присваивается порядковый номер из счётчика `footnote`.

Команду `\footnote` можно использовать только в текстовой моде или в `minipage`. Сноски внутри боксов создаются с помощью команд

`\footnotemark[`*number*`]`
`\footnotetext[`*number*`]{`*text*`}`

Аргументы команд имеют то же назначение, что и у `\footnote`. Команда `\footnotemark` доступна в любой моде. Она печатает только маркёр сноски. Сам текст печатается командой `\footnotetext`, которая должна находится вне бокса.

Команда `\footnote` внутри `minipage` печатает сноски на дне бокса с независимой нумерацией, используя счётчик `mpfootnote`. Для печати примечания внизу страницы надо использовать команду `\footnotemark` для печати маркёра сноски и команду `\footnotetext` вне бокса для печати текста.

Сноски отделяются от основного текста горизонтальной линией, которую рисует команда

`\footnoterule`

Её можно переопределить посредством `\renewcommand`. Расстояние между разделительной линией и первой сноской, а также между самими сносками задаёт невидимая линейка высотой

`\footnotesep`

Её новое значение можно задать посредством `\setlength`.

2.8 Цитирование литературы

Список цитируемой литературы надо размещать внутри командных скобок

```
\begin{thebibliography}{text}
  bibitems
\end{thebibliography}
```

Для печати номеров (или меток) элементов отводится колонка шириной, равной ширине аргумента *text*. Сам аргумент не печатается.

Каждый элемент списка должен начинаться с команды

```
\bibitem[label]{id}
```

id служит идентификатором ссылки. Если опция *label* опущена, то перед элементом печатается в квадратных скобках его порядковый номер в списке. Номер хранится в счётчике `enumiv`. При наличии опции вместо номера печатается метка *label*. При этом значение счётчика `enumiv` не изменяется. Неустойчивые команды в *label* следует защищать командой `\protect`.

В стандартных классах номер элемента в списке литературы печатается в квадратных скобках. Изменение этого формата требует переопределения команды `\@biblabel`, которая не входит в «LaTeX для пользователей». Ниже приведён пример, показывающий, как прямо во входном файле можно задать печать номера элемента без квадратных скобок, но с точкой после него:

```
\makeatletter
\renewcommand{\@biblabel}[1]{#1.}
\makeatother
```

Перед списком литературы печатается заголовок **References** в классе `article` и **Bibliography** в классах `book` и `report`. В классе `article` заголовок хранится в команде

```
\refname
```

а в классах `book` и `report` — в команде

```
\bibname
```

Команды `\refname` и `\bibname` можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет их как **Список литературы** и **Литература**, соответственно.

Команда

```
\cite[text]{id}
```

печатает в квадратных скобках номер или метку элемента с идентификатором *id*. Опция *text* печатается после номера (метки) и запятой. Она служить для уточнения ссылки, например, для указания номера параграфа или страницы.

Одной командой `\cite` можно напечатать сразу несколько ссылок, перечислив их идентификаторы в обязательной аргументе команды через запятую. Номера ссылок печатаются в порядке следования их идентификаторов через запятую даже тогда, когда они составляют диапазон чисел, например, [3, 5, 4]. При подключении пакета cite номера ссылок печатаются в порядке возрастания, а диапазон номеров печатается через короткое тире, например, [1, 3–5] вместо [3, 5, 4, 1].

Внутри командных скобок `thebibliography` можно использовать команду

```
\newblock
```

для разделения на части ссылки, например, на список авторов, название книги, название издательства и т. д. В стандартных классах `\newblock` просто вставляет горизонтальный пробел. При наличии опции `openbib` в декларации `\documentclass` каждый блок печатается с новой строки, а последующие строки в блоке сдвигаются вправо на расстояние

```
\bibindent
```

2.9 Нумерация страниц

По умолчанию страницы нумеруются арабскими цифрами. Изменить формат нумерации можно декларацией

`\pagenumbering{`*format*`}`

с глобальной областью действия. Она начинает отсчёт страниц с единицы. Аргумент декларации *format* может принимать одно из значений:

```
arabic   roman   Roman   alph   Alph
```

Они задают печать номера страницы арабскими цифрами, римскими строчными и прописными цифрами, латинскими строчными и прописными буквами, соответственно. После подключения пакета babel с опцией `russian` можно использовать также значения

```
asbuk   Asbuk
```

для нумерации страниц русскими строчными и прописными буквами, соответственно. Количество нумеруемых страниц не должно, естественно, превышать количества букв в алфавите.

2.10 Печать в две колонки

В стандартных классах опция `twocolumn` устанавливает печать всего документа в две колонки. В одноколоночном режиме переход к печати в две колонки вызывает команда

`\twocolumn[`*preface*`]`

Печать начинается с новой страницы (исполняется команда `\clearpage`), причём сначала в одноколоночном режиме печатается необязательный аргумент *preface*. Команда

`\onecolumn`

вызывает переход в одноколоночный режим печати. Печать начинается с новой страницы (исполняется команда `\clearpage`).

При двухколоночной печати расстояние между колонками задаёт командная длина

`\columnsep`

Командная длина

`\columnseprule`

задаёт ширину вертикальной линии между колонками. По умолчанию она равна нулю. Изменить значение параметров `\columnsep` и `\columnseprule` можно посредством `\setlength`.

2.10.1 Пакет multicol

Команды `\twocolumn` и `\onecolumn` начинают печать текста с новой страницы, что ограничивает область их использования. Пакет multicol из коллекции `tools` вводит окружение

```
\begin{multicols}{n}[preface][skip]
  text
\end{multicols}
```

которое не форсирует переход на новую страницу. Сначала печатается необязательный аргумент *preface* в одну колонку, а затем *text* в *n* колонок. Можно задавать до 10 колонок. Печать начинается с новой страницы только в том случае, когда на текущей странице для многоколоночного текста осталось места меньше, чем задано командной длиной

`\premulticols`

или опцией *skip*. По умолчанию значение `\premulticols` равно 50 pt. Текст после окружения `multicols` печатается с новой страницы, если на текущей странице осталось места меньше, чем задано командной длиной

`\postmulticols`

По умолчанию значение `\postmulticols` равно 20 pt. Перед многоколоночным текстом и после него вставляется эластичным промежуток

`\multicolsep`

с естественной длиной 12 pt. Расстояние между колонками и толщина разделительной линии задаётся обычными параметрами `\columnsep` и `\columnseprule`.

По умолчанию LaTeX делает все колонки одинаковой высоты. Если требуется увеличить высоту всего многоколоночного текста за счёт уменьшения высоты текста в последней колонке, то надо присвоить счётчику

`unbalance`

ненулевое значение. Оно задаёт количество пустых строк, которые вставляются в конце последней колонки.

В `multicols` можно использовать окружения `figure*` и `table*`. Подстрочные примечания печатаются на всю ширину страницы.

2.11 Алфавитный указатель

Алфавитный указатель терминов LaTeX создаёт вместе с пакетом makeidx и программой `makeindex`. Сам процесс создания указателя происходит в четыре этапа.

Первый этап. Каждый термин, подлежащий включению в указатель, надо оформить как аргумент команды

`\index{`*text*`}`

поместив её сразу после термина в тексте документа. *text* может содержать любые символы, включая специальные, с одним ограничением — фигурные скобки всегда должны быть парными. По умолчанию символы `@`, `!` и `|` являются управляющими для `makeindex`. Если перед любым из этих символов поставить двойные кавычки `"`, то он превратится в обычный символ.

Символ `@` необходим тогда, когда для печати термина используются команды. При сортировки терминов `makeindex` использует текст, расположенный до символа `@`, а в алфавитный указатель помещает то, что стоит после него. Пример: `\index{alpha@$\alpha$}`.

Списки термином могут быть вложенными. Символ `!` служит разделителем терминов разного уровня. Пример: `\index{Интегралы!контурные}`, `\index{Интегралы!кратные}`. Поддерживается три уровня вложенности.

Номера страниц у терминов в указателе можно напечатать разными шрифтами, выделяя, скажем, курсивом номер страницы, на которой дано определение термина. В аргументе команды `\index` команды переключения шрифта для печати номера страницы указываются не в своей обычной форме, например, `\emph`, а в виде `|emph`, т. е. признаком команды является не `\`, а `|`. Пример: `\index{Системы уравнений|emph}`.

В алфавитном указателе можно создать ссылку на другой термин. Она оформляется в виде `\index{..|see{..}}` или `\index{..|seealso{..}}` и печатается вместо номера страницы. Пример: `\index{Дионис}\index{Бахус|see{Дионис}}`. `|see` и `|seealso` — это форма описанных ниже команд `\see` и `\seealso` в аргументе команды `\index`.

Второй этап. Включить в преамбулу входного файла, если её там ещё нет, декларацию

`\makeindex`

и обработать его LaTeXом. При наличии декларации `\makeindex` LaTeX, обрабатывая входной файл, скажем, *name*`.tex`, создаёт файл *name*`.idx` и записывает в него входы в указатель в виде

`\indexentry{`*text*`}{`*integer*`}`

где *integer* — номер страницы, на которой находится команда `\index` с аргументом *text*. LaTeX пишет в `idx`-файл *text* в своём внутреннем представлении, в котором русские буквы являются специальными командами, например, буква Я — это команда `\CYRYA`.

Третий этап. Действия на этом этапе зависят от того, входят в указатель термины с русскими буквами или нет. В последнем случае `idx`-файл надо просто обработать программой `makeindex`. Она создаст отсортированный список терминов и запишет его в файл *name*`.ind` в виде окружения

`\begin{theindex}` *items* `\end{theindex}`

Список терминов в окружении `theindex` оформлен следующим образом. Каждому элементу списка в зависимости от уровня, на котором он находится, предшествует одна из команд

`\item` `\subitem` `\subsubitem`

После термина через запятую идут номера страниц документа, на которых встречается этот термин, например, `\emph{13}, 17`. Для ссылок внутри указателя печатается не номер страницы, а одна из команд

`\see{`*label*`}{`*page*`}` `\seealso{`*label*`}{`*page*`}`

Они определены в пакете makeidx. Здесь *label* — термин, на который идёт ссылка, а *page* — номер страницы, на которой находится команда `\index` с этой ссылкой. Перед каждой группой терминов, начинающихся с новой буквы, записывается команда

`\indexspace`

для вставки при печати документа дополнительного вертикального пробела.

Поскольку русские буквы в `idx`-файле записаны в виде команд, `makeindex` будет сортировать термины на русском языке не по буквам, а по именам команд, что неприемлемо. Поэтому `idx`-файл надо обрабатывать не сразу программой `makeindex`, а после перевода русских терминов из представления LaTeXa в «нормальное» представление для текстовых файлов. Правильный `ind`-файл можно получить, обработав `idx`-файл специальным командным файлом, входящим в средства поддержки русского языка `T2`. В среде MS Windows используется файл `rumkidxw.bat`. В командной строке указывается только имя `idx`-файла (без расширения!):

`rumkidxw` *name*

Четвёртый этап. Подключить пакет makeidx, если он всё ещё не подключен. Вставить во входном файле в то месте, где должен быть напечатан алфавитный указатель, определённую в этом пакете команду

`\printindex`

и обработать входной файл LaTeXом. При обработке входного файла команда `\printindex` включает в него посредством команды `\input` содержимое `ind`-файла (если он имеется). Алфавитный указатель печатается с новой страницы в две колонки сразу после заголовка **Index**, который хранится в команде

`\indexname`

Её можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет её как **Предметный указатель**.

Команды `\see` и `\seealso` печатают курсивом соответственно слова *see* и *see also* и затем термин, на который идёт ссылка. Эти слова хранятся в командах

`\seename` `\alsoname`

из пакета makeidx. Опция `russian` пакета babel переопределяет их как *см.* и *см. также*.

Вид алфавитного указателя можно кардинальным образом изменить по сравнению с описанным выше, который установлен по умолчанию. Это можно сделать с помощью подходящего стилевого файла для программы `makeindex` и переопределения должным образом окружения `theindex`.

2.11.1 Дополнительный указатель терминов

В дополнение к указателю Index, LaTeX позволяет создать, скажем, именной указатель или словарь терминов. Процесс его создания аналогичен описанному выше.

Каждый термин надо оформить как аргумент команды

`\glossary{`*text*`}`

При наличии в преамбуле входного файла декларации

`\makeglossary`

LaTeX, обрабатывая входной файл, скажем, *name*`.tex`, создаёт файл *name*`.glo` и записывает в него входы в указатель в виде

`\glossaryentry{`*text*`}{`*integer*`}`

где *integer* — номер страницы, на которой находится команда `\glossary` с аргументом *text*.

Файл *name*`.glo` надо обработать программой `makeindex`. По умолчанию она настроена на создание указателя Index. Поэтому требуется стилевой файл, задающий команды форматирования указателя. Правила создания стилевого файла описаны в документации к `makeindex`. В случае именного указателя можно воспользоваться форматом окружения `theindex`. Тогда достаточно создать файл, скажем, `glos.ist`, следующего содержания:

```
keyword   "\\glossaryentry"
preamble  "\\begin{theglossary}\n"
postamble "\n\n\\end{theglossary}\n"
```

и определить во входном файле окружение `theglossary`:

```
\newenvironment{theglossary}
 {\renewcommand{\indexname}{Именной указатель}\begin{theindex}}
 {\end{theindex}}
```

Командная строка запуска `makeindex` должна иметь вид:

`makeindex -s glos.ist -o` *name*`.gls` *name*`.glo`

`makeindex` создаст отсортированный список терминов и запишет его в файл *name*`.gls` в виде окружения `theglossary`.

Файл *name*`.gls` включается в документ посредством команды `\input`.

3 Счётчики

LaTeX автоматически нумерует страницы, разделы, уравнения, таблицы и т. п. Для каждого типа нумеруемых объектов определён счётчик, имя которого обычно совпадает с названием команды или командных скобок, создающих соответствующие объекты. Так, команде секционирования `\section` соответствует счётчик `section`, а командым скобкам `table` — счётчик `table`. Номер страницы хранится в счётчике `page`, порядковые номера элементов списков `enumerate` разного уровня — в счётчиках `enumi`, `enumii`, `enumiii` и `enumiv`. Номер сноски внутри `minipage` хранит счётчик `mpfootnote`.

Счётчик может быть определён как внутренний по отношению к другому счётчику. В этом случае его значение сбрасывается до нуля при изменении значения последнего. Так, например, счётчик `subsection` является внутренним к счётчику `section`, поэтому нумерация подразделов в каждом разделе начинается с единицы.

Текущее значение счётчика *counter* возвращает команда

`\value{`*counter*`}`

Печатает это значение команда

\the*counter*

Формат, в котором будет напечатано значение счётчика, зависит от того, в каком виде этот счётчик был определён. Команды

\arabic{*counter*}	\roman{*counter*}	\alph{*counter*}
\fnsymbol{*counter*}	\Roman{*counter*}	\Alph{*counter*}

печатают значение счётчика *counter* арабскими цифрами, подстрочными символами (∗, † и т. д. — всего девять символов), римскими строчными и прописными цифрами, латинскими строчными и прописными буквами. Опция `russian` пакета babel определяет команды

\asbuk{*counter*}	\Asbuk{*counter*}

для печати значения счётчика русскими строчными и прописными буквами, соответственно.

Переопределить формат печати значения счётчика можно декларацией `\renewcommand`. Например, после переопределения

```
\renewcommand{\theequation}{\thesection,\alph{equation}}
```

номер первого пронумерованного уравнения во втором разделе будет выглядеть как (2,a).

Новый счётчик *counter* с нулевым значением вводится командой

\newcounter{*counter*}[*out-counter*]

Опция *out-counter* задаёт уже существующий счётчик, по отношению к которому новый счётчик будет внутренним. По умолчанию значение счётчика печатается в формате команды `\arabic`.

Изменить значение счётчика *counter* можно командами

\setcounter{*counter*}{*integer*}
\addtocounter{*counter*}{*integer*}

В команде `\addtocounter` значение *integer* может быть отрицательным числом.

Команды

\stepcounter{*counter*}
\refstepcounter{*counter*}

увеличивают значение счётчика *counter* на единицу, сбрасывая до нуля значения внутренних счётчиков. Кроме того, `\refstepcounter` объявляет свой счётчик текущим: именно его значение печатает команда `\ref`.

4 Командные длины

Командная длина — это команда, значением которой является длина. В качестве абсолютных единиц измерения длины можно использовать `cm` (сантиметр), `mm` (миллиметр), `in` (дюйм, 1 in = 2.54 cm), `pt` (пункт, 72.27 pt = 1 in), `bp` (большой пункт, 72 bp = 1 in) и `pc` (пайка, 1 pc = 12 pt). Можно использовать также относительные единицы измерения длины `ex` и `em`: они зависят от размера текущего шрифта и равны примерно высоте буквы x и ширине буквы M, соответственно. В шрифте Computer Modern Roman 10 pt: 1 ex ≈ 4.3 pt, а 1 em = 10 pt.

Декларация

\newlength{*cmd*}

объявляет новую командную длину *cmd* с нулевым значением. Изменить значение командной длины можно декларациями

\setlength{*cmd*}{*length*}
\addtolength{*cmd*}{*length*}

Первая из них устанавливает значение *cmd* равным *length*, а вторая — изменяет значение *cmd* на величину *length*. Длину можно задавать как в явном виде, например, `-3.14cm` или `2em`, так и используя в качестве единиц измерения другие командные длины, например, `0.5\textwidth`.

Командные длины, задающие вертикальные пробелы, являются, как правило, эластичными. Для них наряду с естественной длиной, задаётся допустимый размер деформации. Например,

```
\setlength{\parskip}{5pt plus 2pt minus 1pt}
```

устанавливает значение длины `\parskip` равным 5 pt, но, если при вёрстке страницы возникнет необходимость, она может быть растянута на 2 pt или сжата на 1 pt.

Командная длина

`\fill`

имеет нулевую естественную длину и неограниченную растяжимость. Командная длина

`\stretch{`*p*`}`

также имеет нулевую естественную длину, но её растяжимость относится к растяжимости `\fill` как *p* к 1. Заметим, что `0.5\fill` является нерастяжимой длиной с нулевым значением, а не эластичной длиной с растяжимостью, вдвое меньшей, чем у `\fill`. Последняя задаётся как `\stretch{0.5}`.

5 Вставка пробелов

5.1 Горизонтальные промежутки

Команды

`\␣` `\,` `\quad` `\qquad`

вставляют горизонтальный пробел соответственно нормального размера и длиной 0.1667 em, 1 em и 2 em. Здесь символ ␣ обозначает пробел в исходном тексте.

Команды

`\hspace{`*length*`}` `\hspace*{`*length*`}`

вставляют горизонтальный пробел длиной *length*. Если, однако, `\hspace` попадёт на край строки, то её пробел будет удалён.

Для команды `\hspace{\fill}`, вставляющей пробел с неограниченной растяжимостью, имеется краткая форма

`\hfill`

Команды

`\hrulefill` `\dotfill`

действуют подобно `\hfill`, заполняя, к тому же, пробел соответственно сплошной и пунктирной линиями на уровне базовой линии строки.

Согласно американской традиции, между предложениями ставится более длинный пробел, чем между словами. Но если перед точкой или знаками ! и ? в конце предложения стоит не строчная буква или цифра, то LaTeX не считает, что предложение заканчивается. Поэтому в подобных предложениях перед знаками пунктуации надо ставить команду

`\@`

В русском языке принято между предложениями делать такой же пробел, что и между словами. Отменить и восстановить дополнительный пробел между предложениями можно с помощью деклараций

`\frenchspacing` `\nofrenchspacing`

соответственно.

5.2 Вертикальные промежутки

Команды

\vspace{*length*} \vspace*{*length*}

вставляют вертикальный пробел величиной *length*, причём внутри абзаца пробел вставляется после заполнения текущей строки. Если, однако, `\vspace` попадёт в начало или в конец страницы, то её пробел будет удалён.

Для команды `\vspace{\fill}`, вставляющей пробел с неограниченной растяжимостью, имеется краткая форма

\vfill

Команда

\addvspace{*length*}

увеличивает вертикальный пробел до величины *length*. Её можно использовать только между абзацами.

Команды

\smallskip \medskip \bigskip

вставляют вертикальный пробел, величину которого задают эластичные командные длины

\smallskipamount \medskipamount \bigskipamount

соответственно. В стандартных классах их естественная длина равна 3 pt, 6 pt и 12 pt.

6 Как управлять форматированием абзацев

В исходном тексте признаком окончания абзаца является пустая строка или команда

\par

Именно они стимулируют LaTeX начинать вёрстку абзаца. Поэтому, если область действия деклараций, например `\small` или `\sloppy`, влияющих на форматирование абзаца, ограничивается группой, то закрывающая фигурная скобка должна стоять *после* пустой строки: LaTeX должен начать вёрстку абзаца раньше, чем закончится действие деклараци.

Перед абзацем вставляется дополнительный к междустрочному интервалу вертикальный промежуток, величина которого задаётся командной длиной

\parskip

В стандартных классах, кроме `letter`, по умолчанию её значение равно нулю.

6.1 Изменение режима форматирования

В режиме вёрстки по умолчанию LaTeX выравнивает строки абзаца на правом краю, варьируя в ограниченных пределах промежутки между словами и делая переносы слов. Если недопускающий разрыва бокс не помещается в строке, а его перенос на следующую строку приводит к слишком большим промежуткам между словами в текущей и предшествующей ей строках, то LaTeX не станет переносить этот бокс и сделает более длинную строку. Что можно сделать, чтобы избавится от длинных строк. Во-первых, переписать абзац так, чтобы бокс оказался внутри строки. Если этого сделать нельзя, то надо ослабить критерий, по которому LaTeX предпочитает длинные строки, а не разреженные. Разреженность строки, которую LaTeX никогда не превышает, задаётся значением TeXовской команды

\tolerance

По умолчанию оно равно 200. Можно увеличить это значение, тогда LaTeX сможет делать все строки более разреженными. Если длинная строка находится в середине или в конце большого абзаца, то за счёт более длинных промежутков между словами в предшествующих ей строках, LaTeX может накопить текст для заполнения этой строки и, следовательно, перенести бокс на следующую строку. Увеличить значение параметра `\tolerance` можно для всего документа сразу, поместив в преамбуле входного файла строку наподобие

```
\tolerance=500
```

или внутри группы, содержащей абзац с длинной строкой.

Более грубый способ избавления от длинных строк задаёт декларация

`\sloppy`

Она разрешает делать практически сколь угодно разреженные строки, поскольку устанавливает максимальное значение параметра `\tolerance` 10000. Декларация `\sloppy` имеет один существенный недостаток. Она, начиная с некоторой величины дополнительных промежутков между словами в разреженной строке, позволяет LaTeXу не увеличивать штраф за дальнейший рост разреженности. В результате, вся разреженность может сконцентрироваться в одной строке. Область действия декларации `\sloppy` можно ограничить группой или командными скобками

`\begin{sloppypar} ... \end{sloppypar}`

Отменяет действие декларации `\sloppy` декларация

`\fussy`

6.2 Отступ в начале абзаца

Первая строка каждого абзаца, за исключением тех, которые идут сразу после команд секционирования, печатается с отступом, величина которого задаётся командной длиной

`\parindent`

Команда

`\noindent`

подавляет отступ в начале абзаца, перед которым она стоит.

Команда

`\indent`

делает горизонтальный пробел, равный значению `\parindent`. Её обычно используют, когда надо сделать стандартный отступ в начале абзаца, в котором LaTeX не сделал этого сам.

Команда `\indent` не работает в первом абзаце, идущим сразу после команд секционирования. В этом случае отступ нужно делать командой `\hspace*{\parindent}`. Отступ в первых абзацах всего документа можно сделать, просто подключив пакет indentfirst из коллекции tools.

6.3 Междустрочный интервал

Расстояние между базисными линиями соседних строк внутри абзаца задаёт командная длина TeX'а `\baselineskip`[2]. Изменять её значение непосредственно во входном файле не рекомендуется, поскольку при смене размера шрифта будет снова автоматически выставлено значение, принятое по умолчанию. Для изменения междустрочного интервала надо использовать каманду

`\baselinestretch`

[2] `\baselineskip` задаёт минимальное расстояние: при вставке внутрь абзаца больших боксов строки автоматически раздвигаются.

Её значение — вещественное число — равно значению междустрочного интервала. В стандартных классах по умолчанию оно равно 1, что соответствует одинарному междустрочному интервалу. Значение команды `\baselinestretch` можно переопределить декларацией `\renewcommand`. Если переопределение выполнено не в преамбуле документа, то оно вступит в силу только после команд смены размера шрифта. Достаточно после переопределения просто поставить команду `\normalsize`. Например, после

```
\renewcommand{\baselinestretch}{2}\normalsize
```

расстояние между строками будет соответствовать двойному интервалу. Область действия переопределения можно ограничить группой или командными скобками. В первом случае закрывающая скобка должна стоять после пустой строки или команды `\par`.

Изменить значение междустрочного интервала для всего документа можно также, поместив в преамбуле документа команду

`\linespread{`*factor*`}`

Как и в случае `\baselinestretch`, значение 1.5 аргумента *factor* почти соответствует полуторному интервалу в текстовом процессоре MS Word, а 2 — двойному.

6.4 Переносы слов

Команда

`\-`

указывает, в каком месте слова LaTeX имеет право делать перенос. Она незаменима, когда в русском тексте используются слова в иностранном написании. Пример: `ter\-ra in\-cog\-ni\-ta`.

Команда `\-` разрешает делать переносы даже в словах, которые печатаются машинописным шрифтом.

Для часто используемых слов правила переноса можно объявить в преамбуле входного файла декларацией

`\hyphenation{`*words*`}`

Слова в аргументе декларации отделяются друг от друга пробелом, а допустимые места переноса указываются дефисом. Так, `\hyphenation{Micro-soft мохер}` разрешает переносить «soft» на новую строку в слове «Microsoft» и запрещает любые переносы в слове «мохер».

6.5 Разрыв строки

Команды

`\linebreak[`*n*`]` `\nolinebreak[`*n*`]`

позволяют регулировать разбиением текста между словами на строки. `\linebreak` стимулирует LaTeX завершить текущую строку, а `\nopagebreak` — продолжить её. Опция *n* — целое число от 0 до 4 — задаёт «силу» команды: чем больше значение *n*, тем выше вероятность выполнения команды. Значение 0 равносильно отсутствию команды, а значение 4 принуждает LaTeX выполнить её. По умолчанию действует значение 4.

Разрыв строки между словами можно запретить, связав их пробелом нормального размера, который делает команда

`~`

Пример: `рис.~13`.

Команды

`\\[`*length*`]` `\\*[`*length*`]`

обрывают вёрстку текущей строки и начинают новую строку, причём $*$-форма запрещает перенос новой строки на следующую страницу. Опция *length* задаёт величину дополнительного вертикального пробела перед новой строкой.

7 Как управлять форматированием страниц

7.1 Переключение режима форматирования

По умолчанию при односторонней печати режим форматирования страниц задаёт декларация

```
\raggedbottom
```

Она разрешает LaTeXу оставлять внизу страницы пустое пространство. При двусторонней печати по умолчанию действует декларация

```
\flushbottom
```

и LaTeX, растягивая эластичные вертикальные промежутки, старается заполнить всю отведённую для размещения текста область страницы (см. рис. 1).

Режим форматирования страниц можно передекларировать в любом месте документа. Новый режим вступает в силу, начиная с текущей страницы.

7.2 Разбиение текста на страницы

Команды

```
\pagebreak[n]    \nopagebreak[n]
```

позволяют регулировать разбиением текста на страницы. `\pagebreak` стимулирует LaTeX завершить вёрстку текущей страницы, а `\nopagebreak` — продолжить её. Внутри абзаца команды исполняются после заполнения текущей строки. Опция *n* — целое число от 0 до 4 — задаёт «силу» команды: чем больше значение *n*, тем выше вероятность выполнения команды. Значение 0 равносильно отсутствию команды, а значение 4 принуждает LaTeX выполнить её. Значение по умолчанию равно 4.

Если попытаться командой `\nopagebreak` сделать чересчур длинную страницу, то LaTeX просто проигнорирует её.

7.3 Изменение высоты области с текстом

Для текущей страницы нижнюю границу области, отведённой для размещения текста (см. рис. 1), можно сместить вверх или вниз с помощью команд

```
\enlargethispage{length}    \enlargethispage*{length}
```

*-форма разрешает LaTeXу максимально сжимать все эластичные вертикальные пробелы на этой странице. Величина смещения — неэластичная длина *length*. При отрицательном значении граница смещается вверх.

7.4 Переход на новую страницу

Обрезать текущую страницу и начать новую можно с помощью команд

```
\newpage    \clearpage    \cleardoublepage
```

Команды `\clearpage` и `\cleardoublepage` заставляют LaTeX сначала напечатать на отдельной странице все находящиеся в очереди на размещение плавающие объекты и только после этого переходить на новую страницу. В случае команды `\cleardoublepage` новая страница должна быть нечётной, поэтому при необходимости перед ней создаётся пустая чётная страница.

При печати в две колонки команда `\newpage` обрезает текущую колонку, а не страницу.

8 Списки

LaTeX поддерживает списки разного типа. Во всех случаях каждый элемент списка должен начинаться с команды

`\item[`*label*`]`

Необязательный аргумент *label* печатается перед содержанием элемента, заменяя собой маркёр или номер, принятые для этого элемента по умолчанию. Каждый элемент одного списка может включать в себя другие списки.

8.1 Маркированные списки

Командные скобки

`\begin{itemize}` *items* `\end{itemize}`

создают маркированный список: перед каждым элементом, введённым в список командой `\item` без аргумента, печатается установленный по умолчанию маркёр. При наличии опции у команды `\item` вместо него печатается *label*. Допускается четыре уровня вложенности маркированных списков. Вид маркёра по умолчанию для списков разного уровня задаётся командами

`\labelitemi` `\labelitemii` `\labelitemiii` `\labelitemiv`

Они печатают • (`\textbullet`), – (`\bfseries\textendash`), ∗ (`\textasteriskcentered`) и · (`\textperiodcentered`), соответственно. Все команды можно переопределить с помощью `\renewcommand`.

8.2 Нумерованные списки

Командные скобки

`\begin{enumerate}` *items* `\end{enumerate}`

создают нумерованный список: перед каждым элементом, введённым в список командой `\item` без аргумента, печатается его порядковый номер. При наличии опции у команды `\item` вместо номера печатается *label*. Допускается четыре уровня вложенности нумерованных списков. Порядковые номера элементов списков разного уровня хранятся в счётчиках

`enumi` `enumii` `enumiii` `enumiv`

Только элементы, введённые в список командой `\item` без аргумента, увеличивают на единицу значение счётчика соответствующего уровня. При наличии опции у команды `\item` значение счётчика не меняется. Номера элементов списков разного уровня печатают команды

`\labelenumi` `\labelenumii` `\labelenumiii` `\labelenumiv`

Они используют перечисленные выше счётчики, поэтому вид номера зависит от определения соответствующего счётчика (см. раздел 3). Элементы списка первого уровня нумеруются арабскими цифрами с точкой: 1., 2., 3. и т. д., второго уровня — строчными латинскими буквами в круглых скобках: (a), (b), (c) и т. д., третьего уровня — строчными римскими цифрами с точкой: i., ii., iii. и т. д. и четвёртого уровня — прописными латинскими буквами с точкой: A., B., C. и т. д.

Чтобы изменить формат, в котором будет напечатано значение счётчика, надо переопределить сам счётчик. Например, в документах на русском языке после переопределения

```
\renewcommand{\theenumii}{\asbuk{enumii}}
```

значение счётчика элементов списка второго уровня будет печататься строчными русскими буквами. Чтобы изменить вид номера, надо переопределить команду, которая печатает этот номер. Например, после переопределения

```
\renewcommand{\labelenumii}{\theenumii).}
```

элементы списка второго уровня будут нумероваться строчными русскими буквами, после которых стоит закрывающая круглая скобка и точка: а)., б)., в). и т. д.

На нумерованные элементы списков можно ссылаться, пометив их, как обычно, командой `\label`. Ссылка, которую печатает команда `\ref`, включает в себя полный номер элемента. Например, для первого элемента списка второго уровня, входящего в состав третьего элемента списка первого уровня, она имеет вид 3а. Для элементов списка третьего уровня ссылка имеет вид типа 3(а)ii.

8.3 Списки описаний

Командные скобки

`\begin{description}` *items* `\end{description}`

создают список, который обычно используется для печати словарных статей. Заголовок темы оформляется как аргумент *label* команды `\item`. По умолчанию он печатается шрифтом с повышенной насыщенностью (действует декларация `\bfseries`). Если *label* сам содержит квадратные скобки, то его надо заключить в фигурные скобки.

8.4 Настраиваемые списки

Для всех типов списков расположение метки (маркёра, номера или заголовка) и текста элементов списка по отношению друг к другу и к внешнему тексту задаётся командными длинами, которые приведены на рис. 2.

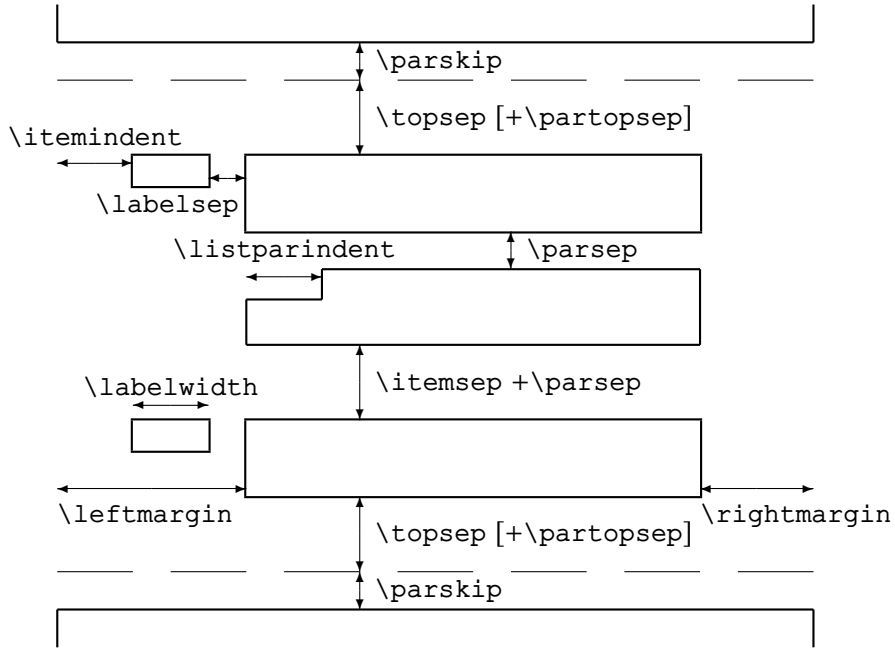

Рис. 2: Макет списка из двух элементов (первый состоит из двух абзацев) с командами, задающими взаимное расположение меток и текста элементов, а также размещение списка по отношению к внешнему тексту.

Вертикальные промежутки. Перед списком и после него вставляется пробел величиной `\topsep`. Если перед списком оставлена пустая строка или имеется команда `\par`, то перед ним и после него вставляется также пробел величиной `\partopsep`.

Дополнительный к междустрочному интервалу промежуток между абзацами внутри элемента задаётся параметром `\parsep` (а не `\parskip`, как в обычном тексте). Между элементами списка в дополнение к `\parsep` вставляется пробел величиной `\itemsep`. В стандартных классах он отличен от нуля, поэтому элементы списки более отделены друг от друга, чем абзацы внутри элемента.

Все командные длины, задающие вертикальные промежутки, являются эластичными.

Горизонтальные промежутки. Параметры `\leftmargin` и `\rightmargin` задают отступ текста элементов списка соответственно от левой и правой границ внешнего текста.

Во всех абзацах элемента списка, кроме первого, задаётся абзацный отступ величиной `\listparindent`. По умолчанию он устанавливается равным нулю. Может иметь отрицательное значение.

Параметр `\labelsep` задаёт расстояние между правым краем метки и левым краем текста, а `\labelwidth` — ширину бокса по умолчанию для метки. Метка в боксе прижимается к его правому краю. Сам бокс можно сдвинуть влево или вправо, используя `\itemindent`. По умолчанию значение этого параметра равно нулю. В стандартных классах ширина бокса устанавливается такой, чтобы его левый край совпадал с левый краем внешнего текста. Если метка шире бокса, то он автоматически расширяется вправо.

Значения всех описанных выше командных длин, принятые по умолчанию для выбранного класса документов, устанавливаются заново при каждом «входе» в список. Поэтому их переопределение перед списком теряет силу внутри списка. В результате, компоновку описанных выше списков `itemize`, `enumerate` и `description` невозможно изменить. Настраиваемый список надо создавать, используя командные скобки

```
\begin{list}{def-label}{settings}
  items
\end{list}
```

Аргумент *def-label* задаёт вид метки по умолчанию, а в аргументе *settings* размещают декларации типа `\setlength`, изменяющие параметры списка. В нём можно также использовать декларацию

```
\usecounter{counter}
```

Она указывает счётчик *counter*, который будет использован для нумерации элементов списка. Аргумент *settings* может быть пустым `{}`. В этом случае используются значения параметров, принятые по умолчанию.

Списки `itemize`, `enumerate` и `description`, описанные в этом разделе, определены через список `list`.

Пример списка `list`, в котором элементы нумеруются как п. 1, п. 2 и т. д.:

```
\newcounter{No}
\begin{list}{п.\,\arabic{No}}{\usecounter{No}}
  \item ...
\end{list}
```

В отличие от `enumerate`, в этом списке ширина правого поля равна нулю, поскольку именно это значение устанавливается для `\rightmargin` по умолчанию.

8.5 Примитивный список

Командные скобки

```
\begin{trivlist}  items  \end{trivlist}
```

создают список, в котором не только правое, но и левое поле списка, а также ширина бокса для меток и его отступ устанавливаются равными нулю. Кроме того, `\parsep` устанавливается равным

\parskip. Остальные параметры первоначально имеют значения, принятые по умолчанию. При «входе» в список они сохраняют свои текущие значения, поэтому переопределения параметров, сделанные перед списком, действуют в нём.

9 Специальные абзацы

9.1 Позиционирование текста в строке

Все строки в командных скобках

```
\begin{center}  ...  \end{center}
```

или в области действия декларации

```
\centering
```

центрируются.

Строки в командных скобках

```
\begin{flushleft}  ...  \end{flushleft}
```

или в области действия декларации

```
\raggedright
```

прижимаются к левому краю внешнего текста.

Строки в командных скобках

```
\begin{flushright}  ...  \end{flushright}
```

или в области действия декларации

```
\raggedleft
```

прижимаются к правому краю внешнего текста.

Во всех случаях: если текст не помещается в одной строке, то он автоматически переносится на следующую строку без переносов в словах. Промежутки между словами фиксированы. Для разбиения текста на строки надо использовать команду \\.

Окружения `center`, `flushleft` и `flushright` определены как список `trivlist` с одним элементом `\item[]` (см. раздел 8.5), поэтому перед и после них вставляется вертикальный пробел величиной `\topsep`, а если им предшествует пустая строка или команда `\par`, то ещё и пробел величиной `\partopsep`. Значение обеих командных длин можно изменить посредством `\setlength` или `\addtolength`.

Первая строка абзаца, следующего за окружениями `center`, `flushleft` и `flushright`, начинается со стандартного отступа, если между ними вставлена пустая строка или команда `\par`.

9.2 Выделение текста

Текст в командных скобках

```
\begin{quote}      ...  \end{quote}
\begin{quotation}  ...  \end{quotation}
```

печатается с дополнительным отступом от левого и правого краёв внешнего текста. Оба окружения определены как список `list` с одним элементом `\item[]` (см. раздел 8.4), поэтому перед и после них вставляется вертикальный пробел величиной `\topsep`, а если им предшествует пустая строка или команда `\par`, то ещё и пробел величиной `\partopsep`. В обоих случаях используются значения параметров, принятые по умолчанию. Поэтому изменение значений этих командных длин во входном файле не влияет на вертикальные пробелы вокруг окружений `quote` и `quotation`.

В `quotation` в начале каждого абзаца делается стандартный отступ, а в `quote` — нет. Поэтому в окружении `quote` между абзацами вставляется дополнительный вертикальный промежуток.

Первая строка абзаца, следующего за окружениями `quote` и `quotation`, начинается со стандартного отступа, если между ними вставлена пустая строка или команда `\par`.

9.3 Буквальное воспроизведение текста

Текст в командных скобках

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

печатается машинописным шрифтом в том виде, в котором он набран во входном файле. В ∗-форме на месте пробелов печатается символ ␣.

Закрывающая командная скобка имеет одну особенность: между `\end` и `{verbatim}` не должно быть пробелов. Пакет verbatim из коллекции `tools` переопределяет окружение `verbatim`. В нём устранена эта особенность. Но зато текст, расположенный до конца строки сразу после закрывающей скобки игнорируется.

Окружение `verbatim` определено как список `trivlist` с одним элементом `\item[]` (см. раздел 8.5), поэтому перед и после него вставляется вертикальный пробел величиной `\topsep`, а если ему предшествует пустая строка или команда `\par`, то ещё и пробел величиной `\partopsep`. Значение обеих командных длин можно изменить посредством `\setlength` или `\addtolength`.

Первая строка абзаца, следующего за окружением `verbatim`, начинается со стандартного отступа, если между ними вставлена пустая строка или команда `\par`.

Команды

```
\verb*text*      \verb*text*
```

печатают внутри абзаца машинописным шрифтов *text* в том виде, в котором он набран во входном файле. Здесь `*` — любой символ, отсутствующий в *text*, например, `|` или `"`. В ∗-форме на месте пробелов печатается символ ␣.

Окружение `verbatim` и команду `\verb` нельзя использовать в аргументах команд.

9.3.1 Пакет shortvrb

Пакет shortvrb, входящий в LaTeX, вводит декларацию

```
\MakeShortVerb{\*}
```

которая делает любой символ `*` специальной «скобкой»: текст, окружённый такими скобками печатается как аргумент команды `\verb`. Например, после объявления `\MakeShortVerb{\+}` вместо `\verb|`*text*`|` достаточно набрать `+`*text*`+`. Можно использовать сразу несколько деклараций.

Декларация

```
\DeleteShortVerb{\*}
```

возвращает символу `*` обычное значение.

9.3.2 Пакет alltt

Пакет alltt, входящий в LaTeX, вводит одноимённые командные скобки

```
\begin{alltt}   ...   \end{alltt}
```

текст в которых печатается машинописным шрифтом в том виде, в котором он набран во входном файле: сохраняются все пробелы и разбиение на строки. Но в отличие от `verbatim`, символы `\`, `{` и `}` остаются специальными, что позволяет использовать внутри окружения `alltt` команды переключения шрифта типа `\textrm{`*text*`}` и математическую моду `\(`*math*`\)`. В *math* верхние и нижние индексы следует оформлять как аргумент соответственно команд `\sp` и `\sb`, поскольку символы `^` и `_` в `alltt` не являются управляющими (как впрочем и `$`).

10 Боксы

10.1 Строковые боксы

Команда

`\makebox[`*width*`][`*position*`]{`*text*`}`

создаёт бокс шириной *width* и печатает в нём *text* в одну строку. Расположение текста в боксе задаётся аргументом *position* с допустимыми значениями `l`, `c` (по умолчанию), `r` и `s`, которые соответствуют размещению текста у левого края, по центру, у правого края и растягиванию его на всю ширину бокса. Опция `s` работает только тогда, когда *text* содержит растяжимые горизонтальные длины, например, обычные пробелы. Если опция *width* опущена, то ширину бокса задаёт сам *text*. В аргументе *width* можно использовать командные длины

`\width` `\height` `\depth` `\totalheight`

Они равны ширине, высоте, глубине и полной высоте бокса аргумента *text*.

Команда

`\framebox[`*width*`][`*position*`]{`*text*`}`

работает как `\makebox`, но рисует вокруг бокса рамку. Толщина линии рамки и расстояние между рамкой и текстом задаются командными длинами

`\fboxrule` `\fboxsep`

соответственно. По умолчанию они равны 0.4 pt и 3 pt.

Команды

`\mbox{`*text*`}` `\fbox{`*text*`}`

эквивалентны командам `\makebox` и `\framebox` с опущенными необязательными аргументами.

10.1.1 Измерение размеров бокса

Декларации

`\settowidth{`*cmd-length*`}{`*text*`}`
`\settoheight{`*cmd-length*`}{`*text*`}`
`\settodepth{`*cmd-length*`}{`*text*`}`

измеряют ширину, высоту и глубину бокса аргумента *text* и запоминают их значения в командных длинах *cmd-length*. Последние должны быть заранее определены командой `\newlength`.

10.1.2 Сдвиг бокса по вертикали

Команда

`\raisebox{`*offset*`}[`*h*`][`*d*`]{`*text*`}`

печатает *text*, сместив его по вертикали на расстояние *offset* от базисной линии строки. Опции *h* и *d* задают фиктивные высоту и глубину текста в боксе. Именно эти значения использует TEX, когда отводит место под бокс.

10.1.3 Предварительное форматирование бокса

Команда

`\newsavebox{`*cmd*`}`

определяет имя новой команды *cmd* с глобальной областью действия для команд

```
\savebox{cmd}[width][position]{text}
\sbox{cmd}{text}
```

и командных скобок

```
\begin{lrbox}{cmd}  text  \end{lrbox}
```

Они запоминают под именем *cmd* сформатированный бокс аргумента *text*. Опции у `\savebox` имеют то же назначение, что и у `\makebox`. Пробелы перед и после *text* в `lrbox` игнорируются, а сам текст может содержать команду `\verb` и командные скобки `verbatim`. Печатает бокс команда

```
\usebox{cmd}
```

10.2 Текстовые боксы

Команда

```
\parbox[align][height][inner-align]{width}{text}
```

печатает *text* в текстовой моде, формируя абзац шириной *width*. Положение бокса в текущей строке задаётся значением аргумента *align*: `t` или `b` — базисная линия верхней или нижней строки бокса совпадает с базисной линией текущей строки; `c` (по умолчанию) — центр бокса выравнивается по центру текущей строки. Если задана высота бокса *height*, то текст внутри бокса позиционируется значением аргумента *inner-align*: `t`, `c` (по умолчанию), `b`, `s` — вверху, по центру, внизу, растягивается на всю высоту бокса. Если опция *inner-align* опущена, то используется значение опции *align*.

Для создания больших текстовых боксов с таблицами, списками, примечаниями и т. п. предназначены командные скобки

```
\begin{minipage}[align][height][inner-align]{width}
  text
\end{minipage}
```

Опции у `minipage` имеют то же назначение, что и у команды `\parbox`.

10.3 Линейка

Команда

```
\rule[offset]{width}{hight}
```

печатает чёрный прямоугольник шириной *width* и высотой *hight*, смещая его по вертикали на расстояние *offset* от базисной линии строки.

11 Таблицы

Командные скобки

```
\begin{tabular}[align]{keys}
  strings
\end{tabular}
```

создают таблицу, которая является боксом и, следовательно, должна умещаться на одной странице (её обычно помещают в окружение `table`, которое создаёт нумерованные плавающие объекты).

Аргумент *align* задаёт расположение таблицы по вертикали в текущей строке. Его допустимые значения: `t`, `c` (по умолчанию) и `b`. По умолчанию таблица позиционируется по центру текущей строки. В случае `t` и `b` — таблица позиционируется так, чтобы базовая линия её соответственно первой и последней строки совпадала с базовой линией текущей строки. Однако, если перед таблицей

или после неё проведена горизонтальная линия (см. ниже), то именно эта линия используется вместо базовых линий строк таблицы.

Количество колонок в таблице задаётся в обязательном аргументе *keys*. Колонки создаются ключами `l`, `c`, `r` и `p{`*width*`}`: по одной колонки на каждый ключ. В случае `l`, `c` и `r` — текст в ячейке печатается в строковой моде, причём ширина колонки устанавливается равной ширине самой широкой ячейки. В остальный ячейках этой колонки текст прижимается к левому (`l`) или к правому (`r`) краю или центрируется (`c`). Ключ `p{`*width*`}` создаёт колонку, в ячейках которой текст печатается в виде текстового бокса шириной *width*, причём базисная линия верхней строки бокса совпадает с базисной линией текущей строки таблицы. В таких ячейках команду `\\` можно использовать только внутри `array`, `tabular`, `minipage`, в аргументе *text* команды `\parbox`, в заданной фигурными скобками области действия деклараций `\centering`, `\raggedleft` и `\raggedright`.

При вёрстке таблицы по обе стороны каждой колонки вставляется пробел величиной

`\tabcolsep`

По умолчанию в стандартных классах, кроме `slides`, он равен 6 pt.

Кроме описанных выше ключей `l`, `c`, `r` и `p{`*width*`}`, в аргументе *keys* используются ключи `|` и `@{`*text*`}`. Ключ `|` задаёт вертикальную разделительную линию между колонками на всю высоту таблицы. Ключ `@{`*text*`}` отменяет вставку пробела между колонками: вместо него во всех строках таблицы печатается *text* в качестве разделителя колонок. Этот ключ с пустым аргументом `@{}` можно использовать для подавления пробела перед первой и после последней колонки.

Ключи `|` и `@{..}` относятся к ячейки, за которой они стоят (кроме первой).

В `@`-выражении можно использовать команду

`\extracolsep{`*length*`}`

которая вставляет пробел величиной *length* с левой стороны всех последующих колонок, кроме первой.

Если таблица содержит несколько наборов колонок одного вида, то в аргументе *keys* их можно задать в виде `*{`*n*`}{`*keys*`}`, где *n* — количество копий колонок, *keys* — набор из описанных выше ключей. Пример аргумента *keys*: `|c|*{3}{p{5cm}|}`.

В каждой строке таблицы переход из одной ячейки в следующую задаётся символом `&`. Область дейстия деклараций, находящихся внутри ячейки, ограничивается самой ячейкой. Количество ячеек в любой строке не должно превышать количества колонок в таблице, которое задано в аргументе *keys*. Ячейки могут быть пустыми. Признаком окончания строки таблицы служит команда `\\` (или её вариант `\\[`*length*`]`) или

`\tabularnewline[`*length*`]`

Здесь *length* — дополнительный вертикальный пробел перед следующей строкой. В конце последней строки команду `\\` или `\tabularnewline` можно не ставить. После них, а также перед первой строкой, доступны команды

`\hline` `\cline{`*i-j*`}`

Первая проводит горизонтальную линию на всю ширину таблицы, а вторая — через колонки с *i*-ой по *j*-ую включительно. Внутри любой ячейки, а также в `@`-выражении, доступна команда

`\vline`

Она проводит вертикальную линию на полную высоту строки.

Несколько ячеек можно слить в одну с помощью команды

`\multicolumn{`*n*`}{`*keys*`}{`*text*`}`

Здесь *n* — количество ячеек ($n = 1, 2, \ldots$).

Расстояние между строками таблицы можно изменить переопределением с помощью декларации `\renewcommand` междустрочного интервала

`\arraystretch`

По умолчанию он равен 1.

Командная длина

```
\arrayrulewidth
```

задаёт толщину разделительных линий, которые чертят в таблице команды `\hline`, `\cline` и `\vline`. По умолчанию в стандартных классах, кроме `slides`, она равна 0.4 pt. Если заданы сразу две линии подряд, то они рисуются на расстоянии

```
\doublerulesep
```

друг от друга. По умолчанию в стандартных классах, кроме `slides`, оно равно 2 pt.

11.1 Пакет array

Пакет array из коллекции `tools` предлагает новую версию окружения `tabular`. В отличие от описанной выше стандартной версии, она содержит ряд новых ключей.

Ключ `|` модифицирован: теперь ширина между колонками увеличивается на ширину вертикальной разделительной линии, что существенно в случае толстых линий.

Ключ `!{`*text*`}` печатает *text* в качестве разделителя колонок. В отличие от `@{..}`, он не подавляет пробелы вокруг колонок.

Ключи `m{`*width*`}` и `b{`*width*`}` создают колонку, в ячейках которой текст печатается в виде текстового бокса шириной *width*, который в случае `m{..}` позиционируется по центру текущей строки таблицы, а в случае `b{..}` располагается так, чтобы базисная линия последней строки бокса совпадала с базисной линией текущей строки таблицы.

Ключи `>{`*before*`}` и `<{`*after*`}` используются соответственно перед и после ключей, создающих колонки (`l`, `c`, `r`, `p{..}`, `m{..}` и `b{..}`) для вставки в начало и в конец каждой ячейки *before* и *after*. Например, текст во всех ячейках колонки типа `>{\bfseries}c` будет напечатан шрифтом повышенной насыщенности, поскольку в начало каждой ячейки будет добавлена декларация `\bfseries`. В свою очередь, текст во всех ячейках колонки типа `>{\(}c<{\)}` будет свёрстан в математической моде.

Вместо неоднократного длинного описания типа колонки можно с помощью декларации

```
\newcolumntype{name}[n]{definition}
```

определить новый ключ *name* с *n* аргументами и замещающим текстом *definition* (аргументы задаются точно так же, как и в `\newcommand`). Пример: `\newcolumntype{B}{>{\bfseries}c}`.

Команда `\showcols` печатает в `log`-файле список нестандартных ключей, определённых во входном файле.

В окружении `tabular` пакета array над строками таблицы вставляется дополнительный вертикальный пробел, равный значению командной длины

```
\extrarowheight
```

По умолчанию он равен нулю.

Базовая линия первой (последней) строки таблицы с опцией `t` (`b`) будет совпадать с базовой линией текущей строки, если горизонтальные линии перед (после) таблицы проведены соответственно командами

```
\firsthline     \lasthline
```

Если линии проведены командой `\hline`, то, как и в стандарной версии `tabular`, именно они, а не строки самой таблицы, используются при её позиционировании по вертикали.

Команды `\firsthline` и `\lasthline` можно использовать для вставки дополнительного вертикального пробела между линией и соответственно первой и последней строками таблицы. Величина пробела задаётся командной длиной

```
\extratabsurround
```

По умолчанию её значение равно 2 pt.

11.2 Таблицы заданной ширины

Командные скобки

```
\begin{tabular*}{width}[align]{keys}
  strings
\end{tabular*}
```

создают таблицу шириной *width*: между колонками должен быть вставлен эластичный пробел, способный растянуть таблицу до заданной ширины. В строке под такие таблицы отводится место согласно заказанной ширине. Если истинная ширина таблицы (определяется содержанием ячеек) больше, чем заказанная, то таблица наедет на соседним с ним текст. Эластичный пробел обычно вставляется с помощью `@{\extracolsep{\fill}}`. В остальном окружение `tabular*` идентично стандартной версии `tabular`, а после подключения пакета array — версии `tabular` из этого пакета.

В пакете tabularx из коллекции `tools` определены командные скобки

```
\begin{tabularx}{width}[align]{keys}
  strings
\end{tabularx}
```

которые создают таблицу шириной *width*, но не за счёт увеличения расстояния между колонками, как в случае `tabular*`, а за счёт подбора ширины колонок. Колонки, ширина которых должна подбираться LaTeXом, помечаются в аргументе *keys* ключом `X`. По умолчанию эти колонки становятся колонками типа `p{..}`. Их можно превратить в колонки типа `m{..}` или `b{..}` из пакета array, который загружается пакетом tabularx. Для этого надо переопределить команду

```
\tabularxcolumn{col.width}
```

которая первоначально определена как `\newcommand{\tabularxcolumn}[1]{p{#1}}`. Например, после переопределения

```
\renewcommand{\tabularxcolumn}[1]{m{#1}}
```

колонки `X` после вычисления их ширины становятся колонками типа `m{..}`.

Если в таблице имеется несколько колонок `X`, то все они устанавливаются одинаковой ширины. Можно, однако, изменяя значение командной длины

```
\hsize
```

создать колонки `X` разной ширины. Например, если в аргументе *keys* вместо `XX` задать

```
>{\setlength{\hsize}{.5\hsize}X>{\setlength{\hsize}{1.5\hsize}X
```

то вторая колонку будет в три раза шире первой. Меняя ширину колонок, надо следить, чтобы их суммарная ширина оставалась неизменной! Колонки `X` разной ширины имеют одно ограничение: их нельзя сливать с помощью `\multicolumn`.

Если `tabularx` используется внутри другой таблицы, то его надо заключить в фигурные скобки. При использовании деклараций `\centering`, `\raggedleft` и `\raggedright`, надо сразу после них поместить команду

```
\arraybackslash
```

В ячейках `tabularx` можно использовать команду `\footnote`.

Ограничение: аргумент команды `\verb` в ячейках `tabularx` не должен содержать непарных фигурных скобок и символов `%`; кроме того, пробелы могут отображаться некорректно.

11.3 Размещение таблицы на нескольких страницах

В пакете longtable определены командные скобки

```
\begin{longtable}[position]{keys}
  strings
\end{longtable}
```

для печати таблиц на нескольких страницах. Печать начинается с новой строки на текущей странице. Положение таблицы по горизонтали задаётся аргументом *position* с допустимыми значениями `l`, `c` (по умолчанию) и `r`. По умолчанию таблица центрируется, а в случае `l` и `r` — прижимается соответственно к левому и правому полю страницы. Колонки и разделители между ними задаются в аргументе *keys*, причём используются те же ключи, что и в стандартной версии `tabular`. После подключения пакета array можно использовать также ключи, введённые в этом пакете.

Обычно первые строки таблицы содержат заголовки колонок. Если таблица расположена на нескольких страницах, то такие строки, как правило, должны дублироваться в начале каждой страницы. Кроме того, в конце страницы, где происходит разрыв таблицы, целесообразно указать, что она продолжается на следующей странице. Поддержка описанных выше требований к длинным таблицам реализована в `longtable` следующим образом. В начале окружения надо задать четыре группы строк, причём любая группа может быть опущена. Группа строк, после которой стоит команда

```
\endfirsthead
```

будет напечатана только в самом начале таблицы. Строки этой группы содержат обычно как заголовок самой таблицы, так и заголовки колонок. Группа строк, после которой стоит команда

```
\endhead
```

будет напечатана в верхней части таблицы на всех страницах, кроме первой. Группа строк, после которой стоит команда

```
\endfoot
```

будет напечатана в нижней части таблицы на всех страницах, кроме последней. И, наконец, группа строк, после которой стоит команда

```
\endlastfoot
```

будет напечатана только в самом конце таблицы.

Подпись к таблице можно напечатать командой

```
\caption[entry]{head}
```

Она, как и подпись, созданная командой `\caption` в окружении `table`, начинается с ключевого слова **Table**, за которым следует порядковый номер таблицы, двоеточие и текст *head*. При наличии опции, текст *entry*, а не *head* включается в список таблиц. В повторяющихся строках таблицы надо использовать команду `\caption` с пустым необязательным аргументом: `\caption[]{`*head*`}`. В этом случае ничего не будет заносится в список таблиц. Можно также воспользоваться командой

```
\caption*{head}
```

Она не нумерует таблицу, а её аргумент не заносится в список таблиц.

По умолчанию таблица центрируется по горизонтали, поскольку командные длины

```
\LTleft     \LTright
```

задающие отступ таблицы соответственно от левого и правого поля, определены как `\fill`. Их можно переопределить посредством `\setlength`. По крайней мере один из отступов или межколоночный пробел должны быть заданы как эластичная длина, чтобы заполнить всю ширину страницы.

Расстояния до и после таблицы задаётся эластичными командными длинами

```
\LTpre     \LTpost
```

По умолчанию они равны `\bigskipamount`. Ширина подписи к таблице задаётся командной длиной

`\LTcapwidth`

По умолчанию она равна 4 дюйма.

12 Плавающие объекты

12.1 Рисунки и таблицы

Чтобы избежать полупустых страниц, любой материал (рисунки, таблицы и т. п.), который не подлежит переносу на следующую страницу по частям, надо оформлять как плавающий объект: LaTeX сам подыщет для него подходящее место, передвигая на следующие страницы. Плавающие объекты создают командные скобки

```
\begin{figure}[placement]  ...  \end{figure}
\begin{table}[placement]   ...  \end{table}
```

При печати в две колонки каждый объект размещается в одной из колонок. Плавающие объекты, созданные командными скобками

```
\begin{figure*}[placement]  ...  \end{figure*}
\begin{table*}[placement]   ...  \end{table*}
```

занимают обе колонки при двухколоночном режиме печати. Опция *placement* уточняет правила размещения объекта. Она может состоять из любой последовательности следующих ключей:

`h` разрешает размещение объекта сразу после заполнения текущей строки текста. Не используется в *-формах при двухколоночной печати.

`t` разрешает размещение объекта в верхней части страницы над текстом.

`b` разрешает размещение объекта в нижней части страницы под текстом.

`p` разрешает размещение объекта на специальной странице, содержащей только плавающие объекты.

`!` снимает ряд описанных ниже ограничений, которые могут препятствовать размещению объекта. Используется только в комбинации с другими ключами.

По умолчанию в стандартных классах используется последовательность `tbp`.

Команды `\clearpage` и `\cleardoublepage` принуждают LaTeX вставить в документ все ещё не включённые в него объекты, не считаясь с ключами.

Декларация

`\suppressfloats[key]`

запрещает размещение на текущей странице плавающих объектов, которые во входном файле следуют за ней. Они перемещаются на следующую страницу. По умолчанию запрет действует на объекты, местоположение которых соответствует ключам `t` и `b`, причём ключ `!` у плавающего объекта отменяет действие декларации и на эти объекты. Опция *key* может принимать значение `t` или `b`, ограничивая действие декларации только на объекты, местоположение которых соответствует такому же ключу.

Подпись к рисунку или таблице печатает команда

`\caption[entry]{head}`

Подпись начинается с ключевого слова: **Figure** в окружениях `figure` и **Table** в окружениях `table`, за которым следует порядковый номер рисунка или таблицы, и затем после двоеточия печатается текст *head*. При наличии опции, *entry*, а не *head* включается в список рисунков и таблиц.

Слова Figure и Table хранятся соответственно в командах

```
\figurename     \tablename
```

Их можно переопределить с помощью `\renewcommand`. Пакет babel с опцией `russian` переопределяет их как **Рис.** и **Таблица**.

В России после номера рисунка или таблицы принято ставить точку, а не двоеточие, как в стандартном LaTeXe. Необходимую замену позволяет сделать пакет caption2. Для этого достаточно переопределить введённую в этом пакете команду

```
\captionlabeldelim
```

следующим образом

```
\renewcommand{\captionlabeldelim}{.}
```

12.1.1 Количество плавающих объектов на странице

Счётчики

```
topnumber   bottomnumber   totalnumber
```

задают максимальное число плавающих объектов, которые могут быть размещены соответственно вверху, внизу и в любом месте страницы. По умолчанию их значения равны 2, 1 и 3. При печати в две колонки эти же счётчики контролируют размещение одноколоночных объектов, а максимальное число занимающих сразу обе колонки объектов, которые могут быть размещены вверху страницы, задаётся счётчиком

```
dbltopnumber
```

По умолчанию его значение равно 2.

Ключ `!` снимает ограничения на размещение объектов, которые устанавливают описанные выше счётчики.

12.1.2 Доля страницы, отводимая под плавающие объекты

Команды

```
\topfraction     \bottomfraction
```

задают максимальную долю страницы, которая может быть занята плавающими объектами в её верхней и нижней частях, соответственно. По умолчанию они равны 0.7 и 0.3. При печати в две колонки эти же команды контролируют размещение одноколоночных объектов, а максимальная доля страницы, которая может быть занята в её верхней части объектами, занимающими сразу обе колонки, задаётся командой

```
\dbltopfraction
```

По умолчанию её значение равно 0.7.

Команда

```
\textfraction
```

задаёт минимальную долю страницы, которая должна быть занята текстом. По умолчанию значение команды равно 0.2.

Команды

```
\floatpagefraction     \dblfloatpagefraction
```

задают минимальную долю специальной страницы, которая должна быть занята соответственно одно- и двухколоночными объектами. По умолчанию она равна 0.5. Следовательно, если плавающие объекты с ключом `p` занимают меньше половины страницы, то для их размещения специальная страница не будет создана.

Все команды, описанные в этом разделе, переопределяются посредством `\renewcommand`.

Ключ `!` снимает ограничения на размещение объектов, которые устанавливают описанные выше команды.

12.1.3 Вертикальные пробелы вокруг плавающих объектов

Все командные длины, описанные в этом разделе, являются эластичными.

Командная длина

`\intextsep`

задаёт вертикальный промежуток перед и после объектов, местоположение которых соответствует ключу `h`. Имеет естественную длину 12 pt.

Командные длины

`\textfloatsep` `\dbltextfloatsep`

задают вертикальный промежуток между текстом и соответственно одно- и двухколоночными объектами, местоположение которых соответствует ключам `t` или `b`. Имеет естественную длину 20 pt.

Командные длины

`\floatsep` `\dblfloatsep`

задают вертикальный промежуток между соответственно одно- и двухколоночными объектами, местоположение которых соответствует ключам `t` или `b`. Имеет естественную длину 12 pt.

12.2 Рисунки и таблицы, обтекаемые текстом

Пакет floatflt вводит окружения `floatingfigure` и `floatingtable` для размещения небольших рисунков и таблиц так, чтобы текст мог обтекать их. Рисунок или таблица располагаются у левого или правого поля страницы, причём их верхний край совпадает с первой строкой того абзаца, в начале которого они размещены в исходном тексте. Пакет имеет три опции: `rflt`, `lflt` и `vflt`. По умолчанию действует `vflt` и обтекаемый объект располагается у левого поля на чётных страницах и у правого поля на нечётных страницах. Опции `rflt` и `lflt` задают размещение всех объектов соответственно только у правого и у левого поля страницы.

Окружение `floatingfigure` имеет следующий синтаксис:

`\begin{floatingfigure}[`*position*`]{`*width*`}` *figure* `\end{floatingfigure}`

Аргумент *width* задаёт ширину бокса, который отводится под рисунок *figure*. Опция *position* позволяет изменить заданное опцией пакета расположение рисунка на странице. Значения `r`, `l` и `p` задают те же правила, что и опции пакета `rflt`, `lflt` и `vflt`. Значение `v` указывает, что для текущего объекта надо использовать опцию пакета.

Подпись к рисунку, как и в окружении `figure`, можно напечатать командой `\caption`.

Окружение `floatingtable` имеет следующий синтаксис:

```
\begin{floatingtable}[position]{tabular environmemt}
[\caption{text}]
\end{floatingtable}
```

Здесь обязательным аргументом является окружение `tabular` (или `\parbox`) и ширина обтекаемого объекта определяется естественной шириной таблицы. Опция *position* имеет то же предназначение, что и у `floatingfigure`.

Подпись к таблице, как и в окружении `table`, можно напечатать командой `\caption`.

Предупреждение LaTeX может выбросить или усечь обтекаемый объект при возникновении проблем с его размещением. Поэтому надо, чтобы всегда было достаточно текста для полного обтекания объекта.

12.3 Заметки на полях

Команда

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

печатает в текстовой моде содержание аргумента *text* в виде заметок на полях. При односторонней печати заметки всегда размещаются на правом поле страницы, а при двусторонней печати — на внешнем от переплёта поле: правом на нечётных и левом на чётных страницах. При двухколоночной печати заметка выносится на ближайшее поле. Для левого поля можно задать с помощью опции *left-text* другую заметку, чем для правого поля. Декларация

`\reversemarginpar`

меняет эти правила на противоположные, а декларация

`\normalmarginpar`

восстанавливает их.

На полях страницы заметка позиционируется так, чтобы её первая строка находилась на уровне строки текста, содержащей команду `\marginpar`. Если команда находится между абзацами, то заметка позиционируется на уровне последней строки предшествующего ей абзаца. Если заметка перекрывает предыдущую заметку, то она смещается вниз. Размер и положение заметок по горизонтали, а также вертикальный промежуток между заметками задаются командными длинами, указанными на рис. 1, расположенном на стр. 7.

13 Текстовые шрифты

13.1 Атрибуты шрифтов

Каждый шрифт в LaTeXe имеет пять атрибутов. Атрибут *encoding* указывает кодировку шрифта, которая задаёт порядок расположения символов, входящих в состав шрифта. По умолчанию используется значение `OT1`. Стандартной кодировкой шрифтов с русскими буквами является `T2A`. Остальные атрибуты шрифта характеризуют его полиграфические свойства: атрибут *family* соответствует гарнитуре шрифта, *series* — насыщенности, *shape* — начертанию, а *size* — кеглю.

Атрибуты шрифта не зависят друг от друга. Можно сменить гарнитуру шрифта, сохраняя при этом установки атрибутов начертания или насыщенности. Если шрифта с заказанной комбинацией значений атрибутов (*encoding*, *family* и т. д.) не окажется, то LaTeX выведет предупреждение и заменит её на ту, которая, по его мнению, ближе всего соответствует заказанной. Набор допустимых комбинаций значений атрибутов для каждого семейства шрифтов можно найти в служебных файлах определения шрифта. Они имеют расширение `fd`. Имена файлов составлены из аббревиатур кодировки и гарнитуры. Например, для шрифтов семейства Computer Modern Roman в кодировке `OT1` это файл `ot1cmr.fd`.

Для пользователей LaTeX имеет целый набор команд для выбора шрифтов с разными комбинациями значений атрибутов. Все они описаны в разделах с 13.2 по 13.5.

13.2 Переключение гарнитуры шрифта

Декларации

`\rmfamily` `\sffamily` `\ttfamily`

объявляют текущим шрифт со значением атрибута *family*, хранящимся соответственно в командах

`\rmdefault` `\sfdefault` `\ttdefault`

По умолчанию им присваиваются значения `cmr`, `cmss` и `cmtt` шрифтов семейства Computer Modern.

Если в преамбуле входного файла объявить переопределение, скажем

```
\renewcommand{\rmdefault}{ptm}
```

то вместо шрифта Computer Modern Roman будет взят шрифт Adobe Times Roman, имеющий значение атрибута *family* `ptm` (поддерживается коллекцией пакетов psnfss).

В командах `\rmdefault`, `\sffamily` и `\ttfamily` принято хранить значение атрибута *family* шрифтов с засечками (типа Times), без засечек (типа Helvetica или Arial) и машинописных (типа Courier).

По умолчанию документ печатается шрифтом, значение атрибута *family* которого хранится в команде `\rmdefault`, поскольку именно оно присваивается команде `\familydefault`.

Для коротких фраз вместо деклараций удобнее пользоваться командами

`\textrm{`*text*`}` `\textsf{`*text*`}` `\texttt{`*text*`}`

Декларации `\fontfamily` и `\usefont`, описанные в разделе 13.6, позволяют выбрать шрифт с произвольной гарнитурой.

13.3 Переключение насыщенности шрифта

Декларации

`\mdseries` `\bfseries`

объявляют текущим шрифт со значением атрибута *series*, хранящимся соответственно в командах

`\mddefault` `\bfdefault`

По умолчанию им присваиваются наиболее распространённые для шрифтов нормальной и полужирной насыщенности значения `m` (medium) и `b` (bold).

По умолчанию документ печатается шрифтом, значение атрибута *series* которого хранится в команде `\mddefault`, поскольку именно оно присваивается команде `\seriesdefault`.

Для коротких фраз вместо деклараций удобнее пользоваться командами

`\textmd{`*text*`}` `\textbf{`*text*`}`

Декларации `\fontfamily` и `\usefont`, описанные в разделе 13.6, позволяют выбрать шрифт с произвольной насыщенность.

13.4 Переключение начертания шрифта

Декларации

`\upshape` `\itshape` `\slshape` `\scshape`

объявляют текущим шрифт со значением атрибута *shape*, хранящимся соответственно в командах

`\updefault` `\itdefault` `\sldefault` `\scdefault`

По умолчанию им присваиваются значения: `n` (normal), `it` (italic), `sl` (slanted) и `sc` (small caps), соответствующие шрифтам прямого, курсивного, наклонного и в стиле заглавных букв начертания.

По умолчанию документ печатается шрифтом, значение атрибута *shape* которого хранится в команде `\updefault`, поскольку именно оно присваивается команде `\shapedefault`.

Для коротких фраз вместо деклараций удобнее пользоваться командами

`\textup{`*text*`}` `\textit{`*text*`}` `\textsl{`*text*`}` `\textsc{`*text*`}`

Декларации `\fontfamily` и `\usefont`, описанные в разделе 13.6, позволяют выбрать шрифт с произвольным начертанием.

Имеется специальная команда

`\emph{`*text*`}`

которая печатает аргумент *text* шрифтом с курсивным начертанием, когда окружающий её текст имеет прямое начертание, и наоборот.

13.5 Переключение размера шрифта

Декларации, изменяющие размер текущего шрифта, приведены в таблице 13.5. Наряду с размером символов, они меняют и расстояние между строками.

Таблица 1: Декларации переключения размера шрифтов и округлённые значения атрибута *size* для опций `10pt`, `11pt` и `12pt` в стандартных классах.

Декларация	*size*			Декларация	*size*		
`\tiny`	5pt	6pt	6pt	`\large`	12pt	12pt	14pt
`\scriptsize`	7pt	8pt	8pt	`\Large`	14pt	14pt	17pt
`\footnotesize`	8pt	9pt	10pt	`\LARGE`	17pt	17pt	20pt
`\small`	9pt	10pt	11pt	`\huge`	20pt	20pt	25pt
`\normalsize`	10pt	11pt	12pt	`\Huge`	25pt	25pt	25pt

Произвольный размер шрифта вместе с расстоянием между строками можно задать описанной в разделе 13.6 декларацией `\fontsize`.

13.6 Включение произвольного шрифта

Декларация

`\fontencoding{`*encoding*`}`

объявляет кодировку *encoding*. В свою очередь, декларации

`\fontfamily{`*family*`}` `\fontseries{`*series*`}` `\fontshape{`*shape*`}`

объявляют гарнитуру *family*, насыщенность *series* и начертание *shape*. Декларация

`\fontsize{`*size*`}{`*baselineskip*`}`

объявляет значение *size* в качестве размера шрифта, а *baselineskip* — в качестве расстояния между строками текста. Обычно значение *baselineskip* на 15–20 % больше, чем *size*. Так, по умолчанию они соответственно равны 12 pt и 10 pt.

Приведённые выше декларации `\font...` только объявляют новые значения атрибутов. Для того, чтобы шрифт с этими значениями атрибутов стал текущим, надо применить декларацию

`\selectfont`

Между `\font...` и `\selectfont` не должно быть текста (допускаются только пробелы). Например, текст в области действия деклараций

```
\fontencoding{U}\fontfamily{psy}\selectfont ...
```

будет напечатан шрифтом Adobe Symbol (поддерживается коллекцией пакетов psnfss).

Декларация

`\usefont{`*encoding*`}{`*family*`}{`*series*`}{`*shape*`}`

сама задаёт все значениями атрибутов шрифта, кроме размера, и объявляет его текущим. Например, текст в области действия декларации

```
\usefont{T1}{pzc}{m}{it} ...
```

будет напечатан шрифтом Adobe Zapf Chancery (поддерживается коллекцией пакетов psnfss).

Команда

`\symbol{`*code*`}`

печатает символ с кодом *code* из текущего шрифта. Код можно задавать десятичным числом (от `0` до `255`), в восьмеричном или шестнадцатеричном исчислениях. Числу в восьмеричном исчислении должен предшествовать апостроф `'`, а в шестнадцатеричном исчислении — двойной апостроф `"`.

13.7 Переключение на основной шрифт документа

Текст в области действия декларации

`\normalfont`

или в аргументе команды

`\textnormal{`*text*`}`

всегда печатается шрифтом, установленным для документа по умолчанию.

14 Новые макроопределения

14.1 Команды

Декларация

`\newcommand{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

объявляет новую команду *cmd*. Замещающим её текстом является *definition*. По умолчанию команда не имеет аргументов. Первая опция — целое число *integer* от 1 до 9 — указывает количество аргументов у команды. При наличии второй опции первый аргумент новой команды становится необязательным и по умолчанию принимает значение *default*. Аргументы команды входят в *definition* в виде `#`*n*, где *n* — порядковый номер аргумента. Например,

```
\newcommand{\F}[2][N]{#2_0,\ldots,#2_{#1}}
```

определяет новую команду `\F` с двумя аргументами, причём первый аргумент по умолчанию принимает значение `N`. В результате, `$\F[k]{x}$` печатает $x_0, \ldots, x_k$, а `$\F{y}$` — $y_0, \ldots, y_N$.

Декларация

`\providecommand{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

также объявляет новую команду *cmd*, но если такая команда уже существует, то остаётся в силе старое определение.

Существующие команды *cmd* можно переопределить с помощью декларации

`\renewcommand{`*cmd*`}[`*integer*`][`*default*`]{`*definition*`}`

14.2 Командные скобки

Декларация

`\newenvironment{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

определяет новое окружение *name*. `\begin{`*name*`}` замещается на *begdef*, а `\end{`*name*`}` — на *enddef*. По умолчанию окружение не имеет аргументов. Первая опция — целое число *integer* от 1 до 9 — указывает количество аргументов у окружения. При наличии второй опции первый аргумент нового окружения становится необязательным и по умолчанию принимает значение *default*. Аргументы входят в *begdef* и *enddef* в виде `#`*n*, где *n* — порядковый номер аргумента.

Существующее окружение *name* можно переопределить с помощью декларации

`\renewenvironment{`*name*`}[`*integer*`][`*default*`]{`*begdef*`}{`*enddef*`}`

14.3 *-форма декларации макроопределений

Все пять описанных выше в этом разделе деклараций имеют $*$-форму. Она считается более подходящей при определении команд с аргументами, исключая случай, когда какой-либо из аргументов содержит целый абзац текста: в аргументах $*$-формы недопустимы пустые строки или команда `\par`. Более того, такой текст рекомендуется оформлять не как аргумент, а как содержание должным образом определённого окружения.

15 Символы

15.1 Служебные символы

Символы #, $, %, &, {, } и _, зарезервированные LaTeXом для служебного пользования, можно напечатать командами

```
\#  \$  \%  \&  \{  \}  \_
```

Значки ˆ и ˜ можно напечатать как акценты `\^` и `\~` над «пустым» символом `{}`. Команды

```
\textasciicircum  \textasciitilde
```

печатают эти значки в ином виде: ^ и ~. Символ \ в текстовой моде печатает команда

```
\textbackslash
```

15.2 Национальные символы европейских алфавитов

LaTeX печатает национальные символы из европейских алфавитов, основанных на латинице. Диакритические знаки (акценты) печатаются командами из таблицы 2, где в каждой колонке слева приведён набор знака для буквы o, а справа — результат действия команды.

Таблица 2: Диакритические знаки (акценты).

`\'{o}`	ò	`\'{o}`	ó	`\^{o}`	ô	`\~{o}`	õ	`\={o}`	ō	`\k{o}`[a]	ǫ
`\.{o}`	ȯ	`\"{o}`	ö	`\c{o}`	o̧	`\u{o}`	ŏ	`\v{o}`	ǒ		
`\H{o}`	ő	`\c{o}`	o̧	`\d{o}`	ọ	`\b{o}`	o̲	`\t{oo}`	o͡o		

[a] Доступна после подключения пакета fontenc с опцией `T1`.

В таблицах 3 и 4 собраны команды для печати символов, отсутствующих в латинском алфавите. Справа от каждой команды показан её результат. Команды из таблицы 4 печатают символы из кодировки `T1`. Они становятся доступными после подключения пакета fontenc с опцией `T1`.

Таблица 3: Особые европейские символы.

`\aa`	å	`\AA`	Å	`\ae`	æ	`\AE`	Æ	`\o`	ø	`\O`	Ø	`!‘`	¡
`\oe`	œ	`\OE`	Œ	`\l`	ł	`\L`	Ł	`\ss`	ß	`\SS`	SS	`?‘`	¿

Таблица 4: Особые европейские символы из кодировки T1.

`\dh`	ð	`\DH`	Ð	`\dj`	đ	`\DJ`	Đ	`\ng`	ŋ	`\NG`	Ŋ	`\th`	þ	`\TH`	Þ

Примеры

Французские слова «chef-d’œuvre» (шедевр) и «tête-à-tête» (с глазу на глаз) набраны соответственно как `chef-d'\oe{}uvre` и `t\^{e}te-\'{a}-t\^{e}te`. Немецкое слово «blöße» (нагота) набрано как `bl\"{o}\ss{}e`.

15.3 Кавычки

В английском языке приняты одинарные и двойные кавычки в виде ‘. . . ’ и “. . . ”. Одинарные левые и правые кавычки набираются как обратный ` и прямой ' апострофы. В свою очередь, двойные левые и правые кавычки набираются как лигатуры ``` и ''. В кодировке `OT1` правые двойные кавычки можно набирать также как текстовые кавычки ".

В России приняты французские («. . . ») и немецкие („. . . “) кавычки (они называются «ёлочки» и «лапки», соответственно). «Лапки» обычно используются внутри «ёлочек», например, «. . . наш гордый „Варяг“ . . . ».

После подключения пакета babel с опцией `russian`, французские левые и правые кавычки набираются как лигатуры `<<` и `>>` или командами `"<` и `">`, а немецкие левые и правые кавычки набираются как лигатуры `,,` и `‘‘` или командами `"‘` и `"’`.

Разнообразные европейские кавычки из кодировки `T1` печатаются командами:

`\guillemotleft`	«	`\guillemotright`	»		
`\guilsinglleft`	‹	`\guilsinglright`	›		
`\quotedblbase`	„	`\quotesinglbase`	‚	`\textquotedbl`	"

Они становятся доступными после подключения пакета fontenc с опцией `T1`.

15.4 Дефис и тире

В английском языке символ `-` используется для набора дефиса в составных словах, таких как «daughter-in-law» или «X-axis». Лигатура `--` предназначена для набора короткого тире, которое используется при указании диапазона чисел, например, «10–13». Лигатура `---` используется для набора длинного тире в тексте. Согласно английской традиции длинное тире печатается без отбивки от текста. На тот случай, когда шрифт не поддерживает лигатуры, припасены команды `\textendash` и `\textemdash` для набора короткого и длинного тире.

Опция `russian` пакета babel вводит три команды для набора тире, которые печатаются согласно российской традиции.

Команда `"---` используется для печати тире в тексте. Оно несколько короче английского длинного тире. Кроме того, команда задаёт небольшую жёсткую отбивку от слова, стоящего перед тире. При этом, само тире не отрывается от слова. После тире следует такая же отбивка от текста, как и перед тире. При наборе текста между словом и командой, за которым она следует, должен стоять пробел.

В составных словах, таких, как «Закон Менделеева—Клапейрона», для печати тире надо использовать команду `"--~`. Она ставит более короткое, по сравнению с английским, тире и позволяет делать переносы во втором слове. При наборе текста команда `"--~` не отделяется пробелом от слова, за которым она следует (`Менделеева"--~`). Следующее за командой слово может быть отделено от неё пробелом или перенесено на другую строку.

Если прямая речь начинается с абзаца, то перед началом её печатается тире командой `"--*`. Она печатает русское тире и жёсткую отбивку нужной величины перед текстом.

15.5 Дополнительные символы

В таблицах 5 и 6 собраны наиболее часто встречающиеся специальные символы стандартного LaTeXa. Справа от каждой команды показан её результат.

Таблица 5: Специальные символы.

`\dag`	†	`\S`	§	`\copyright`	©	`\pounds`	£	`\No`[a]	№
`\ddag`	‡	`\P`	¶	`\textcircled{a}`	ⓐ	`\dots`	…		

[a] Вводится опцией `russian` пакета babel

Таблица 6: Математические символы текстовой моды.

`\textasteriskcentered`	*	`\textless`	<	`\textgreater`	>
`\textperiodcentered`	·	`\textbar`	|	`\textbullet`	•

15.6 Пакет textcomp

В пакете textcomp определены команды для печати разнообразных символов, введённых в кодировке `TS1` (шрифты Text Companion). Наиболее полезные из них собраны в таблицах 7 и 8. Справа от каждой команды показан её результат, напечатанный шрифтами из пакета `txfonts`.

Таблица 7: Математические символы пакета textcomp.

`\textminus`	−	`\textpm`	±	`\textleftarrow`	←	`\textuparrow`	↑
`\texttimes`	×	`\textdiv`	÷	`\textrightarrow`	→	`\textdownarrow`	↓

Таблица 8: Научные символы пакета textcomp.

`\textcelsius`	℃	`\textdegree`	°	`\textohm`	Ω
`\textonehalf`	½	`\textonequarter`	¼	`\textthreequarters`	¾
`\textonesuperior`	¹	`\texttwosuperior`	²	`\textthreesuperior`	³

Алфавитный указатель

Символы[3]

!‘ (¡) 43
"(”) 43
"’ (“) 44
"--* (–) 44
"--- (—) 44
"--~ (–) 44
"< («) 44
"> (») 44
"‘ („) 44
% 6
’ (’) 43
’’ (”) 43
\, ... 20
,, („) 44
\- 23
- (-) 44
-- (–) 44
--- (—) 44
<< («) 44
>> (») 44
?‘ (¿) 43
\@ 20
\␣ 20
\" (акцент ¨) 43
\# (#) 43
\$ ($) 43
\% (%) 43
\& (&) 43
\’ (акцент ´) 43
\. (акцент ˙) 43
\= (акцент ¯) 43
\AA (Å) 43
\AE (Æ) 43
\DH (Ð) 43
\DJ (Đ) 43
\H (акцент ˝) 43
\L (Ł) 43
\NG (Ŋ) 43
\No (№) 44
\O (Ø) 43
\OE (Œ) 43
\P (¶) 44
\S (§) 44
\SS (SS) 43
\TH (Þ) 43
\\ 23
* 23
\^ (акцент ˆ) 43
_ (_) 43
\‘ (акцент `) 43
\aa (å) 43
\ae (æ) 43
\b (акцент ˍ) 43
\c (акцент ¸) 43
\copyright (©) 44
\d (акцент ˙) 43
\dag (†) 44
\ddag (‡) 44
\dh (ð) 43
\dj (đ) 43
\dots (. . .) 44
\guillemotleft («) . . . 44
\guillemotright (») . . 44
\guilsinglleft (‹) . . . 44
\guilsinglright (›) . . 44
\k (акцент ˛) 43
\l (ł) 43
\ng (ŋ) 43
\o (ø) 43
\oe (œ) 43
\pounds (£) 44
\quotedblbase („) 44
\quotesinglbase (‚) . . 44
\ss (ß) 43
\t (акцент ͡) 43
\textasciicircum (^) . 43
\textasciitilde (~) . . 43
\textasteriskcentered (*) 44
\textbackslash (\) . . . 43
\textbar (|) 44
\textbullet (•) 44
\textcelsius (℃) 45
\textdegree (°) 45
\textdiv (÷) 45
\textdownarrow (↓) . . . 45
\textgreater (>) 44
\textleftarrow (←) . . 45
\textless (<) 44
\textminus (−) 45
\textohm (Ω) 45
\textonehalf (½) 45
\textonequarter (¼) . . 45
\textonesuperior (¹) . 45
\textperiodcentered (·) 44
\textpm (±) 45
\textquotedbl (") 44
\textrightarrow (→) . 45
\textthreequarters (¾) 45
\textthreesuperior (³) 45
\texttimes (×) 45
\texttwosuperior (²) . 45
\textuparrow (↑) 45
\th (þ) 43
\u (акцент ˘) 43
\v (акцент ˇ) 43
\{ ({) 43
\} (}) 43
\~ (акцент ˜) 43
‘ (‘) 43
‘‘ (“) 43
~ 23

a

abstract 8
\abstractname 8
\addcontentsline 11
\addtocontents 11
\addtocounter 19
\addtolength 19
\addvspace 21
alltt 29
\Alph 19
\alph 19
\alsoname 17
\and 8
\appendix 10
\appendixname 10
\arabic 19
\arraybackslash 34
\arrayrulewidth 33
\arraystretch 32
\Asbuk 19
\asbuk 19
\author 8

b

\backmatter 10
\baselinestretch 22
\bfdefault 40
\bfseries 40
\bibindent 14
\bibitem 14
\bibname 14
\bigskip 21
\bigskipamount 21
\bottomfraction 37

c

\caption 35, 36
\caption* 35
\captionlabeldelim . . 37
center 28
\centering 28
\chapter 9
\chaptername 9
\cite 14
\cleardoublepage 24
\clearpage 24
\cline 32
\columnsep 15
\columnseprule 15
comment 6
\contentsname 11

[3] В скобках приведён символ, который печатает команда.

d

\date 8
\dblfloatpagefraction 37
\dblfloatsep 38
\dbltextfloatsep 38
\dbltopfraction 37
\DeleteShortVerb 29
\depth 30
description 26
document 4
\documentclass 4
\dotfill 20
\doublerulesep 33

e

\emph 40
\endfirsthead 35
\endfoot 35
\endhead 35
\endinput 4
\endlastfoot 35
\Eng 6
\English 6
\enlargethispage 24
\enlargethispage* ... 24
enumerate 25
\evensidemargin 7
\externaldocument ... 13
\extracolsep 32
\extrarowheight 33
\extratabsurround ... 33

f

\fbox 30
\fboxrule 30
\fboxsep 30
figure 36
figure* 36
\figurename 37
filecontents 7
filecontents* 7
\fill 20
\firsthline 33
floatingfigure 38
floatingtable 38
\floatpagefraction .. 37
\floatsep 38
\flushbottom 24
flushleft 28
flushright 28
\fnsymbol 19
\fontencoding 41
\fontfamily 41
\fontseries 41
\fontshape 41
\fontsize 41
\footnote 13
\footnotemark 13
\footnoterule 13
\footnotesep 13
\footnotesize 41
\footnotetext 13
\footskip 7
\framebox 30
\frenchspacing 20
\frontmatter 10
\fussy 22

g

\glossary 18

h

\headheight 7
\headsep 7
\height 30
\hfill 20
\hline 32
\hrulefill 20
\hsize 34
\hspace 20
\hspace* 20
\Huge 41
\huge 41
\hyphenation 23

i

\include 5
\includeonly 5
\indent 22
\index 16
\indexname 17
\indexspace 17
\input 4
\intextsep 38
\itdefault 40
\item 17, 25
\itemindent 26, 27
itemize 25
\itemsep 26, 27
\itshape 40

l

\label 12
\labelenumi 25
\labelenumii 25
\labelenumiii 25
\labelenumiv 25
\labelitemi 25
\labelitemii 25
\labelitemiii 25
\labelitemiv 25
\labelsep 26, 27
\labelwidth 26, 27
\LARGE 41
\Large 41
\large 41
\lasthline 33
\leftmargin 26, 27
\linebreak 23
\linespread 23
list 27
\listfigurename 11
\listfiles 5
\listoffigures 10
\listoftables 10
\listparindent 26, 27
\listtablename 11
longtable 35
lrbox 31
\LTcapwidth 36
\LTleft 35
\LTpost 35
\LTpre 35
\LTright 35

m

\mainmatter 10
\makebox 30
\makeglossary 18
\makeindex 16
\MakeShortVerb 29
\maketitle 8
\marginpar 39
\marginparpush 7
\marginparsep 7
\marginparwidth 7
\markboth 12
\markright 12
\mbox 30
\mddefault 40
\mdseries 40
\medskip 21
\medskipamount 21
minipage 31
multicols 15
\multicolsep 16
\multicolumn 32

n

\newblock 14
\newcolumntype 33
\newcommand 42
\newcommand* 42
\newcounter 19
\newenvironment 42
\newenvironment* 42
\newlength 19
\newpage 24
\newsavebox 30
\nofiles 4
\nofrenchspacing 20
\noindent 22
\nolinebreak 23
\nopagebreak 24
\normalfont 42
\normalmarginpar 39
\normalsize 41

o

\oddsidemargin 7
\onecolumn 15

p

\pagebreak 24
\pagenumbering 15
\pageref 12
\pagestyle 11
\par 21
\paragraph 10
\parbox 31
\parindent 22
\parsep 26, 27
\parskip 21
\part 9
\partname 9
\partopsep 26
\postmulticols 16
\premulticols 15
\printindex 17
\providecommand 42
\providecommand* 42

q

\qquad 20
\quad 20
quotation 28
quote 28

r

\raggedbottom 24
\raggedleft 28
\raggedright 28
\raisebox 30
\ref 12
\refname 14
\refstepcounter 19
\renewcommand 42
\renewcommand* 42
\renewenvironment ... 42
\renewenvironment* .. 42
\reversemarginpar ... 39
\rightmargin 26, 27
\rmdefault 39
\rmfamily 39
\Roman 19
\roman 19
\rule 31
\Rus 6
\Russian 6

s

\savebox 31
\sbox 31
\scdefault 40
\scriptsize 41
\scshape 40
\section 10
\see 17
\seealso 17
\seename 17
\selectfont 41
\setcounter 19
\setlength 19
\settodepth 30
\settoheight 30
\settowidth 30
\sfdefault 39
\sffamily 39
\sldefault 40
\sloppy 22
sloppypar 22
\slshape 40
\small 41
\smallskip 21
\smallskipamount 21
\stepcounter 19
\stretch 20
\subitem 17
\subparagraph 10
\subsection 10
\subsubitem 17
\subsubsection 10
\suppressfloats 36
\symbol 41

t

\tabcolsep 32
table 36
table* 36
\tablename 37
\tableofcontents 10
tabular 31
tabular* 34
\tabularnewline 32
tabularx 34
\tabularxcolumn 34
\textbf 40
\textcircled 44
\textcyrillic 6
\textfloatsep 38
\textfraction 37
\textheight 7
\textit 40
\textmd 40
\textnormal 42
\textrm 40
\textsc 40
\textsf 40
\textsl 40
\texttt 40
\textup 40
\textwidth 7
\thanks 8
thebibliography 14
\the*counter* 19
theindex 17
\thispagestyle 11
\tiny 41
\title 8
titlepage 8
\tolerance 21
\topfraction 37
\topmargin 7
\topsep 26
\totalheight 30
trivlist 27
\ttdefault 39
\ttfamily 39
\twocolumn 15

u

\updefault 40
\upshape 40
\usebox 31
\usecounter 27
\usefont 41
\usepackage 4

v

\value 18
\verb 29
\verb* 29
verbatim 29
verbatim* 29
\vfill 21
\vline 32
\vpageref 12
\vspace 21
\vspace* 21

w

\width 30

o

опции
- 10pt 5
- 11pt 5
- 12pt 5
- a4paper 5
- a5paper 5
- b5paper 5
- draft 5
- executivepaper 5
- final 5
- fleqn 5
- landscape 5
- legalpaper 5
- leqno 5
- letterpaper 5
- notitlepage 5
- onecolumn 5
- oneside 5
- openany 5
- openbib 5
- openright 5
- titlepage 5
- twocolumn 5

twoside 5

п

пакеты

alltt 29
array 33
babel 6
caption2 37
cite 14
fancyhdr 12
floatflt 38
fontenc 6
indentfirst 22
inputenc 6
layout 8
longtable 35
makeidx 16
multicol 15
shortvrb 29
tabularx 34
textcomp 45
varioref 12
verbatim 6, 29
xr 13

с

счётчики

bottomnumber 37
dbltopnumber 37
enumi 25
enumii 25
enumiii 25
enumiv 25
secnumdepth 9
tocdepth 11
topnumber 37
totalnumber 37
unbalance 16