МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ: «БРЕСТСКИЙ
ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «ЭВМ и системы»

МЕТОДИЧЕСКОЕ ПОСОБИЕ

«Использование системы верстки LaTeX
для оформления учебных работ»

для студентов специальностей
информатики и радиоэлектроники

Брест 2007

УДК 681.3

Методическое пособие содержат краткий теоретический курс по командам системы верстки LaTeX, включающий изложение базовых принципов подготовки документов, а также темы по типовой структуре документов, принципам набора текста, включению математических формул, рисунков и специальных элементов документа.

Методическое пособие предназначено для студентов специальностей информатики и радиоэлектроники и имеет целью снижение трудоемкости оформления учебных работ.

Список лит. **6** назв.

Составители: Костюк Д. А., к.т.н.;
Ильяшевич Д. А.

Рецензент:

Оглавление

1 Элементарное введение

1.1 Что такое TEX и LATEX

TEX — издательская система, созданная американским математиком и программистом Дональдом Кнутом (Donald E. Knuth). TEX был разработан преследуя две основные цели:

- позволить всем создавать качественные публикации с разумными для этого усилиями;
- предоставить инструмент, позволяющий создавать внешне идентичные публикации на всех компьютерах в настоящем и в будущем.

TEX знаменит своей чрезвычайной стабильностью, работой на различных типах компьютеров и практически полным отсутствием ошибок. Номер версии TEX стремится к π и сейчас равен 3,14159.

Во многих технических сообществах, особенно компьютерных, математических, физических и химических, TEX признан стандартом де факто. Огромное количество книг издаются с применением TEXa, включая книги издательств Addison-Wesley, Cambridge University Press, Elsever, Oxford University Press или Springer. Большое количество научных и научно-технических журналов издаются с помощью TEX, позволяя авторам публикаций присылать свои статьи прямо в формате документа TEX.

LATEX (в среде ASCII пишется LaTeX) — язык разметки и система подготовки документов, использующая в качестве механизма для верстки TEX. LATEX автоматизирует многие аспекты верстки документов, такие как автоматическая нумерация страниц, таблиц, иллюстраций, выключных формул, перекрестные ссылки, колонтитулы, предметный указатель, оглавление и список литературы. Система LATEX дополнительно содержит большой набор TEX-макросов, облегчающих создание сложных документов. Первая версия LATEX была написана в 1984 году Лесли Лампортом (Leslie Lamport) и с тех пор стала доминирующим способом подготовки TEX публикаций. Текущая версия системы — LATEX 2_ε (LaTeX2e).

1.2 Назначение LATEX в процессе верстки

Чтобы опубликоваться, авторы отдают свои рукописи в издательство. Один из дизайнеров издательства определяет макет публикации (размер страницы, ширину столбцов, шрифты, интервалы и т. п.). Основываясь на инструкциях дизайнера, верстальщик соответствующим образом оформляет публикацию.

Дизайнер-человек, пытается понять, что автор имел в виду, когда писал свою рукопись. Верстальщик же, в процессе верстки книги руководствуется не своими эстетическими предпочтениями, а жесткими правилами профессиональной полиграфии.

LaTeX берет на себя функции дизайнера книги, используя TeX в качестве верстальщика. Однако, поскольку LaTeX — всего лишь программа, от автора требуется предоставить больше информации о логической структуре документа: указать, что является заголовком, и т. п. Эта информация записывается с помощью команд LaTeX на специальном языке разметки.

Все это в корне отличается от WYSIWYG[1] подхода, применяемого в подавляющем большинстве современных текстовых процессоров, таких как *Microsoft Word* или *OpenOffice Writer*. При работе с такими пакетами автор документа может видеть на экране, как будет выглядеть документ, когда будет напечатан. При использовании LaTeX обычно невозможно увидеть итоговый результат в процессе написания текста. Однако его можно посмотреть после обработки файла LaTeX.

LaTeX позволяет неопытному автору сосредоточиться на содержании текста, а не его оформлении. Это существенно облегчает жизнь в случаях, когда оформление документа должно соответствовать не эстетическим воззрениям автора, а строгому набору правил, диктуемых ГОСТ или каким-либо другим стандартом. LaTeX предотвращает множество ошибок форматирования, так как заставляет автора объявлять логическую структуру документа, а затем берет на себя всю работу над оформлением.

1.3 Преимущества и недостатки

Все издательские системы на базе TeXа обладают достоинствами, заложенными в самом TeXе. Для новичка их можно описать одной фразой: «совсем как в книге». LaTeX как система предоставляет гибкие и удобные средства для достижения книжного качества публикации.

Огромным достоинством систем на базе TeXа является высокое качество и гибкость верстки математических формул. В этом отношении TeX до сих пор не превзойден.

Пользователю необходимо лишь знать несколько основных команд для определения логической структуры. Ему практически никогда не нужно возиться с макетом документа. Готовые, профессионально выполненные макеты, придающие работе действительно качественный вид, уже входят в пакет LaTeX.

[1] What you see is what you get (англ. — Что видишь, то и получишь)

Легко изготавливать даже такие сложные структуры, как указатели, оглавления, библиографии и т. п.

Для решения многих типографских задач, не поддерживаемых напрямую базовым LaTeX, есть свободно распространяемые дополнительные пакеты. LaTeX свободно доступен и работает практически на всех платформах.

Конечно у TeXa есть и недостатки. Первый из них продиктован распространенностью WYSIWYG-систем. Работа с исходным текстом и просмотр того, как текст будет выглядеть на печати — разные операции. Пользователям, привыкшим к WYSIWYG подходу, трудно поверить, что за счет этой особенности время на подготовку публикаций существенно сокращается.

Язык разметки документов, особенно для написания формул, может поначалу показаться достаточно сложным (перед лицом необходимости набрать сотню формул средствами WYSIWYG-процессора типа MS Word это мнение очень быстро меняется).

Хотя стандартные макеты имеют множество относительно легко переопределяемых параметров, достаточно сложно создавать новые макеты документа.

1.4 Работа с системой LaTeX

Для начала автор должен подготовить файл с текстом публикации, который оснащен командами для LaTeXa. По традиции такой файл должен иметь расширение .tex (так называемый TeX-файл).

После составления TeX-файла (или его части) можно с помощью программы-транслятора преобразовать его в формат dvi (**d**e**v**ice **i**ndependent (англ.) — не зависящий от устройства). Для исправления обнаруженных ошибок придется вернуться к редактированию исходного TeX-файла, затем снова преобразовать его в dvi и просмотреть. Окончательная доводка может потребовать многократного повторения этого шага.

Далее, dvi-документ можно преобразовать в формат PostScript или PDF (последний может быть распечатан или просмотрен с помощью программы Adobe Acrobat Reader).

1.5 Основные понятия

Абзацы разделяются между собой пустой строкой. Любое количество подряд идущих пустых строк эквивалентно одной. Любое количество подряд идущих пробелов или табуляций эквивалентно одному про-

белу. LaTeX игнорирует ваше оформление текста пробелами и переводами строк, поэтому вы можете не заботиться о внешнем виде TeX-файла. Однако, гораздо удобнее, если ваш файл будет удобно просматривать и редактировать. Вы можете использовать пробелы, пустые строки и табуляции на ваше усмотрение для удобства чтения и изменения вашего документа.

Команды используются в тех случаях, когда необходимо изменить оформление, вставить специальный символ, создать новый раздел и т. п. Команда начинается обратной косой чертой \, за которой следует имя команды. Имя команды может содержать несколько латинских букв (прописные и строчные различаются), либо один символ, не являющийся ни буквой, ни цифрой. Последовательности \% и \dots — это команды.

Команда может содержать обязательный аргумент (или несколько). Обязательный аргумент заключается в фигурные скобки ({}), следующий сразу за именем команды. В обязательный аргумент допустимо включать вложенные команды. Изменения внутри аргумента прекращают свое действие сразу же после закрывающей скобки. Иными словами, область действия команды с обязательным аргументом ограничена фигурными скобками. В LaTeXe некоторые команды могут иметь еще и необязательный аргументы. В отличие от обязательных, они заключаются в квадратные скобки ([]). Необязательные аргументы, если они указываются, должны идти перед перед обязательными, сразу же после названия команды.

Если команда не имеет обязательных аргументов, то сразу за ней должен идти символ, не являющийся латинской буквой. Обычно в таких случаях ставят пробел, который не будет выведен на печать. Чтобы напечатать пробел сразу после команды без аргументов необходимо добавить команду *обязательного пробела* (пробел после обратной косой черты), либо добавить пустой обязательный аргумент ({}).

Формулы в TeX делятся на два вида: внутри текста и *выключные*, т. е. вынесенные в отдельную строку. Внутритекстовые формулы TeX автоматически старается ужать, чтобы они занимали как можно меньше места по вертикали. Внутритекстовые формулы окружаются с обеих сторон знаками $. Выключные формулы окружаются знаками $$ или парой команд \[и \]. Формулы, заключенные в $$ всегда центрируются по горизонтали.

Окружение — это фрагмент текста, заключенный между командами \begin{*env*} и \end{*env*}, где *env* — имя окружения. Окружение указывает, что данному фрагменту текста следует применить некоторый специальный тип оформления. Например, окружение equation — автоматически нумеруемая выключная формула, itemize — маркированный перечень, и т. д.

Таблица 1.1 — Единицы длины

pt	пункт ≈ 0.35 миллиметра
pc	пика $= 12$ pt
mm	миллиметр
cm	сантиметр $= 10$ mm
in	дюйм $= 25{,}4$ mm
dd	пункт Дидо $\approx 1{,}07$ pt
cc	цицеро $= 12$ dd

Разумное форматирование исходного текста делает его более удобным для чтения и упрощает процесс редактирования. Хотя вопросы удобства индивидуальны и зависят от используемых текстовых редакторов, рискнем привести некоторые советы по форматированию:

- Избегайте слишком длинных строк. Если позволяет ваш текстовый редактор, установите автоматический перенос строк после 72–80 символов. Не во всех редакторах удобно просматривать и редактировать длинные строки.
- Команды \begin, \end, \section и его аналоги, \item, \newpage набираются отдельной строкой.
- Внутритекстовые формулы, за исключением самых коротких, набираются отдельной строкой.

2 Базовая структура документов LaTeX

2.1 Структура исходного текста

Когда LaTeX обрабатывает входной файл, он ожидает следования определенной структуре документа. Так, каждый входной файл должен начинаться с команды, определяющей тип документа. Тип документа задается командой \documentclass[*опции*]{*класс*}.

Здесь *класс* определяет тип создаваемого документа. Таблица 2.1 перечисляет основные классы. В состав LaTeX 2_ε входят дополнительные классы для других видов документов, включая письма и слайды. Также вы можете создавать и свои классы.

Таблица 2.1 — Классы документов

Класс	Назначение
article	Для статей в научных журналах, презентаций, коротких отчетов, программной документации и т. п.
report	Для более длинных отчетов, небольших книжек, презентаций
book	Для настоящих книг

Параметр *опции* изменяет поведение класса документа. Опции должны разделяться запятыми. В таблице 2.2 перечислены самые употребительные опции стандартных классов документов.

Например, если в начале TeX-файла стоит, \documentclass[11pt,twoside,a4paper]{article}, это заставляет LaTeX оформлять документ как *статью*, с базовым размером шрифта в *одиннадцать пунктов* и форматировать документ для *двусторонней* печати на бумаге *формата A4*.

В процессе написания вашего документа, вы, вероятно, обнаружите, что в некоторых областях базовый LaTeX не может решить ваши задачи. Если вы захотите включить в документ графику, цветной текст или исходный код программы из внешнего файла, вам нужно будет расширить возможности LaTeX. Такие расширения называются *пакетами* Пакеты активизируются командой \usepackage[*опции*]{*пакет*}, где *пакет* — это имя пакета, а *опции* — список ключевых слов, активирующих специальные свойства пакета. Некоторые пакеты уже включены в состав LaTeX 2_ε (см. таблицу 2.3). Другие предоставляются отдельно.

Итак, в начале TeX-файла указывается его тип и подключаются дополнительные пакеты. Затем следует сам документ, заключенный в окружение \begin{document} . . . \end{document}. Все, что находится перед этим окружением называется *преамбулой* документа. Кроме этого, в преамбуле документа могут указываться команды, имеющие действие на весь документ и команды-настройки для подключенных пакетов.

Таблица 2.2 — Опции классов документов

Опция	*Назначение*
10pt, 11pt 12pt	Устанавливает размер *основного* шрифта документа. Если ни одна из этих опций не указана, подразумевается 10pt.
a4paper, letterpaper, . . .	Определяет размер листа. По умолчанию подразумевается letterpaper. Так же могут быть указаны a5paper, b5paper, executivepaper и legalpaper.
twocolumn	Заставляет LaTeX набирать документ в две колонки.
twoside, onseside	Выбирает одно- или двусторонний вывод. По умолчанию классы article и report используют односторонний вывод, класс book — двусторонний вывод.
openright, openany	Делает главы начинающимися или только на правой странице, или на первой доступной. Это не работает с классом article, так как он ничего не знает о главах. Класс report по умолчанию начинает главы на следующей странице, а класс book — на правой.

Например, в преамбуле можно задать стиль страницы. LaTeX поддерживает три предопределенных комбинации верхнего и нижнего колонтитула — так называемые стили страницы. Параметр *стиль* команды \pagestyle{*стиль*} определяет, какой из них использовать. Предопределенные стили страницы перечислены в таблице 2.4.

Пример каркаса типичной статьи, оформленной в LaTeX, приведен на рисунке 2.1.

```
\documentclass[12pt,a4paper]{article}
\usepackage[cp1251]{inputenc}
\usepackage[T2A]{fontenc}
\usepackage[english,russian]{babel}
\frenchspacing
\begin{document}
Вот тут начинается моя замечательная статья.

\ldots{} а тут она кончается.
\end{document}
```

Рисунок 2.1 — Пример каркаса типичной статьи

Таблица 2.3 — Некоторые из распространяемых с LaTeX пакетов

Пакет	*Назначение*
fontenc	Указывает, какую кодировку шрифта должен использовать LaTeX
syntonly	Обрабатывает документ, не печатая его. Это удобно для быстрой проверки на ошибки
inputenc	Позволяет указать входную кодировку документа. Например, cp1251 (принята в ОС Windows), koi8-r (принята в Unix-системах), cp866 (принята в ОС DOS)
indentfirst	Пакет абзацных отступов первой строки

Таблица 2.4 — Предопределенные стили страницы LaTeX

Стиль	*Назначение*
plain	Печатает номера страниц внизу страницы в середине нижнего колонтитула. Этот стиль установлен по умолчанию
headings	Печатает название текущей главы и номер страницы в верхнем колонтитуле каждой страницы, а нижний колонтитул остается пустым
empty	Делает и верхние, и нижние колонтитулы пустыми

2.2 Типы файлов, генерируемые LaTeX

Начав работать с LaTeX вы вскоре обнаружите большое количество создаваемых LaTeX файлов с различным расширением. В таблице 2.5 перечислены основные типы файлов, используемые при работе с TeX.

2.3 Разбиение исходного файла на части

При работе с большими документами намного удобнее разделить входной файл на несколько частей. LaTeX содержит две команды, которые позволяют это сделать. \include{*файл*} — эту команду можно использовать в теле документа, чтобы включить в него содержимое какого-либо файла. Заметьте, что указывать во вставляемом файле преамбулу и окружение *document* не нужно. LaTeX добавит текст из включаемого файла, начав новую страницу.

В преамбуле документа можно указать какие из включаемых файлов добавлять в документ, командой \includeonly{*filename, filename, . . .*}. Команды \include с файлами, не указанными в \includeonly будут проигнорированы.

Таблица 2.5 — Типы файлов, используемые при работе с TEX

Тип файла	*Назначение*
.dvi	Device Independent file (англ. — файл, не зависящий от устройства). Это — основной результат запуска LATEX
.log	Содержит детальный отчет о том, что происходило в последний прогон компиляции
.toc	Хранит заголовки всех разделов. Читается в следующий проход компиляции и используется для создания оглавления
.aux	Еще один файл, передающий информацию при повторном проходе компиляции. Кроме всего прочего, используется для создания перекрестных ссылок

В случае, если вы не хотите, чтобы LATEX начинал новую страницу при вставке файла, вместо \include используйте команду \input{*файл*}. Она просто включает содержимое указанного файла. Действие команды \includeonly на команду \input не распространяется.

3 Набор текста

3.1 Поддержка кириллицы

Если планируется создавать документы не на английском языке, то LaTeX должен быть сконфигурирован соответствующим образом:

1) Все генерируемые автоматически текстовые строки[1] должны быть переведены на другой язык. Также должны быть использованы правила переноса для соответствующего языка. Для многих языков эти изменения достигаются использованием пакета babel (автор Johannes Braams).
2) Должны соблюдаться специфические для языка типографские правила.

Если ваша система LaTeX уже соответствующим образом сконфигурирована, вы можете активизировать пакет babel командой \usepackage [*язык*] {babel} после команды \documentclass. Если форматный файл LaTeX не содержит правил переноса для выбранного вами языка, babel будет работать, но запретит переносы, что негативно скажется на качестве выводимого документа.

Для правильного вывода текста, набранного кириллицей, необходимо указать кодировку входного файла командой \usepackage[*кодировка*]{inputenc}. Допустимые кодировки для русского языка указаны в таблице 2.3.

Если вы планируете создание многоязычного документа, воспользуйтесь пакетом ucs и многобайтовой кодировкой utf8, где каждый символ кодируют последовательностью от одного до четырех байт (см. рисунок 3.1).

```
\usepackage{ucs}
\usepackage[utf8]{inputenc}
```

Рисунок 3.1 — Включение режима многоязыкового документа

Иной случай — с кодировкой шрифта. Множество входных кодировок можно отобразить на одну кодировку шрифта, что уменьшает число необходимых наборов шрифтов. Кодировки шрифтов обрабатываются пакетом fontent: \usepackage[*кодировка*]{fontenc}, где *кодировка* — требуемая кодировка шрифта. Можно одновременно загружать несколько кодировок.

Если вам потребуется использовать кириллицу в математическом режиме, загрузите до пакета fontenc пакет mathtext

[1] Содержание, Глава, Список иллюстраций, ...

В преамбулу желательно добавить команду \frenchspacing. Дело в том, что в западных типографских правилах принято добавлять бóльший пробел в конце предложения (после точки, восклицательного или вопросительного знака). Команда \frenchspacing отключает этот режим в TEX.

3.2 Правила набора текста

В системе LATEX эти и только эти символы отображаются очевидным образом:

```
A B C ... X Y Z
a b c ... x y z
А Б В ... Э Ю Я
а б в ... э ю я
0 1 2 3 4 5 6 7 8 9 , . ; : @ |
? ! ( ) [ ] < > - + * = / ’ ’ "
```

```
A B C ... X Y Z
a b c ... x y z
А Б В ... Э Ю Я
а б в ... э ю я
0 1 2 3 4 5 6 7 8 9 , . ; : @ |
? ! ( ) [ ] < > – + ∗ = / ’ ’ "
```

Символы, которые можно напечатать, подставив перед ними \:

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

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

Символы, которые нельзя обычным способом отобразить в тексте:

```
\ ~ ^
```

```
\verb"\" \verb"~" \verb"^"
```

Если символ % употреблен в тексте не в составе комбинации \%, то он является «символом комментария»: все символы, расположенные на строке после него, TEX игнорирует. С помощью символа % в исходный текст можно вносить пометки «для себя»:

Это поучительный пример.
Жил-был у бабушки серенький козлик.

```
Это % глупый
% Лучше:
поучительный пример.

Жил–был у бабушки
сере%
    нький козлик.
```

Скобки всех видов набираются вплотную к тексту, который они окружают:

Верно: (текст) [текст] {текст} ⟨текст⟩
Неверно: (текст) [текст]

```
Верно: (текст) [текст] \{текст\}
  $\langle$текст$\rangle$
Неверно: ( текст ) [ текст ]
```

Знаки препинания набираются слитно с предшествующим текстом и отдельно от последующего:

Верно: тт. тт, тт; тт: тт! тт? тт% тт...
Неверно: ноль , один ,два .

```
Верно: тт. тт, тт; тт: тт! тт?
  тт\% mm\dots
Неверно: ноль , один ,два .
```

Дефис, длинное тире (em-dash), короткое тире (en-dash) и минус — это совершенно разные знаки:

дефисы в словах: δ-функция
диапазоны чисел: страницы 3–7
тире в предложениях: Это — тире.
минусы в формулах: $-f(-x) = f(x)$

```
дефисы в~словах: $\delta$-функция
диапазоны чисел: страницы~3--7
тире в~предложениях: Это~--- тире.
минусы в~формулах: $-f(-x)=f(x)$
```

Знаки № и § набираются слитно с последующим текстом:

Верно: №12 №№12–14 §12 §§12–14
Неверно: № 12 § 12

```
Верно: \No12 \No\No12--14 \S12
  \S\S12--14
Неверно: \No~12 \S~12
```

В отличие от пишущей машинки, книжный набор использует различные знаки для открывающей и закрывающей кавычек (вместо нейтрального знака "). В английских текстах открывающая кавычка изображается во входном тексте двумя подряд идущими обратными апострофами, закрывающая — двумя апострофами.

В русских текстах употребляются кавычки типа «елочки» и „лапки“. Если в тексте встречаются кавычки внутри кавычек, то, согласно типографским правилам, внутренние кавычки должны отличаться от внешних: в английских текстах снаружи ставятся двойные кавычки, задаваемые как `` и '', а внутри одинарные, задаваемые как ` и '; в русских текстах можно, например, снаружи поставить «елочки», а внутри „лапки“. Если при этом наружная и внутренняя кавычка соседствуют, их надлежит разделить дополнительным небольшим пробелом. В LaTeX'е для этой цели служит команда \,. В русификации LaTeX'а PSCyr, использованной при наборе этого пособия, «елочки» задаются командами << и >>, а „лапки“ — командами „ и “):

«„Карова“ или „корова“, как правильно?» — спросил он.

```
<<\,,,Карова'' или ,,корова'',
как правильно?>>~--- спросил он.
```

Для многоточия есть специальная команда \ldots или \dots.

Вместо «...» пишем:
Нет, что-то здесь не так...

```
Вместо <<...>> пишем:
Нет, что-то здесь не так\ldots
```

3.3 Диакритические знаки

Во многих языках используются буквы с дополнительными значками, размещающимися над или под буквой (они называются диакритическими знаками). Кроме того в ряде языков, использующих латинский

алфавит, есть специальные дополнительные буквы. В TEXe имеются команды для набора букв с диакритическими знаками из почти всех европейских языков. Команды для получения диакритических знаков собраны в таблице 3.1, где знаки проставлены, для примера, при букве «e».

Таблица 3.1 — Диакритические знаки

Набрано	*Вышло*
\‘e	è
\’e	é
\^e	ê
\~e	ẽ
\=e	ē

Набрано	*Вышло*
\.e	ė
\u{e}	ĕ
\v{e}	ě
\H{e}	ő
\"e	ë

Набрано	*Вышло*
\c{e}	ȩ
\d{e}	ẹ
\b{e}	e̲
\t oo	o͡o

Особые символы и буквы, употребляемые в текстовом режиме, представлены в таблице 3.2.

Таблица 3.2 — Буквы специального вида

Набрано	*Вышло*
\dag	†
\ddag	‡
\O	Ø
\OE	Œ
\AE	Æ
\AA	Å

Набрано	*Вышло*
\S	§
\P	¶
\o	ø
\oe	œ
\ae	æ
\aa	å

Набрано	*Вышло*
\No	№
\copyright	©
\i	ı
\j	
\ss	ß

Обратите внимание на команды \i и \j в этой таблице: они нужны для того, чтобы ставить диакритические знаки над буквами i и j. Если просто сказать, допустим, \=i, то получится ī, а это не то, что требуется. Правильно писать ī. Вот несколько примеров.

```
Oui, c'est peut-\^etre \c{c}a. \\
\"Ежик под \"елкой.
```

Oui, c’est peut-être ça.
Ёжик под ёлкой.

Над буквами в математических формулах также часто приходится ставить надстрочные знаки, но описанные в настоящем разделе команды для этого непригодны; команды, делающие это в формулах, приведены в таблице 4.11 на странице 42.

3.4 Промежутки между словами

Как и на пишущей машинке, в LATEX пробел остается пробелом — небольшим горизонтальным отступом. Только следует учитывать правила русской профессиональной полиграфии: при переносах на другую строку предлоги не должны отрываться от следующего слова, а тире

не отрывается от предыдущего слова. Для этого используется жесткий пробел ~.

Слова, состоящие из одной или двух букв, обычно желательно переносить на новую строку вместе с последующими словами (кроме случая, когда такое слово стоит в конце строки: здесь лучше переносить на новую строку два последних слова). Список коротких слов: я, ты, мы, вы, не, ни, на, но, в, во, до, от, и, а, ее, он, с, со, о, об, ну, к, ко, за, их, из, ей, ой, ай. После таких слов в тексте желательно заменять обычный пробел жёстким.

Следует отличать предлоги, местоимения и некоторые другие слова от частиц типа «ли» (последние желательно оставлять на одной строке с предшествующим словом или переносить их оба). Жесткий пробел желательно ставить *непосредственно перед* ними: ли, ль, же, ж, бы, б.

В правилах русской полиграфии запрещено отрывать инициалы от фамилии и переносить их на следующую строку. К тому же, между инициалами и фамилией желательно ставить не обычный жесткий пробел, а укороченный в два раза. Такой пробел задается командой \,.

Верно: И. И. Иванов, и т. д., т. е., и др.
Лучше: И. И. Иванов, и т. д., т. е.
Неверно: И.И. Иванов, И.И.Иванов

```
Верно: И.~И.~Иванов, и~т.~д., т.~е.,
  и~др.
Лучше: И.\,И.\,Иванов, и~т.\,д.,
  т.\,е.
Неверно: И.И.~Иванов, И.И.Иванов
```

3.5 Переключение шрифтов

Начнем с предупреждения, которое не требуется профессиональным полиграфистам, но не повредит остальным: Не увлекайтесь переключением шрифтов! Чем *меньше* различных видов шрифта использовано в тексте, тем легче его читать и тем красивее он выглядит.

LaTeX выбирает подходящее начертание и размер шрифта, основываясь на логической структуре документа (разделы, сноски, . . .). Иногда требуется сменить шрифт вручную, можно воспользоваться командами, перечисленными в таблицах 3.3 и 3.4. Действительный размер каждого шрифта определяется дизайном и зависит от класса и опций документа. Таблица 3.5 показывает абсолютные размеры, соответствующие этим командам в стандартных классах документов.

Шрифты гарнитуры \texttt обладают специфическими свойствами: все символы в них имеют одинаковую ширину, промежутки между словами жестко фиксированы и не обладают растяжимостью. В словах, набранных этими шрифтами, не делается автоматических переносов при верстке абзацев. Применяются эти шрифты для изображения исходных

Таблица 3.3 — Шрифты

Команда	*Шрифт*
\textrm{...}	Обычный шрифт (roman)
\texttt{...}	Имитация пишущей машинки (typewriter)
\textsf{...}	Рубленый шрифт (sans serif)
\textbf{...}	**Полужирный шрифт (boldface)**
\textit{...}	*Курсив (italic)*
\textsl{...}	*Наклонный шрифт (slanted)*
\textsc{...}	Капитель (Small Caps)

текстов компьютерных программ, фрагментов TEXовских исходных текстов и т. п.

Таблица 3.4 — Размеры шрифта

Команда	*Название размера*
\tiny	Крошечный
\scriptsize	Очень маленький
\footnotesize	Довольно маленький
\small	Маленький
\normalsize	Нормальный
\large	Большой
\Large	Еще больше
\LARGE	Очень большой
\huge	Огромный
\Huge	Громадный

Команды, меняющие размер, одновременно устанавливают гарнитуру roman («обычный» шрифт). Поэтому полужирный большой шрифт требует не команд \bf\large, а команд \large\bf.

В связи с командами смены размера шрифта заметную роль играют фигурные скобки. Они используются для построения *групп*. Группы ограничивают область действия большинства команд LATEX.

Маленький, **полужирный**, большой, *курсив*.

```
{\small Маленький,
  {\bf полужирный},
  {\Large большой},
  {\it курсив}.}
```

Если вы хотите применить команду изменения размера к одноми или нескольким абзацам текста, для этого лучше использовать синтаксис окружения.

Таблица 3.5 — Абсолютные размеры шрифтов в стандартных классах

Размер	*10pt (по умолчанию)*	*опция 11pt*	*опция 12pt*
\tiny	5pt	6pt	6pt
\scriptsize	7pt	8pt	8pt
\footnotesize	8pt	9pt	10pt
\small	9pt	10pt	11pt
\normalsize	10pt	11pt	12pt
\large	12pt	12pt	14pt
\Large	14pt	14pt	17pt
\LARGE	17pt	17pt	20pt
\huge	20pt	20pt	25pt
\Huge	25pt	25pt	25pt

Длинный текст большими буквами.

```
\begin{Large}
Длинный текст большими буквами.
\end{Large}
```

Это избавит вас от подсчета множества фигурных скобок.

3.5.1 Опасность

Имейте в виду, что усеивать документы явными командами, вроде только что описанных, опасно, потому что это противоречит основной идее LaTeX: разделению логической и визуальной разметки. Это значит, что, если вы пользуетесь одними и теми же командами смены шрифта в разных местах для верстки специального вида информации, вы должны использовать \newcommand (см. раздел 5.6 на странице 53) и определить команду, «оборачивающую» в себя команду смены шрифта.

Не **входите** в эту комнату. Она занята **машиной** неизвестного назначения.

```
% в преамбуле или пакете
\newcommand{\danger}[1]{\textbf{#1}}
% в документе
Не \danger{входите} в эту комнату.
Она занята \danger{машиной}
неизвестного назначения.
```

Этот подход имеет то преимущество, что если вы позже решите использовать другое визуальное представление опасности, нежели \textbf, не понадобится пробираться через весь документ, отыскивая все вхождения \textbf и определяя, отмечает ли каждое из них опасность или что-нибудь другое.

3.5.2 PSCyr

PSCyr — коллекция русифицированных PostScript-шрифтов в формате Type1. В настоящее время получили распространение два пакета, позволяющие использовать векторные (Type1) варианты базовых TEX'овских шрифтов (Computer Modern) для набора русских текстов: cmcyr и wncyr. Работа с пакетом PSCyr (версии 0.4d) позволяет получить доступ ещё к *пятнадцати* семействам шрифтовых гарнитур в формате Type1. Возможные варианты представлены в таблицах 3.6–3.9.

Таблица 3.6 — Шрифты с засечками

Команда	*Шрифт, начертание*
\textac	Академическая (AcademyPSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*
\textaq	Антиква (AntiquaPSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*, ***Полужирное Курсивное***
\textha	Балтика (HandbookPSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*
\textco	Бодони (CollegePSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*
\textjn	Журнальная (JournalPSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*
\textlz	Лазурская (Lazurski), начертание Прямое нормальное
\textsv	Сувенир (SouvenirPSCyr), начертания Прямое нормальное, **Полужирное**
\texttm	Таймс (TimesNewRomanPSMT), начертания Прямое нормальное, **Полужирное**, *Курсивное*, ***Полужирное Курсивное***

При необходимости набрать часть текста другим шрифтом, соответствующий блок текста помещается в аргумент команды переключения шрифта: например, блок текста внутри команды \textac{...} будет набран шрифтом Academy. Если же вы хотите набирать весь текст другим шрифтом, то разумнее будет сразу же в преамбуле документа установить соответствующий шрифт по умолчанию. Например, команда \renewcommand{\rmdefault}{ftm} установит в качестве шрифта с засечками шрифт Times. Внутренние имена всех доступных шрифтов коллекции указаны в файле pscyr.sty; они кодируются тремя буквами, первой из которых является буква f (что значит free), а две последующие буквы получаются из сокращенного названия шрифта.

Таблица 3.8 — Рубленые шрифты (без засечек)

Команда	*Шрифт, начертание*
\textar	Ариал (ArialMT), начертания Прямое нормальное, **Полужирное**, *Курсивное*, ***Полужирное Курсивное***, **Жирное**
\texttx	Букварная (TextbookPSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*
\textma	Журнальная Рубленая (MagazinePSCyr), начертания Прямое нормальное, **Полужирное**, *Курсивное*

Пакет позволяет пользоваться стандартными для LaTeX 2$_\varepsilon$ командами смены шрифтов (\tiny ... \Huge) и смены начертания (\textbf ... \textrm).

Примеры: КАПИТЕЛЬ АНТИКВА, *наклонный и* **полужирный Таймс**, *большой наклонный Ариал*, малюсенькая Журнальная рубленая, **маленькая полужирная Букварная**.

3.6 Сноски

Чтобы сделать сноску к какому-то месту в тексте, достаточно использовать команду \footnote с одним обязательным аргументом — текстом сноски. В стандартных стилях LaTeXа сноски[2] нумеруются подряд на протяжении всей главы или даже (в стиле article) всего документа. Исходный текст предыдущего фрагмента представлен на рисунке 3.2.

```
...сноски\footnote{Вроде этой} нумеруются...
```

Рисунок 3.2 — Пример применения сноски

Если после слова, к которому делается сноска, должен стоять знак препинания, то в исходном тексте его надо поставить *после* закрывающей фигурной скобки, ограничивающей аргумент команды \footnote.

[2]Вроде этой

Таблица 3.9 — Моноширные шрифты

Команда	*Шрифт, начертание*
\textcr	Курьер (CourierNewPSMT), начертания Прямое нормальное, **Полужирное,** *Курсивное,* ***Полужирное Курсивное***
\texter	ER Курьер (ERKurierPSCyr), начертания Прямое нормальное, **Полужирное,** *Курсивное,* ***Полужирное Курсивное***

Если сноску нужно присоединить к тексту в таблице, названию раздела и т. п., следует использовать формат задания сноски, представленный на рисунке 3.3.

```
...сноски\footnotemark{}\footnotetext{Вроде
этой} нумеруются...
```

Рисунок 3.3 — Пример альтернативного задания сноски

3.7 Абзацы

Чтобы TEX сверстал абзац, никаких специальных усилий прилагать не нужно: достаточно оставить в исходном тексте пустую строку, указывающую TEXу на конец абзаца, или в конце абзаца вставить команду \par.

Обычно абзацы делаются выровненными по ширине; при необходимости промежутки между словами растягиваются или сжимаются, а в словах делаются переносы. TEX выбирает из всех вариантов разбиения текста абзаца на строки оптимальный; при этом и для сжатия, и для растяжения промежутков между словами есть пределы, которые TEX старается не превышать. В случаях, когда необходимы «необычные» абзацы, используются специальные команды изменения вида абзаца, которые мы опишем в этом разделе.

3.7.1 Подавление абзацного отступа

Иногда возникает необходимость создать абзац, в котором нет абзацного отступа. Для этой цели удобно воспользоваться командой \noindent. В том абзаце, отступ в котором вы хотите подавить, эта команда должна идти первой (до любого текста):

В этом абзаце отступа не будет.

В этом абзаце отступ будет присутствовать.

```
\noindent В этом абзаце
отступа не будет.

В этом абзаце отступ
будет \noindent
присутствовать.
```

Команда \noindent действует только на тот абзац, который с нее начинается; если ее поместить внутри абзаца, то вообще ничего не произойдет (что и иллюстрирует второй из абзацев в нашем примере). Между \noindent и абзацем, к которому она относится, не должно быть пустой строки (иначе получится, что \noindent относится к «пустому абзацу», заканчивающемуся этой пустой строкой).

В большинстве случаев, когда разумно сделать абзац без отступа, LaTeX заботится об этом сам, так что вам не придется пользоваться командой \noindent чересчур часто.

3.7.2 Вертикальные промежутки

Большинство вертикальных промежутков (например, между заголовком раздела и его текстом) LaTeX устанавливает самостоятельно. Иногда возникает необходимость сделать дополнительный вертикальный промежуток между абзацами. Подобно тому, как внутри абзацев для задания промежутков вручную разумнее пользоваться не командами, явно задающими размер промежутка, а командами вроде \, или \quad, так и для задания промежутков между абзацами в первую очередь полезны такие команды:

- \smallskip задает такой промежуток;
- \medskip задает такой промежуток;
- \bigskip задает такой промежуток.

Проще всего поставить эти команды непосредственно после пустой строки или команды \par, завершающей абзац:

После этого абзаца мы оставим дополнительный пробел.

А теперь начнем новый абзац.

```
После этого абзаца мы оставим
дополнительный пробел.\par\smallskip
А теперь начнем новый абзац.
```

Конкретная величина промежутков, задаваемых этими командами, зависит от класса документа.

Если вы хотите задать размер вертикального промежутка в явном виде, можно воспользоваться командой \vspace. У нее есть один обязательный аргумент — величина промежутка. Например, можно написать \vspace{2ex}.

Команду \vspace удобнее всего ставить после конца абзаца (подобно таким командам, как \smallskip).

Можно поставить команду \vspace (или \smallskip и т. п.) не после пустой строки или \par, а непосредственно перед ними, после всего текста абзаца. Если поставить какую-либо из этих команд внутри абзаца, то дополнительный вертикальный пробел получится не между абзацами, а между строками абзаца.

3.7.3 Принудительный разрыв страницы

Для принудительного разрыва страниц в LaTeXe существует несколько способов. Первый и самый простой — команда \newpage. Под действием этой команды текущая страница завершается и дополняется снизу пустым пространством, если высота страницы получается меньше, чем надо.

Команда clearpage также предназначена для принудительного разрыва страницы. Если пользоваться только теми средствами LaTeXa, которые были описаны до этого момента, то она будет работать в точности так же, как \newpage. В том же случае, если к моменту подачи этой команды остались так называемые «плавающие» иллюстрации или таблицы (см. раздел 5.4 на странице 50), то перед выдачей новой страницы они будут напечатаны.

Команда \cleardoublepage делает то же, что и \clearpage, но при этом в некоторых классах документов (в тех, которые предусматривают разные поля для страниц с четным и нечетным номером — см. таблицу 2.2 на странице 11 по поводу опции twoside класса документа) новая страница обязательно имеет нечетный номер (если необходимо, при этом создается дополнительная пустая страница).

Если поставить подряд две команды \newpage (или \clearpage), то в печатном тексте чистая страница не получится. Чтобы создать чистую страницу, надо LaTeX немного обмануть: между двумя командами для разрыва страницы добавить, например, жёсткий пробел.

3.7.4 Центрирование и прижатие текста к краю

Для этих целей используются окружения center для центрирования, а также flushleft и flushright для выравнивания по левому и правому краю соответственно.

Внутри каждого из этих окружений можно в принципе набирать и самый обычный текст, стандартным образом разбитый на абзацы с помощью пустых строк, но при этом каждая строка получающегося абзаца будет центрирована (для окружения center) или выровнена по левому или правому краю (для flushleft и flushright соответственно).

```
левый
марш
```

```
\begin{flushleft}
левый\\
марш
\end{flushleft}
```

```
наше дело
правое
```

```
\begin{flushright}
наше дело\\
правое
\end{flushright}
```

```
а вот мы позиционируемся
в центристской части
политического спектра
```

```
\begin{center}
а вот мы позиционируемся\\
в центристской части\\
политического спектра
\end{center}
```

3.7.5 Перечни

Для печати перечней используются окружения itemize (для простейших перечней), enumerate (для нумерованных перечней) и description (для перечней, в которых каждый пункт имеет заголовок — например, словарных статей или иных описаний). В любом случае элементы перечня вводятся командой \item (иногда с необязательным аргументом). Разберем последовательно, как работают указанные окружения.

Простейшие перечни (itemize). Каждый элемент перечня вводится командой \item без аргумента.

- Перечни могут быть вложенными друг в друга:
  - Максимальная глубина вложенности равна 4.
  - Отступы и символы перед элементами выбираются автоматически.
- При попытке вложить пять таких окружений LaTeX выдаст сообщение об ошибке.

То, каким предшествующий текст был в исходном файле, показано на рисунке 3.4.

Внутри окружения itemize до первой команды item не должно идти никакого текста или команд, генерирующих текст. Если вы попытаетесь проигнорировать этот запрет, LaTeX выдаст сообщение об ошибке. Другие команды (например, команды смены шрифта) могут идти и до первого item.

```
\begin{itemize}
\item
Перечни могут быть вложенными друг в~друга:
  \begin{itemize}
  \item
  Максимальная глубина вложенности равна~4.

  \item
  Отступы и~символы перед элементами выбираются
  автоматически.
  \end{itemize}

\item При попытке вложить пять таких окружений \LaTeX{}
выдаст сообщение об~ошибке.
\end{itemize}
```

Рисунок 3.4 — Пример оформления простого перечня с вложением

Окружение itemize можно использовать также для создания перечней, в которых каждый элемент имеет короткий заголовок. Для создания такого заголовка надо задать команде \item необязательный аргумент. При наличии у этой команды необязательного аргумента стандартный значок, отмечающий элемент перечня, не печатается, а вместо него печатается текст, заданный в необязательном аргументе:

- Этот элемент перечня помечен стандартно.

Раз Здесь мы сами задали заголовок.

Два Здесь тоже.

```
\begin{itemize}
\item
Этот элемент перечня помечен
стандартно.

\item[\sf Раз]
Здесь мы~сами задали заголовок.

\item[Два]
Здесь тоже.
\end{itemize}
```

Обратите внимание, что заголовки, заданные нами в необязательных аргументах команд \item, печатаются выровненными по правому краю, а команды смены шрифта в этих аргументах не распространяются на дальнейший текст.

Если заголовок, заданный в необязательном аргументе команды \item, будет слишком длинен, то он наложится на левое поле. В таких случаях лучше пользоваться окружением description, о котором речь пойдет ниже.

Нумерованные перечни (enumerate). В таких перечнях каждый элемент также вводится командой \item без аргумента, но на печати он будет отмечен не значком, а номером (эти номера создаются LaTeXом автоматически; если вы переставите какие-то элементы перечня, что-то

добавите или удалите, нумерация автоматически изменится).

1) В окружении enumerate элементы списка нумеруются цифрами или буквами.
2) Нумерация производится автоматически.
3) Перечни могут быть вложенными друг в друга:
   а) Максимальная глубина вложенности равна 4.
   б) Отступы и обозначения для элементов выбираются автоматически.
4) При попытке вложить пять таких окружений LaTeX выдаст сообщение об ошибке.

То, каким предшествующий текст был в исходном файле, показано на рисунке 3.5.

```
\begin{enumerate}
\item
В~окружении \comm{enumerate} элементы списка нумеруются
цифрами или буквами.

\item
Нумерация производится автоматически.

\item
Перечни могут быть вложенными друг в~друга:

  \begin{enumerate}
  \item
  Максимальная глубина вложенности равна~4.

  \item
  Отступы и~обозначения для элементов выбираются
  автоматически.
  \end{enumerate}

\item При попытке вложить пять таких окружений \LaTeX{}
выдаст сообщение об~ошибке.
\end{enumerate}
```

Рисунок 3.5 — Пример оформления нумерованного перечня

Внутри окружения enumerate до первой команды \item не должно идти никакого текста или команд, генерирующих текст.

В окружении enumerate команда \item может иметь необязательный аргумент, который работает так же, как в окружении itemize.

Перечни с заголовками (description). В этих перечнях каждый элемент, как уже было сказано, снабжен заголовком. Поэтому элементы перечня вводятся командой \item с необязательным аргументом, представляющим собой этот заголовок.

Заголовки элементов перечня оформляются в окружении description полужирным шрифтом. Если вас не устраивает этот шрифт, можно аргумент команды \item начать с команды переключения шрифта, скажем, \rm или \sl.

Внутри окружения description до первой команды \item не должно идти никакого текста или команд, генерирующих текст.

Летом можно собирать разные ягоды:

черника: темно-синие, очень вкусные, хороши в свежем виде, варенье тоже получается хорошее;

голубика: синие, более водянистые, чем черника, и не такие вкусные;

брусника: ярко-красные, из них получается очень вкусное варенье.

```
Летом можно собирать разные ягоды:

\begin{description}
\item[черника:]
темно-синие, очень вкусные,
хороши в~свежем виде, варенье
тоже получается хорошее;

\item[голубика:]
синие, более водянистые, чем
черника, и~не~такие вкусные;

\item[брусника:]
ярко-красные, из~них получается
очень вкусное варенье.
\end{description}
```

3.8 Разделы документа, титульный лист, оглавление

Чтобы помочь читателю ориентироваться, вы должны разделять ее на главы, разделы и подразделы. LaTeX поддерживает это специальными командами, принимающими в качестве аргумента заголовок раздела.

Класс article включает следующие команды секционирования: \section{...} (раздел), \subsection{...} (подраздел), \subsubsection{...} (подподраздел), \paragraph{...} (параграф), \subparagraph{...} (подпараграф), \appendix (приложение).

В классах report и book вы можете использовать две дополнительные команды: \part{...} (часть) и \chapter{...} (глава).

Так как глав в классе article нет, статьи довольно легко добавлять в книгу в качестве глав. Интервалы между разделами, нумерация и размер шрифта заголовков устанавливаются LaTeX автоматически.

Две из команд секционирования — особенные:

- Команда \part не влияет на последовательность нумерования глав.
- Команда \appendix аргумента не имеет. Она просто начинает нумеровать главы буквами вместо цифр.

LaTeX создает оглавление, беря заголовки разделов и номера страниц из предыдущего цикла компиляции документа. Команда \tableofcontents вставляет оглавление в то место, где она вызвана. Чтобы получить правильное оглавление, новый документ должен быть обработан LaTeX дважды.

Все перечисленные команды секционирования существуют также в вариантах со звездочкой. Такой вариант получается добавлением * к имени команды. Они генерируют заголовки разделов, которые не нумеруются и не включаются в оглавление. Например, команда \section{Справка} становится \section*{Справка}.

Обычно заголовки разделов появляются в оглавлении точно в том же виде, в каком они вводятся в тексте. Иногда это невозможно из-за того, что заголовок слишком длинен для оглавления. Элемент оглавления может в этом случае указываться необязательным аргументом перед собственно заголовком.

Титульный лист документа в целом генерируется при помощи команды \maketitle.

Его содержимое должно быть определено командами \title{. . . }, \author{. . . } и \date{. . . } до момента вызова \maketitle. Аргумент команды \author может содержать несколько имен, разделенных командами \and.

3.9 Создание списка литературы

LaTeX предоставляет возможность оформить список литературы, элементы которого нумеруются автоматически; в тексте при этом надо ссылаться не на номера, которые могут измениться в процессе работы над документом, а на установленные вами условные обозначения для элементов списка литературы.

Список литературы оформляется как окружение thebibliography. Это окружение имеет обязательный аргумент — номер элемента библиографии, который займет больше всего места на печати (в стандартных шрифтах все цифры имеют одинаковую ширину, так что достаточно привести в качестве аргумента, например, номер 99, если в списке литературы будет заведомо меньше 100 источников).

Каждый элемент списка литературы вводится командой \bibitem. У нее есть один обязательный аргумент — ваше условное обозначение. В качестве такого обозначения можно использовать любую последовательность из букв и цифр.

В тексте ссылка на элемент списка литературы делается с помощью команды \cite. У нее есть обязательный аргумент — условное обозначение того источника, на который вы ссылаетесь. Можно сослаться

сразу на несколько источников — для этого в аргументе команды \cite надо указать через запятую обозначения для тех источников, на которые вы хотите сослаться. Команда \cite может иметь необязательный аргумент: он ставится перед обязательным; в квадратных скобках записывается текст, который будет через запятую напечатан после номеров ссылок.

В [3, гл. 1] описана встреча Винни-Пуха с несколькими пчелами. В [1, 2] приведены другие сведения о медведях.

1) М. Е. Салтыков-Щедрин. Медведь на воеводстве.
2) Л. Н. Толстой. Три медведя.
3) А. А. Милн. Винни-Пух.

```
В~\cite[гл.~1]{Winnie} описана
встреча Винни--Пуха с несколькими
пчелами.
В~\cite{voevoda,med3} приведены
другие сведения о медведях.
\begin{thebibliography}{99}
\bibitem{voevoda}
М.\,Е.\,Салтыков--Щедрин. Медведь
на воеводстве.
\bibitem{med3} Л.\,Н.\,Толстой.
Три медведя.
\bibitem{Winnie} А.\,А.\,Милн.
Винни--Пух.
\end{thebibliography}
```

Как это обычно и происходит с автоматически генерируемыми LaTeXом ссылками, вам может потребоваться запустить обработку исходного файла дважды.

4 Набор математических формул

4.1 Основные принципы

LaTeX включает в себя специальный режим для верстки математики. Математика может быть набрана внутри абзаца, но может и разбивать абзац выделенной формулой. Математический текст внутри абзаца вводится между \(и \), между $ и $ или между \begin{math} или \end{math}.

Складывая a в квадрате с b в квадрате, получаем c в квадрате. Или излагая языком математики: $c^2 = a^2 + b^2$

```
Складывая $a$ в квадрате с
$b$ в квадрате, получаем
$c$ в квадрате. Или
излагая языком математики:
$c^2 = a^2 + b^2$
```

TeX произносится как $\tau\epsilon\chi$.

```
\TeX{} произносится как
\(\tau\epsilon\chi\).
```

100 м^3 воды.

```
100~м$^3$ воды.
```

Это исходит от моего ♡

```
Это исходит от моего
\begin{math}\heartsuit\end{math}
```

Большие математические уравнения или формулы предпочтительнее «выключать», то есть верстать их на отдельных строчках. Для этого заключайте их между \[и \] или между \begin{displaymath} и \end{displaymath}.

Складывая a в квадрате с b в квадрате, получаем c в квадрате. Или излагая языком математики:

$$c^2 = a^2 + b^2$$

или вы можете выразить это короче:

$$a + b = c$$

```
Складывая $a$ в квадрате с
$b$ в квадрате, получаем
$c$ в квадрате. Или
излагая языком математики:
\begin{displaymath}
c^{2}=a^{2}+b^{2}
\end{displaymath}
или вы можете выразить это
короче: \[a+b=c\]
```

Если вы хотите, чтобы LaTeX нумеровал уравнения, используйте окружение equation.

$$\varepsilon > 0 \qquad (4.1)$$

```
\begin{equation}
\varepsilon > 0
\end{equation}
```

Обратите внимание на разницу в стиле верстки выражений в абзацах и выключных формулах:

$\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k^2}=\frac{\pi^2}{6}$

```
$\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}$
```

$$\lim_{n\to\infty} \sum_{k=1}^{n} \frac{1}{k^2} = \frac{\pi^2}{6}$$

```
\begin{displaymath}
\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}
\end{displaymath}
```

Как вы могли уже заметить, для написания дроби в математическом режиме используется команда \frac с двумя обязательными параметрами, для числителя и знаменателя. Допускается вложенность:

$$\frac{1}{\frac{x}{x^2}}$$

```
$$\frac{
  1
}{
  \frac{x}{x^2}
}$$
```

Квадратный корень вводится как \sqrt, корень *n*-ой степени печатается при помощи \sqrt[*n*]. Размер знака корня выбирается LaTeX автоматически. Если нужен только знак, используйте \surd.

$\sqrt{x^2 + \sqrt{y}}$ $\sqrt[3]{2}$ $\surd[x^2 + y^2]$

```
$\sqrt{x^2 + \sqrt{y}}$ \qquad
$\sqrt[3]{2}$ \qquad
$\surd[x^2 + y^2]$
```

Математический режим отличается от *текстового режима*. Например, в математическом режиме:

1) Большинство пробелов и возвратов каретки не принимаются во внимание, так как все пробелы либо выводятся из логики математических выражений, или должны в явном виде задаваться командами вроде \,, \quad или \qquad. Размер \quad примерно соответствует ширине буквы «M» в текущем шрифте, а \qquad — в два раза больше \quad.
2) Пустые строчки недопустимы. Каждая формула занимает только один абзац.
3) Каждая буква считается именем переменной и верстается в этом качестве. Если вы хотите в формулу ввести нормальный текст (нормальный прямой шрифт с нормальными пробелами), то вам нужно вводить его командами \textrm{...} (см. также раздел 3.5 на странице 18).

$$\forall x \in \boldsymbol{R}: \qquad x^2 \geq 0 \qquad (4.2)$$

```
\begin{equation}
\forall x \in \mathbf{R}:
\qquad x^{2} \geq 0
\end{equation}
```

$$x^2 \geq 0 \qquad \text{для всех } x \in \boldsymbol{R} \qquad (4.3)$$

```
\begin{equation}
x^{2} \geq 0\qquad
\textrm{для всех }x\in
\mathbf{R}
\end{equation}
```

Математики бывают очень строги к используемым символам: здесь будет удобно использовать «ажурные полужирные символы», которые получаются командой \mathbb из пакетов amsfonts или amssymb. Последний пример теперь выглядит так:

$x^2 \geq 0 \qquad$ для всех $x \in \mathbb{R}$

```
\begin{displaymath}
x^{2} \geq 0\qquad
\textrm{для всех }x\in
\mathbb{R}
\end{displaymath}
```

Большинство команд математического режима действует только на следующий символ. Так что, если вы хотите, чтобы команда влияла на несколько символов, вам нужно сгруппировать их вместе при помощи фигурных скобок: {. . . }.

$$a^x + y \neq a^{x+y} \qquad (4.4)$$

```
\begin{equation}
a^x+y \neq a^{x+y}
\end{equation}
```

Имейте в виду, что расстановка пробелов и переносов строки не влияет на внешний вид формулы.

Коммутативность: $x + y = y + x$
Коммутативность: $x + y = y + x$

```
Коммутативность: $x+y=y+x$
Коммутативность: $ x + y = y + x $
```

4.2 Таблицы символов, операций и знаков

Имя команды, задающей строчную греческую букву, совпадает с английским названием этой буквы (например, буква α задается командой \alpha). Исключение составляет буква o (она называется «омикрон»): по начертанию она совпадает с курсивной латинской о, так что специальной команды для нее не предусмотрено, и для ее набора достаточно просто написать о в формуле. Некоторые греческие буквы имеют по два варианта начертаний; это также отражено в таблице 4.1.

Обратите внимание на требования русского математического набора:

греческая $\varkappa$ вместо κ
греческая ε вместо ϵ
греческая φ вместо ϕ

```
греческая $\varkappa$
  вместо $\kappa$
греческая $\varepsilon$
  вместо $\epsilon$
греческая $\varphi$
  вместо $\phi$
```

Имя команды, задающей прописную греческую букву, пишется с прописной буквы (например, буква Ψ задается командой \Psi). Некоторые прописные греческие буквы («альфа», например) совпадают по на-

Таблица 4.1 — Строчные греческие буквы

Набрано	*Вышло*
\alpha	α
\delta	δ
\zeta	ζ
\vartheta	ϑ
\lambda	λ
\xi	ξ
\rho	ρ
\varsigma	ς
\phi	ϕ
\psi	ψ

Набрано	*Вышло*
\beta	β
\epsilon	ϵ
\eta	η
\iota	ι
\mu	μ
\pi	π
\varrho	ϱ
\tau	τ
\varphi	φ
\omega	ω

Набрано	*Вышло*
\gamma	γ
\varepsilon	ε
\theta	θ
\kappa	κ
\nu	ν
\varpi	ϖ
\sigma	σ
\upsilon	υ
\chi	χ

чертанию с латинскими, и для них специальных команд нет — надо просто набрать соответствующую латинскую букву прямым шрифтом (см. 3.5 по поводу того, как это сделать). Не надо использовать греческие буквы Σ и Π из этой таблицы в качестве знаков суммы и произведения: для этих целей есть специальные команды, о которых пойдет речь дальше. Прописные греческие буквы, не совпадающие по начертанию с латинскими, представлены в таблице 4.2.

Таблица 4.2 — Прописные греческие буквы

Набрано	*Вышло*
\Gamma	Γ
\Lambda	Λ
\Sigma	Σ
\Psi	Ψ

Набрано	*Вышло*
\Delta	Δ
\Xi	Ξ
\Upsilon	Υ
\Omega	Ω

Набрано	*Вышло*
\Theta	Θ
\Pi	Π
\Phi	Φ

Следующая серия символов — символы бинарных операций (наподобие знаков сложения, умножения и т. д.). TEX оставляет в формуле небольшие пробелы по обе стороны этих знаков — кроме случаев, когда есть основания считать, что эти знаки используются не для обозначения операций, а для других целей (если, например, стоят два плюса подряд, то дополнительного пробела между ними не будет). Список бинарных операций отображен в таблице 4.3.

В таблице 4.4 собраны символы бинарных отношений. Команда \mid в этой таблице определяет вертикальную черточку, рассматриваемую как знак бинарного отношения; ее *не* следует употреблять, если вертикальная черточка употребляется как аналог скобки (например, как знак абсолютной величины).

Обратите внимание, что в правилах русской полиграфии операции $\leq$ и $\geq$ принято писать в виде $\leqslant$ и $\geqslant$. Для этого используются команды \leqslant и \geqslant соответственно.

Таблица 4.3 — Бинарные операции

Набрано	*Вышло*
+	$+$
\pm	$\pm$
\div	$\div$
\circ	$\circ$
\cup	$\cup$
\sqcup	$\sqcup$
\oplus	$\oplus$
\odot	$\odot$
\triangleright	$\triangleright$
\wr	$\wr$
\ddagger	$\ddagger$
\bigtriangledown	$\bigtriangledown$

Набрано	*Вышло*
-	$-$
\mp	$\mp$
\setminus	$\setminus$
\bullet	$\bullet$
\uplus	$\uplus$
\vee	$\vee$
\ominus	$\ominus$
\oslash	$\oslash$
\amalg	$\amalg$
\star	$\star$
\bigcirc	$\bigcirc$

Набрано	*Вышло*
*	$*$
\times	$\times$
\cdot	$\cdot$
\cap	$\cap$
\sqcap	$\sqcap$
\wedge	$\wedge$
\otimes	$\otimes$
\triangleleft	$\triangleleft$
\diamond	$\diamond$
\dagger	$\dagger$
\bigtriangleup	$\bigtriangleup$

В таблице 4.5 собраны стрелки различных видов.

Следующей большой группой рассмотрим команды для математических операций типа $\sin$, $\log$ и т. п., обозначаемых последовательностью букв, набранных прямым шрифтом. К любой из этих команд можно поставить верхний и/или нижний индекс (см. пример формулы 4.4 на странице 34).

Теперь обсудим, как можно было бы получить, скажем, формулу:

$$\sum_{i=1}^{n} n^2 = \frac{n(n+1)(2n+1)}{6}$$

```
$$\sum_{i = 1}^n n^2 =
  \frac{n (n + 1) (2n + 1)}{6}$$
```

с дополнительными записями над и под знаком операции. В данной формуле эти записи называются «пределы суммирования», поэтому в TEXнической терминологии записи над и под знаком операции принято называть «пределами» (по-английски limits). В исходном тексте «пределы» обозначаются точно так же, как индексы.

Существует различие в отображении «пределов» для выключной и внутритекстовой формулы. В первом случае, «пределы» печатаются сверху и снизу математической операции. Во внутритекстовой формуле они отображаются в том же виде, что и индексы.

В таблице 4.7 отображен список операций, ведущих себя так же, как \sum:

Обратите внимание, что для экономии места пределы интегрирования (\int и \oint) печатаются не сверху и снизу от знаков интеграла, а по бокам (даже в выключных формулах):

$$\int_0^1 x^2\,dx = 1/6$$

```
$$\int_0^1 x^2\,dx = 1/6$$
```

Таблица 4.4 — Бинарные отношения

Набрано	*Вышло*
<	$<$
:	$:$
\ne	$\ne$
\approx	$\approx$
\ll	$\ll$
\parallel	$\parallel$
\notin	$\notin$
\subseteq	$\subseteq$
\approx	$\approx$
\succ	$\succ$
\preceq	$\preceq$
\sqsupseteq	$\sqsupseteq$
\dashv	$\dashv$
\mid	$\mid$
\propto	$\propto$

Набрано	*Вышло*
>	$>$
\le	$\le$
\sim	$\sim$
\cong	$\cong$
\gg	$\gg$
\perp	$\perp$
\ni	$\ni$
\supset	$\supset$
\cong	$\cong$
\prec	$\prec$
\asymp	$\asymp$
\models	$\models$
\smile	$\smile$
\bowtie	$\bowtie$

Набрано	*Вышло*
=	$=$
\ge	$\ge$
\simeq	$\simeq$
\equiv	$\equiv$
\doteq	$\doteq$
\in	$\in$
\subset	$\subset$
\supseteq	$\supseteq$
\equiv	$\equiv$
\succeq	$\succeq$
\sqsubseteq	$\sqsubseteq$
\vdash	$\vdash$
\frown	$\frown$
\Join	$\Join$

Если необходимо, чтобы пределы интегрирования стояли над и под знаком интеграла, то надо непосредственно после \int записать команду \limits, а уже после нее — обозначения для пределов интегрирования:

$$\int\limits_0^1 x^2\,dx = 1/6$$

```
$$\int\limits_0^1 x^2\,dx = 1/6$$
```

Тот же прием с командой \limits можно применить, если хочется, чтобы во внутритекстовой формуле «пределы» у оператора стояли над и под ним, а не по бокам.

Если, с другой стороны, надо, чтобы «пределы» у какого-либо оператора стояли не над и под знаком оператора, а сбоку, то после команды для знака оператора надо записать команду \nolimits, а уже после нее — обозначения для «пределов».

В последней таблице этого подраздела приведем список символов, которые трудно отнести к какому-либо из вышеприведенных списков.

Последние шесть символов (от † до $) можно использовать не только в формулах (математическом режиме), но и в тексте.

Таблица 4.5 — Стрелки различных видов

Набрано	*Вышло*
\to	$\to$
\Rightarrow	$\Rightarrow$
\hookrightarrow	$\hookrightarrow$
\longmapsto	$\longmapsto$
\gets	$\gets$
\Leftarrow	$\Leftarrow$
\hookleftarrow	$\hookleftarrow$
\longleftrightarrow	$\longleftrightarrow$
\Longleftrightarrow	$\Longleftrightarrow$
\Uparrow	$\Uparrow$
\Downarrow	$\Downarrow$
\Updownarrow	$\Updownarrow$
\searrow	$\searrow$
\nwarrow	$\nwarrow$
\rightharpoonup	$\rightharpoonup$

Набрано	*Вышло*
\longrightarrow	$\longrightarrow$
\Longrightarrow	$\Longrightarrow$
\mapsto	$\mapsto$
\leadsto	$\leadsto$
\longleftarrow	$\longleftarrow$
\Longleftarrow	$\Longleftarrow$
\leftrightarrow	$\leftrightarrow$
\Leftrightarrow	$\Leftrightarrow$
\uparrow	$\uparrow$
\downarrow	$\downarrow$
\updownarrow	$\updownarrow$
\nearrow	$\nearrow$
\swarrow	$\swarrow$
\leftharpoonup	$\leftharpoonup$
\leftharpoondown	$\leftharpoondown$

Таблица 4.6 — Операции типа синус

Набрано	*Вышло*
\log	$\log$
\arg	$\arg$
\hom	$\hom$
\sin	$\sin$
\arccos	$\arccos$
\ctg	$\ctg$
\csc	$\csc$

Набрано	*Вышло*
\lg	$\lg$
\ker	$\ker$
\deg	$\deg$
\arcsin	$\arcsin$
\tg	$\tg$
\arcctg	$\arcctg$
\sinh	$\sinh$

Набрано	*Вышло*
\ln	$\ln$
\dim	$\dim$
\exp	$\exp$
\cos	$\cos$
\arctg	$\arctg$
\sec	$\sec$
\cosh	$\cosh$

4.3 Скобки переменного размера

Если заключенный в скобки фрагмент формулы занимает много места по вертикали (за счет дробей, степеней и тому подобного), то и сами скобки должны быть больше обычных. В TEXe на этот случай предусмотрен механизм автоматического выбора размера скобок.

В формуле

$$e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$$

```
$$e = \lim_{n \to \infty}
  \left(
    1 + \frac{1}{n}
  \right)^n$$
```

скобки обычного размера вокруг $1+\frac{1}{n}$ смотрелись бы плохо; поэтому при ее наборе надо поставить команду \left перед открывающей скобкой и команду \right перед закрывающей.

Таблица 4.7 — Операции с «пределами»

Набрано	*Вышло*
\sum	$\sum$
\bigcap	$\bigcap$
\bigotimes	$\bigotimes$
\bigwedge	$\bigwedge$
\lim	$\lim$
\max	$\max$
\inf	$\inf$
\gcd	$\gcd$

Набрано	*Вышло*
\prod	$\prod$
\coprod	$\coprod$
\bigodot	$\bigodot$
\biguplus	$\biguplus$
\limsup	$\limsup$
\min	$\min$
\det	$\det$
\int	$\int$

Набрано	*Вышло*
\bigcup	$\bigcup$
\bigoplus	$\bigoplus$
\bigvee	$\bigvee$
\bigsqcup	$\bigsqcup$
\liminf	$\liminf$
\sup	$\sup$
\Pr	$\Pr$
\oint	$\oint$

Таблица 4.8 — Другие символы, используемые в математическом режиме

Набрано	*Вышло*
\partial	∂
\infty	∞
\emptyset	$\emptyset$
\prime	$\prime$
\imath	$\imath$
\surd	$\surd$
\natural	$\natural$
\wp	$\wp$
\backslash	$\backslash$
\clubsuit	$\clubsuit$
\mho	$\mho$
\dag	$\dagger$
\ddag	$\ddagger$

Набрано	*Вышло*
\triangle	$\triangle$
\forall	$\forall$
\neg	$\neg$
\hbar	$\hbar$
\jmath	$\jmath$
\flat	$\flat$
\top	$\top$
\Re	$\Re$
\\|	$\|$
\diamondsuit	$\diamondsuit$
\Box	$\Box$
\S	$\S$
\P	$\P$

Набрано	*Вышло*
\angle	$\angle$
\exists	$\exists$
\aleph	$\aleph$
\nabla	∇
\ell	ℓ
\sharp	$\sharp$
\bot	$\bot$
\Im	$\Im$
\spadesuit	$\spadesuit$
\heartsuit	$\heartsuit$
\Diamond	$\Diamond$
\copyright	$\copyright$
\pounds	$\pounds$

Если перед одной скобкой стоит \left, а перед другой скобкой стоит \right, то на печати размер этих скобок будет соответствовать высоте фрагмента формулы, заключенного между \left и \right.

Конструкция с \left и \right применима не только к круглым скобкам. В таблице 4.9 перечислены скобки и некоторые другие символы, которые с помощью \left и \right автоматически принимают нужный размер. Кроме знаков, перечисленных в этой таблице, менять свои размеры под действием \left и \right могут и вертикальные стрелки из таблицы 4.5 на странице 38.

Вместе с каждой командой \left в формуле должна присутствовать соответствующая ей команда \right, в противном случае TEX выдаст сообщением об ошибке. Вместе с тем TEX вовсе не требует, чтобы ограничители (например, скобки) при командах \left и \right были расположены сколько-нибудь осмысленно с математической точки зрения: вполне можно написать что-нибудь вроде \left(... \right], или даже, вопреки

Таблица 4.9 — Ограничители

Набрано	*Вышло*
(	(
]	]
\lfloor	⌊
\rceil	⌉
|	∣
\backslash	\

Набрано	*Вышло*
)	)
{	{
\rfloor	⌋
\langle	⟨
\\|	∥

Набрано	*Вышло*
[	[
}	}
\lceil	⌈
\rangle	⟩
/	/

смыслу слов left и right, \left) ... \right(— за правильность своих формул отвечаете только вы, и TEX тут не помощник.

Вместо ограничителя после команды \left или \right можно поставить точку. На месте этой точки ничего не напечатается, а другой «ограничитель» будет необходимого размера.

4.4 Набор матриц и систем уравнений

Хотя подробное рассмотрение набора формул в математическом режиме выходит за рамки данного пособия, рассмотрим еще одну часто используемую тему.

Чтобы набрать с помощью LATEXа матрицу, надо воспользоваться окружением array. Прежде, чем описывать, как это окружение работает, разберем такой пример:

$$\left(\begin{array}{cccc} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{array}\right)$$

```
$$
\left(\begin{array}{cccc}
a_{11} & a_{12} & \ldots & a_{1n} \\
a_{21} & a_{22} & \ldots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{n1} & a_{n2} & \ldots & a_{nn}
\end{array}\right)
$$
```

Посмотрим, как устроен исходный текст, давший на печати эту матрицу. Всякая матрица состоит из *строк* и *столбцов*; строки матрицы разделяются с помощью команды \\ (последнюю строку заканчивать командой \\ не обязательно), а элементы внутри одной строки, относящиеся к разным столбцам, отделяются друг от друга с помощью символа &. Далее, после \begin{array}, открывающего окружение, должна следовать (в фигурных скобках, поскольку это параметр окружения array) так называемая *преамбула* матрицы, описывающая, сколько и каких столбцов должно быть в матрице. В нашем случае преамбула представляет собой четыре буквы c. Это значит, что в матрице 4 столбца (по букве на столбец), и что содержимое каждого из этих столбцов должно быть

расположено по центру столбца (поскольку каждая из букв — буква c). Кроме c, в преамбуле может стоять буква l, означающая, что соответствующий столбец будет выровнен по левому краю, или r, означающая, что столбец будет выровнен по правому краю.

В примере одной строке исходного текста соответствовала одна строка матрицы. Такое расположение мы выбрали только для удобства чтения, но, вообще говоря, оно совершенно не обязательно: бывает, что на протяжении нескольких строк приходится тянуть текст, который пойдет в одну строку на печати, а иногда в одной строке исходного текста помещается несколько строк матрицы.

Разберем еще один типичный пример — верстку системы уравнений с помощью окружения array:

$$\left\{\begin{array}{rcl} x^2+y^2 & = & 7 \\ x+y & = & 3. \end{array}\right.$$

```
$$
  \left\{\begin{array}{rcl}
  x^2 + y ^2 & = & 7 \\
  x + y      & = & 3.\\
  \end{array}\right.
$$
```

Мы отвели по одному столбцу на левую часть каждого уравнения, на знак равенства и на правую часть. При этом мы попросили, чтоб левые части уравнений были выровнены по правому краю (отсюда r в преамбуле), правые части — по левому краю (l в преамбуле), а знак равенства располагался по центру своей колонки (поэтому вторая буква в преамбуле — буква c). Для создания фигурной скобки, охватывающей всю систему, мы воспользовались командами \left и \right, причем при команде \right стоит «пустой ограничитель» — точка.

4.5 Важные мелочи

В математическую формулу можно включить фрагмент обычного текста с помощью LaTeXовской команды \mbox или \textrm.

$\sqrt{x^3} = x$ для всех x.

```
$$\sqrt{x^3} = x \qquad
  \mbox{для всех}x.$$
```

Чтобы получить в математической формуле изображение перечеркнутого символа, надо перед командой, генерирующей этот символ, поставить команду \not.

Часто требуется поставить горизонтальную черту над или под любым фрагментом формулы. Для этого можно воспользоваться командами \overline и \underline соответственно.

Кроме стандартного курсивного шрифта в математической формуле может понадобиться набрать какую-нибудь часть формулы другим,

специальным, образом. Для этого используются измененные математические шрифты, приведенные в таблице 4.10.

Таблица 4.10 — Математические шрифты

Команда	*Пример*
ABCdef	\mathrm{ABCdef}
ABCdef	\mathit{ABCdef}
$ABCdef$	\mathnormal{ABCdef}
$\mathcal{ABC}$	\mathcal{ABC}
$\mathfrak{ABCdef}$	\mathfrak{ABCdef}
$\mathbb{ABC}$	\mathbb{ABC}

В математическом режиме также используются специальные значки над символами, называемые акцентами. Варианты акцентов отображены в таблице 4.11.

Таблица 4.11 — Акценты математического режима

Набрано	*Вышло*
\hat{a}	$\hat{a}$
\acute{a}	$\acute{a}$
\ddot{a}	$\ddot{a}$
\vec{a}	$\vec{a}$

Набрано	*Вышло*
\check{a}	$\check{a}$
\grave{a}	$\grave{a}$
\breve{a}	$\breve{a}$
\widehat{a}	$\widehat{a}$

Набрано	*Вышло*
\tilde{a}	$\tilde{a}$
\dot{a}	$\dot{a}$
\bar{a}	$\bar{a}$
\widetilde{a}	$\widetilde{a}$

Если поставить значок над буквой i, так, чтобы сохранилась и точка над буквой, то это будет некрасиво. Поэтому ставить акценты следует не прямо над этой буквой, а над символом $\imath$ (\imath).

5 Специальные возможности

5.1 Использование цвета

Пакет color позволяет выбирать по своему усмотрению цвет текста и фона как для отдельного блока на странице, так и для всей страницы печатного документа.

Все команды переключения цвета имеют опцию, в которой указывается цветовая модель, и обязательный аргумент, в котором задается спецификация цвета. Синтаксис спецификации зависит от выбранной модели. Рассмотрим все цветовые модели, которые поддерживает пакет color.

Цветовая модель named является моделью по умолчанию. Спецификация цвета в этой модели задается по *имени* цвета. В пакете color определены имена 68 цветов, приведенные в таблице 5.1 вместе с самими цветами[1].

Для того, чтобы вывести текст каким-либо цветом, необходимо воспользоваться командой \textcolor[*модель*]{*цвет*}{*текст*}, где *модель* — используемая цветовая модель. Если модель не указана — используется named.

Ярко-коричневый x^2

```
\textcolor{Sepia}{
  \textbf{Темно–коричневый} $x^2$}
```

В модели rgb любой цвет получается в результате смешения трех базовых цветов: красного (red), зеленого (green) и синего (blue). Поэтому спецификация цвета в этой модели задается тремя перечисленными через запятую числами, каждое от 0 до 1, которые соответствуют абсолютной интенсивности базовых составляющих света. Например, желтый цвет задается как 1,1,0, а белый — как 1,1,1. 0,0,0 дает черный цвет.

Задавая цвет, надо помнить, что его восприятие зависит не только от длины волны света, но и от насыщенности и яркости излучения. Например, свет, содержащий лучи только красного цвета от тусклого источника, воспринимается не как бледно-красный, а как черный. Бледно-красный цвет получается при большой интенсивности света, в котором красная составляющая несколько превышает остальные.

Серый (gray) цвет получается при смешении в равных пропорциях базовых цветов из модели rgb с интенсивностью меньше единицы. Спецификация цвета в модели gray задается одним числом от 0 до 1, например, 0.5 вместо 0.5,0.5,0.5 в модели rgb. Черному цвету соответствует 0, а белому — 1.

[1] Восприятие цвета зависит от выходного устройства: даже на экран монитора GhostScript и Acrobat Reader выводят разные цвета с одной и той же спецификацией.

Таблица 5.1 — Цвета модели named

Имя	*Цвет*
GreenYellow	
Dandelion	
Melon	
BurntOrange	
Mahogany	
Red	
WildStrawberry	
Magenta	
Mulberry	
Lavender	
DarkOrchid	
Violet	
Periwinkle	
MidnightBlue	
Blue	
ProcessBlue	
TealBlue	
Emerald	
Green	
LimeGreen	
OliveGreen	
Brown	
Black	

Имя	*Цвет*
Yellow	
Apricot	
YellowOrange	
Bittersweet	
Maroon	
OrangeRed	
Salmon	
VioletRed	
RedViolet	
Thistle	
Purple	
RoyalPurple	
CadetBlue	
NavyBlue	
Cerulean	
SkyBlue	
Aquamarine	
JungleGreen	
ForestGreen	
YellowGreen	
RawSienna	
Tan	
White	

Имя	*Цвет*
Goldenrod	
Peach	
Orange	
RedOrange	
BrickRed	
RubineRed	
CarnationPink	
Rhodamine	
Fuchsia	
Orchid	
Plum	
BlueViolet	
CornflowerBlue	
RoyalBlue	
Cyan	
Turquoise	
BlueGreen	
SeaGreen	
PineGreen	
SpringGreen	
Sepia	
Gray	

Если на лист бумаги нанести красную краску и затем осветить его белым светом, то только красная составляющая света отразится от бумаги. Нанесем теперь на бумагу красную краску густо посаженными точками, а затем заполним все промежутки между красными точками зеленой краской. Теперь от листа бумаги отразится как красный, так и зеленый свет. Издали различить отдельные точки нельзя, поэтому глаз получит смесь красного и зеленого света и бумага будет выглядеть желтой[2].

На этом принципе основана цветовая модель cmyk. В этой модели спецификация цвета задается четырьмя перечисленными через запятую числами от 0 до 1, которые соответствуют «количеству» голубой (cyan), пурпурной (magenta), желтой (yellow) и черной (black) краски на белом листе бумаги. Теоретически при смешении первых трех цветов в равной пропорции должен получиться черный цвет. В действи-

[2]Если сначала смешать красную и зеленую краски, а затем нанести смесь на бумагу, то получится темный цвет с красноватым оттенком.

тельности краски поглощают свет не полностью и поэтому смесь трех основных цветов выглядит темно-коричневой. По этой причине в модель введена еще и черная краска. Из описанного выше ясно, что в модели cmyk белому цвету соответствует спецификация 0,0,0,0 (на белом листе бумаги нет никакой краски), а черному — 0,0,0,1.

Красный,
опять красный,
и еще раз красный!

```
\textcolor{red}{Красный,}
\textcolor[cmyk]{0,1,1,0}{
  опять красный,}
\textcolor[rgb]{1,0,0}{и
  еще раз красный!}
```

Команда \definecolor{*имя*}{*модель*}{*цвет*} позволяет определить *имя* для любого цвета. Здесь *модель* — цветовая модель, *цвет* — спецификация цвета. Используя *имя* в качестве спецификации цвета, цветовую модель можно не указывать, поскольку такой способ переключения цвета относится к модели по умолчанию named.

Блеклый...

```
\definecolor{faded}{gray}{0.7}
\textcolor{faded}{Блеклый\dots}
```

Цвет фона текста можно задать командой \colorbox[*модель*]{*цвет*}{*текст*}.

Текст

```
\colorbox{Yellow}{Текст}
```

Изменить цвет страницы можно командой \pagecolor[*модель*]{*цвет*}. Область действия команды не ограничивается никакими окружениями. Чтобы вернуть белый цвет страниц, надо вызвать команду \pagecolor{white}.

5.2 Верстка таблиц

Публикации в научно-технических изданиях часто требуют оформления данных, в виде таблиц. LaTeX предлагает несколько пакетов для представления таблиц. Рассмотрим создание таблиц с помощью окружения — tabular.

Основные особенности окружения tabular:

- Окружение tabular применимо в любом режиме: оно создает таблицу в виде прямоугольной области, которую можно поместить в середину формулы или строки текста. Поэтому с помощью окружения tabular можно строить таблицы с очень сложной структурой, вкладывая одну таблицу в другую.
- LaTeX не может начать новую страницу в середине текста, форматируемого окружением tabular. Поэтому таблицу, созданную tabular, обычно размещают в виде плавающего объекта с помощью

окружений table или figure (см. раздел 5.4 на странице 50). Очень длинные таблицы печатают на нескольких страницах, используя пакет longtable.

- LaTeX автоматически устанавливает ширину колонок.

Окружение tabular позволяет легко рисовать границы таблиц, проводить разделительные вертикальные и горизонтальные линии между элементами. Строки в таблице разделяются командой \\ или \tabularnewline, а каждая строка обрабатывается в строковом режиме, то есть не переносится на новую строку, если не вмещается в текущую.

Планета	Среднее удаление от Солнца, 10^6 км
Меркурий	58
Венера	108
Земля	150
Марс	227
Юпитер	778
Сатурн	1 424
Уран	2 874
Нептун	4 497
Плутон	5 894

```
\begin{tabular}{|l|p{3,5cm}|} \hline
Планета & Среднее удаление от
Солнца, $10^6$~км \\ \hline
Меркурий & 58 \\ \hline
Венера & 108 \\ \hline
Земля & 150 \\ \hline
Марс & 227 \\ \hline
Юпитер & 778 \\ \hline
Сатурн & 1\,424 \\ \hline
Уран & 2\,874 \\ \hline
Нептун & 4\,497 \\ \hline
Плутон & 5\,894 \\ \hline
\end{tabular}
```

В обязательном параметре указывается количество и вид колонок. В примере {|l| p{3,5cm}|} мы указываем, что в таблице будет три колонки, причем содержимое первой должно быть выравнено по левому краю, а второй колонки — в виде параграфа шириной 3,5 см. Символы | показывают, что между колонками нужно провести вертикальные линии на всю высоту таблицы. Горизонтальные линии на всю ширину таблицы проводит команда \hline.

Как процедура tabular может форматировать колонки переменной ширины, показывает еще один пример.

Imparfait		*Plus-que-parfait*		
j’	étais	j’	avais	été
tu	étais	tu	avais	été
il	était	il	avait	été
nous	étions	nous	avions	été
vous	étiez	vous	aviez	été
ils	étaient	ils	avaient	été

```
\begin{tabular}{|rl|l|rcl|}
\cline{1-2} \cline{4-6}
\multicolumn{2}{|c|}{\it Imparfait} & \hspace{5mm} &
\multicolumn{3}{|c|}{\it Plus-que-parfait} \\
\cline{1-2} \cline{4-6}

j’ & \’etais & & j’ & avais & \’et\’e \\
```

```
\cline{1-2} \cline{4-6}

tu & \'etais & & tu & avais & \'et\'e \\
\cline{1-2} \cline{4-6}

il & \'etait & & il & avait & \'et\'e \\
\cline{1-2} \cline{4-6}

nous & \'etions & & nous & avions & \'et\'e \\
\cline{1-2} \cline{4-6}

vous & \'etiez & & vous & aviez & \'et\'e \\
\cline{1-2} \cline{4-6}

ils & \'etaient & & ils & avaient & \'et\'e \\
\cline{1-2} \cline{4-6}
\end{tabular}
```

Пробел между таблицами в 5 мм описан как пустая колонка. Ширина колонки определяется командой горизонтального отступа \hspace{5mm} в первой строке исходного текста таблицы. Команда \cline проводит горизонтальную линию через колонки (в данном случае, через колонки 1–2 и 4–6). Команда \multicolumn объединяет элементы нескольких ячеек одной строки таблицы в одну ячейку, позволяя, как в данном примере, создавать общую ячейку. Первый параметр указывает, сколько колонок нужно объединить. Второй параметр должен содержать описание параметров форматирования этой ячейки. Третий параметр — текст этой объединенной ячейки.

Итак, в обязательном параметре окружения tabular указываются количество способы форматирования колонок. Основные возможные значения в этом параметре:

- l — колонка с выравниванием по левой границе;
- c — колонка с выравниванием по центру;
- r — колонка с выравниванием по правой границе;
- p{wd} — колонка, шириной в wd с выравниванием по ширине строки;
- | — вертикальная линия между колонками на всю высоту таблицы.

Материал, набираемый в окружении tabular, всегда верстается на одной странице. Если вам нужно набирать длинные таблицы, используйте окружения supertabular или longtable.

Пакет colortbl, используя пакеты color и tabular, позволяет раскрашивать таблицы. Цвета задаются точно так же, как в пакете color через цветовую модель и спецификацию цвета.

Команду \columncolor[*модель*]{цвет}[*отступ слева*][*отступ справа*] можно использовать в спецификации >{...} в преамбуле окружения

tabular. Она раскрашивает весь столбец таблицы цветом, который задается параметром *цвет*. Опции *отступ слева* и *отступ справа* задают расстояния, соответственно, слева и справа между краями окрашенной области и текстом. Если указан только один аргумент, то он задает оба расстояния.


```
\begin{tabular}{%
|>{\color{white}\columncolor{%
black}}l|>{\columncolor{yellow%
}[.6\tabcolsep]}c|>{%
\columncolor[gray]{.8}}r|}
один & два & три \\
четыре & пять & шесть
\end{tabular}
```

Команда \rowcolor имеет такие же параметры, что и команда \columncolor и может использоваться для задания цвета целой строки таблицы. Ее место — в самом начале строки таблицы.


```
\begin{tabular}{|l|c|}
\rowcolor[gray]{.9} один & два \\
\rowcolor[gray]{.6} три & четыре
\end{tabular}
```

Приведем теперь пример таблицы со слитыми ячейками в строке. В этом случае цвет слитых ячеек нужно задавать в спецификации команды \multicolumn. Сделаем это, введя новый тип колонки.


```
\newcolumntype{H}{>{%
  \columncolor{magenta}}c}
\begin{tabular}{%
|>{\columncolor{yellow}}l|
>{\color{white}\columncolor{%
black}}l|}
\multicolumn{2}{|H|}{один} \\
два & три \\ четыре & пять
\end{tabular}
```

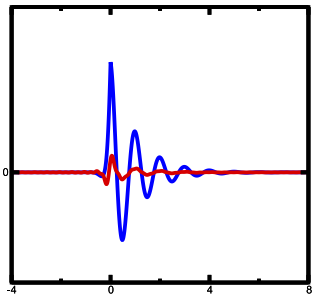
Глобальная команда \arrayrulecolor[*модель*]{цвет} задает цвет горизонтальных и вертикальных линий, разделяющих ячейки в таблицах.

5.3 Включение рисунков

Для включения рисунков в LaTeX в подавляющем большинстве случаев используют пакет graphicx. Пакет позволяет включать в документ рисунки из графических файлов, поворачивать, растягивать или сжимать изображения.

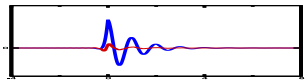
В пакете graphix определена команда \includegraphics[*параметры*] {*файл*} для вставки в документ рисунка из графического файла. В необязательном параметре может указываться целый список ключей, изменяющих свойства вставляемого рисунка. Значения параметров задаются в виде *параметр=значение*, а в списке они перечисляются через запятую. Команда \includegraphics не заканчивает абзац, поэтому позволяет вставлять небольшие рисунки прямо внутрь текста.

При использовании *pdflatex* можно вставлять изображения в форматах *png*, *pdf* и *jpeg*. Когда расширение файла в команде \includegraphics не указано, драйвер последовательно (слева направо по списку) добавляет к имени файла все известные ему расширения и ищет файл уже по полному имени. Поиск прекращается, как только находится первый подходящий файл.




```
\includegraphics{graph}
```

Можно принудительно изменить размер рисунка, задав его параметрами width и height (ширина и высота рисунка, соответственно).



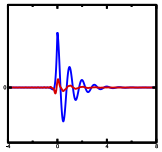
```
\includegraphics[width=4cm,%
  height=1cm]{graph}
```

Параметр keepaspectratio обеспечивает сохранение ширины к высоте самого рисунка, если заданные значения ширины и высоты области, выделенной под рисунок, нарушают, как в предыдущем примере, это отношение.

```
\includegraphics[width=4cm,%
  height=1cm,%
  keepaspectratio]{graph}
```

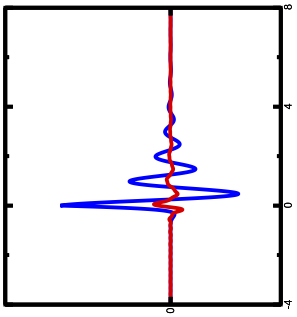
Как видно из примера, при использовании параметра keepaspectratio принимается во внимание только один параметр, определяющий размер изображения (с меньшим значением), а второй просто игнорируется.

Параметр scale=*scale* изменяет «натуральный» размер рисунка в *scale* раз.



```
\includegraphics[scale=0.5]{graph}
```

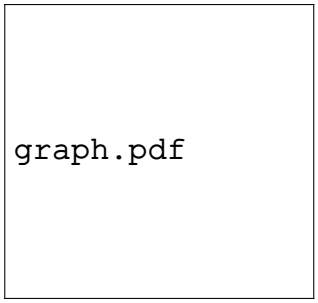
Параметр angle=*angle* поворачивает рисунок на *angle* градусов против часовой стрелки. Ось отсчета можно определить параметром origin=*позиция*. Возможные значения: *l* — на левом крае области изображения, *c* — по центру области изображения, *r* — на правом крае области изображения. К этим значениям можно добавить *t* — верхний край области изображения и *b* — нижний край области изображения. В результате получаются следующие допустимые значения *pos*: *lb*, *l*, *lt*, *b*, *c*, *t*, *rb*, *r*, *rt*.



```
\includegraphics[angle=90,%
  origin=c]{graph}
```

По умолчанию LaTeX ищет файлы с рисунками в каталоге, в котором находится входной файл. Можно указывать относительный или полный путь к файлу прямо в команде \includegraphics, но проще указать каталоги, в которых будут находиться ваши изображения. Команда \graphicspath{список каталогов} позволяет расширить область поиска. В параметре *списка каталогов* каждая из директорий заключается в фигурные скобки. Если указать, например \graphicspath{{images/}{d:/images/}}, то LaTeX будет искать файлы с рисунками также в подкаталоге images текущего каталога и в директории d:/images/.

Указание параметра draft в команде \documentclass в преамбуле документа или в параметре рисунка приводит к тому, что вместо рисунка будет начерчена рамка и напечатано внутри него имя файла рисунка.




```
\includegraphics[draft]{graph}
```

Если параметр draft указан в \documentclass, его можно принудительно отменить для отдельных рисунков параметром final.

5.4 Плавающие объекты

Большинство публикаций содержат множество иллюстраций и таблиц. Эти элементы нуждаются в специальном обращении с ними, так как они не могут быть разбиты между страницами. Одним из выходов было бы начинать новую страницу каждый раз, если иллюстрация или

Таблица 5.2 — Допустимые ключи размещения плавающих объектов

Ключ	*Позволяет размещать объект*
h	*здесь же*, в том самом месте текста, где он появился (обычно используется для маленьких объектов)
t	*наверху* страницы
b	*внизу* страницы
p	на *специальной* странице, содержащей только плавающие объекты
!	принуждать LaTeX выбирать только из заданных вами параметров

таблица слишком велика, чтобы поместиться на текущей странице. Однако, тогда страницы оставались бы частично пустыми, что смотрится очень плохо.

Любая иллюстрация или таблица, не умещающаяся на текущей странице, может «плавать», перемещаясь на следующую страницу в процессе заполнения текстом текущей. LaTeX предлагает для плавающих объектов два окружения, одно для таблиц и одно для иллюстраций. Чтобы полностью использовать их преимущества, важно примерно представлять, как LaTeX обрабатывает плавающие объекты. Иначе они могут стать источником разочарования из-за того, что LaTeX помещает их не туда, куда вы хотите. Вначале рассмотрим команды, предоставляемые LaTeX для плавающих объектов.

Любой материал, включенный в окружения figure или table, трактуется как плавающий. Оба окружения имеют необязательный параметр, называемый *спецификацией размещения*. Этот параметр указывает LaTeX, куда можно перемещать плавающий объект. Спецификация размещения конструируется путем собирания в строчку ключей размещения плавающего объекта. Допустимые ключи размещения приведены в таблице 5.2.

Например, таблицу можно начать следующей строкой: \begin{table}[!hbp]. Спецификация размещения [!hbp] позволяет LaTeX разместить таблицу прямо по месту (h), или внизу той же страницы (b), или на выделенной странице (p) — даже если это будет смотреться не так уж хорошо (!). Если никакой спецификации размещения не задано, стандартные классы предполагают [tbp].

LaTeX размещает каждый встреченный плавающий объект в соответствии с заданной автором спецификацией. Если объект нельзя поместить на текущей странице, он откладывается, помещаясь в очередь иллюстраций или в очередь таблиц. Когда начинается новая страница,

LaTeX проверяет, можно ли заполнить специальную страницу плавающими объектами из очередей. Если нет, то первый объект из каждой очереди считается только что встретившимся в тексте: LaTeX снова пытается разместить их в соответствии с их спецификациями (за исключением *h*, что уже невозможно). Новые встреченные в тексте плавающие объекты помещаются в соответствующие очереди. LaTeX сохраняет порядок, в котором встретились плавающие объекты соответствующего типа. Поэтому иллюстрация, которую не удается разместить, отталкивает все дальнейшие иллюстрации к концу документа. Следовательно, если LaTeX не размещает плавающие объекты, как вы этого ожидаете, вероятно какой-то объект устроил затор в одной из очередей.

Хотя и возможно задавать в LaTeX конкретную спецификацию размещения плавающего объекта, это может вызвать проблемы. Если объект не помещается в указанном месте, он «застревает», блокируя последующие плавающие объекты. В частности, никогда не используйте ключ [h]; это настолько плохо, что в современных версиях LaTeX он автоматически заменяется [ht].

Командой \caption{текст заголовка} вы можете задать заголовок для объекта. Увеличивающийся номер и строка «Рисунок» или «Таблица» добавляются LaTeXом автоматически.

В определенных случаях может быть необходимо использовать команду \clearpage или даже \cleardoublepage. Она указывает LaTeX немедленно разместить все плавающие объекты, остававшиеся в очередях, и затем начать новую страницу. \cleardoublepage, помимо этого, начинает новую правостороннюю страницу.

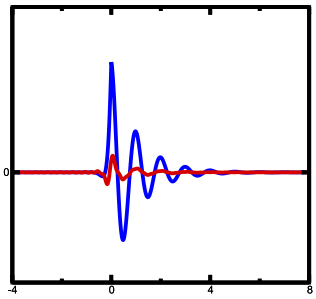


Рисунок 5.1 — Пример вставки рисунка

Листинг 5.1 — Исходный текст примера вставки рисунка

```
\begin{figure}[ht]
  \centering
  \includegraphics{graph}
  \caption{Пример вставки рисунка}
\end{figure}
```

5.5 Перекрестные ссылки

В книгах, отчетах и статьях часто встречаются перекрестные ссылки на иллюстрации, таблицы и отдельные части текста. Для этого LaTeX предоставляет следующие команды: \label{метка}, \ref{метка} и \pageref{метка}, где *метка* — выбранный пользователем идентификатор. LaTeX заменяет \ref номером раздела, подраздела, иллюстрации, таблицы или формулы, где была использована соответствующая команда \label. \pageref печатает номер страницы, соответствующей команде \label. Для правильной обработки ссылок может потребоваться повторный проход TeX-файла (иначе вместо номера-метки будут стоять знаки вопроса).

$$\pi \approx 3,14159265358979324 \quad (5.1)$$

Ссылка на этот раздел выглядит так: «см. раздел 5.5 на стр. 53.»
Ссылка на формулу работает так же — 5.1.

```
\begin{equation}
\pi \approx 3,14159265358979324
\label{equ:equation}
\end{equation}

Ссылка на~этот
раздел~\label{sec:this}
выглядит так: <<см.
раздел~\ref{sec:this}
на~стр.~\pageref{sec:this}.>>

Ссылка на~формулу работает так
же~---~\ref{equ:equation}.
```

5.6 Создание новых команд и окружений

Чтобы добавить собственные команды, воспользуйтесь специальной командой \newcommand{*название*}[*число*]{*определение*}. Обычно эта команда требует двух аргументов: название и определение команды, которую вы создаете. Аргумент — число в квадратных скобках не обязателен. Он применяется для создания новых команд, которые, в свою очередь, принимают до 9 аргументов.

Внимание! Теперь этот текст заметят
Ложь: вот теперь этот текст заметят

```
\newcommand{\warning}{\bf
  Внимание! \rm}
\newcommand{\wordwarn}[2]{\bf #1%
  \rm \footnotesize{~#2}}
\warning{Теперь этот текст
  заметят} \\
\wordwarn{Ложь:}{вот теперь этот
  текст заметят}
```

LaTeX не позволит вам создать новую команду, которая бы изменяла уже существующую. Но для случая, когда вы явно хотите изменить

существующую команду, есть специальная команда: \renewcommand. Она имеет тот же синтаксис, что и команда \newcommand.

Аналогично команде \newcommand, существует команда для создания вашего собственного окружения, \newenvironment{*название*}[*номер*]{*начало*}{*конец*}. Подобно команде \newcommand, \newenvironment можно использовать с необязательным аргументом или без него. Материал, заключенный в аргумент *начало*, обрабатывается до обработки текста внутри окружения. Материал, заключенный в аргумент *конец*, обрабатывается, когда встречается команда \end{*название*}.

5.7 Включение исходных текстов программы

Для качественного отображения исходных текстов программ с выделением ключевых слов, строковых констант, комментариев и т. п. существует пакет listings. В этом пакете присутствуют определения автоматического распознавания ключевых слов для многих языков программирования и разметки, поэтому в большинстве случаев достаточно просто правильно определить язык исходного текста. Среди огромного количества поддерживаемых языков присутствуют:

- Assembler (x86masm);
- C (ANSI, Objective, Sharp);
- C++ (ANSI, GNU, ISO, Visual);
- Delphi;
- HTML;
- Java;
- make (empty, gnu);
- Matlab;
- Pascal (Borland6, Standard, XSC);
- Perl;
- PHP;
- Python;
- Ruby;
- SQL;
- TeX (AlLaTeX, common, LaTeX, plain, primitive);
- VHDL;
- XML.

Для вставки исходного текста используется окружение lstlisting. В случаях, если исходный код находится в отдельном файле, удобнее просто ссылаться на него командой \lstinputlisting[аргументы]{*имя файла*}. В остальном команда \lstinputlisting аналогична окружению lstlisting.

```
/* hello.c */
#include <stdio.h>

int main() {
  printf("Hello, world!\n");

  return 0;
}
```

```
\begin{lstlisting}[language={[ANSI]C}]
/* hello.c */
#include <stdio.h>

int main() {
  printf("Hello, world!\n");

  return 0;
}
\ end{lstlisting}
```

Пакет listings содержит огромное количество параметров и настроек, что не позволяет подробно рассмотреть их в рамках данного пособия. Для более детальной информации обратитесь к документации пакета listings.

Литература

1) Говорухин В., Цибулин В. Компьютер в математическом исследовании. М: Питер, 2001.
2) Грэтцер Г. Первые шаги в LaTeX'e. — М.: Мир, 2000.
3) Гуссенс М., Миттельбах Ф., Самарин А. Путеводитель по пакету LaTeX. — М.: Мир, 1999.
4) Кнут Д. Все про TeX. М: Вильямс, 2003.
5) Рыжиков Ю. Работа над диссертацией по техническим наукам. М: BHV, 2005.
6) Спивак М. Восхитительный AMS-TeX. Руководство по комфортному изготовлению научных публикаций в пакете AMS-TeX. М: Мир, 1993.

УЧЕБНОЕ ИЗДАНИЕ

Составители: Костюк Д. А., к.т.н.;
Ильяшевич Д. А.

МЕТОДИЧЕСКОЕ ПОСОБИЕ

«Использование системы верстки LaTeX
для оформления учебных работ»

для студентов специальностей
информатики и радиоэлектроники

Ответственный за выпуск:

Редактор Строкач Т.В.