Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Алтайский государственный университет»

Международный институт экономики, менеджмента и информационных систем

Кафедра цифровых технологий и бизнес-аналитики

Издательская система LaTeX

Учебное пособие

Барнаул

Издательство
Алтайского государственного
университета
2021

УДК 004.439
ББК 32.97
Ж91

Автор: канд. физ.-мат. наук *Журенков О. В.*
Рецензент: канд. физ.-мат. наук *Юдинцев А. Ю.*

Ж91 **Журенков, Олег Викторович**

Издательская система LaTeX : учеб. пособие / О. В. Журенков ; Министерство науки и высшего образования РФ, Алтайский государственный университет. – Барнаул : Изд-во Алт. ун-та, 2021. — 166 с.

ISBN 978-5-7904-2576-9.

Пособие содержит подробное описание управляющих команд издательской системы LaTeX, а также некоторых низкоуровневых команд (TeX'а) и утилит, используемых вместе с этой издательской системой. Наряду со справочной информацией пособие содержит ряд лабораторных работ, последовательное выполнение которых позволит в освоить LaTeX на уровне верстальщика (наборщика текста), что вполне достаточно для автора научных публикаций любой сложности.

Издательская система LaTeX является свободно-распространяемым кроссплатформенным приложением и может использоваться для преподавания дисциплин, связанных с издательскими системами, редакционно-издательской деятельностью, подготовкой научных публикаций.

Пособие может использоваться в дисциплинах, рассматривающих издательские системы и технологии; адресовано студентам, магистрантам, аспирантам, научным работникам, инженерам и всем, желающим освоить подготовку публикаций высокого качества.

УДК 004.439
ББК 32.97

ISBN 978-5-7904-2576-9

© О. В. Журенков, 2021
© О. В. Журенков, оформление, 2021
© Алтайский государственный университет, 2021

Содержание

1 Издательские системы **8**
1. Обзор издательских систем (ИС) 8
1.1. Системы визуального проектирования 8
1.2. Системы логического проектирования 9
1.3. LaTeX 10
2. Автоматизация процесса документирования 11
3. Doxygen 12
Лабораторная работа №1. Документирование программного кода 17

2 Введение в LaTeX **18**
1. Рабочий процесс LaTeX 18
2. Диагностика ошибок 21
2.1. Сообщения об ошибках, выдаваемые LaTeX'ом 24
2.2. Сообщения об ошибках, выдаваемые TeX'ом 28
2.3. Предупреждающие сообщения LaTeX'а 32
2.4. Предупреждающие сообщения TeX'а 33
3. Команды и аргументы 34
3.1. Логосы и декларации 35
3.2. Правила скобок 35
4. Буквы и символы 36
4.1. Специальные символы 36
4.2. Акценты 36
4.3. Тире, кавычки, многоточия 39
5. Шрифты 40
5.1. Пользовательские команды выбора шрифтов 40
5.2. Пользовательские команды изменения размера шрифтов 40
6. Низкоуровневые команды задания шрифтов 42
6.1. Кодировка 42
6.2. Непосредственное задание символа 43
6.3. Гарнитура 43
6.4. Насыщенность 44
6.5. Начертание 46
6.6. Кегль 46
6.7. Переключение шрифтов 47

Лабораторная работа №2. Простой текст 48

3 Вёрстка и форматирование **51**
1. Управление строками 51
   1.1. Горизонтальные пробелы 51
   1.2. Переносы 53
   1.3. Разрыв строки 53
   1.4. Абзацы 54
2. Управление страницами 55
   2.1. Вертикальные пробелы 55
   2.2. Страницы 56
Лабораторная работа №3. Строки, абзацы, страницы 58
3. Форматирование текста 61
   3.1. Выравнивание 61
   3.2. Цитаты и стихи 61
   3.3. Списки 62
   3.4. Неформатированный текст 62
   3.5. Сноски и заметки 62
4. Боксы 63
   4.1. Строковые боксы 63
   4.2. Текстовые боксы 64
   4.3. Линейные боксы 65
   4.4. Сохранение бокса 66
5. Команды в LaTeX 2_ε 67
   5.1. Командные длины 67
   5.2. Счётчики 68
   6.1. Определение новых команд 71
   6.2. Определение новых окружений 72
   6.3. Определение команд типа «теорема» 73
Лабораторная работа №4. Форматирование текста 74

4 Печатный документ **76**
1. Структура документа 76
   1.1. Преамбула 76
   1.2. Титульный лист 76
   1.3. Аннотация (abstract) 77
   1.4. Секционирование (рубрикация) 77
   1.5. Оглавление (содержание) 78
   1.6. Библиография (список литературы) 79

2. Механизм перекрёстного цитирования 80
3. Большие документы 81
3.1. Условная компиляция 81
3.2. Включение файлов 82
Лабораторная работа №5. Структура документа 83
4. Стиль документа 85
4.1. Стиль страницы 85
4.2. Стандартные классы 89
Лабораторная работа №6. Стиль документа 91
5. Определение новых команд 92

5 Математика в LaTeX'е 93
1. Основные понятия 93
1.1. Математические моды (режимы) 93
1.2. Пробелы в математических формулах 95
2. Алфавит математики 96
2.1. Математические акценты 96
2.2. Греческие буквы 97
2.3. Бинарные операторы 97
2.4. Символы сравнения 97
2.5. Большие операторы и символы переменного размера 98
2.6. Разделители 100
2.7. Стрелки 102
2.8. Функции 102
2.9. Прочие символы 103
3. Основные структуры 104
3.1. Верхние и нижние индексы 104
3.2. Корни 105
3.3. Дроби 105
3.4. Биномиальные коэффициенты 106
3.5. Размещение объектов друг над другом 107
3.6. Знаки пунктуации и многоточия 108
3.7. Формулы в рамке 109
3.8. Условный выбор 109
3.9. Матрицы 110
4. Большие формулы 112
4.1. Уравнения 112
4.2. Сложные формулы 113
4.3. Позиционирование 114

4.4. Нумерация и ссылки 115
5. Шрифты 116
5.1. Классы $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX 117
Лабораторная работа №7. Математика в LaTeXe 118

6 Таблицы в LaTeX'е 119
1. Табулятор 119
2. Таблица 121
2.1. Параметры настройки 123
3. Размещение таблиц 123
Лабораторная работа №8. Таблицы 126

7 Графика в LaTeX 2_ε 127
1. Основные понятия 127
2. Импортирование графики 128
3. Размещение рисунков 133
4. Псевдографика 136
4.1. Позиционирование 136
4.2. Линии 137
4.3. Круги 137
4.4. Овалы 138
4.5. Боксы 139
5. Цвет 140
5.1. Определение цвета 141
5.2. Использование цветов 141
Лабораторная работа №9. Графика в LaTeXe 143

A BibTeX 146
1. Введение в BibTeX 146
2. Библиографическая база данных 148
2.1. Записи в базе данных 149
2.2. Текст поля 151
2.3. Сокращения 153
2.4. Склеивание строк 154
2.5. Преамбула базы данных 154
2.6. Перекрёстное цитирование 155
2.7. Записи в базе данных 156

B Утилита MakeIndex **159**
1. Использование утилиты MakeIndex 159
2. Вход в указатель 160
   2.1. Диапазоны страниц и перекрёстные ссылки 161
   2.2. Оформление элементов указателя 162
   2.3. Печать специальных символов 162
3. Параметры MakeIndex 163

Список литературы **165**

Глава 1

Издательские системы

1. Обзор издательских систем (ИС)

Все издательские системы можно разделить на два большие класса, которые отличаются принципом управления разметкой и форматированием.

1.1. Системы визуального проектирования

Системы *визуального проектирования* WYSIWYG (What You See Is What You Get):

- QuarkXpress;
- PageMaker, FrameMaker, InDesign (Adobe);
- Corel Word Perfect, Corel Draw (Corel);
- Scribus;
- TeXmacs, LyX;
- ...

MS Word — не является ИС! ИС от Microsoft — MS Office Publisher. Однако многие издательства принимают небольшие статьи в формате *RTF* (формата `doc` не существует, есть такое расширение файлов, и разные программы сохраняют файлы с таким расширением с разными недокументированными форматами).

Для обмена используется *XML* или файлы формата *RTF*.

> *RTF (Rich Text Format, «формат обогащённого текста») — свободный межплатформенный формат хранения размеченных текстовых документов, предложенный* Microsoft *и др.*

Первая версия стандарта *RTF* появилась в 1987 г., с тех пор спецификация формата несколько раз изменялась, поэтому *имеет место несовместимость* `rtf`*-файлов разных форматов. RTF*-документы поддерживаются большинством современных текстовых редакторов (под Microsoft Windows это, как правило, осуществляется с помощью стандартных библиотек, входящих в состав операционной системы).

Доступность издательского ПО с функцией WYSIWYG вытеснила большинство языков разметки и логического проектирования среди обычных пользователей, хотя серьёзная издательская работа по-прежнему использует разметку для специфических не визуальных структур текста, а WYSIWYG-редакторы сейчас чаще всего сохраняют документы в форматах, основанных на языках разметки.

1.2. Системы логического проектирования

Системы *логического проектирования* WYSIWYM (What You See Is What You Mean), основанные на текстовом процессоре TEX:

- PlainTEX;
- LATEX, SLiTEX, $\mathcal{A}\mathcal{M}\mathcal{S}$-TEX, LATEX 2_ε, ConTEXt, Omega;
- fpTEX, teTEX, TEXLive;
- emTEX (MS DOS, OS/2);
- MiKTEX (Windows);
- ozTEX, MacTeX, gwTeX (Apple/Mac OS X);
- pcTEX (Y&Y Inc.);
- Scientific Word (TCI Software Research Inc.);

- Personal TEX;
- TrueTEX;
- ...

Среди всего множества сборок TEX'а, TEXLive является мультиплатформенным и наилучшим (по поддержке и обновлениям).

1.3. LATEX

В начале 80-х гг. *Лесли Лампортом* (*Leslie Lamport*) была разработана издательская система на базе *TEX*а, названная им LATEX...

Преимущества ИС LATEX:

- соответствие стандарту *SGML*;
- полное разделение содержания документа с его оформлением благодаря концепции *общей разметки* (основываясь на опыте профессиональных типографских дизайнеров);
- совершенное полиграфическое качество;
- большое количество выходных форматов;
- полная совместимость для разных платформ;
- свободное распространение.

Последняя версия — LATEX 2_ε(предыдущая — LATEX2.09, будущая — LATEX3).

LATEX относится к свободному программному обеспечению и распространяется под лицензией LATEX Project Public License.

Во многих развитых компьютерных аналитических системах, таких, как Maple, Mathematica, Maxima, Reduce, R/RStudio, Gnuplot возможен экспорт документов в формат `.tex`. Для представления формул в Википедии также используется TEX-нотация.

LATEX используется не только для подготовки научных статей, аналитических отчётов и прочей документации со сложной разметкой (формулами, большими таблицами, графикой), но и для получения документации программных систем. Очень часто программисты начинают знакомство с LATEXом в связи с документированием программного кода.

2. Автоматизация процесса документирования

Наиболее полной и согласованной получается документация при автоматизации процесса документирования. *Автоматизированное документирование* — одна из самых важных сторон разработки программного кода. Идея автоматизированного документирования заключается в том, что в исходном коде пишутся комментарии в специальном формате. Далее запускается специальная утилита, которая разбирает исходный код, вытаскивая комментарии и объединяя их в единый структурированный документ. Таким образом, с плеч программиста снимается вся черновая работа по созданию программной документации. Кроме того, такой способ документирования позволяет более строго контролировать версии программного кода и документации, поскольку при изменении программного кода программист должен изменять расположенные рядом комментарии.

Для документирования программного кода используют *программы-генераторы*, сканирующие файлы проекта в поисках комментариев. Без комментариев даже небольшую программу трудно понять не только постороннему программисту, но и самому автору.

Многие среды разработки (Zend, Visual Studio, Netbeans) уже включают в себя средства документирования, но никакого общего стандарта до сих пор нет. В своём большинстве, цель генераторов — составить нечто типа энциклопедии всех функций, классов, объектов, иерархий классов, связей между файлами, используемые компоненты и третьи программные продукты.

Генераторы можно условно разделить на:

- Универсальные
  - Doxygen
  - Natural Docs На выходе — HTML.
  - AsciDoc (GNU GPLv2) На выходе — HTML(4,5), XHTML, DocBook, LaTeX, PDF, EPUB, DVI, PS, man, HTML Help, txt.
  - Doc-O-Matic для C/C++, C#, Delphi, VB.NET, IDL, Java, PHP, JavaScript, ASPX, JSP, MatLab.

Специфичные

— javadoc (Free) для Java.

— phpDocumentor (Open Source) для PHP.

— Docutils (Open Source) для Python.

— NDoc, Sandcastle, GhostDoc для .NET.

— PasDoc (GNU GPLv2) для Pascal (Object Pascal) На выходе — HTML, HtmlHelp, LaTeX, ...

Генераторы, как правило, по-своему хранят данные и комментарии, завися от языка, однако последние тенденции, например в Visual Studio, показывают популяризацию xml. DocBook — одна из предлагаемых универсальных схем.

3. Doxygen

Doxygen — мощная и распространённая утилита для документирования программного кода, написанного на C, C++, Objective-C, Java, Python, IDL (Corba и Microsoft) и некоторых версий PHP, C#, и D. Выходная документация представляет собой файлы в формате HTML, XML, LaTeX (в частности, для дальнейшего создания PDF-файлов), RTF, справочного руководства в стиле кроссплатформенной библиотеки Qt, справочного руководства в стиле Unix — man и т. д.

Даже, если не оформлять специальным образом исходные тексты, Doxygen сгенерирует из них замечательные справочники программного интрефейса приложения. В выходной документ могут быть включены разделы с описанием используемых пространств имён, классов, файлов и их указатели на соответствующие разделы. В случае использования пакета визуализации графов — Graphviz, описание классов может сопровождаться диаграммами UML.

Распространяется по лицензии GNU GPL.

Существует два вида комментариев:

Комментарии для автодокументирования исходного кода. Эти комментарии оформляются с помощью специальных форматов (doxygen, javadoc, MS XML Documentation Comments).

Комментарии исходного кода — сопровождающие исходный код, объясняющие его работу, раскрывающие непонятные и сомнительные моменты.

В отличие от комментариев для автоматизированного документирования, эти комментарии используются в любом программном коде. *Стоимость владения* исходным кодом уменьшается прямо пропорционально количеству и качеству таких комментариев. Это связано с тем, что хорошо структурированный и комментированный код легче поддерживается и сопровождается. То есть программистам просто требуется меньше времени для того, чтобы разобраться с ним. Для анализа качества исходного кода можно использовать специализированные инструменты (например, together), которые подсчитывают различные метрики кода, описывающие его качество.

Doxygen позволяет из комментариев кода, записанных в специальном формате, генерировать документы HTML, LaTeX, RTF, PDF с ссылками, диаграммами классов и прочими удобствами.

В Doxygen существуют несколько вариантов комментариев для автоматизированного документирования:

1. блочный стиль JavaDoc

```
/**
комментарий
*/
```

2. блочный стиль Qt

```
/*!
комментарий
*/
```

3. строчный стиль C++

```
/// комментарий
```

4. строчный стиль C++

```
//! комментарий
```

Всё, что написано внутри комментариев, оформленных таким образом, выносится в документацию. Для единообразия лучше использовать `///` для коротких комментариев и `/** */` для длинных.

Служебные слова

В Doxygen есть служебные слова, которые задают правила отображения частей документации (выделение в отдельный раздел, специальное форматирование и шрифт для выделения примера кода, вынесение текста на заглавную страницу и т.п.).

Автор — `@author` *имя*

Текст на главной странице — `@mainpage`

Создание новой страницы — `@page` *идентификатор название*

Например:

- `@page pagereq Требования к системе`

Вставка примера кода — `@code` текст программы `@endcode`

Создание раздела — `@section` *идентификатор название*

Например:

- `@section common Общие требования`

Заметка — `@note`

Предупреждение — `@warning`

Описание к файлу — `@file`

Краткое описание — `@brief` Можно настроить файл опций таким образом, чтобы в краткое описание выносилось первое предложение до точки подробного описания (по умолчанию описание считается подробным). Для этого нужно включить опцию JAVADOC_AUTOBRIEF — в утилите `doxywizard` она находится на первой странице.

Подробное описание — `@full`

Именованная группа — `@name` *имя*. После этого группа в скобках @{ ... @}

Внутри парного тега `\f` можно задавать формулу в нотации LaTeX. Этот тег будет проигнорирован, если в качестве выходного формата используется не LaTeX. Например:

■

```
Расстояние между \f$(x_1,y_1)\f$ и \f$(x_2,y_2)\f$
вычисляется, как \f$\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}\f$.
```

Расстояние между (x_1, y_1) и (x_2, y_2) вычисляется, как $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$.

Документирование функций `@function` *имя-функции*
Например:

■

```
  /**
  @function MyFunction

  Тестовая функция, имеет два входных параметра.

@param in\_Parameter1 --- параметр, отвечающий за что-то
@param out\_Parameter2 --- выходной параметр
@return true если всё хорошо, false если всё плохо
  */

bool MyFunction (int in\_Parameter1,
            std::string out\_Parameter2);
```

Схема работы Doxygen (совместно с Doxywizard) представлена на рис. 1.1.

Синтаксис имеет следующий вид:

1. Для генерации шаблона конфигурационного файла

```
doxygen [-s] -g [configName]
```

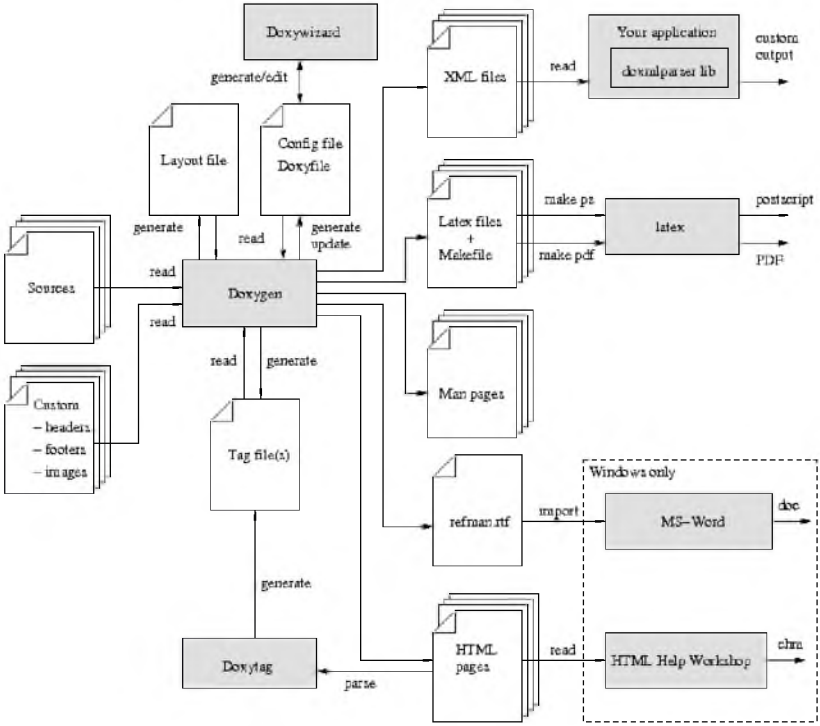


Рис. 1.1. Информационные потоки Doxygen (диаграмма с сайта www.doxygen.nl)

2. Для обновления шаблона конфигурационного файла

```
doxygen [-s] -u [configName]
```

3. Генерация документации на основе конфигурационного файла

```
doxygen [configName]
```

4. Для создания файла шаблона управления макетом при генерации документации

```
doxygen -l layoutFileName.xml
```

5. Для генерации файла документации с заданным расширением (на примере `.rtf`)

```
RTF:   doxygen -e rtf extensionsFile
```

6. Для создания шаблона стиля для RTF, HTML или LaTeX

```
RTF:   doxygen -w rtf styleSheetFile
HTML:  doxygen -w html headerF footerF styleSheetF [configFile]
LaTeX: doxygen -w latex headerF footerF styleSheetF [configFile]
```

Файл настроек можно редактировать в любом текстовом редакторе или с помощью специальных утилит с графическим интерфейсом. Одна из них, Doxywizard, поставляется вместе с Doxygen.

Лабораторная работа №1
Документирование программного кода

Задание 1. Добавление комментариев

Изучите справочную информацию по Doxygen.

Возьмите любую свою программу, откройте программный код в редакторе (любом). Дополните код комментариями для атодокументирования. При этом используйте синтаксис Doxygen (см., например, doxygenorg.ru).

Задание 2. Генерация документации

Сгенерируйте документацию, используя Doxygen (рекомендуется использовать GUI утилиту Doxywizard). Получите документацию в формате HTML и PDF (через LaTeX).

Отредактируйте файл в формате LaTeX(добавьте изменения в текст, измените класс документа).

Глава 2

Введение в LaTeX

1. Рабочий процесс LaTeX

Исходный LaTeX'овский файл является обычным текстовым файлом, содержащим, кроме текста, управляющие *команды*. Самая первая команда в исходном файле — `\documentclass{класс}` — определение *класса* документа.

Команда `\usepackage{пакет}` подключает дополнительные *пакеты*.

Сам текст документа должен быть написан между двумя командами:

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

Подобные команды называются *окружениями* (environment), или *процедурами*.

В исходном файле можно комментировать строки знаком '%'. , закомментированные строки не компилируются. **Преамбула** — вводная часть исходного файла, предшествующая самому документу.

Например:

■

```
\documentclass[a4paper,10pt]{article}
\usepackage[cp1251]{inputenc}
\usepackage[russian]{babel}

\title{Пример статьи} \author{Журенков~О.\,В.}

\begin{document}
\maketitle
\begin{abstract}
Очень простой пример
\end{abstract}

\section{Первый раздел} Какой-то текст.
\end{document}
```

Рассмотрим, как же работает такая издательская система. На рис. 2.1 показана логическая схема взаимодействия файлов, драйверов, программ и устройств.

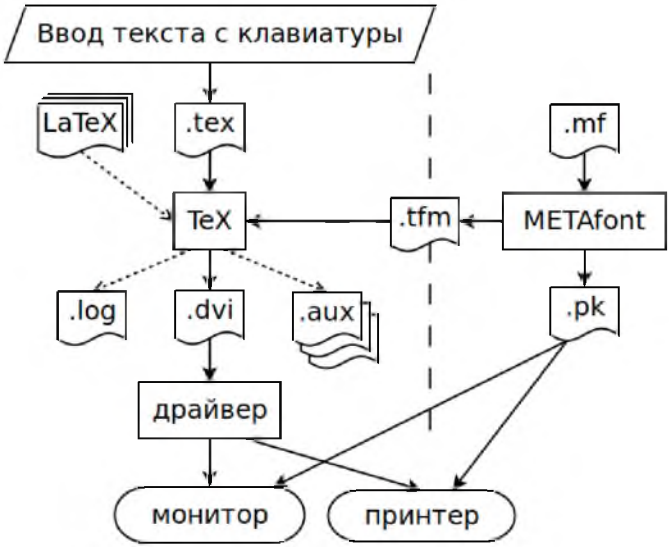


Рис. 2.1. Взаимодействие компонентов издательской системы LaTeX

Как уже было сказано …«Теховский» файл представляет из себя текстовый файл, включающий в себя, кроме обычного текста, набор управляющих команд, все они начинаются со знака ‘\’. Кроме этого знака существует ещё 9 служебных символов: ‘{’, ‘}’, ‘%’, ‘&’, ‘$’, ‘#’, ‘~’, ‘^’, ‘_’.

В .log файл выводится протокол компиляции (немного подробнее, чем на экран), который полезно уметь читать. В начале выводится имя компилятора, его версия, дата. Далее читается компилируемый файл: о начале чтения любого файла свидетельствует открывающаяся круглая скобка ‘(’, когда файл полностью прочитан, скобка закрывается ‘)’.

Например:

■

```
% Символ % служит для комментирования текста до конца
%строки.

\documentclass[a4paper]{article}  % Класс и опции печатного
                                  % документа.
\usepackage[cp1251]{inputenc}     % Подключение
\usepackage[russian]{babel}       % дополнительных пакетов.

\title{Пример статьи}             % Заголовок документа.
\author{Журенков~О.\,В.}          % Автор документа.
%\date{}                          % Вы можете сами задать
%                                 % дату.

\begin{document}                  % Здесь заканчивается
                                  % преамбула и начинается
                                  % текст.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\maketitle                        % Делает титульный лист.

\section{Первый раздел}           % Это заголовок раздела.
 Какой-то текст. Простой текст.

Слова разделяются одним или     несколькими пробелами,
конец строки считается тоже пробелом.
 Абзацы
    разделяются одной или несколькими строками. Добавление
лишних пробелов или строк во входном файле не влияет на
результат.
```

```
Одинарые кавычки печатаются как здесь: ‘выделенный текст’.
Двойные кавычки печатаются как здесь: ‘‘выделенный текст’’.
%
Длинный дефис (интервал) в предложениях печатается, как
здесь (37--43). Длинное тире  в предложениях печатается ---
как здесь.

Курсив (italic) выглядит так: \textit{это курсив}. Жирный
шрифт (bold) выглядит так: \textbf{это жирный шрифт}.

\end{document}                    % Здесь заканчивается текст.
Что либо ещё игнорируется.
```

2. Диагностика ошибок

Даже у опытного пользователя возможны ошибки в исходном файле. Тогда при компиляции будут выдаваться на экран (и в `.aux` файл) сообщения об ошибках. Кроме того, неизбежно будут появляться предупреждающие сообщения.

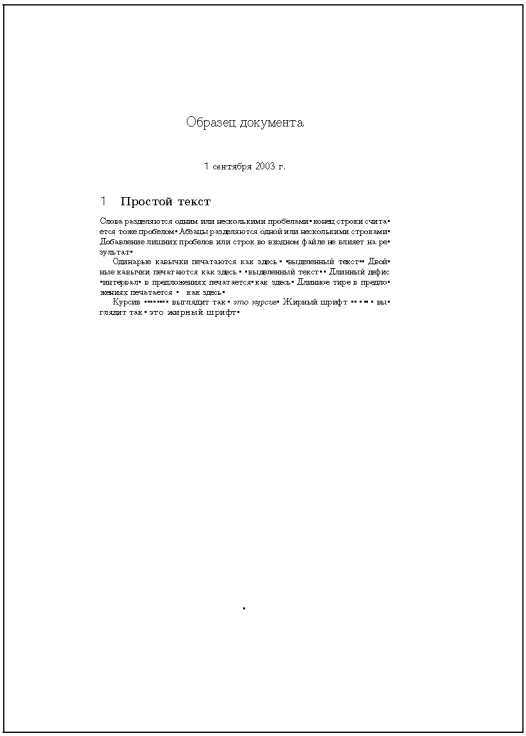
Сообщения об ошибках и предупреждения бывают двух видов: генерируемые TEX'ом и генерируемые LATEX'ом.

Сообщение, об ошибке всегда начинается со знака '!'. Если ошибка обнаруживается TEX'ом, то сообщение следует сразу после знака '!', например:

```
! Missing } inserted.
<inserted text>
                }
l.246 ... k^{'2} \tau} e^{-\lambda \tau}=$
                                         $
```

Из этого сообщения видно, что в исходном тексте пропущена скобка (}) и TEX сам её вставил.

Строка в исходном файле, на которой допущена ошибка указана ниже и начинается с буквы 'l'. Далее выводится часть этой строки, причём, в том месте, где предполагается ошибка, строка разрывается.



Образец документа

1 сентября 2003 г.

1 Простой текст

Слова разделяются одним или несколькими пробелами• конец строки считается тоже пробелом• Абзацы разделяются одной или несколькими строками• Добавление лишних пробелов или строк во входном файле не влияет на результат•

Одинарные кавычки печатаются как здесь • "выделенный текст"• Двойные кавычки печатаются как здесь • "выделенный текст"• Длинный дефис "интервал" в предложениях печатается как здесь• Длинное тире в предложениях печатается • — как здесь•

Курсив •••••••• выглядит так • *это курсив*• Жирный шрифт •••••• выглядит так • это жирный шрифт•

Рис. 2.2. Результат компиляции (.dvi файл) простого текста

Сообщения об ошибках LaTeX'а начинаются с фразы «LaTeX Error:», например:

```
! LaTeX Error: \begin{document} ended by \end{equation}.

See the LaTeX manual or LaTeX Companion for explanation.
Type  H <return>  for immediate help.
 ...

l.47 \end{equation}

Your command was ignored.
Type I <command> <return> to replace it with another
command, or <return> to continue without it.
```

При возникновении ошибки, компиляция останавливается, компилятор приглашает к диалогу знаком '`?`'. При этом можно ввести '`h`' (помощь), '`e`' или '`x`' (выход, прекращение компиляции), '`s`' (продолжение компиляции без остановки), клавиша'RETURN' (продолжение компиляции до следующей ошибки), '`r`' (то-же, что и '`s`', но без вывода на экран), '`i`' (ввод исправленной команды).

Если ошибка не критическая, то возможно продолжение компиляции, в противном случае выводится знак '`*`', тогда можно только прекратить компиляцию, нажав предварительно '`Ctrl C`'.

Наиболее часто встречающиеся ошибки:

- пропущена команда начала или конца документа;
- ошибка в имени команды, окружения, опции;
- пропущен аргумент команды;
- несоответствие открывающих и закрывающих скобок или командных скобок (начало, конец окружения)[1];
- пропущен один из разделителей математической моды;
- команда использована в недопустимой для нее моде.

Далее приводятся полный перечень сообщений об ошибках и предупреждений.

[1] Современные текстовые редакторы, как правило имеют функцию проверки скобок.

2.1. Сообщения об ошибках, выдаваемые LaTeX'ом

- `Bad \line or \vector argument`

  Неправильный аргумент команды `\line` или `\vector`, задающий наклон линии или вектора, соответственно.

- `Bad math environment delimiter`

  Неправильный разделитель математической моды: вероятно пропущен один из разделителей, включающих/выключающих математическую моду (`$`, `$$`, `\[`, `\]`, `\(`, или `\)`).

- `Bad use of \\`

  Неправильное использование команды \ (используется между абзацами, где она не имеет смысла).

- `\begin{env1} ended by \end{env2}`

  Нарушено соответствие командных скобок: `\begin{env1}` закончилось `\end{env2}`. Возможная причина — опечатка в имени, а также пропуск или лишняя вставка `\begin` или `\end` командных скобок.

- `Can be used only in preamble`

  Команда LaTeX'а, которая может появляться только в преамбуле документа, использована в теле документа (например, такие команды, как `\documentclass`, `\usepackage`, `\nofile`, `\includeonly`, `\makeindex`, `\makeglossary`). Это сообщение появляется и при наличии лишней команды `\begin{document}`.

- `Command name COM already used`

  Команда `COM` уже определена. Эта ошибка вызвана использованием одной из команд определения новой команды или окружения `newcounter`, `\newcommand`, `\newenvironment`, `\newtheorem`, `\newlist`, `\newsavebox` или `\newlength` с именем, которое уже определено.

- `Command COM not provided in base NFSS`

  Команда `COM` не реализована в базовой схеме NFSS. В NFSS не входят некоторые символы, входящие в стандарт LaTeX 2.09:

`\mho`, `\Join`, `\Box`, `\Diamon`, `\leadsto`, `\sqsubset`,`\sqsupset`, `\lhd`, `\unlhd`, `\rhd`, `\unrhd`. В LaTeX 2_ε эти символы определены в пакете latexsym.

- `Counter too large`

  Слишком большое значение счётчика. Возникает при использовании *алфавитного* стиля нумерации (максимальное значение символов 26 для английского алфавита) или специального (счётчик `fnsymbol`, максимальное значение — 9).

- `Environment ENV undefined`

  Используются окружение с неизвестным именем (в команде `\begin{ENV}` неизвестное имя `ENV`).

- `File FILENAME not found`

  Файл `FILENAME` не найден. LaTeX приостанавливает компиляцию и ожидает ввода с клавиатуры правильного имени файла (без расширения). При нажатии клавиши Enter компиляция продолжится, даже если не писать имя файла.

- `Float(s) lost`

  Командные скобки плавающих элементов (`figure, table`) или команда `\marginpar` помещены внутри бокса. Если в тексте много плавающих элементов, LaTeX, возможно, обнаружит эту ошибку, далеко уйдя от содержащей её команды, и найти соответствующее место в тексте будет нелегко. Один из таких элементов будет потерян, но необязательно тот, который ошибочно помещен в бокс.

- `Illegal character in array arg`

  Неправильный символ в аргументе окружения `array` или `tabular` или во втором аргументе команды `\multicolumn`.

- `Illegal use of \verb command`

  Неправильное использование команды `\verb`, вероятно она используется в аргументе другой команды.

- `\include cannot be nested`

Команда `\include` не может быть вложена, вероятно `\include` используется в файле, подключаемом командой `\include` из главного файла.

- `LaTeX2e command COM in LaTeX 2.09 document`

  Команда `COM` версии LaTeX 2_ε используется в документе LaTeX 2.09.

- `Lonely \item--perhaps a missing list environment`

  Отдельная команда `\item`, вероятно пропущено окружение-список.

- `Missing \begin{document}`

  Пропущена команда `\begin{document}`, либо в преамбуле содержится текст или команды, выводящие текст на печать.

- `Missing p-arg in preamble`

  В аргументе командных скобок array или tabular, или во втором аргументе команды `\multicolumn`, используемой внутри этих командных скобок, найден символ 'p' без следующего за ним выражения.

- `Missing @-exp in preamble`

  В аргументе командных скобок array или tabular, или во втором аргументе команды `\multicolumn`, используемой внутри этих командных скобок, найден символ '@' без следующего за ним @-выражения.

- `No such counter`

  Указание несуществующего счетчика в одной из команд работающей со счетчиками, например `\setcounter` или `\addtocounter`. Если ошибка появляется при обработке файла с расширением. aux, это может означать, что определение счетчика командой `\newcounter` находится вне преамбулы (т.е. после `\begin{document}`).

- `Not in outer par mode`

Командные скобки плавающих элементов (figure, table) или команда `\marginpar` используются в математической моде либо в боксе. Плавающие элементы можно использовать только во внешней абзацной моде.

- `\pushtabs and \poptabs don't match`

  В пределах командных скобок tabbing обнаружена команда `\poptabs` без предшествующей команды `\pushtabs`; или в момент обработки команды `\end{tabbing}` оказалось, что одна или несколько команд `\pushtabs` лишние.

- `Something's wrong---perhaps a missing \item`

  Скорее всего, пропущена команда `\item` в командных скобках, создающих список. Такое сообщение выдается, если пропущен аргумент в командных скобках `thebibliography`.

- `Tab overflow`

  Превышен предел для числа позиций табулятора. Увеличение этого предела может быть достигнуто только модификацией LaTeX'a.

- `There's no line here to end`

  Команды `\newline` или `\\` используются между абзацами, где они не имеют смысла. Для получения дополнительных промежутков между абзацами следует использовать команды `\vspace` или `\vskip`.

- `This is a LaTeX bug`

  Не повезло. Это следствие ошибки в самом LaTeX'e. В таком случае рекомендуется показать документ эксперту.

- `Too deeply nested`

  В документе более шести вложенных списков или более четырех вложенных списков с автоматическим перечислением. (Командные скобки `quote` и `verse` также рассматриваются как списки.)

- `Too many unprocessed floats`

не хватило памяти для плавающих элементов, создаваемых командными скобками `figure`, `table` или командой `\marginpar`. Ошибка возникает либо при попытке создать слишком много таких объектов на одной странице, либо когда в командных скобках figure или table заданы параметры размещения, превышающие возможности одной страницы.

- `Undefined tab position`

  Попытка перемещения табулятура на не определенную позицию при использовании одной из следующих команд: `>`, `\+`, `\-` или `<`.

- `\< in mid line`

  Команда '`<`' появилась в середине строки в командных скобках `tabular`. Эта команда может находиться только в начале строки.

2.2. Сообщения об ошибках, выдаваемые TEX'ом

- `Counter too large`

  Сноски нумеруются буквами или специальными символами, число которых, разумеется, ограничено. Эта ошибка может возникнуть, например, при использовании слишком большого числа команд `\thanks`.

- `Double subscript`

  В математической формуле обнаружена неоднозначная комбинация для получения двойного нижнего индекса.

- `Double superscript`

  В математической формуле обнаружена неоднозначная комбинация для получения двойного верхнего индекса.

- `Extra alignment tab has been changed to \cr`

  Количество символов '&' в строке массива или таблицы превышает заданное количество столбцов.

- `Extra \}, or forgotten \$`

Группирующие скобки не соответствуют друг другу, или пропущен (либо задан лишний) ограничитель выражения в математической моде.

- `Font ... not loaded. Not enough room left`

  Документ использует больше шрифтов, чем позволяет TEX. Можно попробовать обработать документ по частям.

- `I can't find file '... '`

  T^X не может найти указанный входной файл. Это может быть результатом ошибки при вводе имени файла при запуске TEX'а или указанием несуществующего файла в команде `\input` или `\documentstyle`. Если файл не найден, TEX выдает приглашение:

  Please type another input file name: и ждет ввода надлежащего имени файла.

- `Illegal unit of measure (pt inserted)`

  При задании длины указана безразмерная величина.

- `Illegal parameter number in definition of ...`

  При использовании команд `\newcommand` или `\renewcommand` сделана ссылка на аргумент, номер которого больше числа аргументов, определенных в этой команде. Если вместо многоточия в сообщении указана команда `\@gtempa`, это может означать, что в аргумент команды `\label` помещён символ '`#`'.

- `Misplaced alignment tab character \&`

  Использование специального символа '&' в обычном тексте, а не в таблице.

- `Missing control sequence inserted`

  В определении новых команд с помощью `\newcommand`, `\renewcommand` или `\newlength` есть такое, в котором первый аргумент не является правильно заданным именем команды.

- `Missing number, treated as zero`

Не задано числовое значение параметра (размерного или безразмерного). Скорее всего пропущен обязательный аргумент команды, или квадратные скобки в тексте помешены так, что были приняты за начало необязательного аргумента.

- `Missing \{ inserted`
  `Missing \} inserted`

  Пропущена одна из группирующих скобок.

- `Missing \$ inserted`

  Ряд команд TEX'а могут быть использованы только в математической моде. Встречая такие команды вне математической моды, TEX выдает данное сообщение об ошибке и при этом автоматически переключается в математическую моду.

- `Not a letter`

  Ошибка в аргументе команды `\hyphenation`.

- `Runaway argument?`
  `Paragraph ended before ... was complete`

  Зарегистрирован признак конца абзаца (команда `\par` или пустая строка) в аргументе команды, в котором не может быть нескольких абзацев. Причиной ошибки может быть так же отсутствие закрывающей скобки в конце аргумента или разделителя в команде `\verb`.

- `TeX capacity exceeded, sorry [...]`

  TEX'у не хватило памяти, и он прекратил работу. В квадратных скобках будет указан параметр, определяющий возможные причины ошибки, именно:

  `buffer size` В аргументах команд рубрикации или команд `\caption`, `\addcontentsline`, или `\addtocontents` задан слишком длинный фрагмент текста. Сообщение может быть выдано при работе команды `\end{document}`, если подобная ошибка имела место при исполнении команд `\tableofcontents`, `\listoffigures` или `\listoftables`.

`exception dictionary` TEX'у задано слишком много исключений для переноса слов командами `\hyphenation`. Удалите из этих команд редко встречающиеся в тексте слова, используя для указания в них места переноса команду `\-`.

`hash size` Определено слишком много команд и(или) меток для перекрестных ссылок.

`main memory size` Создана настолько сложная страница, что TEX не может хранить всю информацию, необходимую для ее формирования. Для того чтобы выявить причину, по которой переполняется память, рекомендуется вставить команду `\clearpage` перед местом возникновения ошибки. Затем, возможно, придется изменить исходный файл в месте сбоя, упростив построение страниц (например, перенеся подальше друг от друга рисунки и таблицы).

pool size Возможно, вы используете слишком много перекрестных ссылок и(или) определяете слишком много новых команд. Точнее, у ваших меток и команд слишком большая суммарная длина в символах. Попробуйте сделать имена ваших команд и меток короче.

Такая диагностика может также появиться, если пропущена закрывающая скобка, отделяющая аргументы таких команд, как `\setcounter`, `\newenvironment` или `\newtheorem`.

save size Команды, командные скобки и группы определений имеют слишком глубокую вложенность. Чаше всего это следствие заликливания команд, когда команда вызывает сама себя, возможно через несколько уровней.

- `Text line contains an invalid character`

Строка содержит запрещенные символы. Скорее всего текст имеет неправильную кодировку.

- `Undefined control sequence`

Использована неопределенная команда. Наиболее вероятная причина — наличие ошибки в имени команды или имени мак-

ропакета. Возможно не задан файл описания стиля, содержащий определение данной команды.

- `Use of COM doesn't match its definition`

  Если COM — команда LaTeX'а, скорее всего это одна из команд, используемых в пределах окружения `picture`, аргумент которой задан в неверном формате.

  Если `COM` — `\@array`, неправильно задано @-выражение в окружении `array`.

- `You can't use 'macro parameter character \#' in ... mode`

  Использование специального символа '`#`' в обычном тексте. Повидимому, требовалось ввести '`\#`'.

2.3. Предупреждающие сообщения LaTeX'а

В отличие от процедуры выдачи сообщения об ошибке, при выдаче предупреждающих сообщений LaTeX не приостанавливает работу. Все LaTeX'овские предупреждения начинаются с текста: «LaTeX Warning». Ниже приводятся в алфавитном порядке предупреждения, выдаваемые LaTeX'ом.

- `Citation '... ' on page ... undefined`

  Метка цитируемой работы в команде `\cite`, используемой при построении списка литературы, не определена командой `\bibitem`.

- `Label '... ' multiply defined`

  Несколько команд `\label` или `\bibitem` имеют один и тот же аргумент.

- `Label(s) may have changed. Rerun to get cross-references right`

  Числа, выводимые командами `\ref`, `\pageref` или `\cite` могут оказаться неверными. Рекомендуется выполнить еще одну компиляцию.

- `Marginpar on page ... moved`

  Сноска на полях, создаваемая командой `\marginpar`, будет сдвинута вниз во избежание вывода ее поверх предыдущей

сноски. Таким образом, данная сноска не окажется напротив той строки, на которой встретилась команда `\marginpar`.

- `Oval too small`

  Команде `\oval`, используемой в окружении `picture`, поручено нарисовать настолько маленький овал, что выполнить это не представляется возможным. Овал поэтому будет несколько больше заданного.

- `Reference '... ' on page ... undefined`

  Аргумент команды `\ref` или `\pageref`, используемых при установлении перекрестных ссылок, не был определен командой `\label`.

2.4. Предупреждающие сообщения TEX'а

TEX свои предупреждения никак не предваряет и работу не приостанавливает. Ниже приводятся предупреждения, выдаваемые TEX'ом.

- `Overfull \hbox ...`

  Текст не помещается в горизонтальный бокс. Зачастую такая ситуация возникает, когда TEX не может удачно разбить абзац на строки. Исправить ситуацию можно несколькими способами:

  1. задать дополнительные места разрешенных переносов слов;
  2. поставить принудительный или предпочтительный разрыв строк в некоторых местах;
  3. если таких предупреждений возникает слишком много, можно увеличить ширину страницы.

  В этом сообщении выводится также информация о том, насколько переполнен бокс. Если это переполнение не очень велико, сообщение можно игнорировать.

- `Overfull \vbox ...`

TEX не может найти удачного места для перехода на новую страницу, поэтому на страницу выводится текст, превышающий размер страницы. Рекомендуется использовать соответствующие команды для указания наиболее удачных мест для перехода на следующую страницу.

- `Underfull \hbox`

  Горизонтальный бокс почти или вовсе не заполнен. Такое сообщение может быть вызвано, например, использованием команды `\linebreak` в строке, заполненной менее чем наполовину.

- `Underfull \vbox`

  TEX не может найти удачного места для перехода на новую страницу, и создает страницу с недостаточным количеством текста на ней. В таком случае рекомендуется использовать соответствующие команды для указания наиболее удачных мест для перехода на следующую страницу.

3. Команды и аргументы

Команды имеют один или несколько аргументов, исключение составляют только команды ввода символов. *Обязательные* аргументы пишутся в фигурных скобках после имени команды.

Замечание: *Если обязательный аргумент пропущен, то в его качестве используется первый символ (не считая пробела), следующий за командой. Поэтому, при написании команд с аргументом из одного символа, скобки можно не писать.*

Необязательные аргументы пишутся в квадратных скобках, перед обязательными аргументами. Если аргументов несколько, то они отделяются, чаще всего, запятыми, реже — пробелами.

Некоторые команды (в основном это команды, работающие в графической моде) имеют ещё аргументы-координаты, записываемые в круглых скобках.

Команда `\underline{текст}` служит для подчёркивания *текста*.

С помощью команды `\textcircled{символ}` можно обвести окружность вокруг любого *символа*(!)

Строго говоря, команды можно разделить на *логосы*, *декларации*, собственно *команды* и *окружения*.

3.1. Логосы и декларации

> ***Логосы***, — *команды, вставляющие текст (набор символов), обычно форматированные определённым образом.*

Это в первую очередь имена разделов документа: `\partname`, `\chaptername`, `\contentsname`, `\listfigurename`, `\listtablename`, `\abstractname`, `\appendixname`, `\refname`, `\bibname`, `\indexname` а также некоторые «зарезервированные» слова: `\figurename`, `\tablename`, `\pagename`, `\seename`, `\alsoname`.

Кроме того LaTeX имеет свои логотипы:

```
\TeX, \LaTeX, \LaTeXe.
```

Многоточие — `\ldots` — тоже является логосом, как и команда текущей даты — `\today`.

Декларации ничего не печатают, а изменяют какой-либо режим (например, для изменения шрифтов, выравнивания текста, вставки отступа, ...).

Декларация `\protect` защищает следующую за ней команду, используется в подвижных аргументах (когда аргумент передаётся в другую команду, как при создании содержания, ссылки и т. п.).

3.2. Правила скобок

1. *TeX проверяет соответствие скобок { и }*, которые служат для группировки; окружения (`\begin ... \end`) тоже выполняют роль скобок (называемых командными).

2. *Декларации действуют в пределах фигурных и командных скобок*, значит декларации можно писать в аргументах команд и в теле окружений.

4. Буквы и символы

LaTeX распознаёт строчные и прописные буквы, десять цифр и 16 знаков препинания.

4.1. Специальные символы

Существует 10 символов, которые в LaTeX'е имеют особое назначение. Полный список этих символов и команд их ввода представлен в табл. 2.1.

Таблица 2.1.

Спецсимволы, требующие особого ввода

Символ	Назначение	Ввод
\	Признак команды	`$\backslash $`
{	Начало группы	`\{`
}	Конец группы	`\}`
%	Комментарий	`\%`
&	Табулятор	`\&`
~	Неразрываемый пробел	`\~{}`
$	Начало и конец математики	`\$`
^	Верхний индекс	`\^{}`
_	Нижний индекс	`\_{}`
#	Признак подстановки в макросах	`\#`

4.2. Акценты

Как правило, для того, чтобы поставить акцент на какой-нибудь букве, перед буквой печатается соответствующее **командное слово** или **командный символ** (буквы с акцентами широко используются во многих европейских языках, в русском языке — это 'ё' и й).

Большинство акцентов печатается с помощью командных символов такой же формы. Несколько из них печатается с помощью командных слов, содержащих одну букву.

Существует командное слово `\i` (для получения ı — строчной буквы i без точки над ней) и `\j` (для получения ȷ — строчной бук-

Таблица 2.2.

Акцентирование с помощью командных символов

Название	Ввод	Результат
grave	\‘o	ò
acute	\’o	ó
circumflex	\^o	ô
umlaut/dieresis/trémat	\"o	ö
tilde	\~o	õ
macron	\=o	ō
dot	\.o	ȯ

вы j без точки над ней). Эти команды позволяют поставить другой знак акцента над верхней частью соответствующей буквы.

Командные символы акцентирования приведены в табл. 2.2.

Командные слова для акцентирования приведены в табл. 2.3.

LaTeX позволяет печатать нетерминальные символы из европейских алфавитов. Эти команды приведены в табл. 2.4.

Многие из этих символов являются лигатурами латинского алфавита.

> ***Лигатýра*** *(лат. ligatura — связь), — это знак любой системы письма или фонетической транскрипции, образованный путём соединения двух и более терминальных (клавиатурных) символов.*

По не слившимся частям (иногда изменяющим свою форму) обычно можно видеть, какие буквы входят в состав лигатуры. В некоторых системах письма лигатуры многих буквенных сочетаний вошли во всеобщее употребление, являясь как бы сложными буквенными знаками для изображения определённых звуковых комплексов.

В ИС LaTeX для ввода некоторых часто используемых нетерминальных символов используются **лигатуры**, вводимые терминальными символами. В основном, это тире и кавычки. Некоторые

Таблица 2.3.

Акцентирование с помощью очень коротких командных слов

Название	Ввод	Результат
cedilla	`\c{o}`	o̧
underdot	`\d{o}`	ọ
underbar	`\b{o}`	o̱
háček	`\v{o}`	ǒ
breve	`\u{o}`	ŏ
tie	`\t{oo}`	o͡o
Hungarian umlaut	`\H{o}`	ő

Таблица 2.4.

Иностранные нетерминальные символы

Пример	Ввод	Результат
Ægean, æsthetics	`\AE, \ae`	Æ, æ
Œuvres, hors d'œuvre	`\OE, \oe`	Œ, œ
Ångstrom	`\AA, \aa`	Å, å
Øre, København	`\O, \o`	Ø, ø
Łodz, łódka	`\L, \l`	Ł, ł
Nuß	`\ss`	ß
?‘Si?	`?‘`	?‘
!‘Si!	`!‘`	!‘

сочетания символов латиницы тоже образуют лигатуры (когда вместо нескольких вводимых символов получается один типографский символ).

Например:

■ ff ff fi fi

```
ff f{f}     fi f{i}
```

4.3. Тире, кавычки, многоточия

Используется четыре типа тире:

дефис — в сложных словах;

интервал — для обозначения числовых интервалов;

тире — как грамматический знак в предложениях;

минус — используемый с отрицательными числами.

Способы ввода этих символов представлен в табл. 2.5.

Таблица 2.5.

Различные типы тире

Название	Ввод	Результат
hyphen	`-`	-
en-dash	`--`	–
em-dash	`---`	—
minus sign	`$-$`	$-$

Тире различной длины вводятся лигатурами.

В тексте часто используются кавычки и многоточия. Способы ввода этих символов представлен в табл. 2.6. Как и тире, эти символы вводятся, в основном, **лигатурами**.

В русскоязычном (кириллическом) тексте принято использовать французские кавычки («. . . »).

Таблица 2.6.

Кавычки и многоточие

Название	Ввод	Результат
левые кавычки	`` `	‘
правые кавычки	`` '	’
левые двойные кавычки	`` ``	“
правые двойные кавычки	`` ''	”
левые французские кавычки[1]	`<<|«|<|\<`	«
правые французские кавычки[1]	`>>|»|>|\>`	»
многоточие	`\ldots`	...

[1]Знак ‘|’ используется здесь в качестве союза «или», а способ ввода зависит от конкретной русификации.

5. Шрифты

Для *выделения текста* используется команда `\emph{текст}`.

5.1. Пользовательские команды выбора шрифтов

В большинстве случаев пользователю достаточно небольшого набора шрифтов, отличающихся разными характеристиками. Команды для выбора таких шрифтов представлены в табл. 2.7.

5.2. Пользовательские команды изменения размера шрифтов

Кроме шрифта, иногда возникает необходимость изменить размер шрифта. LaTeX в своём арсенале имеет набор пользовательских команд для изменения размера шрифтов, они представлены в табл. 2.8.

Таблица 2.7.

Пользовательские команды выбора шрифтов

Название шрифта	**Команда**	**Декларация**	**Образец печати**
Прямой	`\textrm{текст}`	`\rmfamily` | `\rm`	roman
Средний	`\textmd{текст}`	`\mdseries`	medium
Жирный	`\textbf{текст}`	`\bfserise` | `\bf`	**boldface**
Курсив	`\textit{текст}`	`\itshape` | `\it`	*italic*
Наклонный	`\textsl{текст}`	`\slshape` | `\sl`	*slanted*
Без отточий	`\textsf{текст}`	`\sffamily` | `\sf`	sans serifed
Строчные, как прописные	`\textsc{текст}`	`\scshape` | `\sc`	Small Caps
Равноширинный	`\texttt{текст}`	`\ttfamily` | `\tt`	`type-writer`

Таблица 2.8.

Пользовательские команды задания размера шрифтов

Команда	**Образец**
`\tiny`	Sample text.
`\scriptsize`	Sample text.
`\footnotesize`	Sample text.
`\small`	Sample text.
`\normalsize`	Sample text.
`\large`	Sample text.
`\Large`	Sample text.
`\LARGE`	Sample text.
`\huge`	Sample text.
`\Huge`	Sample text.

6. Низкоуровневые команды задания шрифтов

После создания **NFSS** (New Font Selection Schem), называемой ещё «ортогональной схемой выбора шрифтов», стало возможным задание ещё большего многообразия шрифтов.

Каждый шрифт характеризуется определённым набором параметров: **кодировка**, **гарнитура** (**семейство**), **насыщенность**, **начертание** и **размер**.

Теперь каждый параметр можно изменять независимо от других, чего нельзя было делать в LaTeX 2.09.

6.1. Кодировка

В TeX'е введена своя кодировка, называемая *внутренней* (общепринятая кодировка или кодовая страница, в таком случае является *внешней*).

Первая кодировка (TeX text encoding), введённая Д. Кнутом, в LaTeX 2_ε называется **OT1** (Old Text 1) и содержит всего 128 символов.

В 1990 г., на конференции пользователей TeX'а была принята кодировка TeX extended text encoding, называемая ещё «корковской»[1]. Она содержит 256 символов, в неё включены все символы алфавитов романской группы (западно-европейских стран).

Задаётся кодировка с помощью команды `\fontencoding{enc}`. Список допустимых кодировок *enc* представлен в табл. 2.9.

Например:

```
\fontencoding{OT1}
```

Для русских шрифтов часто используется кодировка LCY или LH. Но местные (локальные) кодировки LaTeX не знает. Lxx кодировки следует подключать через пакет fontenc[2], например:

[1] Название происходит от места проведения конференции, — ирландского городка Корк (Cork)

[2] Пакеты русификации уже содержат такие команды, так-что пользователю, обычно, не надо задумываться о кодировке.

Таблица 2.9.

Стандартные внутренние кодировки LaTeX 2_ε

enc	**Кодировка**	**Примечание**
OT1	TeX text encoding	Кодировка Д. Кнута
T1	TeX extended text encoding	«Корковская» кодировка
U	Unknown	Неизвестная кодировка
Lxx	A Local encoding	Местная кодировка

`\usepakage[LCY]{fontenc}`.

Старый способ задания русских шрифтов, основывался на кодировке OT1, дополненной 128 символами, содержащими символы русского алфавита.

В современных версиях используются официальные русские шрифты T2.

6.2. Непосредственное задание символа

Команда `\symbol{код}` вставляет в текст символ с кодом «*код*» внутренней кодовой таблицы. Код можно задать десятичным числом, восьмеричным (начинающемся с апострофа `'`) или шестнадцатеричным (начинающемся с кавычек `"`).

Такой способ задания символов обычно используется в определении новых команд.

6.3. Гарнитура

> ***Гарнитура (семейство шрифтов)*** *определяет дизайнерское решение, общее для разных шрифтов.*

В группе шрифтов Computer Modern, созданных Д. Кнутом, есть три гарнитуры: Roman, Serif и `Typewriter`. Основные отличительные черты гарнитуры, это *контрастность*, *пропорциональность* и *засечки*.

Контрастность *определяется отношением толщины вертикальных и горизонтальных линий символов.*

Например:

■ Гарнитура Computer Modern Roman более контрастная, чем Computer Modern Serif.

Пропорциональность *задаётся отношением ширины к высоте буквы 'M'.*

Пропорциональность тесно связана с шириной букв. В пропорциональных шрифтах занимаемое буквой место пропорционально действительной ширине буквы, как например, в Computer Modern Roman.

Кроме того, шрифт может быть **моноширинный** (когда каждая буква занимает одинаковое по ширине место). Пример **моноширинного** шрифта — `Computer Modern Typewriter`.

Засечки *располагают на вертикальных линиях букв, такие шрифты ещё называют* ***романскими****.*

Шрифты без засечек называют ***рублёными****.*

Например:

■ Гарнитура Computer Modern Roman имеет засечки, что отображено в названии, а Computer Modern Serif является рублёным шрифтом.

Задаётся гарнитура командой `\fontfamily{family}`.

Основные семейства кириллических шрифтов в LaTeX 2_ε приведены в табл. 2.10.

6.4. Насыщенность

Насыщенность — *относительная толщина линий букв.*

В LaTeX 2_ε насыщенность задаётся с помощью команды `\fontseries{series}`. Как видно из табл. 2.11, некоторые серии задают не только насыщенность но и пропорциональность.

Таблица 2.10.

Наиболее распространённые семейства шрифтов в LaTeX 2ε

family	**Гарнитура**	**Примечание**
cmr	Computer Modern Roman	Романский
cmss	Computer Modern Sans	Рублёный
cmtt	Computer Modern Typewriter	Машинописный
cmdh	Computer Modern Dunhill	Данхил
cmfib	**Computer Modern Fibonacci**	**Фиббоначи**
cmfr	Computer Modern Funny	Забавный

Таблица 2.11.

Стандартные значения параметра *series*

series	**Насыщенность**	**Примечание**
l	Light	Светлый шрифт
m	Medium	Нормальный шрифт
b	**Bold**	**Жирный шрифт**
bx	**Bold extended**	**Широкий жирный шрифт**
sb	Semi-bold	Полужирный шрифт
c	Condenced	Узкий шрифт

6.5. Начертание

***Начертание** определяет форму шрифта.*

Задаётся начертание командой `\fontshape{shape}`. Основные начертания LaTeX 2ε приведены в табл. 2.12.

Таблица 2.12.

Стандартные значения параметра *shape*

shape	**Начертание**	**Примечание**
n	Normal	Прямой шрифт
it	*Italic*	*Курсивный шрифт*
sl	*Slanted (oblique)*	*Наклонный шрифт*
sc	Caps and small caps	Капитель
ui	Upright italic	Прямой курсивный шрифт

6.6. Кегль

Кегль или размер шрифта измеряют в печатных пунктах (pt). В одном дюйме содержится 72,27 pt.

Задать размер шрифта можно с помощью команды `\fontsize{size}{baselineskip}`. Эта команда, кроме размера *size*, содержит ещё один параметр — *baselineskip*, который определяет расстояние между строками.

Значения параметров могут быть любыми, в произвольных единицах измерения (например, `\fontsize{15pt}{5mm}`), однако, делать это надо осторожно.

В профессиональных издательских системах используются кегли 7, 8, 9, 10, 11, 12, 14, 16, 18, 20, 24, 30, 36 pt.

Шрифты **Computer Modern** спроектированы для размеров 5, 6, 7, 8, 9, 10, 12, 17 pt, другие размеры получаются линейным масштабированием.

В большинстве случаев можно обойтись пользовательскими командами задания размера шрифтов.

Пользовательские декларации переключения размера шрифтов задают линейку кеглей 5, 6, 7, 8, 9, 10, 10.95, 12, 14.4, 17.28, 20.74, 24.88 pt.

6.7. Переключение шрифтов

Любое изменение шрифта вступает в силу только после декларации `\selectfont`, которая должна следовать сразу же после изменения параметров шрифтов.

Существует альтернативный вариант задания шрифта, — с помощью команды `\usefont{enc}{family}{series}{shape}`, задающей четыре параметра (все, кроме размера).

Все эти команды действуют только в текстовой *и* строковой *модах.*

В *математической* моде существуют свои, математические шрифты, которые определяются своими командами.

Графические символы, используемые в *графической* моде, задаются специальными кодами.

Лабораторная работа №2
Простой текст

1. Создайте новый файл (обязательно с расширением “tex”). Напишите в нём преамбулу:

   ```
   \documentclass{article}
   ```

   Напишите в теле документа фразу «Всё началось с TeX’a» (слово TeX следует набирать латинскими буквами). Откомпилируйте файл и посмотрите результат.

2. Измените преамбулу:

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

   Откомпилируйте файл и посмотрите результат. Обратите внимание на символ, соответствующий букве ‘ё’.

3. Измените преамбулу:

   ```
   \documentclass{article}
   \usepackage[cp1251]{inputenc}
   \usepackage[russian]{babel}
   ```

   Откомпилируйте файл и посмотрите результат.

4. Прочитайте `.log` файл, соответствующий исходному файлу, и напишите (после команды `\end{document}`):

   4.1. какой компилятор используется;

   4.2. класс документа и его версию;

   4.3. файлы, загружаемые пакетом Babel и их авторов, (если они указаны).

5. Сделайте умышленную ошибку в слове `begin` команды `\begin{document}`. На какой строке при компиляции будет указана ошибка; к каким ошибкам она относится (ошибкам TeXa или LaTeXa)?

6. Сделайте умышленную ошибку в слове `document` команды `\begin{document}`. На какой строке при компиляции будет указана ошибка; к каким ошибкам она относится?

7. Используя команды задания акцентов и символов, пользовательские команды выбора шрифтов, наберите следующий текст.

Введение

Всё началось с TEX'а (в русской транскрипции произносится «тех», по словам автора, это название взято из греческого) — программа форматирования и вёрстки текстов, написанная Дональдом Кнутом (*Donald E. Knuth*), известным математиком и программистом. Всемирную известность он приобрёл после публикации его монографии **«Искусство программирования»**. В начале 80-х Лесли Лампортом (*Leslie Lamport*) была разработана издательская система на базе TEXа, названная им LATEX...

...“Теховский” файл представляет из себя текстовый файл, включающий в себя, кроме обычного текста, набор управляющих команд, все они начинаются со знака ‘\’. Кроме этого знака существует ещё 9 служебных символов: ‘{’, ‘}’, ‘%’, ‘&’, ‘$’, ‘#’, ‘~’, ‘^’, ‘_’.

Кроме обычных символов можно использовать и иностранные, экзотические символы, например: Schrödinger; Čerenkov; Genève; soupçon; naïve; l'électron; Å.

ⓓ 15 мая 2021 г.

ⓒ *О. В. Журенков*

Дополнительно

- Напишите ниже свою фамилию (на английском) шрифтом «Данхил» (“Dunhill”).
- Поставьте рядом свой знак зодиака, используя шрифты wasy (пакет wasysym).
- Напишите ниже свою фамилию широким жирным прямым курсивом размером 16 pt.
- Прочитайте .log файл и напишите:

1. формат компилятора (LaTeX2.09 или LaTeX2e) и в какой строке он указан;
2. опции класса документа и их версии;
3. что делает пакет lh;
4. имена файлов (.fd), определяющих русские шрифты;
5. тип кодировки wasy.

Глава 3

Вёрстка и форматирование

1. Управление строками

> ***Вёрстка*** *— размещение контента, монтаж полос оригинал-макета из составных элементов (текста, заголовков, таблиц, иллюстраций и пр.).*

Под **вёрсткой** понимают также результат этого процесса, то есть готовые полосы.

Строки в LaTeX'е формируются автоматически, но встречаются ситуации, когда требуется «ручное вмешательство».

1.1. Горизонтальные пробелы

Для формирования красивого, более удобочитаемого текста, TeX использует технологии **кернения** и **трекинга**.

> ***Кернение*** *— изменение расстояния между определённой парой символов.*

Например:

■

Где лучше? — без кернения
Где лучше? — с кернением

***Трекинг** — изменение расстояния между символами в зависимости от кегля (размера шрифта).*

Команды управления пробелами в тексте представлены в табл. 3.1.

Таблица 3.1.

Пробелы в LaTeX 2ε

`\␣`	пробел нормального размера	_
`\,`	маленький пробел	_
`~`	неразрывный пробел	_
`\/`	корректирующий пробел	
`\quad`	пробел, шириной с букву ‘M’	__
`\qquad`	пробел, шириной как две ‘M’	____

`\hspace{длина}` — вставляет пробел заданной *длины*.

`\hspace*{длина}` — вставляет пробел даже на краю строки.

Единицы измерения, используемые в TeX’е, представлены в табл. 3.2.

Таблица 3.2.

Единицы измерения в TeXe

`mm`	миллиметр ≈ 1/25 дюйма	
`cm`	сантиметр = 10 mm	
`in`	inch = 25,4 mm	
`pt`	пункт ≈ 1/72 дюйма ≈ $\frac{1}{3}$ mm	
`em`	примерная ширина буквы ‘M’ текущего шрифта	
`ex`	примерная высота буквы ‘x’ текущего шрифта	

`\hfil` — вставляет пробел длины `fil`.

`\hfill` — вставляет пробел длины `fill`.

`\@` — увеличивает пробел в конце предложения.

`\frenchspacing` — декларирует подавление дополнительного пробела в конце предложения.

`\nofrenchspacing` — декларирует дополнительный пробел в конце предложения, действует по умолчанию.

`\fussy` — декларирует «плотные строки».

`\sloppy` — декларирует «разрежённые строки», этой декларации соответствует окружение `sloppypar`.

`\emergencystretch` — длина, которая задаёт степень разрежённости строк (изменяется в преамбуле). Например:

■ `\emergencystretch=1cm`.

1.2. Переносы

Запретить переносы можно с помощью строкового бокса `\mbox{`*текст*`}` (см. на стр. 4.1.).

Подсказать правильные переносы можно с помощью декларации `\-`, вставленной в месте предполагаемого переноса. Например:

■ `ку\-соч\-но-не\-пре\-рыв\-ная функция`.

В преамбуле указываются правила переноса для часто используемых слов с помощью команды `\hyphenation{`*сло-во сло-во ...*`}`.

Пакет babel поддерживает около 200 языков.

1.3. Разрыв строки

Повлиять на разрыв строк можно с помощью следующих команд:

- `\linebreak[`*N*`]` — поощряет с «силой» N ($N = 0 \div 4$) разрыв строки, причём текущая строка выравнивается по правому

краю $N = 0$ соответствует отсутствию команды, а $N = 4$ — обязательному разрыву (действует по умолчанию).

- `\nolinebreak[N]` — препятствует с «силой» N разрыву строки $N = 0$ соответствует отсутствию команды, а $N = 4$ — запрету разрыва (действует по умолчанию).

- `\\[высота]` — начинает новую строку, не завершая абзац. Необязательный аргумент *высота* задаёт дополнительный вертикальный пробел, вставляемый перед новой строкой. Модификация этой команды (`\\*[высота]`) запрещает перенос новой строки на следующую страницу.

- `\newline` — действует аналогично `\\`.

При формировании строк могут появляться предупреждения TEX'а:

- "`Underfull \hbox (badness ...`" — незаполненный горизонтальный бокс (в скобках указана относительная величина разрежённости);

- "`Overfull \hbox (...`" — переполненный горизонтальный бокс (в скобках указана длина переполнения в пунктах).

1.4. Абзацы

Абзацы при наборе разделяются пустыми строками. В некоторых случаях удобнее пользоваться командой `\par`.

`\indent` — вставляет абзацный отступ. Эта команда равносильна `\hspace{\parindent}`, где `\parindent` — длина абзацного отступа.

`\noindent` — подавляет абзацный отступ (ставится в начале абзаца).

2. Управление страницами

Страницы в LaTeX'е, как и строки, формируются автоматически, при этом также встречаются ситуации, когда требуется «ручное вмешательство».

2.1. Вертикальные пробелы

`\vspace{высота}` — вставляет между строками пробел заданной *высоты*.

`\vspace*{высота}` — вставляет пробел заданной *высоты* даже на краю страницы.

Для длин `fil` и `fill` имеются аббревиатуры: `\vfil` и `\vfill`.

`\addvspace{высота}` — добавляет отступ, необходимый для достижения вертикального пробела между абзацами высотой *высота*. *Применяется только в текстовой моде и пишется* обязательно на отдельной строке.

Для увеличения вертикального отступа между абзацам существуют команды:

- `\smallskip` — малый отступ (высотой `\smallskipamount`).
- `\medskip` — отступ высотой `\medskipamount` (средний, в 2 раза больше малого).
- `\bigskip` — отступ высотой `\bigskipamount` (большой, в 2 раза больше среднего).

`\baselinestretch` — определяет расстояние между строками (как в команде `\fontsize`), *пишется в преамбуле.*

Например:

■ Например, для задания интервала 1,5 надо написать `\renewcommand{\baselinestretch}{1.5}`.

2.2. Страницы

`\flushbottom` — включает режим, когда все страницы выровнены по нижнему краю (действует по умолчанию при включении опции класса `twoside`).

`\raggedbottom` — включает режим, когда страницы не выровнены.

При формировании страниц могут появляться предупреждения компилятора:

- "`Underfull \vbox (badness ...`" — незаполненный вертикальный бокс (в скобках указана относительная величина разрежённости);
- "`Overfull \vbox (...`" — переполненный вертикальный бокс (в скобках указана высота переполнения в пунктах).

Повлиять на разбиение страниц можно с помощью следующих команд:

- `\nopagebreak[N]` — препятствует с «силой» N разрыву страницы($N = 0 \div 4$). $N = 0$ соответствует отсутствию команды, а $N = 4$ — запрещает разрыв (действует по умолчанию).
- `\samepage` — запрещает разрыв страницы, так же как и `\nopagebreak`.
- `\pagebreak[N]` — поощряет с «силой» N разрыв страницы $N = 0$ соответствует отсутствию команды, а $N = 4$ — обязательному разрыву (действует по умолчанию).
- `\newpage` — начинает новую страницу, так же как и `\pagebreak` (при двухколоночной печати эти команды начинают новую колонку). *При этом текущая страница получается укороченной, даже если включена декларация `\flushbottom`.*
- `\clearpage` — выводит на печать оставшиеся **плавающие объекты** и начинает новую страницу.

- `\cleardoublepage` — действует как и команда `\clearpage`, но при двусторонней печати, при необходимости, добавляет пустую левую страницу.

- `\enlargethispage{`*высота*`}` — увеличивает высоту страницы на величину *высота*.

  Модификация команды (`\enlargethispage*{`*высота*`}`) при этом сжимает все сжимаемые пробелы.

Лабораторная работа №3

Строки, абзацы, страницы

Создайте новый файл и наберите следующий текст.

TEX автоматически вставляет пробелы между словами растяжимой длины. Кроме того, в арсенале LATEXа имеется большой набор команд, задающих различные пробелы:

А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе
А и Б сидели на трубе

TEX имеет совершенный алгоритм разбиения слов на слоги, способный найти 80–90% правильных переносов. В некоторых случаях требуется запрещать переносы (для этого можно использовать неразрывный пробел или строковый бокс), а в некоторых наоборот, подсказать правильные переносы (hyphenation).

Абзацы могут не иметь абзацный отступ.

При желании можно поставить двойной отступ.

Расстояние между абзацами можно увеличить вот так.

Или так.

Или так.

Здесь между абзацами вставлен вертикальный пробел 1 см.

А здесь расстояние между абзацами увеличено до 1 см.
Можно вставить пробел длиной fil.

Можно вставить пробел длиной fill.
При необходимости можно увеличить длину страницы.
Можно в любом месте начать страницу или запретить разрыв страниц. При этом нижний край может быть поджат или выровнен по нижней границе.

1

Верх
Середина
Низ

2

Верх

Середина

Низ

3

3. Форматирование текста

***Форматирование** (текста) — процесс придания тексту определённого вида.*

В этом деле LaTeX использует логическую разметку. Физическую разметку, через задание параметров шрифтов, мы рассмотрели ранее, см. 2.7 раздел.

3.1. Выравнивание

По умолчанию, текст выравнивается TeX'ом по ширине. Другие варианты выравнивания представлены в табл. 3.3.

Таблица 3.3.

Команды выравнивания текста

	Окружение	Декларация	Команда
по центру	`center`	\centering	\centerline{*текст*}
влево	`flushleft`	\raggedright	\leftline{*текст*}
вправо	`flushright`	\raggedleft	\rightline{*текст*}

Команды выравнивания используются для коротких строк, в них текст не переносится.

3.2. Цитаты и стихи

Для выделения абзацев равными отступами слева и справа существует два окружения:

- `quote` — используется для небольших цитат (фрагментов текста);
- `quotation` — используется для больших фрагментов текста, состоящих из нескольких абзацев.

Для форматирования стихов используется окружение `verse`. Каждая строка должна заканчиваться командой `\\`, а строфы отделяться пустыми строками.

3.3. Списки

LaTeX имеет в своём арсенале три вида списков, которые задаются соответствующими окружениями:

1. `enumerate` — нумерованный список;
2. `itemize` — маркированный (ненумерованный) список;
3. `description` — список-описание.

Каждый элемент списка начинается с команды `\item[метка]`, необязательный аргумент *метка* можно использовать для задания номера или маркера (для первых двух списков). *В третьем списке этот аргумент* обязателен, *в нём задаётся описываемое слово.*

3.4. Неформатированный текст

Для изображения текста без форматирования используется окружение `verbatim`. Модификация `verbatim*`, отображает пробелы символом ␣.

Для отображения небольшой части неформатированного текста определена команда `\verb<c>{текст}<c>`, в качестве `<c>` можно использовать любой, отсутствующий в тексте, символ.

Команда `\verb*<c>{текст}<c>` так-же отображает пробелы.

Например:

■
```
\verb|\LaTeX   <-- logos|  \verb*|\LaTeX   <-- logos|
\LaTeX   <-- logos         \LaTeX␣␣␣<--␣logos
```

3.5. Сноски и заметки

Текст, который пишется внизу страницы, называется ***сноской***[1].

Сноски формируются командой `\footnote[N]{текст}`, она ставится *сразу же после слова, без пробела.* Сноски нумеруются автоматически. С помощью N можно изменить нумерацию.

`\footnotemark[N]` — ставит только маркер, увеличивая значение своего счётчика.

`\footnotext[N]{текст}` — пишет *текст* подстрочного примечания.

Кроме сносок в тексте можно использовать **заметки**, которые пишутся на полях, напротив нужного места в тексте. Для **заметок** используется команда `\marginpar[текст слева]{текст}`.

Необязательный аргумент *текст слева* используется при двусторонней печати, если *текст* заметки должен быть разным при появлении заметки на левой и правой странице.

Декларация `\reversmarginpar` меняет текст левых и правых заметок, а `\normalmarginpar` восстанавливает нормальный режим.

4. Боксы

Бокс *— это прямоугольная область, с которой TEX оперирует как с единым, неделимым объектом.*

Параметры **бокса** задаются размерными величинами *width*, *depth*, *height*, *totalheight*.

4.1. Строковые боксы

Внутри строкового бокса TEX работает в *строковой моде*, т. е. *текст не может быть разорван.*

`\mbox{текст}` — помещает аргумент *текст* в бокс, размеры бокса чуть больше фактических размеров аргумента.

[1] Синоним — подстрочные примечания.

Рис. 3.1. Бокс и его параметры

`\fbox{текст}` — создаёт такой-же бокс как и команда `\mbox`, но ещё и окружает бокс рамкой.

`\makebox[ширина][позиц.]{текст}` — помещает аргумент *текст* в бокс, размеры которого можно задать с помощью аргумента *ширина*. Если задан аргумент *ширина*, то можно задействовать аргумент *позиц.*, задающий позиционирование содержимого бокса.

Аргумента *позиц.* может принимать следующие значения:

`l` — текст сдвинут к левому краю бокса;

`c` — текст расположен по центру бокса;

`r` — текст сдвинут к правому краю бокса;

`s` — текст растянут по всему боксу.

По умолчанию действует опция `c`.

`\framebox[ширина][позиц.]{текст}` — создаёт такой-же бокс как и команда `\makebox`, но ещё и окружает бокс рамкой.

`\raisebox{высота}[height][depth]{текст}` — изменяет положение текста относительно базисной линии, — поднимает на высоту *высота* (или опускает, если *высота* отрицательная). Необязательные аргументы *height* и *depth* изменяют естественные значения этих параметров.

4.2. Текстовые боксы

Внутри текстового бокса TEX работает в *текстовой моде.*

`\parbox[внеш.позиц.][высота][позиц.]{ширина}{текст}`

создаёт бокс шириной *ширина*, форматирующий *текст* в виде одного абзаца. Аргумент *внеш.позиц.* задаёт положение бокса относительно текущей базисной линии:

`t` — по текущей базисной линии выравнивается базисная линия *верхней строки*;

`c` — по текущей базисной линии выравнивается *центр бокса*;

`b` — по текущей базисной линии выравнивается базисная линия *нижней строки.*

По умолчанию действует опция `c`.

Аргумент *высота* изменяет высоту бокса.

Вертикальное позиционирование текста внутри бокса задаётся аргументом *позиц.*:

`t` — текст сдвинут к верхнему краю бокса;

`c` — текст расположен по центру бокса;

`b` — текст сдвинут к нижнему краю бокса;

`s` — текст растянут по всему боксу за счёт растяжимых вертикальных пробелов (если таковые имеются).

По умолчанию действует опция, указанная в аргументе *внеш.позиц.*

Окружение `minipage[внеш.позиц.][высота][позиц.]{ширина}` создаёт текстовый бокс шириной *ширина* со всеми возможностями обычного текста.

Аргумент *внеш.позиц.* так-же задаёт положение бокса относительно текущей базисной линии.

С помощью аргумента *высота* можно изменить высоту бокса, а с помощью аргумента *позиц.* так-же задать и вертикальное позиционирование текста внутри бокса.

4.3. Линейные боксы

С помощью линейных боксов можно создавать декоративные или невидимые линейки.

`\rule[смещение]{ширина}{высота}` — создаёт линейный бокс с размерами *ширина* и *высота*. С помощью аргумента *смещение* задаётся смещение по вертикали относительно базисной линии.

`\hrulefill` — создаёт тонкую горизонтальную линию длиной fill.

`\dotfill` — создаёт пробел fill, заполненный точками . . . . . .

`\strut` — создаёт бокс нулевой ширины («распорку»), высота и глубина которого максимальны для букв текущего шрифта.

Например:

■

```
\underline{без} \underline{распорки} \qquad
\underline{с\strut} \underline{распоркой\strut}
```

без распорки с распоркой

В математической моде для вставки «распорки» имеется аналогичная команда `\mathstrut`.

4.4. Сохранение бокса

Кроме рассмотренных команд в LaTeX'е существует ряд команд для измерения размеров боксов, для определения, сохранения и использования новых боксов.

Повторяющийся в документе сложный текст (рисунки, таблицы, трудоёмкое форматирование) лучше отформатировать один раз, «запомнить» его в виде бокса и использовать в уже «готовом» виде. Именно таким образом задаются логосы.

Для работы с такими боксами имеются следующие команды:

`\newsavebox{\имя}` — задаёт *имя* бокса.

`\sbox{\имя}{текст}` — форматирует *текст* в строковой моде (действует как `\mbox`, но не выводит на печать содержимое бокса).

Окружение `lrbox{\имя}` действует аналогично команде `\sbox`, с тем отличием, что все пробелы в начале и конце текста будут проигнорированы. Кроме того, в тексте могут быть неформатированные участки (созданные командой `\verb` или окружением

verbatim).

`\savebox[ширина][позиц.]{\имя}{текст}` — форматирует *текст* в строковой моде. Можно задать *ширину* бокса и *позиционирование текста* (действует как `\makebox`, но не выводит на печать содержимое бокса). Опция *позиц.*, может принимать те же значения:

`l` — текст сдвинут к левому краю бокса;

`c` — текст расположен по центру бокса;

`r` — текст сдвинут к правому краю бокса;

`s` — текст растянут по всему боксу.

По умолчанию действует опция `c`.

`\usebox{\имя}` — печатает бокс *имя*.

Например:

■

```
\newsavebox{\Help}
\savebox{\Help}[2em]{\fbox{\texttt{F1}}}
Клавиша \usebox{\Help} --- помощь.
```

Клавиша $\boxed{\texttt{F1}}$ — помощь.

5. Команды в LaTeX 2_ε

5.1. Командные длины

Командные длины — команды, выражающие определённые длины.

Для работы с произвольными командными длинами существуют следующие команды:

- `\newlength{\имя}` — задание новой командной длины.
- `\setlength{\имя}{длина plus длина minus длина}` — установка длины команды *имя*.

- `\addtolength{\имя}{длина plus длина minus длина}` — увеличение длины команды *имя*.

Команды для измерения строковых боксов:

- `\settowidth{\длина}{текст}` — измеряет ширину строкового бокса, образованного аргументом *текст*. Измеренная длина присваивается аргументу *длина*.

- `\settoheight{\длина}{текст}` — измеряет высоту строкового бокса, образованного аргументом *текст*.

- `\settodepth{\длина}{текст}` — измеряет глубину строкового бокса, образованного аргументом *текст*.

Например:

■

```
\newlength{\bx}
\settoheight{\bx}{\fbox{F8}}
\addtolength{\bx}{2\fboxsep plus 2pt}

Клавиша \framebox[\bx]{F1}
--- помощь.
```

Клавиша $\boxed{\text{F1}}$ — помощь.

5.2. Счётчики

Основные счётчики LaTeX'а представлены в табл. 3.4.

Внутренние счётчики подчинены (подчинённые счётчики) своим внешним счётчикам и автоматически сбрасываются при изменении внешних счётчиков.

Например:

■ Счётчик `subsection` является внешним для счётчика `subsubsection` и внутренним для `section`.

Таблица 3.4.

Основные счётчики LaTeX'а

Имя	Что нумерует
page	страницы
part	части
chapter	главы
section	разделы
subsection	подразделы
subsubsection	подподразделы
paragraph	параграфы
subparagraph	подпараграфы
enumi	элементы списка 1-го уровня
enumii	элементы списка 2-го уровня
enumiii	элементы списка 3-го уровня
enumiv	элементы списка 4-го уровня
footnote	подстрочные примечания
mpfootnote	подстрочные примечания в министраницах
equation	уравнения
table	таблицы
figure	рисунки

Большинство счётчиков устанавливаются в нуль в начале компиляции. Команды и окружения, работающие со счётчиками, сначала увеличивают значение счётчика, а потом уже используют его.

Для работы непосредственно со счётчиками существуют команды:

`\the`*счётчик* — печатает значение *счётчика* в формате, определённом для этого счётчика.

Всего имеется 6 форматов: *arabic*, *roman*, *Roman*, *alph*, *Alph*, *fnsymbol*. Им соответствуют следующие команды:

`\arabic{`*счётчик*`}` — печатает значение *счётчика* арабскими цифрами;

`\roman{`*счётчик*`}` — печатает значение *счётчика* строчными римскими цифрами;

`\Roman{счётчик}` — печатает значение *счётчика* прописными римскими цифрами;

`\alph{счётчик}` — печатает значение *счётчика* строчными буквами английского алфавита (значение счётчика $\leqslant 26$);

`\Alph{счётчик}` — печатает значение *счётчика* прописными буквами английского алфавита (значение счётчика $\leqslant 26$);

`\fnsymbol{счётчик}` — печатает значение *счётчика* специальными символами, часто используемыми для подстрочных примечаний в `minipage` (значение счётчика $\leqslant 9$).

`\setcounter{счётчик}{N}` — задаёт значение *счётчика* равное N (N — целое, возможно отрицательное).

`\addtocounter{счётчик}{N}` — увеличивает значение *счётчика* на N.

`\value{счётчик}` — возвращает «чистое» значение счётчика. Может использоваться в любых командах, где LaTeX ожидает число, в основном — в аргументе N команд `\setcounter` и `\addtocounter`. Эта команда устойчивая, её никогда не следует защищать командой `\protect`.

`\stepcounter{счётчик}` — наращивает (увеличивает на 1) значение *счётчика*, при этом все внутренние счётчики сбрасываются (устанавливаются в начальное значение).

`\refstepcounter{счётчик}` — выполняет тоже, что и предыдущая команда, кроме того, ещё декларирует, что текст, генерируемый командой `\the`*счётчик*, будет использован в месте употребления команды перекрёстного цитирования `\ref`.

`\newcounter{имя}[счётчик]` задаёт счётчик с именем *имя*. Начальное значение счётчика равно 0, а формат `\the`*имя* — *arabic*. Необязательный аргумент задаёт внешний *счётчик*. При этом,

при наращивании значения этого внешнего *счётчика* (командами stepcounter или refstepcounter) все подчинённые счётчики (в т. ч. и новый счётчик *имя*) сбрасываются в нуль.

Замечание: *эта команда не может присутствовать в файлах, включаемых командой* `\include`!

Например:

■

```
\thepage\ \Roman{page}
```

71 LXXI

```
\thesubsection
\setcounter{subsection}{\value{page}}
\thesubsection
\stepcounter{section} \thesubsection
```

5.2. 5.71. 6.0.

```
\fnsymbol{mpfootnote} \arabic{mpfootnote}
\stepcounter{mpfootnote} \fnsymbol{mpfootnote}
\setcounter{mpfootnote}{9} \fnsymbol{mpfootnote}
```

0 * ‡‡

6.1. Определение новых команд

`\newcommand{\имя}[N пар.][знач.]{определение}` — определение новой команды *имя*. Здесь *N пар.* — число параметров (аргументов), *знач.* — значение по умолчанию. В *определении* в месте появления параметра пишется знак # и порядковый номер параметра.

`\renewcommand{\имя}[N пар.][знач.]{определ.}` — переопределение существующей команды.

`\providecommand{\имя}[N пар.][знач.]{определ.}` — определение новой команды; если такая команда существует, то сохраняется её предыдущее значение.

Например:

■

```
\providecommand{\No}{\textnumero}
\renewcommand{\No}{\textnumero}

Правило \No1: Всякая команда заканчивается
небуквенным символом.
```

Правило №1: Всякая команда заканчивается небуквенным символом.

6.2. Определение новых окружений

`\newenvironment{имя}[N пар.][знач.]{нач.}{кон.}` — определение нового окружения. *Нач.* и *кон.* — команды, выполняемые в начале и в конце окружения.

`\renewenvironment{имя}[N пар.][знач.]{нач.}{кон.}` — переопределение существующего окружения.

Например:

■

```
\newenvironment{Squote}[1][\checkmark]{%
 \raggedright\rightmargin=5em\begin{itemize}
 \item[#1]\rmfamily\fontshape{ui}\selectfont}
 {\end{itemize}}

\begin{Squote}
Издательская система логической вёрстки
\end{Squote}
\begin{Squote}[\copyright]
\LaTeXe
\end{Squote}
```

✓ *Издательская система логической вёрстки*

© LaTeX 2_ε

6.3. Определение команд типа «теорема»

`\newtheorem{имя}{Заголовок}[счётчик]` — определяет новую теорему *имя*. Необязательный аргумент задаёт внешний *счётчик*.

`\newtheorem{имя}[теорема]{Заголовок}` — определение новой теоремы по прототипу (определяемая теорема будет нумероваться так же как и *теорема*-прототип).

Теоремы применяются также, как окружения, при этом возможен один необязательный аргумент.

Например:

■

```
\subsection[Искусство научных исследований]
{Искусство научных исследований\footnote{Из
 книги А.~Блоха <<Закон Мерфи>>}}

\newtheorem{rules}{Правило}
\newtheorem{low}{Закон}[section]

\begin{low}[Майерса]Если факты не подтверждают
теорию, от них надо избавиться.\end{low}

\begin{rules}[точности] Работая над решением
задачи, всегда полезно знать ответ.\end{rules}

\begin{low}[лаборатории Фетта] Никогда не пытайтесь
повторить удачный эксперимент.\end{low}
```

Искусство научных исследований[1]

Закон 6.1 (Майерса). *Если факты не подтверждают теорию, от них надо избавиться.*

Правило 1 (точности). *Работая над решением задачи, всегда полезно знать ответ.*

Закон 6.2 (лаборатории Фетта). *Никогда не пытайтесь повторить удачный эксперимент.*

[1] Из книги А. Блоха «Закон Мерфи»

Лабораторная работа №4

Форматирование текста

Создайте новый файл и наберите следующий текст.

Текст выровнен по центру. Текст выровнен по центру. Текст выровнен по центру. Текст выровнен по центру.

Текст выровнен влево. Текст выровнен влево. Текст выровнен влево. Текст выровнен влево.

Текст выровнен вправо. Текст выровнен вправо. Текст выровнен вправо. Текст выровнен вправо.

Так можно выделить один абзац или короткую цитату.

Так можно выделить несколько абзацев. В этом случае автоматически вставляется абзацный отступ[1].

Так-же можно выделить и большую цитату.

Для стихов используется специальное окружение:

Годы, люди и народы
Убегают навсегда
Как текучая вода.
В гибком зеркале природы
Звезды — невод, рыбы — мы,
Боги — призраки у тьмы.

Велимир Хлебников
1916

LaTeX в своём арсенале имеет три вида списков:

нумерованный — все элементы списка пронумерованы;

маркированный — все элементы списка помечены маркерами (значками);

описание — все элементы списка состоят из выделенного текста (ключевого слова) и его толкования, как например этот список.

Пример нумерованного списка:

1. Первая запись первого уровня.
2. Вторая запись первого уровня.
   (a) Первая запись второго уровня.
   (b) Вторая запись второго уровня.
      i. Первая запись третьего уровня.
      ii. Вторая запись третьего уровня.
         A. Первая запись четвёртого уровня.
         B. Вторая запись четвёртого уровня.

[1] Если, конечно-же, во всём тексте абзацы имеют абзацный отступ.

C. Третья запись четвёртого уровня.

iii. Третья запись третьего уровня.

(c) Третья запись второго уровня.

3. Третья запись первого уровня.

Пример ненумерованного (маркированного) списка:

- Первая запись первого уровня.
- Вторая запись первого уровня.
  - – Первая запись второго уровня.
  - – Вторая запись второго уровня.
    - ∗ Первая запись третьего уровня.
    - ∗ Вторая запись третьего уровня.
      - · Первая запись четвёртого уровня.
      - · Вторая запись четвёртого уровня.
      - · Третья запись четвёртого уровня.
    - ∗ Третья запись третьего уровня.
  - – Третья запись второго уровня.

§ Третья запись первого уровня.

Неформатированный текст используют для отображения частей исходного текста LaTeXа или текстов программ. Например:

Здесь использована модификация с отображением пробелов

```
Неформатированный␣текст␣используют
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣для
отображения␣частей␣исходного␣текста␣␣␣␣␣␣\LaTeX␣a
или␣текстов␣программ.
```

Для коротких фрагментов, например чтобы отобразить команду `\footnotesize` используется другой способ.

Можно изменить размер и положение одного абзаца

А можно изменить размер и положение одного большого участка текста.
Внутри этого окружения сохраняются все возможности форматирования текста[a].

[a] Ширина текста взята 4 см.

Короткие фрагменты текста можно выделить так или так.
Можно текст поднять или опустить.

С помощью «линеек» можно сделать такое:

2

Глава 4

Печатный документ

1. Структура документа

1.1. Преамбула

> ***Преамбула*** *— вводная часть исходного файла, предшествующая самому документу.*

`\documentclass[опции]{класс}[дата]` — определяет *класс* документа, *опции* — дополнительные параметры, *дата* указывает дату выпуска наиболее старой версии класса, пригодного для компиляции (записывается в формате `гггг/мм/дд`).

`\usepackage[опции]{пакет}[дата]` — подключает дополнительные *пакеты*. Эта команда действует аналогично предыдущей.

1.2. Титульный лист

Для автоматической генерации титульного листа используется декларация `\maketitle`. Информация для титульного листа берётся из команд:

- `\title{Заголовок}` — **обязательна**;

- `\author{Автор}` — **обязательна**, можно между «авторами» вставлять декларацию `\and`;

- `\date{дата}` — если не указать, то ставит текущую дату.

В аргументах всех этих команд можно использовать команду `\thanks{примечания}`.

Для составления титульного листа вручную используется окружение `titlepage`, которое создаёт новую страницу, устанавливает счётчик страниц на 1, но не печатает номер страницы.

1.3. Аннотация (abstract)

Для аннотации используется окружение `abstract`, оно начинает новую страницу, если включена опция документа titlepage. Текст внутри окружения выделяется дополнительными отступами слева и справа.

1.4. Секционирование (рубрикация)

Таблица 4.1.

Стандартные команды рубрикации

Команда	Рубрика
`\part[альт.загол.]{загол.}`	Часть
`\chapter[альт.загол.]{загол.}`	Глава
`\section[альт.загол.]{загол.}`	Раздел
`\subsection[альт.загол.]{загол.}`	Подраздел
`\subsubsection[альт.загол.]{загол.}`	Подподраздел
`\paragraph[альт.загол.]{загол.}`	Параграф
`\subparagraph[альт.загол.]{загол.}`	Подпараграф

Обязательный аргумент *загол.* — заголовок рубрики, необязательный *альт.загол.* — альтернативный заголовок. Если задан аль-

тернативный заголовок, то он будет фигурировать в содержании и в колонтитулах.

Все эти рубрики имеют разновидность со звёздочкой. Команда со звёздочкой (*) создаёт рубрику без номера и не имеет дополнительного аргумента.

Аргумент, который передаётся в содержание (колонтитул), является *подвижным*, поэтому *неустойчивые команды*, используемые в таких аргументах, должны быть защищены с помощью декларации `\protect`.

Нумерация рубрик устроена иерархически (за исключением `\part`, эта рубрика необязательна и никак не влияет на нумерацию глав). В классах **article** и **proc** не определена рубрика `\chapter`, для них самой старшей и обязательной рубрикой является `\section`. Класс **letter** вообще не имеет рубрик.

Счётчик `secnumdepth` определяет уровень самого младшего раздела, использующего нумерацию.

Декларация `\appendix` обозначает приложения. Она изменяет способ нумерации, а при использовании рубрики `\chapter` пишет слово «Приложение» вместо «Глава».

Ключевые слова рубрик задаются следующими командами-логосами: `\abstractname`, `\partname`, `\chaptername`, `\appendixname`, которые должны переопределяться в файле русификации.

1.5. Оглавление (содержание)

Оглавление, списки рисунков и таблиц делают соответствующие декларации:

- `\tableofcontents` — оглавление;
- `\listoffigures` — список рисунков;
- `\listoftables` — список таблиц.

При этом создаются *служебные файлы* `.toc`, `.lof` и `.lot`, соответственно.

Служебные файлы создаются/обновляются LaTeXом по мере необходимости, если только, в преамбуле не стоит декларация `\nofiles`. Обычно эта декларация используется на самом последнем этапе подготовки документа, для внесения последних поправок (если в этом есть необходимость).

Для «ручной» коррекции содержаний используются следующие команды:

- `\addcontentsline{файл}{формат}{запись}` — добавление *записи* в соответствующий *файл* в указанном *формате*. Форматы могут быть следующими:
  - — для файлов `.toc` — *part*, *chapter*, *section*, *subsection*, *subsubsection*, *paragraph*, *subparagraph*;
  - — для файлов `.lof` — *figure*;
  - — для файлов `.lot` — *table*.
- `\addtocontents{файл}{запись}` — добавление *записи* в соответствующий *файл*. Здесь в качестве *записи* можно использовать любой текст в т. ч. и форматирующие команды.

Для внесения записи с произвольным номером (например, номером рубрики, рисунка, таблицы) можно использовать команду `\numberline{номер}{запись}` с предварительной командой `\protect`.

Для составления оглавления полезно использовать счётчик `tocdepth`, который определяет уровень самого младшего раздела, входящего в оглавление.

Замечание: *Запись в приведённых командах является подвижным аргументом!*

1.6. Библиография (список литературы)

Список литературы формируется с помощью окружения `thebibliography`:

```
\begin{thebibliography}{текст}
   \bibitem[метка]{имя} данные
```

```
    \bibitem[метка]{имя} данные
    .
    .
    .
    \bibitem[метка]{имя} данные
\end{thebibliography}
```

Обязательный аргумент *текст* служит для задания левого отступа, равного ширине этого *текста*.

Если *метка* в аргументе `\bibitem` не задана, то записи списка нумеруются порядковыми номерами.

Для задания *имени* литературного источника можно использовать любые символы, кроме ‘,’.

При записи данных литературного источника можно использовать декларацию `\newblock`, которую следует вставлять между логическими блоками. При использовании стандартных классов в этом месте будет добавлен дополнительный пробел. При наличии опции класса openbib каждый блок будет напечатан с новой строки, а строки внутри блока наделены дополнительным левым отступом на величину `\bibindent`.

Цитирование литературы в тексте производится командой `\cite[текст]{имя}`, при этом, если указан *текст*, то он вставляется, через запятую, после индекса источника. Можно в одной команде указывать несколько источников, перечисляя их через запятую.

Для цитирования литературы используется *механизм перекрёстного цитирования* (файл `.aux`).

Для более удобной работы с библиографией существует утилита BibTeX.

2. Механизм перекрёстного цитирования

Для того, чтобы сослаться на ту или иную часть текста, необходимо поставить *метку* в этой части с помощью команды `\label{метка}`.

Для ссылок существуют две стандартные команды:

- `\ref{метка}` — вставляет значение счётчика изменённого перед соответствующей командой `\label`.

- `\pageref{метка}` — вставляет номер страницы, на которой стоит соответствующая команда `\label`.

Метку надо ставить *сразу же после помечаемого объекта!* Например:

■

```
\@writefile{lot}{\contentsline {table}{
  \numberline {1}{\ignorespaces Стандартные
  команды рубрикации}}{2}{table.0.1}}
\newlabel{sect}{{1}{2}{Секционирование (рубрикация)
  \relax }{table.0.1}}
\bibcite{latexe}{\hyper@link[Cite]{}{\hyper@hash
  latexe}{1}}
\bibcite{latex}{\hyper@link[Cite]{}{\hyper@hash
  latex}{2}}
\newlabel{ex4}{{}{1}{\relax }{task.0..4}}
\newlabel{step}{{{\bf  8.}}{2}{Структура документа
  \relax }{enumi.0.8}}
```

3. Большие документы

3.1. Условная компиляция

Часть документа, для которой выполняется *условная компиляция* надо окружить командными скобками if: `\iftrue` ... `\fi` или `\iffalse` ... `\fi`.

Первые командные скобки разрешают компиляцию, вторые — запрещают.

Можно внутри скобок if использовать «оператор» `\else`, который имеет тот-же смысл (инверсия), что и в программировании.

3.2. Включение файлов

Включать файлы в основной документ можно двумя способами.

1. Можно использовать команду `\input{файл}`.

   Имя *файла* можно указать как локальное или как полное, при этом, для описания пути используется слэш '/'. Если файл имеет расширение `.tex`, то его можно не писать.

   Действие этой команды аналогично тому, как если-бы вместо команды было написано содержимое файла.

2. Другой способ заключается в том, что документ разбивается на логически законченные части, которые записываются в отдельных файлах.

   Создаётся *главный документ*, который включает в себя преамбулу, окружение **document**, при необходимости — команды генерации содержаний и команды `\include{файл}` для каждого используемого файла.

   В преамбуле пишется `\includeonly{список файлов}`, — команда в которой перечисляются те файлы, которые должны быть откомпилированы.

   Содержимое каждого откомпилированного файла будет начинаться с новой страницы, при этом, сохранится вся необходимая информация для перекрёстного цитирования.

Например:

■ Пример использования метода "include":

```
\documentclass[a4paper,openbib]{report}
\usepackage[cp1251]{inputenc}
\usepackage[russian]{babel}
    \includeonly{%
%tex/format,
%tex/ex3,
tex/doc,
%tex/ex4,
    }
\begin{document}
```

```
   \tableofcontents
\include{tex/format}
\include{tex/ex3}
\include{tex/doc}
\include{tex/ex4}
\end{document}
```

Лабораторная работа №5
Структура документа

1. Создайте титульный лист с заголовком «Основы издательской системы LaTeX».

2. Предыдущие задания должны быть в отдельных файлах. В начале каждого файла поместите команду самой старшей рубрики (только не часть), дав соответствующие названия:

   text1 — Введение

   text2 — Строки, абзацы, страницы

   text3 — Форматирование

3. Создайте проект (главный документ) и подключите имеющиеся файлы с помощью команды `input`.

4. Создайте содержание (в начале документа).

5. Создайте bib-файл, содержащий библиографические данные об источниках литературы (добавьте данные для своих источников). О библиографической базе данных и утилите BibTeX можно узнать в приложении A.

6. Включите библиографию в главный файл. Используйте стиль GOST.

7. Создайте файл с *заключением*, причём рубрики «Введение» и «Заключение» должны быть без номеров, но присутствовать в содержании. Текст «Заключения» следующий:

> Изложенный здесь материал не является полным, однако этого достаточно для обычного пользователя. Более глубокое изложение издательской системы LaTeX 2_ε можно изучить в [1] или в [2].

8. Допишите в заключение примерно следующую фразу (используя для этого *механизм перекрёстных ссылок*):

> В разделе 1, на с. 3 показаны примеры горизонтальных и вертикальных пробелов.

9. Выполните компиляцию *раздельно*, по одному файлу.

10. Выполните *условную* компиляцию (произвольно).

11. Подключите имеющиеся файлы с помощью команды *include* и выполните компиляцию *раздельно*, по одному файлу, как в п. 9

Дополнительно

Утилита BibTeX

- Получите в .pdf файле полный список литературы в стиле *plain* и проверьте правильность составления базы.
- Получите список литературы с опцией *openbib.*
- Откомпилируйте свой файл с использованием различных стандартных стилей (*plain*, *unsrt*, *alpha*, *abbrv*).

Список литературы

1. *Гуссенс, М.* Путеводитель по пакету LaTeX и его расширению LaTeX 2_ε [Текст]: [Пер. с англ.] / М. Гуссенс, Ф. Миттельбах, А. Самарин; под ред. И. А. Маховой. — М.: Мир, 1999. — 606 с. — ISBN 5-03-003325-4 (русск.). — ISBN 0-201-54199-8 (англ.). — РФФИ №97-01-14165.

2. *Котельников, И. А.* LaTeX по русски [Текст] / И. А. Котельников, П. З. Чеботаев. — 3-е, переработанное и дополненное изд. — Новосибирск: Сибирский хронограф, 2004. — 496 с. — ISBN 5-87550-195-2. — на обложке: «Настольная издательская система LaTeX по русски».

3. *Гуссенс, М.* Путеводитель по пакету LaTeX и его графическим расширениям. Иллюстрирование документов при помощи TeX'а и PostScript'а [Текст]: [Пер. с англ.] / М. Гуссенс, С. Ратц, Ф. Миттельбах; под ред. А. И. Лебедева. — М.: Мир, Бином ЛЗ, 2002. — 621 с. — ISBN 5-03-003388-2 («Мир»).

4. *Львовский, С. М.* Работа в системе [Электронный ресурс] / С. М. Львовский. — М.: Интернет-Университет Информационных Технологий, 2007. Режим доступа: http://biblioclub.ru/index.php?page=book&id=234150.

4. Стиль документа

4.1. Стиль страницы

Страница состоит из *верхнего колонтитула*, *тела страницы* и *нижнего колонтитула*. Для задания стиля используются декларации:

- `\pagestyle{стиль}` — распространяется на весь текст, начиная с текущей страницы;
- `\thispagestyle{стиль}` — распространяется только на текущую страницу.

Стиль может иметь одно из 4 значений:

empty — пустые колонтитулы (по умолчанию использует класс **letter**);

plain — верхний колонтитул пустой, в нижнем колонтитуле — номер страницы (по умолчанию использует все классы, кроме **book** и **letter**);

headings — нижний колонтитул пустой, а в верхнем колонтитуле — номер страницы и другая информация, в зависимости от класса документа, например, название текущего раздела (по умолчанию использует класс **book**);

myheadings — нижний колонтитул пустой, а в верхний колонтитул записывается информация, определяемая командами `\markboth{лев.}{прав.}` и `\markright{прав.}`.

Стиль номера страницы задаётся `\pagenumbering{стиль}`, здесь *стиль* может принимать, уже знакомые по счётчикам, значения: *arabic*, *roman*, *Roman*, *alph* и *Alph*.

4.1.1. Формат страницы

Формат страницы определяется многими параметрами — **командными длинами**. Для удобства, можно подключить пакет `layout`, который по команде `\layout` выводит макет страницы с указанием всех длин и их значений.

Далее приводится перечень этих параметров.

`\paperheight` — высота листа бумаги.

`\paperwidth` — ширина листа бумаги.

`\topmargin` — расстояние между верхним краем листа и верхним колонтитулом **минус 1 дюйм**.

`\oddsidemargin` — расстояние между левым краем листа и левым краем текста на правой странице **минус 1 дюйм**.

`\evensidemargin` — то же самое для левой страницы.

`\textheight` — высота текста, т. е. вертикальный размер тела страницы (без колонтитулов).

`\textwidth` — ширина текста, т. е. горизонтальный размер колонтитулов и тела страницы. Переопределяется внутри процедур, изменяющих правую и/или левую границы текста.

`\linewidth` — ширина строки; равна значению `\textwidth` за исключением строк внутри процедур форматирования абзацев, таких как **quote** или `\itemize`. Значение `\linewidth` не должно изменяться командами, изменяющими длину.

`\topskip` — минимальное расстояние между верхним краем тела страницы и базисной линией первой строки текста. Соответствует `\baselineskip` для первой строки страницы.

`\headheight` — высота верхнего колонтитула.

`\headsep` — вертикальное расстояние между верхним колонтитулом и телом текстовой страницы.

`\footheight` — высота нижнего колонтитула.

`\footskip` — вертикальное расстояние между нижним краем текста и нижним краем нижнего колонтитула; измеряется между базисными линиями последних строк текста и нижнего колонтитула.

`\parindent` — ширина отступа в начале абзаца. Внутри `\parbox` устанавливается равной нулю. Может быть изменена в любом месте.

`\parskip` — дополнительный вертикальный пробел, вставляемый перед абзацем.

`\marginparwidth` — ширина заметок на полях.

`\marginparsep` — горизонтальное расстояние между внешним краем текста и заметкой на полях.

Таблица 4.2.

Эффективные значения \baselinestretch для различных размеров шрифта

интервал	**10 pt**	**11 pt**	**12 pt**
полтора	1,25	1,213	1,241
два	1,667	1,618	1,655

\baselinskip — минимальное расстояние между базисными линиями последовательных строк текста. Расстояние между некоторыми строками может быть больше, если они содержат высокие объекты. Значение \baselineskip устанавливается декларациями, изменяющими размер шрифта.

\baselinestretch — действительное число, равное величине межстрочного интервала; по умолчанию равно 1. Значение \baselinestretch изменяется командой \renewcommand.

Например:

■

Например, поставленная в преамбулу декларация

\renewcommand{\baselinestretch}{1.5} приведёт к тому, что весь печатный документ будет напечатан через 1,5 интервала + \baselineskip текущего шрифта.

Пакет doublespace *Стефана Пейджа* позволяет изменять интерлиньяж локально в k-раз посредством окружения spacing{k}.

Пакет setspace *Гоффри Тобина* основан на пакете **doublespace**, он автоматически рассчитывает \baselinestretch для текущего шрифта. В пакете определены команды \singlespacing, \onehalfspacing и \doublespacing — для глобального изменения интерлиньяжа, а также, соответствующие им окружения:

`singlespace`, `onehalfspace`, и `doublespace` — для локального изменения интерлиньяжа.

Текст можно разместить в две колонки. Переход в двухколоночный режим осуществляется по команде `\twocolumn[текст]`. При этом начинается новая страница, если имеется необязательный аргумент *текст*, то он будет написан сверху страницы в одноколоночном режиме.

Переход в одноколоночный режим выполняется по команде `\onecolumn`.

Если основная часть документа должна быть свёрстана в две колонки, то применяется *опция документа* twocolumn. В классе **proc** определён только двухколоночный режим.

4.2. Стандартные классы

На сегодняшний день в LaTeXe существует 6 стандартных классов:

report — *отчёт*, предназначен для написания небольших документов.

article — *статья*, предназначен для написания небольших документов, разделы начинаются с `\section`, хотя раздел `\part` определён.

proc — *доклад*, предназначен для написания небольших документов (статья в журнал, в сборник трудов), разделы начинаются с `\section`, в отличии от **article**, печатается только в две колонки.

book — *книга*, предназначен для написания больших документов (книга, журнал, сборник трудов). Отличается от **report** оформлением страниц и тремя дополнительными командами: `\frontmatter` — для вводной части, `\mainmatter` — для основной части и `\backmatter` — для заключительной части.

slides — *слайд*, предназначен для подготовки презентаций, слайд-шоу. Отличается увеличенными шрифтами, имеет свою до-

полнительную опцию и набор команд и окружений для подготовки и печати слайдов, оверлеев и заметок.

letter — *письмо*, предназначен для написания разного рода деловых писем. Имеет большой набор своих команд а многие стандартные команды отсутствуют.

Опции стандартных классов

10pt | 11pt | 12pt — устанавливают размер основного шрифта. По умолчанию используется 10pt.

Эти опции отсутствуют в классе ***slides****.*

letterpaper | legalpaper | executivpaper | a4paper | a5paper | b5paper — устанавливают размер листа бумаги:

Letter	8,5 д	× 11 д
Legal	8,5 д	× 14 д
Executive	7,25 д	× 10,5 д
A4	210 мм	× 297 мм
A5	148 мм	× 210 мм
B5	176 мм	× 250 мм

По умолчанию установлена опция letterpaper.

Опции a5paper и b5paper отсутствуют в классе ***proc****.*

landscape — устанавливает альбомное расположение листа (ландшафт), при этом ширина и высота листа меняются местами.

final | draft — устанавливают режим печати (чистовой или черновой). В черновом режиме строки, выходящие за правый край помечаются чёрными маркерами. По умолчанию используется опция final.

oneside | twoside — устанавливают формат документа для односторонней / двусторонней печати. При двусторонней печати форматы чётной и нечётной (левой и правой) страниц отличаются. При односторонней печати все страницы считаются правыми. По умолчанию используется опция oneside во всех классах, кроме **book**.

Эти опции отсутствуют в классе ***slides****.*

openright | openany — устанавливают режим печати глав, соответственно, на правой странице (используется по умолчанию в классе **book**) или на любой странице (используется по умолчанию в классе **report**).

В других классах эти опции отсутствуют.

onecolumn | twocolumn — устанавливают одноколоночный / двухколоночный формат документа. По умолчанию onecolumn используется для всех классов, кроме **proc**.

*Эти опции отсутствуют в классах **slides** и **letter**, а в классе **proc** не поддерживается опция onecolumn.*

titlepage | notitlepage — устанавливают режим печати титульной страницы (сгенерированной автоматически, декларацией `\maketitle`) и аннотации: на отдельных страницах (titlepage) или перед текстом, на той-же странице. По умолчанию titlepage используется для всех классов, кроме **article**.

*Эти опции отсутствуют в классе **letter**, а в классе **proc** не поддерживается опция titlepage.*

openbib — устанавливает режим форматирования списка литературы в *открытом стиле*, когда элементы записи, написанные в виде отдельных блоков печатаются с новой строки.

*Эта опция отсутствуют в классах **slides** и **letter**.*

leqno — устанавливает формат печати номеров формул слева, а не справа.

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

Лабораторная работа №6
Стиль документа

1. Поочерёдно примените к своему документе стили, задаваемые классами book, report, article, proc, slides.

2. Оформите разные части Вашего документа в разных стилях, соответствующих классам документа:

report — Введение

proc — Строки, абзацы, страницы

article — Форматирование

book — Заключение

3. Посмотрите действующие параметры страницы (с помощью пакета layout).

4. Задайте следующие параметры страницы:

   - Ширина текста — 17 см.
   - Высота текста — 23 см.
   - Левое поле — 3,5 см.
   - Верхнее поле — 3 мм.
   - Высота верхнего колонтитула — 17 мм.
   - Расстояние между верхним колонтитулом и телом текстовой страницы — 10 мм.
   - Межстрочный интервал — 1,5.

5. Создайте файл, озаглавленный «*Определение новых команд*», **с использованием определения *окружения* для следствий и *теоремы* для законов**, содержащий следующий текст:

5. Определение новых команд

Закон 5.1 (Мерфи). *Если какая-нибудь неприятность может случиться, она случается.*

Закон 5.2 (Чизхолма). *Когда дела идут хорошо, что-то должно случиться в ближайшем будущем.*

Следствия

1. Когда дела идут хуже некуда, в самом ближайшем будущем они пойдут ещё хуже.

2. Если вам кажется, что ситуация улучшается, значит, вы чего-то не заметили.

6. Задайте в этом разделе нумерацию страниц римскими цифрами.

Глава 5

Математика в LaTeX'е

1. Основные понятия

Математические формулы записываются в исходном файле в виде логических структур, а LaTeX уже строит формулу в соответствии со всеми требованиями и стандартами.

1.1. Математические моды (режимы)

Математический режим можно включить несколькими способами, при этом, возможны два варианта расположения математических формул:

- *Внутри строки* (**in-line**), когда математический текст является частью текстовой строки. Этот режим задаётся тремя способами: с помощью специального окружения `math` (`\begin{math} . . . \end{math}`), или `\( . . . \)`, или `$ . . . $` (последняя команда происходит из TeXа). В этом режиме формулы переносятся автоматически, как текст.

- В явном виде (**display**), когда математический текст печатается на отдельной строке (так называемая **вынесенная формула**). Этот режим тоже можно задать тремя способами: окружением `displaymath` (`\begin{displaymath} . . .`

`\end{displaymath}`), или `\[. . . \]`, или `$$ . . . $$` (эта команда тоже происходит из TEX'а).

Кроме того, для вынесенных формул имеется специальное окружение. Окружение `equation` формирует вынесенную формулу и автоматически её нумерует, вариант `equation*` отменяет нумерацию. *В вынесенном режиме формулы не переносятся автоматически.*

Одни и те же математические формулы выглядят по-разному в этих двух режимах.

При наборе сложных формул рекомендуется пользоваться $\mathcal{A}\mathcal{M}\mathcal{S}$-LATEX'ом (amsmath) (см. раздел 4.2.). Тогда, кроме обычного **equation** доступны окружения:

- **multline** — для длинных выражений;
- **align** (**flalign**, **alignat**) — для выравнивания выражений;
- **gather** — для центрирования выражения;
- **split** — для произвольного выравнивания (используется внутри другого окружения, обычно **equation**).

Кроме того, этот пакет вводит ряд новых, более удобных команд, а также решает некоторые проблемы с расположением отдельных элементов в математической моде.

Следует различать обычный (t) и математический (t) текст.

В свою очередь, в математических выражениях единицы измерения, химические элементы, в русскоязычном тексте ещё и запятая (как разделитель дробной части), пишутся в текстовой моде.

Для того, чтобы простой текст внутри математической формулы выглядел правильно, можно использовать команду `\mbox{текст}`, а лучше `\text{текст}` из пакета **amsmath**.

Например:

■

$$S_5 = 1{,}09 \cdot 10^{-18} \left(\frac{10^5\ \text{ГэВ}}{E} \right)^{3{,}02}$$

```
$$S_5=1{,}09\cdot 10^{-18}
\left(\frac{10^5\mbox{ГэВ}}E\right)^{3{,}02}$$
```

$t = 1{,}8$ км в. э.	`$t=1$,8~км~в.~э.`
$B_{\text{кон}} = 100$ нТл	`$B_{\text{кон}}=100\;\text{нТл}$`
$n_0 = 100$ см$^{-3}$	`$n_0=100$ см$^{-3}$`

1.2. Пробелы в математических формулах

В математических текстах, как в тех, которые располагаются в текстовой строке, так и в тех, которые печатаются в вынесенном режиме, межсимвольные пробелы устанавливаются только LaTeX'ом. Добавление лишних пробелов во входном файле никак не влияет на расположение символов в математических формулах.

Для ручного управления пробелами существуют специальные команды, перечисленные в табл. 5.1.

Таблица 5.1.

Пробелы внутри математических текстов

Название пробела	Команда	←Размер→
Двойной математический	`\qquad`	
Математический	`\quad`	
Межсимвольный	`\␣`	
Большой	`\;`	
Средний	`\:`	
Малый	`\,`	
Отрицательный малый	`\!`	
Отрицательный средний	`\negmedspace`	
Отрицательный большой	`\negthickspace`	

Первые три пробела и малый пробел можно использовать и в обычном тексте, остальные — только в математическом.

Для точной настройки горизонтальных пробелов в математике имеется команда `\mspace{длина}`, где *длина* измеряется *только* в математических единицах (mu, 1mu= 1/18em).

2. Алфавит математики

Литеры, используемые в формулах можно разделить на два класса: *алфавитно-цифровые символы* и *математические*.

Терминальные (клавиатурные) символы `+ - / * = ' | < > ( )` в математических формулах выглядят следующим образом: $+ - / * = ' | < > ( )$.

2.1. Математические акценты

Акценты, используемые в текстовой и строковой моде не работают в математической моде. Их, конечно, можно использовать в аргументах команд `\mbox{}` и `\text{}`). Для математики существуют свои акценты, которые задаются специальными командами (см. табл. 5.2), действующими только в математической моде.

Таблица 5.2.

Математические акценты

Команда	Пример	
`\hat`	`\hat a`	$\hat{a}$
`\check`	`\check a`	$\check{a}$
`\breve`	`\breve{a}`	$\breve{a}$
`\acute`	`\acute{a}`	$\acute{a}$
`\grave`	`\grave{a}`	$\grave{a}$
`\tilde`	`\tilde{a}`	$\tilde{a}$
`\bar`	`\bar{a}`	$\bar{a}$
`\vec`	`\vec{a}`	$\vec{a}$
`\dot`	`\dot{a}`	$\dot{a}$
`\ddot`	`\ddot{a}`	$\ddot{a}$
`\dddot`	`\dddot{a}`	$\dddot{a}$
`\ddddot`	`\ddddot{a}`	$\ddddot{a}$

Последние две команды доступны в пакете **amsmath**.

Кроме простых акцентов, в LaTeX'е имеется два растяжимых акцента, — `\widehat{}` и `\widetilde{}`, их ширина подбирается автоматически, от одного до трёх символов. Например:

■ $\widehat{ABC} = \widetilde{AB} + \widetilde{BC}$

`$\widehat{ABC}=\widetilde{AB}+\widetilde{BC}$`.

Пакет amsxtra вводит команды, задающие акценты как верхние индексы имена таких команд образованы добавлением приставки “sp” (см. табл. 5.3).

Таблица 5.3.

Математические акценты AMS-LaTeX

Команда	Пример	
\sphat	(AmSxtra)\sphat	$(AmSxtra)$
\spcheck	(AmSxtra)\spcheck	$(AmSxtra)$
\spbreve	(AmSxtra)\spbreve	$(AmSxtra)$
\sptilde	(AmSxtra)\sptilde	$(AmSxtra)$
\spdot	(AmSxtra)\spdot	$(AmSxtra)$
\spddot	(AmSxtra)\spddot	$(AmSxtra)$
\spdddot	(AmSxtra)\spdddot	$(AmSxtra)$

2.2. Греческие буквы

Имена команд для букв греческого алфавита составлены из их английских транскрипций (см. табл. 5.4).

В этом наборе отсутствует буква «омикрон», т. к. она совпадает с латинской буквой ‘o’. По той-же причине отсутствуют специальные команды для некоторых прописных букв.

По умолчанию, прописные буквы пишутся прямым шрифтом, а строчные — курсивом.

Некоторые строчные буквы имеют два варианта написания.

2.3. Бинарные операторы

Команды для бинарных операторов представлены в табл. 5.5.

LaTeX окружает эти символы дополнительными пробелами (за исключением индексов). Команда \not перечёркивает символ (любой).

2.4. Символы сравнения

Команды для символов сравнения представлены в табл. 5.6.

Например:

■ $x \not\in Y$ `$x\not\in Y$`

2.5. Большие операторы и символы переменного размера

Команды для больших символов операторов представлены в табл. 5.7.

Символы, начинающиеся с `big` имеют меньшие аналоги среди знаков бинарных операций, они обычно пишутся с индексами.

Такие символы, как знаки суммы, произведения, интегралы меняют свой размер в зависимости от типа математической моды, при этом пределы (индексы) автоматически размещаются справа от символа или сверху и снизу.

Для ручного управления размещением пределов имеются две команды:

- `\limits` — расположение пределов сверху и снизу от символа;

Таблица 5.4.

Команды для букв греческого алфавита

α	`\alpha`	β	`\beta`	γ	`\gamma`
δ	`\delta`	ϵ	`\epsilon`	ε	`\varepsilon`
ζ	`\zeta`	η	`\eta`	θ	`\theta`
ϑ	`\vartheta`	ι	`\iota`	κ	`\kappa`
$\varkappa$[1]	`\varkappa`	λ	`\lambda`	μ	`\mu`
ν	`\nu`	ξ	`\xi`	o	`o`
π	`\pi`	ϖ	`\varpi`	ρ	`\rho`
ϱ	`\varrho`	σ	`\sigma`	ς	`\varsigma`
τ	`\tau`	υ	`\upsilon`	ϕ	`\phi`
φ	`\varphi`	χ	`\chi`	ψ	`\psi`
ω	`\omega`	Γ	`\Gamma`	Δ	`\Delta`
Θ	`\Theta`	Λ	`\Lambda`	Ξ	`\Xi`
Π	`\Pi`	Σ	`\Sigma`	Υ	`\Upsilon`
Φ	`\Phi`	Ψ	`\Psi`	Ω	`\Omega`

Таблица 5.5.

Команды для бинарных операторов

$\pm$	\pm	$\cap$	\cap	$\vee$	\vee
$\mp$	\mp	$\cup$	\cup	$\wedge$	\wedge
$\setminus$	\setminus	$\uplus$	\uplus	$\oplus$	\oplus
$\cdot$	\cdot	$\sqcap$	\sqcap	$\ominus$	\ominus
$\times$	\times	$\sqcup$	\sqcup	$\otimes$	\otimes
$*$	\ast	$\triangleleft$	\triangleleft	$\oslash$	\oslash
$\star$	\star	$\triangleright$	\triangleright	$\odot$	\odot
$\diamond$	\diamond	$\wr$	\wr	$\dagger$	\dagger
$\circ$	\circ	$\bigcirc$	\bigcirc	$\ddagger$	\ddager
$\bullet$	\bullet	$\bigtriangleup$	\bigtriangleup	$\amalg$	\amalg
$\div$	\div	$\bigtriangledown$	\bigtriangledown		

Таблица 5.6.

Команды для символов сравнения (отношений)

$\leq$	\leq	$\geq$	\geq	$\equiv$	\equiv
$\prec$	\prec	$\succ$	\succ	$\sim$	\sim
$\preceq$	\preceq	$\succeq$	\succeq	$\simeq$	\simeq
$\ll$	\ll	$\gg$	\gg	$\asymp$	\asymp
$\subset$	\subset	$\supset$	\supset	$\approx$	\approx
$\subseteq$	\subseteq	$\supseteq$	\supseteq	$\cong$	\cong
$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq	$\bowtie$	\bowtie
$\in$	\in	$\ni$	\ni	$\propto$	\propto
$\vdash$	\vdash	$\dashv$	\dashv	$\models$	\models
$\smile$	\smile	$\mid$	\mid	$\doteq$	\doteq
$\frown$	\frown	$\parallel$	\parallel	$\perp$	\perp

- `\nolimits` — расположение пределов справа от символа.

Для глобального управления размещением пределов, в пакете **amsmath** определены опции:

sumlimits | nosumlimits — для управления пределами в суммах,

intlimits | nointlimits — для управления пределами в интегралах.

Таблица 5.7.

Команды для больших операторов

$\sum$	`\sum`	$\bigcap$	`\bigcap`	$\bigodot$	`\bigodot`
$\prod$	`\prod`	$\bigcup$	`\bigcup`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`	$\bigoplus$	`\bigoplus`
$\int$	`\int`	$\bigvee$	`\bigvee`	$\biguplus$	`\biguplus`
$\oint$	`\oint`	$\bigwedge$	`\bigwedge`	$\iint$	`\iint`[1]
$\iiint$	`\iiint`[a]	$\iiiint$	`\iiiint`[a]	$\int\cdots\int$	`\idotsint`[a]

2.6. Разделители

В качестве разделителей в математике используются различные скобки, уголки, вертикальные стрелки (см. табл. 5.8).

Таблица 5.8.

Команды для символов-разделителей

$($	`(`	$[$	`[`	$\{$	`\{`
$)$	`)`	$]$	`]`	$\}$	`\}`
$\langle$	`\langle`	$/$	`/`	$\vert$	`|`
$\rangle$	`\rangle`	$\backslash$	`\backslash`	$\Vert$	`\\|`
$\lfloor$	`\lfloor`	$\lceil$	`\lceil`		
$\rfloor$	`\rfloor`	$\rceil$	`\rceil`		
$\uparrow$	`\uparrow`	$\downarrow$	`\downarrow`	$\updownarrow$	`\updownarrow`
$\Uparrow$	`\Uparrow`	$\Downarrow$	`\Downarrow`	$\Updownarrow$	`\Updownarrow`

Для автоматического управления высотой разделителей следует использовать команды `\left` (для левой скобки) и `\right` (для правой) перед соответствующим разделителем. Для непарных разделителей можно использовать одну из этих команд, однако, надо иметь в виду, что в каждой строке количество команд `\left` и `\right` должно совпадать.

Для этого можно использовать вместо разделителя точку: команды `\left.` и `\right.` ничего не пишут.

Например:

■

$$H_c=\frac{n_1!\,n_2!}{n_1+n_2}\sum_i\left[\binom{n_1}{i}\binom{n_2}{n_1+i}+\right.$$
$$\left.+\binom{n_1-1}{i}\binom{n_2-1}{n_1+i}\right].$$

```
\begin{multline*}
H_c=\frac{n_1!\,n_2!}{n_1+n_2}\sum_i \left[
    \binom{n_1}{i}\binom{n_2}{n_1+i}+\right.\\
+\left.\binom{n_1-1}{i}\binom{n_2-1}{n_1+i}
    \right].
\end{multline*}
```

Иногда автоматический выбор высоты не удовлетворяет авторским замыслам. Для ручного управления высотой разделителей используют команды из табл. 5.9.

Таблица 5.9.

Команды, увеличивающие разделители

Слева	В центре	Справа	Символ
\bigl	\bigm	\bigr	\big
\Bigl	\Bigm	\Bigr	\Big
\biggl	\biggm	\biggr	\bigg
\Biggl	\Biggm	\Biggr	\Bigg

Здесь записаны команды в порядке возрастания. Команды из второй колонки относятся к непарным разделителям, а команды из последней колонки рекомендуется применять к символам (при этом окружающие пробелы будут несколько меньше, чем для непарного разделителя). Например:

■

$$\left\{\left[A/B\right]=C\right\};\quad A\Big/B=C;\quad A\Big/B=C;$$

```
$$ \Biggl\{\Bigl[A/B\Bigr]=C\Biggr\};
\quad A\Biggm/B=C; \quad A\Bigg /B=C; $$
```

2.7. Стрелки

Команды для символов-стрелок представлены в табл. 5.10.

Таблица 5.10.

Команды для символов-стрелок

$\leftarrow$	\leftarrow	$\Leftarrow$	\Leftarrow
$\rightarrow$	\rightarrow	$\Rightarrow$	\Rightarrow
$\longleftarrow$	\longleftarrow	$\Longleftarrow$	\Longleftarrow
$\longrightarrow$	\longrightarrow	$\Longrightarrow$	\Longrightarrow
$\uparrow$	\uparrow	$\Uparrow$	\Uparrow
$\downarrow$	\downarrow	$\Downarrow$	\Downarrow
$\leftrightarrow$	\leftrightarrow	$\Leftrightarrow$	\Leftrightarrow
$\longleftrightarrow$	\longleftrightarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\updownarrow$	\updownarrow	$\Updownarrow$	\Updownarrow
$\nearrow$	\nearrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\nwarrow$	\nwarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto
$\hookleftarrow$	\hookleftarrow	$\hookrightarrow$	\hookrightarrow
$\leftharpoonup$	\leftharpoonup	$\rightharpoonup$	\rightharpoonup
$\rightharpoondown$	\rightharpoondown	$\leftharpoondown$	\leftharpoondown
$\rightleftharpoons$	\rightleftharpoons		

Вертикальные стрелки автоматически изменяют свою высоту, когда используются в качестве разделителей.

2.8. Функции

Функции (их ещё называют логарифмоподобными) принято писать в математике прямым шрифтом (см. табл. 5.11).

Некоторые функции могут иметь индексы.

Если требуется ввести новую функцию, то лучше это сделать с помощью команд пакета amsopn:

- \DeclareMathOperator{*имя*}{*написание*};
- \DeclareMathOperator*{*имя*}{*написание*}.

Таблица 5.11.

Команды для функций

sin	`\sin`	arcsin	`\arcsin`	sec	`\sec`
cos	`\cos`	arccos	`\arccos`	csc	`\csc`
tan	`\tan`	arctan	`\arctan`	cot	`\cot`
sinh	`\sinh`	cosh	`\cosh`	max	`\max`
tanh	`\tanh`	coth	`\coth`	min	`\min`
log	`\log`	ln	`\ln`	lg	`\lg`
exp	`\exp`	sup	`\sup`	inf	`\inf`
lim	`\lim`	lim sup	`\limsup`	lim inf	`\liminf`
arg	`\arg`	ker	`\ker`	hom	`\hom`
det	`\det`	dim	`\dim`	gcd	`\gcd`
deg	`\deg`	Pr	`\Pr`		

Здесь *имя* и *написание* имеют тот-же смысл, что и при определении новых команд. Определять новые функции следует в преамбуле. Например:

■

```
\DeclareMathOperator{\tg}{tg}|
\DeclareMathOperator*{\Lim}{Lim}
  $$\Lim_{x\rightarrow k\pi}\tg^2x=0,
    \forall k=0,1,\dotsc$$
```

$$\operatorname*{Lim}_{x\rightarrow k\pi}\operatorname{tg}^2 x=0,\forall k=0,1,\dotsc$$

2.9. Прочие символы

В этой группе (см. табл. 5.12) собраны все остальные символы. Если этих символов оказалось недостаточно для записи выражения, можно воспользоваться символами $\mathcal{AMS}$-LaTeX'а.

Таблица 5.12.

Команды для разных математических символов

$\aleph$	\aleph	$'$	\prime	$\forall$	\forall
$\hbar$	\hbar	$\emptyset$	\emptyset	$\exists$	\exists
$\imath$	\imath	∇	\nabla	$\neg$	\neg
$\jmath$	\jmath	$\surd$	\surd	$\flat$	\flat
ℓ	\ell	$\top$	\top	$\natural$	\natural
wp	\wp	$\bot$	\bot	$\sharp$	\sharp
$\Re$	\Re	$\|$	\\|	$\clubsuit$	\clubsuit
$\Im$	\Im	$\angle$	\ angle	$\diamondsuit$	\diamondsuit
∂	\partial	$\triangle$	\triangle	$\heartsuit$	\heartsuit
∞	\infty	$\backslash$	\backslash	$\spadesuit$	\spadesuit

3. Основные структуры

3.1. Верхние и нижние индексы

Символы '_' и '^' обозначают, что следующий за ними символ является, соответственно, нижним или верхним индексом. Если в индексе должно быть несколько символов, то их нужно взять в фигурные скобки (сгруппировать). Например:

■

`$x^{2y}$`	x^{2y}	`${x_y}_1$`	x_{y_1}
`$x_{2y}$`	x_{2y}	`$x^y^1$`	ошибка
`$x^y_1$`	x^y_1	`$x^{y^2}$`	x^{y^2}
`$x_y^1$`	x_y^1	`${x^y}^2$`	x^{y^2}
`$x_y_1$`	ошибка	`см$^2$`	см^2
`$x_{y_1}$`	x_{y_1}	`$x_{рад}$`	x

`${_i^jA_\|^{2y}}$`	${}^j_i A^{2y}_{\|}$
`$A^k{}_{ij}{}^l$`	$A^k{}_{ij}{}^l$

Пакет **amsmath** вводит команду `\substack{выражения}` для записи многострочных индексов. Для разбиения *выражения* на строки используется команда \\. Например:

■

$$\sum_{\substack{0\le i\le m\\ 0<j<n}} P(i,j)$$

```
\begin{equation*}
\sum_{\substack{0\le i\le m\\
0<j<n}} P(i,j)
\end{equation*}
```

3.2. Корни

`\sqrt[n]{арг.}` — корень из *арг.* степени n.

Для корректировки положения указателя в пакете **amsmath** имеются команды `\leftroot{n}` для сдвига влево (вправо, если n отрицательно) на n математических длин и `\uproot{n}` для сдвига вверх (вниз, если n отрицательно). Например:

■

$$\sqrt{x} \quad \sqrt[3]{x} \quad \sqrt[\beta]{k} \quad \sqrt[\leftroot{-2}\uproot{2}\beta]{k}$$

```
$$ \sqrt{x} \quad \sqrt[3]{x} \quad
\sqrt[\beta]{k} \quad
\sqrt[\leftroot{-2}\uproot{2}\beta]{k} $$
```

3.3. Дроби

Небольшие дроби можно писать, используя знак '/'.

`\frac{числ.}{знам.}` — дробь.

В $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX'е имеются дополнительные команды для более гибкого представления дробей: `\dfrac{числ.}{знам.}` и `\tfrac{числ.}{знам.}`, которые форматируют дроби как в вынесенной формуле и в текстовой строке, соответственно. Например:

■ $\frac{a}{b}\; \dfrac{a}{b}\; \tfrac{a}{b}$ `\frac{a}{b}\; \dfrac{a}{b}\; \tfrac{a}{b}`

Непрерывную дробь можно получить с помощью команды `\cfrac[сдвиг]{числ.}{знам.}`. В необязательном параметре можно указать *сдвиг* числителя: `l` — влево, `r` — вправо.

Например:

■

$$\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cfrac{1}{\sqrt{2}+\cdots}}}}}$$

```
\cfrac{1}{\sqrt{2}+
 \cfrac{1}{\sqrt{2}+
  \cfrac[l]{1}{\sqrt{2}+
   \cfrac[r]{1}{\sqrt{2}+
        \cfrac{1}{\sqrt{2}+\dotsb
}}}}}
```

3.4. Биномиальные коэффициенты

Биномиальные коэффициенты можно получить с помощью команды `\choose`. Например:

■ $\binom{n}{n-k}$ `$n\choose {n-k}$`

В $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX'е имеются более удобные команды

- `\binom{числ.}{знам.}`,
- `\dbinom{числ.}{знам.}`,
- `\tbinom{числ.}{знам.}`.

Последние две команды имеют тот-же смысл, что и аналогичные команды для дробей. Например:

■ $\binom{n}{n-k}$ `$\binom{n}{n-k}$`

3.5. Размещение объектов друг над другом

В математических выражениях часто используют символы, расположенные друг над другом, как, например, акценты или в бинарных операциях, операциях сравнения, в выражениях со стрелками. Для размещения объектов друг над другом имеется команда `\stackrel{верхний}{нижний}`. Например:

■

$$\sin\alpha \stackrel{\alpha\rightarrow 0}{\rightarrow} 0$$

```
$$ \sin\alpha
\stackrel{\alpha\rightarrow0}
{\rightarrow} 0 $$
```

Пакет **amsmath** предоставляет дополнительные команды:

- `\overset{верхний}{нижний}` — для расположения *верхнего* объекта над *нижним* символом;
- `\underset{нижний}{верхний}` — для расположения *нижнего* объекта под *верхним* символом;
- `\sideset{слева}{справа}` — для расположения объектов *слева* и *справа* от символа, используется, в основном для индексов.

Все эти команды можно использовать в сочетании друг с другом и другими командами.

`\overline{выражение}` и `\underline{выражение}` используют для надчёркивания и подчёркивания *выражения*, соответственно. Команда подчёркивания может применяться и в текстовом режиме.

`\overbrace{выражение}` и `\underbrace{выражение}` вставляют, соответственно, над и под *выражением* горизонтальную фигурную скобку. Например:

■

$$\overbrace{\underbrace{A \rightarrow B}_{\forall x \in \Re}}^{\leftrightarrow}$$

```
$$\overset{\looparrowright}{\underset{\forall
 x\in\Re}{\underbrace{A\rightharpoonup B}}}$$
```

Для рисования левой и правой стрелки над *выражением* используются команды `\overleftarrow{выражение}` и `\overrightarrow{выражение}`, соответственно.

Пакет **amsmath** дополняет их следующими командами:

- `\underleftarrow{выражение}` и `\underrightarrow{выражение}` — для стрелок под *выражением*;
- для двунаправленных стрелок над и под *выражением* — `\overleftrightarrow{выражение}` и `\underleftrightarrow{выражение}`;
- для растяжимых стрелок — `\xleftarrow[ниж.]{верх.}` и `\xrightarrow[ниж.]{верх.}`, обязательный аргумент *верх.* – верхний индекс, необязательный *ниж.* — нижний.

Например:

■

$$\overleftrightarrow{A \xrightarrow[x\subsetneq B]{\forall x\in\Re} B}$$

```
$$\overleftrightarrow{{A\xrightarrow[x
 \varsubsetneq B]{\forall x\in\Re}B}}$$
```

3.6. Знаки пунктуации и многоточия

LaTeX вставляет небольшие пробелы после знаков пунктуации, но *в математическом режиме* точка, двоеточие *и* восклицательный знак *не являются знаками пунктуации*, для этих знаков есть специальные команды:

, ,	; ;	: \colon	. \ldotp	· \cdotp

Для многоточий в математической моде имеются свои команды:

… \ldots	⋯ \cdots	⋮ \vdots	⋱ \ddots

Горизонтальные многоточия, в большинстве случаев, AMS-LaTeX правильно поставит при использовании команды `\dots`, используемой и в текстовом режиме. В тех случаях, когда многоточие заканчивает формулу, следует явно указать тип многоточия: `\dotsc`, `\dotsb`, `\dotsi`, `\dotsm`, — соответственно после *запятой*, *бинарного оператора*, знака *интеграла* и *точки умножения*.

3.7. Формулы в рамке

Формулы в рамке создаёт команда из пакета **amsmath** `\boxed{формула}`. Например:

$$\boxed{r!\,(-1)^{(r-j)/2}}$$

```
\begin{equation*}
\boxed{r!\,(-1)^{(r-j)/2}}
\end{equation*}
```

3.8. Условный выбор

Условный выбор в математике записываются с помощью левой фигурной скобки, после которой следуют возможные варианты, в зависимости от условий. Для написания таких конструкций в пакете **amsmath** имеется окружение `cases`: Например:

$$P_{r-j} = \begin{cases} 0, & \text{если } r-j \text{ не чётно;} \\ r!\,(-1)^{(r-j)/2}, & \text{если } r-j \text{ чётно.} \end{cases}$$

```
\begin{equation*}
```

```
\label{eq:C} P_{r-j}=
  \begin{cases}
   0,&  \text{если $r-j$ не чётно};\\
   r!\,(-1)^{(r-j)/2},&\text{если $r-j$ чётно}.
  \end{cases}
\end{equation*}
```

3.9. Матрицы

LaTeX имеет гибкое окружение `array` для написания многострочных выражений типа *матриц*.

```
\begin{array}{колонки}
... & ... \\
... & ... \\
... & ...
\end{array}
```

В параметре *колонки* задаётся формат колонок (l, c, r), литерал формата задаёт выравнивание соответствующей колонки. Внутри колонки разделяются табулятором '&', а срока заканчивается знаком \\. Например:

■

$$\mathbf{X} = \left\langle \begin{array}{lcr} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right\rangle$$

```
\mathbf{X} = \left\langle
\begin{array}{lcr}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array}
\right\rangle
```

Для написания матриц в пакете **amsmath** имеется следующий набор окружений:

- `pmatrix` — матрица в круглых скобках ( );

- `bmatrix` — матрица в квадратных скобках [ ];
- `vmatrix` — матрица в вертикальных разделителях ;
- `Vmatrix` — матрица в двойных вертикальных разделителях ;
- `matrix` — матрица без скобок.
- `smallmatrix` — маленькая матрица в текстовой строке, скобки, при необходимости, нужно вставлять явно.

Внутри матриц доступна команда `\hdotsfor[`*расст.*`]{`n`}`, вставляющая строку точек, занимающую n колонок. Необязательный аргумент *расст.* служит для изменения расстояний между точками.

Например:

■ В этом примере команда `\hdotsfor[2]{4}` рисует точки шириной в четыре колонки с увеличенным расстоянием между точками в два раза.

$$\begin{pmatrix} D_1t & -a_{12}t_2 & \dots & -a_{1n}t_n \\ -a_{21}t_1 & D_2t & \dots & -a_{2n}t_n \\ \dots & \dots & \dots & \dots \\ -a_{n1}t_1 & -a_{n2}t_2 & \dots & D_nt \end{pmatrix},$$

```
\begin{pmatrix}
D_1t       & -a_{12}t_2 & \dots & -a_{1n}t_n \\
-a_{21}t_1 & D_2t       & \dots & -a_{2n}t_n \\
\hdotsfor[2]{4}\\
-a_{n1}t_1 & -a_{n2}t_2 & \dots & D_nt
\end{pmatrix}
```

4. Большие формулы

4.1. Уравнения

Большие формулы пишутся в вынесенном режиме. Для этого используется окружение `equation`, которое автоматически вставляет номер уравнения, `equation*` — без номера. Например:

■

$$f(x) = \cos x$$

$$f(x) = \cos x \tag{1}$$

```
\begin{equation*}\label{eq:simple}
f(x) = \cos x
\end{equation*}
\begin{equation}\label{eq:simple}
f(x) = \cos x
\end{equation}
```

Для многострочных выражений имеются окружения `eqnarray` и `eqnarray*`, которые, как и **equation**, включают математическую моду. Они работают, как таблица **array**, состоящая из трёх столбцов формата `{rcl}`. Средний столбец, обычно, используется для знака отношения. Команда \\ разрывает строки. Например:

■

$$\begin{aligned} f(x) &= \cos x \\ f'(x) &= -\sin x \\ \int_{0}^{x} f(y)dy &= \sin x \end{aligned}$$

```
\begin{eqnarray*}
f(x) & = & \cos x \\
f'(x) & = & -\sin x \\
\int_{0}^{x} f(y)dy & = & \sin x
\end{eqnarray*}
```

Здесь \\ используется также, как и в текстовой моде.

4.2. Сложные формулы

Рассмотрим окружения для вынесенных формул пакета **amsmath**.

Например:

```
\begin{equation}\label{split}
\begin{split}
a& =b+c-d+\\
 & \quad +e-f\\
 & =g+h\\
 & =i
\end{split}
\end{equation}
```

$$\begin{aligned} a &= b + c - d + \\ &\quad + e - f \\ &= g + h \\ &= i \end{aligned} \tag{2}$$

Например:

```
\label{multline}
\begin{multline}
a+b+c+d+\\
+i+j+k+l
\end{multline}
```

$$\begin{multline} a + b + c + d + \\ + i + j + k + l \end{multline} \tag{3}$$

Например:

```
\begin{gather}\label{gather}
a_1=b_1+c_1\\
a_2=b_2+c_2-d_2+e_2
\end{gather}
```

$$\begin{gather} a_1 = b_1 + c_1 \tag{4} \\ a_2 = b_2 + c_2 - d_2 + e_2 \tag{5} \end{gather}$$

Например:

```
\begin{align}\label{align}
a_1& =b_1+c_1\\
a_2& =b_2+c_2-d_2+e_2
\end{align}
```

$$\begin{aligned} a_1 &= b_1 + c_1 && (6) \\ a_2 &= b_2 + c_2 - d_2 + e_2 && (7) \end{aligned}$$

```
\begin{align}\label{align2}
a_{11}& =b_{11}&
a_{12}& =b_{12}\\
a_{21}& =b_{21}&
a_{22}& =b_{22}+c_{22}
\end{align}
```

$$\begin{aligned} a_{11} &= b_{11} & a_{12} &= b_{12} && (8) \\ a_{21} &= b_{21} & a_{22} &= b_{22} + c_{22} && (9) \end{aligned}$$

Например:

```
\begin{alignat}{2}\label{alignat}
a_{11}& =b_{11}&
a_{12}& =b_{12}\\
a_{21}& =b_{21}&
a_{22}& =b_{22}+c_{22}
\end{alignat}
```

$$\left|\begin{aligned} a_{11} &= b_{11} & a_{12} &= b_{12} && (10)\\ a_{21} &= b_{21} & a_{22} &= b_{22}+c_{22} && (11)\end{aligned}\right|$$

Например:

```
\begin{flalign*}
a_{11}& =b_{11}&
a_{12}& =b_{12}\\
a_{21}& =b_{21}&
a_{22}& =b_{22}+c_{22}
\end{flalign*}
```

$$\left|\begin{aligned} a_{11} &= b_{11} & a_{12} &= b_{12}\\ a_{21} &= b_{21} & a_{22} &= b_{22}+c_{22}\end{aligned}\right|$$

Для вставки текста в многострочную формулу используется команда `\intertext{текст}`.

В сложных формулах, определённых в $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX'е, автоматический разрыв страницы не производится. Для управления разрывом определены 2 способа. Команда `\displaybreak[N]` (ставится перед `\\`) действует аналогично `\pagebreak`. Декларация `\allowdisplaybreaks[N]` может располагаться в любом месте и действует внутри скобок.

4.3. Позиционирование

Команды, регулирующие размеры бокса, часто бывают полезны, особенно в математических формулах:

- `\mathstrut` — бокс, имеющий высоту круглой скобки и нулевую ширину.

- `\phantom{формула}` — бокс, имеющий ширину *формулы* и нулевую высоту.

- `\vphantom{формула}` — бокс, имеющий высоту *формулы* и нулевую ширину.

- `\smash[t|b]{формула}` — бокс с *формулой*, имеющий нулевую высоту. Возможны опции: t — зануляется только верхняя часть бокса (выше базисной линии); b — зануляется нижняя часть бокса (глубина).

Все эти команды могут использоваться и в текстовой моде.

4.4. Нумерация и ссылки

Большие формулы часто нумеруются, причём нумерация происходит автоматически. В многострочных выражениях бывают ситуации, когда номера нужны лишь для некоторых строк.

Для ручного управления имеются команды:

- `\tag{метка}` — ставит *метку* (номер);

- `\tag*{метка}` — ставит *метку* без окружающих скобок;

- `\notag` (или LaTeX'овская команда `\nonumber`) — убирает номер.

Для подчинённой нумерации (формул внутри группы) используется окружение `subequations`, как в примерах (12a), (12b) и (12c).

Например:

■

$$f(x) = \cos x \tag{12a}$$

$$f'(x) = -\sin x \tag{12b}$$

$$\int_0^x f(y)dy = \sin x \tag{12c}$$

```
\begin{subequations}
\begin{align}
\label{f} f(x) & = \cos x \\
\label{f'} f'(x) & = -\sin x \\
```

```
\label{F} \int_{0}^{x} f(y)dy & = \sin x
\end{align}
\end{subequations}
```

Ссылки на формулы пишутся, как и их номера, в круглых скобках. В пакете **amsmath** для ссылок имеется команда `\eqref{имя}`. Например:

■ Нумерация формул показана в примерах (1)–(10), а подчинённая — в примерах (12a), (12b) и (12c).

```
Нумерация формул показана в примерах
\eqref{eq:simple}--\eqref{alignat},
а подчинённая --- в примерах
\eqref{f}, \eqref{f'} и \eqref{F}.
```

5. Шрифты

Математические шрифты, как и текстовые, различаются гарнитурой, насыщенностью, начертанием и размером. *Однако, в математических шрифтах нельзя изменять отдельные атрибуты.*

Шрифты определяются математической версией (командой `\mathversion{версия}`). Изначально существует две *версии*: `bold` и `normal`, которые в LaTeX 2_ε задаются, соответственно, декларациями `\boldmath` и `\unboldmath`. *Переключать версию можно только вне математической моды.*

Внутри математической моды можно изменять *математические алфавиты*, они определяют начертание букв, цифр, прописных греческих букв и акцентов.

В табл. 5.7 представлены математические шрифты и соответствующие команды. В качестве аргумента используется следующий `текст`:

```
АБВабв, \widehat{ABC}\vec{a}\ddot{b}\breve{c},
123, \Delta\Omega\delta\zeta\omega
```

Пакет `amsbsy` (загружаемый пакетом **amsmath**) вводит команду `\boldsymbol{выражение}`, которая выводит *выражение* в **bold** версии математических шрифтов.

Может оказаться, что для какого-либо символа отсутствует

Таблица 5.13.

Математические алфавиты

`\mathrm{текст}`	, $\widehat{\mathrm{ABC}}\tilde{\mathrm{a}}\ddot{\mathrm{b}}\check{\mathrm{c}}$, 123, $\Delta\Sigma\Omega\delta\zeta\omega$
`\mathbf{текст}`	, $\widehat{\mathbf{ABC}}\tilde{\mathbf{a}}\ddot{\mathbf{b}}\check{\mathbf{c}}$, **123**, $\mathbf{\Delta\Sigma\Omega}\delta\zeta\omega$
`\mathsf{текст}`	, $\widehat{\mathsf{ABC}}\tilde{\mathsf{a}}\ddot{\mathsf{b}}\check{\mathsf{c}}$, 123, $\Delta\Sigma\Omega\delta\zeta\omega$
`\mathtt{текст}`	, $\widehat{\mathtt{ABC}}\tilde{\mathtt{a}}\ddot{\mathtt{b}}\check{\mathtt{c}}$, 123, $\Delta\Sigma\Omega\delta\zeta\omega$
`\mathit{текст}`	, $\widehat{ABC}\tilde{a}\ddot{b}\check{c}$, *123*, $\Delta\Sigma\Omega\delta\zeta\omega$
`\mathnormal{текст}`	, $\widehat{ABC}\tilde{a}\ddot{b}\check{c}$, 123, $\Delta\Sigma\Omega\delta\zeta\omega$
`\mathcal{текст}`	, $\widehat{\mathcal{ABC}}\tilde{\mathcal{a}}\ddot{\mathcal{b}}\check{\mathcal{c}}$, $\infty\in\ni$, $\cdot\pm\otimes\delta\zeta\omega$

`bold`-вариант. Для этого случая имеется команда `\pmb{символ}`, рисует символ несколько раз, с небольшими смещениями.

$\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX имеет коллекцию дополнительных алфавитов (готические, калиграфические, контурные шрифты). Например:

■

$$\mathbb{LATEX} \ne \mathfrak{latex} \ne \mathfrak{LATEX} \ne \mathcal{LATEX}.$$

```
$$ \mathbb{LATEX}\ne\mathfrak{latex}\ne
\mathfrak{LATEX}\ne\mathcal{LATEX}. $$
```

5.1. Классы $\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX

$\mathcal{A}\mathcal{M}\mathcal{S}$-LaTeX поставляется с двумя своими классами: `amsart` и `amsbook`, соответствующие классам **article** и **book**. При их использовании не надо отдельно загружать пакет **amsmath**.

Лабораторная работа №7
Математика в LaTeXe

Создайте новый файл на своём диске и наберите следующий текст.

Скорость потерь пакетов $\lambda'(t)$ вычисляется по формуле:

$$\lambda'(t) = \frac{1}{R}\left[1 - \left(1 - \frac{\lambda(t)}{A(t)}\right)^{RA}\right], \qquad (1)$$

где

$$\lambda(t) = \frac{W(t)}{R}p(q),$$

$$p(q) = \begin{cases} 0, & \text{если } q < B_C, \\ \frac{q-B_C}{B_E-B_C}, & \text{если } B_C q B_E, \\ 1, & \text{если } q > B_E. \end{cases}$$

Запишем обратные преобразования Фурье в виде суммы вычетов:

$$R_\alpha^{(2)}(x) = \frac{1}{2\pi i}\int\limits_{-\infty}^{\infty} \hat{R}_\alpha^{(2)}(s)x^{-s}ds = \sum\left\{x^{-s}\left[A(s)F_\gamma^{(2)}(s) + \right.\right.$$

$$\left.\left. + B(s)F_\alpha^{(2)}(s)\right]/\varphi(s)\right\}; \qquad (2)$$

$$\hat{R}_a^{\text{II}}(t,s) = \sum_b \int\limits_0^t \hat{G}_{a\to b}(t-t',s)\hat{F}_b^{\text{II}}(t',s)dt'. \qquad (3)$$

Глава 6

Таблицы в LaTeX'е

1. Табулятор

Табулятор создаёт окружение `tabbing`.

Внутри используются следующие управляющие команды:

`\=` — устанавливает табулятор;

`\>` — переход к следующему табулятору справа;

`\\` — начинает новую строку;

`\kill` — переход на новую строку без печати текста текущей строки с сохранением табуляторов, обычно используется для задания шаблона;

`\+` — передвигает левое поле последующих строк на один табулятор вправо, как бы вставляя в начало строк контрольный символ `\>`, несколько таких команд дают суммарный эффект;

`\-` — передвигает левое поле последующих строк на один табулятор влево, отменяя тем самым одну, выданную ранее, команду `\+`;

`\<` — переход к предыдущему табулятору слева, применимо лишь после употребления одной или нескольких команд `\+`;

`\pushtabs` — запоминает текущее положение табуляторов, чтобы восстановить их по команде `\poptabs`;

`\poptabs` — восстанавливает положение табуляторов, сохранённое по команде `\pushtabs`;

`\'` — сдвигает в колонке текст, предшествующий команде, вправо, последующий текст выровнен как обычно — влево;

`\'` — сдвигает весь последующий текст к правой границе *страницы*, действует только на *последнюю* колонку;

`\a=`, `\a'`, `\a‘` — заменяют, переопределённые внутри **tabbing** — команды акцентирования `\=`, `\'`, `\‘`.

Например:

■

```
Деканат МИЭМИС\quad \= 29--65--47
\quad\= c $10^{00}$ до $\infty$ \kill
\textbf{Подразделение} \> \textbf{Телефон}
\> \textbf{Часы приёма} \\
Кафедра ЦТиБА \> 29--65--25
\> c $9^{00}$ до $\infty$ \\
Деканат МИЭМИС \> 29--65--47
\> c $8^{00}$ до $17^{00}$ \\
```

Подразделение	**Телефон**	**Часы приёма**
Кафедра ЦТиБА	29–65–25	с 9^{00} до ∞
Деканат МИЭМИС	29–65–47	с 8^{00} до 17^{00}

2. Таблица

Таблица в LaTeX'е — это бокс из последовательных рядов, элементы которых выровнены в столбцах.

Текст внутри ячеек обрабатывается в текстовой моде, т. е., при необходимости, разбивается на строки.

Таблица может быть разлинована. В LaTeX'е имеется три разновидности таблиц.

`array[положение]{формат}` — используется только в математической моде.

`tabular[положение]{формат}` — используется в любой моде.

`tabular*{ширина}[положение]{формат}` — таблица заданной *ширины*.

Аргумент *ширина* используется только в **tabular*** и задаёт ширину таблицы. Необязательный аргумент задаёт *положение* таблицы относительно текущей базисной линии:

t — по текущей базисной линии выравнивается базисная линия *верхней строки*;

c — по текущей базисной линии выравнивается *центр таблицы* (действует по умолчанию);

b — по текущей базисной линии выравнивается базисная линия *нижней строки*.

Аргумент *формат* задаёт количество и формат колонок. Можно использовать следующие управляющие символы:

l — текст сдвинут к левому краю колонки.

c — текст расположен по центру колонки.

r — текст сдвинут к правому краю колонки.

p{*ширина*} — текст выровнен по ширине в колонке шириной *ширина* (как в `\parbox`).

| — вертикальная линия на полную высоту и глубину бокса (два символа подряд создают двойную линию).

@{*текст*} — вставка *текста* во все строки, *текст* является подвижным аргументом, в окружении **array** он обрабатывается в математической моде. Такая конструкция называется ещё @-выражением. @-выражение подавляет горизонтальный пробел, вставляемый между колонками.

Команда `\extracolsep{ширина}` в @-выражении вставляет дополнительный пробел величиной *ширина* слева всех последующих колонок (кроме первой).

***{*n*}{*формат*}** — мультиформат, задаёт n повторений формата колонок *формат*.

Внутри окружений записываются строки таблиц, где можно использовать следующие команды и управляющие символы:

& — табулятор (разделитель колонок), количество ячеек в строке должно совпадать с количеством столбцов, заданном в формате таблицы;

`\\[высота]` — начинает новую строку, может иметь необязательный параметр *высота*, увеличивающий расстояние между этой и следующей строкой на заданную величину;

`\tabularnewline[высота]` — аналогична предыдущей команде, используется для отличия в исходном файле конца строки таблицы от конца строки в других командах внутри таблицы (`\parbox`, окружениях **minipage**, **tabular**, **array**, при использовании деклараций `\centering`, `\raggedright`, `\raggedleft`, которые можно использовать в ячейках таблицы);

`\vline` — вставляет вертикальную линию на полную высоту и глубину строки (может использоваться в @-выражениях);

`\hline` — вставляет горизонтальную линию на всю ширину бокса (две подряд команды `\hline` рисуют двойную линию с зазором, непересекаемую вертикальными линиями);

`\cline{i-j}` — проводит горизонтальную линию через колонки с i-й по j-ю;

`\multicolumn{n}{формат}{текст}` — создаёт одну ячейку шириной n колонок с форматом *формат*. При употреблении крайних вертикалей в аргументе *формат* надо следить за положением вертикальных линий, обычно для всех, кроме крайних ячеек не следует задавать левую вертикаль, если она задана в формате таблицы.

2.1. Параметры настройки

Следующие команды представляют из себя командные длины, которые можно изменять как за пределами таблиц так и внутри ячейки. В последнем случае обязательно следует ограничивать область действия этих команд.

`\arraycolsep` — половина ширины горизонтального пробела между колонками в окружении **array**;

`\tabcolsep` — половина ширины горизонтального пробела между колонками в окружениях **tabular** и **tabular***;

`\arrayrulewidth` — толщина линий;

`\doublerulesep` — ширина зазора между двойными линиями.

`\arraystretch` — коэффициент, определяющий интервал между строками. Действительный интервал получается умножением этого коэффициента на стандартное расстояние между строками, определённое для используемых в строке шрифтов. Значение коэффициента изменяется с помощью команды `\renewcommand` (как и коэффициент `\baselinestretch`).

3. Размещение таблиц

При использовании окружения **tabbing** таблица обрабатывается в текстовой моде, в том месте, где она написана и обрабатывается по строкам, т. е., при необходимости переносится на следующую страницу.

При использовании окружения **tabular** создаётся неразрывный бокс в месте написания таблицы а содержимое таблицы обрабатывается не последовательно, как текст, а целиком, после прочтения всей таблицы. Во первых, это приводит к увеличению времени компиляции (по сравнению с **tabbing**), а во вторых, может привести к переполнению страницы (когда таблица не помещается по высоте страницы). Поэтому, чтобы не замедлять работу компилятора, следует, по мере возможностей использовать окружение **tabbing**.

Для правильного расположения сложных таблиц надо их создавать, как *плавающий объект.*

> ***Плавающим*** *называется объект, расположением которого (в выходном файле) занимается сам компилятор.*

В LaTeX 2_ε имеется три таких объекта: *таблица*, *рисунок* и *заметка на полях.*

Для задания плавающих таблиц в LaTeX'е применяются два окружения: `table[положение]` и `table*[положение]`. В одноколоночной печати эти окружения не различаются, а в двухколоночной — первое окружение всегда размещает таблицу в одной колонке, а второе — в двух (на страницу, как в одноколоночной печати).

Аргумент *положение* задаёт расположение бокса на странице:

t — (top) расположение *вверху страницы*;

b — (bottom) расположение *внизу страницы*, в случае неудачи, LaTeX будет пытаться разместить объект вверху, затем внизу последующих страниц;

p — (page) расположение *на отдельной странице*, содержащей только плавающие объекты;

h — (here) расположение, *по возможности на месте*, сразу же после заполнения текущей строки, в случае неудачи, `h` заменяется на `t` (не допускается в **table*** при печати в две колонки);

! — расположение *на указанном месте*, игнорируя большинство запретов, используется перед дескрипторами `h`, `t` и `b` (хотя бы с одним).

Опции действуют независимо от порядка перечисления. По умолчанию действует опция `[tbp]`.

Обычно, при использовании `\caption` ставят метку (`\label`), ставить её нужно обязательно после `\caption` (см. табл. 6.1). Например:

```
(см.␣табл.~\ref{pi})
```

Таблица 6.1.

Пример таблицы

Выражение с π	Значение
π	3.1416
π^{π}	36.46
$(\pi^{\pi})^{\pi}$	80662.7

Например:

```
\begin{table}[htb]
\centering
\caption{Пример таблицы}\label{pi}
\renewcommand{\arraystretch}{1.2}
\begin{tabular}{||c|r@{.}l||}
\hline
Выражение с $\pi$       &
\multicolumn{2}{c||}{Значение} \\
\hline\hline
$\pi$                  & 3&1416  \\
$\pi^{\pi}$            & 36&46   \\[1mm]
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\hline
\end{tabular}
\end{table}
```

Лабораторная работа №8

Таблицы

Создайте новый файл и наберите следующий текст.

С помощью окружения tabbing можно делать несложные таблицы без обрамления:

раз два три четыре
    два
        три
            четыре
        три
            четыре
    два
раз два три четыре
гиппопотам аллигатор
раз два
три четыре
раз два
три четыре
one two three four
un deux trois quatre

Для построения более сложных таблиц используется окружение tabular, чтобы сделать настоящую таблицу применяют окружение table, например табл. 1.

Таблица 1.

Пример таблицы

Программное обеспечение			
Год	Цена		Комментарий
	мин.	макс.	
1971	97–245		Ориентация, в основном, на большие ЭВМ
1980	180–320		Появляются первые микрокомпьютеры
1990	300–800		Ориентация на мобильные операционные системы

Для небольших, узких таблиц лучше использовать окружение floatingtable из пакета floatflt.

1984	десятичная
11111000000	двоичная
3700	восьмеричная
7C0	шестнадцатеричная

Текст текст

1

Глава 7

Графика в LaTeX 2_ε

1. Основные понятия

Способы воспроизведения графики в LaTeX'е:

1. Набор шрифтов с элементарными линиями (окружение **picture** в LaTeX'е, пакет XY-pic).
2. Макропакеты для рисования, дополняющие и расширяющие окружение **picture** (**epic**, **eepic**, **feynman**, , **bg**, **bidding**, **crossword**, **cwpuzzle**).
3. Шрифты с картинками, созданные и встраиваемые METAFONT'ом или METAPOST'ом и основанные на них пакеты (**mfpic**, FeynMF, **timing**, **circ**, , **chess**, **bdfchess**, **cchess**, **go**).
4. ASCII-рисование — рисование кривых точками (PICTEX,).
5. Встраиваемый PostScript (PSTrics, **axodraw**).
6. Включение графических файлов разных форматов (в зависимости от драйверов и набора графических фильтров), созданных графическими редакторами (пакеты **epsf**, **epsfig**, **graphics**, **graphicx**).
7. Создание изображений в графических редакторах с последующей конвертацией в какой-либо внутренний формат LaTeX'а (m4, GNUPlot, MAPLE, MatLab, Mathematica,...).

Таблица 7.1.

Наиболее распространённые драйверы

Драйвер	Автор	ОС
`dvips`	Tom Rokicki	все ОС
`dview`	Василий К. Малышев	Windows
`dviwin`	Hippocrates Sendoukas	Windows
`emtex`	Eberhard Mattes	MS DOS, OS/2
`xdvi`	Paul Vojta	UNIX
`windvi`	Fabrice Popineau	Windows

2. Импортирование графики

Вставка (импорт) графических файлов в TEX-документ осуществляется с помощью внешних *программ-драйверов* (см. табл. 7.1), выполняющих вывод на внешние устройства (монитор, принтер).

Для непосредственного обращения к драйверам существует команда `\special`, имеющая множество аргументов, специфичных для каждого драйвера. Чтобы упростить работу пользователя, создано много пакетов для работы с внешними устройствами, в том числе и вставки графических файлов.

На сегодня широко известны такие пакеты, как **epsf** *Тома Рокицки* (*Tom Rokicki*), **epsfig** *Себастьяна Раца* (*Sebastian Rahtz*), **graphics** и **graphicx** *Дэвида Карлисли* (*David Carlisle*) и *Себастьяна Раца*.

Последние два пакета являются наиболее современными и развитыми, они способны вставлять графические файлы любых форматов, поддерживаемых драйверами, а не только .eps файлы (как первые два пакета). Кроме того, коллекция graphics содержит ещё некоторые пакеты для работы с внешними устройствами.

Пакеты **graphics** и **graphicx** осуществляют одинаковый доступ к драйверам и вводят одинаковый набор команд, они различаются лишь способом задания необязательных аргументов.

Наиболее совершенным и удобным пакетом является **graphicx** из коллекции graphics

При подключении пакета можно выбрать *драйвер* из следующего списка:

`dvips` (используется по умолчанию), `xdvi`, `dvipdf`, `pdftex`, `dvipsone`, `dviwindo`, `emtex`, `dviwin`, `pctexps`, `pctexwin`, `pctexhp`, `pctex32`, `truetex`, `tcidvi`, `oztex`, `textures`.

Импортирование графических файлов осуществляется с помощью команды `\includegraphics[параметры]{файл}` или `\includegraphics*[параметры]{файл}`. Модифицированная команда обрезает часть рисунка, выходящую за пределы заданного бокса.

Чтобы включить в печатный документ рисунок, записанный в графическом *файле*, достаточно в нужное место в исходном файле вставить команду `\includegraphics{файл}`. В качестве разделителя директорий используется не '\' а '/', как в UNIX.

Например:

■

`\includegraphics{zip}`

Параметры отделяются друг от друга запятой. Параметры записываются в виде равенства, в левой части которого стоит наименование параметра, а в правой — его значение. Многие параметры имеют размерные величины. По умолчанию используется единицы измерения **bp** (большие пункты, 1 in=72 bp).

bb — параметр, используемый для задания всех координат углов ограничивающего бокса (**BoundingBox**). Значение параметра должно содержать четыре числа, разделённых пробелами.

Например, `bb=0 0 1in 2in` задаёт ограничивающий бокс с координатами нижнего левого угла (0,0) и с координатами верхнего правого угла — (72,144).

bbllx, bblly — x- и y-координаты нижнего левого угла ограничивающего бокса.

bburx, bbury — x- и y-координаты верхнего правого угла ограничивающего бокса.

Набор ключей `bbllx=a, bblly=b, bburx=c, bbury=d` эквивалентен `bb= a b c d`.

natwidth, natheight — естественная ширина и высота рисунка. Эти параметры удобны для установки координат верхнего правого угла в случае, когда обе координаты нижнего левого угла равны нулю.

Набор ключей `natwidth=w, natheight=h` эквивалентны `bb= 0 0 w h`.

hiresbb — указывает, что LaTeX должен считывать размеры ограничивающего бокса из строки, начинающейся с `%%HIResBoundingBox`, а не из строки `%%BoundingBox`.

`hiresbb=true` эквивалентен простому заданию параметра `hiresbb` без значения.

В случае, когда по умолчанию считываются размеры изображения с высоким разрешением, ключ `hiresbb=false` позволяет импортировать первый рисунок.

clip — может принимать значения `true` или `false`. Если в параметрах присутствует запись `clip=true` (или просто `clip`), то часть рисунка, выходящая за границы видимой области отсекается.

viewport — задаёт видимую часть рисунка. Подобно параметру `bb`, видимая часть рисунка определяется двумя парами координат нижнего левого угла и верхнего правого угла видимой области.

Координаты углов видимой области отсчитываются от левого нижнего угла ограничивающего бокса.

trim — альтернативный способ задания видимой части рисунка. Положительные числа соответствуют смещению границы видимой области внутрь ограничивающего бокса, отрицательные числа расширяют видимую область. Например, `trim= 1mm 2mm 3mm 4mm` отрезают от рисунка 1 мм слева, 2 мм снизу, 3 мм справа и 4 мм сверху.

draft — устанавливает черновой режим импортирования, когда вместо рисунка выводится рамка с размерами ограничивающего бокса, а внутри неё написано название импортированного файла. Параметр `draft` эквивалентен заданию `draft=true`.

Если включена опция `draft` пакета graphicx или класса документа, то параметр задание `draft=false` позволяет реально импортировать отдельный рисунок.

scale — изменение размера рисунка по заданному масштабу. Ключ `scale=2` увеличивает рисунок в два раза относительно его естественного размера. Пропорции рисунка сохраняются.

width — требуемая ширина рисунка.

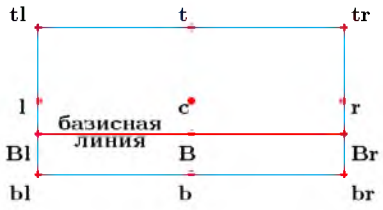
hieght|totalheight — требуемая *высота* / *полная высота* рисунка, соответственно. Задание одного из параметров `width`, `height`, `totalheight` приводит к изменению масштаба рисунка с сохранением пропорций.

keepaspectratio — означает `keepaspectratio=true`, задаёт режим масштабирования рисунка с сохранением пропорций. Рисунок увеличивается настолько, насколько это возможно при сохранении его аспектного отношения без выхода изображения за заданные размеры видимой области.

angle — угол поворота рисунка (в градусах), положительное значение задаёт вращение против часовой стрелки.

origin — определяет положение центра вращения рисунка, по умолчанию рисунок вращается относительно точки привязки.

Значение ключа может состоять из одной или двух букв, фиксирующих положение центра вращения. Горизонтальная координата центра вращения задаётся одним из трёх спецификаторов: `l` — слева, `c` — центр, `r` — справа, в то время как вертикальная координата определяется одним из четырёх спецификаторов: `t` — вверх, `c` — центр, `b` — низ, `B` — базисная линия.



Например, `origin=rb` задаёт в качестве точки вращения правый нижний угол бокса, `origin=lt` задаёт левый верхний угол. Комбинация `cB` задаёт середину отрезка базисной линии, проходящей через вращаемый бокс, а `lc` задаёт среднюю по высоте точку с левой стороны бокса. Точке привязки соответствует `lb`. Всего таким способом можно задать 12 точек. Порядок следования спецификаторов не имеет значения: `br` эквивалентно `rb`. Если только задан один спецификатор, то вторым предполагается `c`.

type — задаёт тип рисунка. Все файлы одного типа обрабатываются при помощи одних и тех же внутренних команд `\spesial` (конкретная форма которых зависит от выбранного драйвера).

Например, файлы с расширениями `.ps` и `.eps` рассматриваются драйвером `dvips` как файлы одного типа векторной графики *EPS*, а файлы с расширениями `.bmp` и `.pcx` считаются принадлежащими к типу растровой графики *BMP*. Поэтому следует задать ключ `type=eps`, если рисунок *PostScript* почему-то записан в файл с расширением, не совпадающим с `.eps` и `.ps`.

ext — определяет расширение имени графического файла. Например, если задан ключ `ext=eps`, то драйвер выходного устройства попытается загрузить файл с именем *файл*`.eps`.

read — определяет расширение имени файла, в котором записаны размеры ограничивающего бокса в виде строки `%%BoundingBox: llx lly urx ury`, который считывается LaTeX'ом при компиляции исходного текста печатного документа. Для графического файла с расширением `.eps` размеры считываются из этого же файла, поэтому применение ключа `read=.eps` повторяет правило, используемое по умолчанию.

command — задаёт команду, которую программа-драйвер должна применить к графическому файлу перед его импортированием. С eps-файлом не требуется производить каких-либо предварительных действий, поэтому этот ключ задавать не нужно. Обычно его используют при импортировании сжатых файлов.

Запомните: *Если параметры пропущены, то рисунок будет вставлен с естественными размерами*, т. е. размерами ограничивающего бокса (в .eps файле размеры записываются в параметре BoundingBox).

Например:

■ Здесь вставлен рисунок, не превысив заданных размеров и не изменив пропорции.

```
\includegraphics[keepaspectratio,
  width=2cm, height=10cm]{rosette}
```

Высота рисунка не достигла заказанной величины в 10 сантиметров дюйма, т. к. ширина рисунка превысила бы заказанное значение в 2 сантиметра или были бы искажены пропорции рисунка.

В случаях, когда в печатный документ (используя драйвер `dvips`) нужно вставить *растровый* рисунок, необходимо указать его размеры в аргументе команды `\includegraphics` или в специальном текстовом файле. С помощью этих размеров осуществляется и масштабирование.

3. Размещение рисунков

Рисунки, вставляемые в текст из внешних файлов и рисунки, созданные средствами самого LaTeX'а образуют бокс заданного размера и обрабатываются как все боксы.

Часто требуется рисунок отделить от текста, снабдить его подписью и разместить нужным образом на странице. Для автоматизации этого процесса имеется специальное окружение, помещающее рисунок (вообще — любое содержимое) в **плавающий бокс**.

Плавающий рисунок задаётся, подобно плавающим таблицам, окружениями `figure[положение]` и `figure*[положение]`. В двухколоночной печати, первое окружение всегда размещает рисунок в одной колонке, а второе — в двух (на всю ширину страницы, как в одноколоночной печати).

Необязательный аргумент *положение* задаёт расположение бокса на странице, он может состоять из следующих символов:

h — (here) расположение на месте, сразу же после заполнения текущей строки (не допускается в *-форме при печати в две колонки);

t — (top) расположение вверху страницы;

b — (bottom) расположение внизу страницы;

p — (page) расположение на отдельной странице, содержащей только плавающие объекты.

! — расположение *на указанном месте*, игнорируя большинство запретов, используется перед дескрипторами `h`, `t` и `b` (хотя бы с одним);

Подпись создаётся командой `\caption{подпись}`, которая ставится внутри окружения **figure** после самого рисунка, при этом LaTeX сформирует соответствующую подпись с номером рисунка, как, например рис. 7.1.

Например:

■

```
\begin{figure}[hbtp]
\centering
\includegraphics[width=1.\textwidth,
  bb=59 366 507 519,clip]{pic1}
\caption{Результаты классификации}
\label{SamplePic}
\end{figure}
```

Подробнее о плавающих объектах написано в разделе 6, описывающем таблицы.

Для вставки рисунка, обтекаемого текстом (справа или слева), существуют специальные пакеты:

- `floatflt` — для плавающих рисунков;
- `wrapfig` — для неплавающих рисунков.

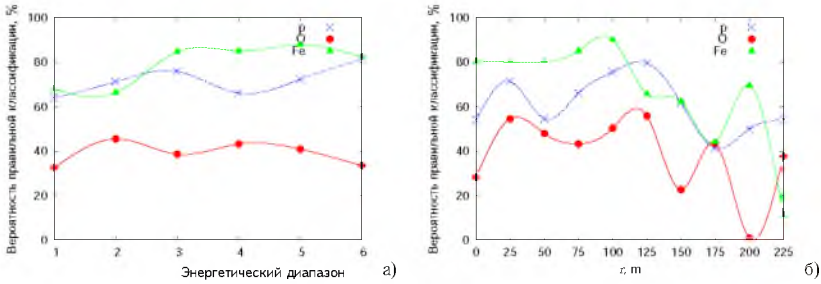
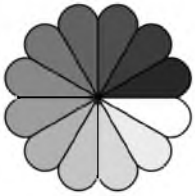


Рис. 7.1. Результаты классификации

Так, с помощью пакета floatflt, можно вставить обтекаемый рисунок, сформированный окружением `floatingfigure`:



Текст текст.

```
\begin{floatingfigure}{.3\textwidth}
\includegraphics[width=.23\textwidth,height=.25\textheight,
keepaspectratio]{rosette}
\end{floatingfigure}
Текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст.
```

4. Псевдографика

LaTeX имеет несколько шрифтов псевдографики (элементарных линий разных размеров), которые используются исключительно в **графическом режиме** (графической моде). Для перехода в этот режим служит окружение `picture(ширина,высота)($x_0, y_0$)`. *Ширина* и *высота* задают размеры графического бокса и являются обязательными, а x_0 и y_0 задают координаты левого нижнего угла и являются необязательными, по умолчанию $x_0 = 0$, $y_0 = 0$.

Аргументы окружения **picture** (как обязательные, так и необязательные) задаются в круглых скобках, чтобы подчеркнуть, что они являются парами длин (по осям X и Y).

Размерность не указывается а используется текущая единица измерения, задаваемая длиной `\unitlength` (по умолчанию она равна 1 pt), которая переопределяется до окружения **picture**. Такие же правила задания длин и координат действуют для всех команд в графическом режиме.

Графические элементы могут изображаться линиями разной толщины:

- `\thinlines` — тонкие линии (используется по умолчанию);
- `\thicklines` — толстые линии.

Команда `\linethickness{толщина}` служит для произвольного задания *толщины* горизонтальных и вертикальных линий.

4.1. Позиционирование

Любой объект в графике должен быть помещён в определённую точку, задаваемую парой координат (x, y) (иначе объект будет помещён в точку (0,0)). Команда `\put($x, y$){объект}`, размещает *объект* так, чтобы его точка привязки находилась в точке с координатами (x, y).

Для размещения многократно повторяющегося объекта существует команда `\multiput($x, y$)($dx, dy$){$n$}{объект}`, — создаёт n

копий *объекта*, размещая их так, чтобы точка привязки i-й копии находилась в точке с координатами $(x + [i-1]dx, y + [i-1]dy)$.

4.2. Линии

Для задания *отрезков* (прямых линий) используется команда `\line(`l_x, l_y`){`*длина*$_x$`}`, где пара (l_x, l_y) задают относительные проекции линии на соответствующие координатные оси, тем самым определяя наклон линии. l_x и l_y могут принимать только целые значения из диапазона $[-6;\ 6]$ и не могут иметь общего делителя. Длина линии определяется проекцией на ось X, задаваемом в параметре *длина*$_x$.

Кроме линий существует возможность рисовать *векторы* (линии со стрелками на конце). Векторы задаются командой `\vector(`l_x, l_y`){`*длина*$_x$`}`, её аргументы имеют такой же смысл как и в команде `\line`, с единственным отличием: l_x и l_y могут принимать значения из диапазона $[-4;\ 4]$.

В LaTeXe имеется возможность построения кривых второго порядка. Команда `\qbezier[`N`](`x_A, y_A`)(`x_B, y_B`)(`x_C, y_C`)` рисует квадратичную кривую Безье. Для задания кривой Безье второго порядка необходимо задать начальную (A) и конечную точки (C) а также контрольную точку (B), в которой пересекаются касательные, проведённые к параболе в граничных точках.

Необязательный аргумент N задаёт количество точек, используемых для аппроксимации кривой. По умолчании это число определяется параметром `\qbeziermax`, его можно изменить с помощью команды `\renewcommand`.

4.3. Круги

Команда `\circle{`*диаметр*`}` рисует окружность ○.

Команда `\circle*{`*диаметр*`}` рисует круг ●.

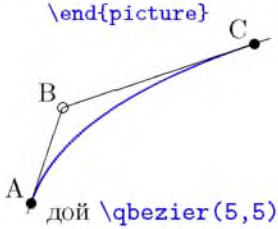
Точкой привязки этих объектов служит центр окружности. *Диаметр* может задаваться любым положительным числом, однако реальный диаметр окружности будет выбран LaTeX'ом из

набора имеющихся шрифтов, наиболее близкий к заданному. Максимальный диаметр для окружности обычно составляет 40 pt а для круга — в два раза меньше.

Например:

■

```
\begin{picture}(100,40)(10,10)
{\color{MyBlue}\qbezier(5,5)(15,35)(75,55)}
\put(5,5){\circle*{3}}
\put(3,7){\makebox(0,0)[br]{A}}
\put(15,35){\circle{3}}
\put(13,37){\makebox(0,0)[br]{B}}
\put(75,55){\circle*{3}}
\put(73,57){\makebox(0,0)[br]{C}}
\put(15,35){\line(-1,-3){11}}
\put(15,35){\line(3,1){65}}
\end{picture}
```



Здесь кривая Безье задана командой `\qbezier(5,5)(15,35)(75,55)`.

4.4. Овалы

Овал *(в LaTeX'е) — это прямоугольник с максимально скруглёнными углами* ⬭.

Команда `\oval(ширина,высота)[часть]` строит овал с заданной *шириной* и *высотой*. Точка привязки овала находится в его центре.

Необязательный аргумент задаёт *часть* овала, он может состоять из одного или двух спецификаторов:

`l` — левый; `r` — правый;
`t` — верхний; `b` — нижний.

Если задан один спецификатор, то рисуется соответствующая половина овала, если заданы два совместимых спецификатора, то — соответствующая четверть овала.

Чтобы нарисовать скругление меньшего радиуса, используются четверти овалов соответственно меньшего размера.

4.5. Боксы

В графической моде текстовые боксы имеют другой синтаксис и в этих боксах не добавляются окружающие пробелы вокруг текста. Команды выглядят следующим образом:

`\makebox(ширина,высота)[позиц.]{текст}`;

`\framebox(ширина,высота)[позиц.]{текст}`.

Позиционирование может состоять из одного или двух спецификаторов:

l — текст сдвинут к левому краю бокса;

c — текст расположен по центру бокса;

r — текст сдвинут к правому краю бокса;

s — текст растянут на всю ширину боксу;

t — текст сдвинут к верхнему краю бокса;

b — текст сдвинут к нижнему краю бокса.

По умолчанию текст центрируется.

Кроме перечисленных боксов, в графическом режиме добавляются ещё несколько команд:

`\dashbox{штрих}(ширина,высота)[позиц.]{текст}` — для боксов, обведённых пунктирным контуром, длина пунктира задаётся параметром *штрих*, измеряемом в единицах `\unitlength`.

`\frame{текст}` — упрощённый вариант команды framebox, аналог команды `\fbox` в текстовой моде.

`\shortstack[позиционирование]{текст}` — формирует стек. Текст внутри бокса располагается в одну колонку, строки разделяются командой `\\`. Опция позиционирование позволяет выравнивать текст по горизонтали, допустимые значения: `r`, `l`, `c` (используется по умолчанию).

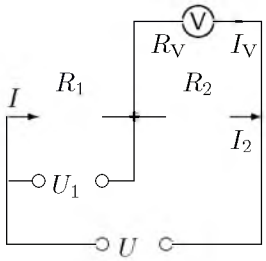
Эта команда может использоваться не только в графическом, но и в любом другом режиме.

Для сохранения бокса также существует «графический вариант»: `\savebox{имя}(ширина,высота)[позиц.]{текст}`. Здесь *имя* также задаёт имя сохраняемого бокса. Рекомендуется использовать эту команду для сложных рисунков из повторяющихся графических элементов, т. к. с этой командой компиляция происходит быстрее, чем с использованием вложенных команд `\multiput`.

Например:

■

```
\begin{picture}(80,100)(0,-50)
\put(10,-20){\line(1,0){8}}
\put(50,-20){\line(-1,0){8}}
\put(10,-40){\line(1,0){28}}
\put(90,-40){\line(-1,0){28}}
\put(20,-20){\circle{4}}
\put(10,3){$I$}
\put(40,-20){\circle{4}}
\put(40,-40){\circle{4}}
\put(45,-44){$U$}
\put(60,-40){\circle{4}}
\put(10,0){\line(1,0){10}}
\put(10,0){\vector(1,0){10}}
\put(40,0){\line(1,0){20}}
\put(24,-24){$U_1$}
\put(80,0){\line(1,0){10}}
\put(80,0){\vector(1,0){10}}
\put(80,-10){$I_2$}
\put(50,0){\circle*{3}}
\put(50,-20){\line(0,1){50}}
\put(90,30){\line(0,-1){70}}
\put(10,0){\line(0,-1){40}}
\put(50,30){\line(1,0){15}}
\put(75,30){\line(1,0){15}}
\put(75,30){\vector(1,0){10}}
\put(80,20){$I_{\text{V}}$}
\thicklines
\put(20,-5){\framebox(20,10){}}
\put(25,8){$R_1$}
\put(60,-5){\framebox(20,10){}}
\put(65,8){$R_2$}
\put(70,30){\circle{10}}
\put(70,30){\makebox(0,0){\small\sf V}}
\put(55,20){$R_{\text{V}}$}
\end{picture}
```



5. Цвет

Для вывода в цвете TEX'у также нужно указать драйвер устройства. Чтобы задействовать цвет надо подключить пакет color из

коллекции graphics. Представление цвета зависит не от LaTeX'а (пакета **color**), а от используемого драйвера.

Обычно все драйверы поддерживает 4 цветовые модели:

gray — задаётся числом от 0 до 1.

rgb — задаётся тройкой чисел от 0 до 1.

cmyk — задаётся четвёркой чисел от 0 до 1.

named — задаётся именем. Драйвер `dvips` содержит определения 68 имён цветов.

5.1. Определение цвета

Любой драйвер поддерживает цвета `black`, [redacted], red, green, blue. cyan, magenta и [redacted],

Для определения произвольного цвета существует команда `\definecolor{имя}{модель}{описание}`.

Например:

■

```
\definecolor{MyBlue}{rgb}{.6,.8,1}
```

Цвет `RoyalBlue` (rgb(65,105,225)) определён только в драйвере `dvips`.

```
\definecolor{RBlue}{rgb}{.255,.412,.88}
```

5.2. Использование цветов

Для задания цвета текста имеется декларация

- `\color{цвет}` или
- `\color[модель]{описание}`

и команда

- `\textcolor{цвет}{текст}` или

- `\textcolor[модель]{описание}{текст}`.

Цвет страницы изменяет декларация `\pagecolor{цвет}`, где вместо имени *цвета* также можно задать цвет используя его *описание* в к.-л. цветовой *модели*.

Цветные текстовые боксы задаются командами

- `\colorbox{цвет фона}{текст}` и
- `\fcolorbox{цвет рамки}{цвет фона}{текст}`,

которые действуют и регулируются как команда `\fbox`.

Цвета фона и *рамки* также можно указывать явно, в описаниях цветовой *модели*.

Например:

■ Текст цвета cyan с вкраплением зелёного на фоне цвета Salmon.

```
{\pagecolor[named]{Salmon}  \color{cyan}Текст
цвета \colorbox[named]{OliveGreen}{\texttt{cyan}}
 с вкраплением \textcolor{green}{\tt{зёленого}}
 на фоне цвета \fcolorbox{green}{cyan}%
  {\color[named]{Salmon}Salmon}.  }
```

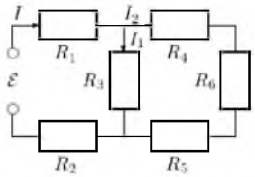
Для более сложной работы с цветом можно использовать пакет для цветных таблиц (colortbl) и пакеты для подготовки слайдов (seminar, а лучше — класс **beamer**).

Лабораторная работа №9

Графика в LaTeXe

Создайте новый файл и наберите следующий текст.

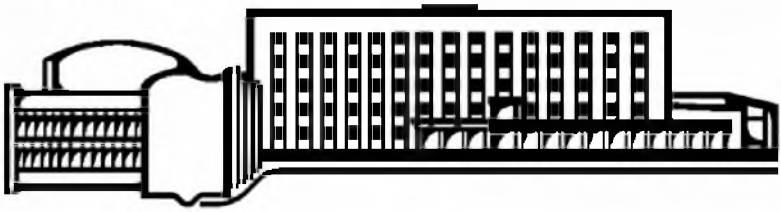
В окружении picture можно рисовать простые картинке прямо в LaTeX'е.



Пользуясь пакетом graphicx можно вставлять графические файлы различных форматов.

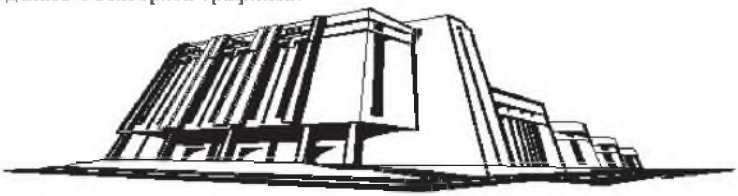
Файл P:/tex/GRAPH/ASU.eps:

Файл P:/tex/GRAPH/ASUd.png:



Файл P:/tex/GRAPH/ASUlogo.jpg:

Кроме того, их можно масштабировать и поворачивать, только лучше это делать с векторной графикой.



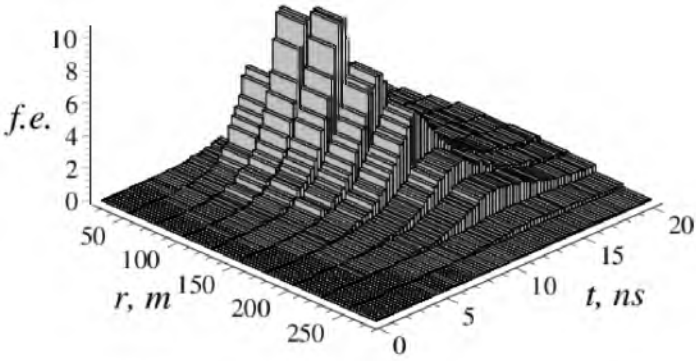


Рис. 1. Пример рисунка `P:/tex/SAMPLES/fe5m.eps`

Для вставки рисунков, обтекаемых текстом можно использовать окружение floatinfigure из пакета floatflt.

Текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст
текст текст текст текст текст текст

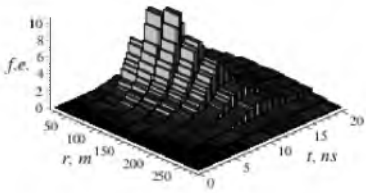


Рис. 2. Тот же пример, но окружённый текстом

текст текст текст текст текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст текст текст текст текст
текст текст текст текст текст текст текст текст текст текст текст текст
текст текст текст текст текст текст

Приложение А

BibTeX

1. Введение в BibTeX

BibTeX — это утилита, предназначенная для создания библиографического списка (списка цитируемой литературы). Программу написал Орен Паташник (*Patashnik, Oren*).

BibTeX собирает затребованный библиографический список; вся необходимая информация берётся из файлов библиографических баз данных (имеющих расширение bib), оформляется список в одном из заданных стилей (стилевые файлы имеют расширение bst). На рис. 1.1 продемонстрирована работа BibTeXа:

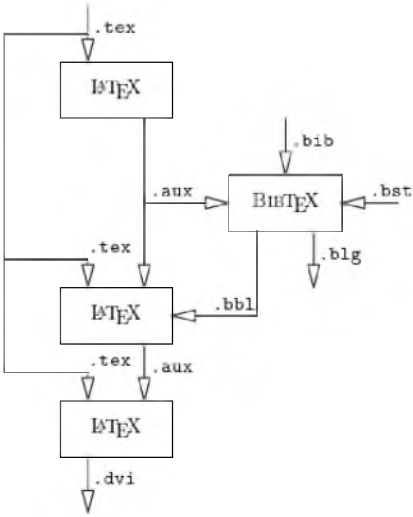


Рис. 1.1. Схема работы утилиты BibTeX

- после первой компиляции LaTeXа в .aux файл записываются ссылки, образованные командой cite;
- на основе этого файла

формирует файл-список, с расширением bbl (кроме него создаётся ещё файл протокола компиляции — .blg);

- после второй компиляции LaTeXа содержимое файла bbl включается в документ;
- после третьей компиляции LaTeXа проставляются ссылки на литературу.

Для работы с BibTeXом необходимо в исходном файле, в месте размещения библиографического списка, вставить команду `\bibliography{bib-файлы}`, где *bib-файлы* — имена .bib файлов (библиографических баз данных), разделённые запятыми. В месте написания этой команды появится в .dvi файле список литературы.

Библиографический стиль (используемый для оформления списка) задаётся командой `\bibliographystyle{bib-style}`, которая обязательно должна стоять раньше команды `\bibliography`, так как BibTeX не имеет стиля «по умолчанию».

Существует 4 стандартных BibTeXовских стиля:

plain *Открытый стиль.* Список формируется в алфавитном порядке по именам авторов, затем по годам, далее по названиям работ. Каждая запись помечается порядковым номером. Ссылку на публикацию печатает команда стандартная команда `\cite[примечание]{метка}`, используя *метку*, которым публикация помечена в базе данных.

unsrt *Несортирующий стиль.* Отличается от plain тем, что список литературы упорядочивается в порядке цитирования в тексте.

alpha *Алфавитный стиль.* Отличается от plain тем, что публикации вместо номеров помечаются аббревиатурой, состоящей из фамилии автора и года издания (например: “Knu83”). Библиография сортируется по этим меткам, затем по именам авторов, по годам изданий, по названиям.

abbrv *Аббревиатурный стиль.* Отличается от plain тем, что вместо полных имён авторов, названий месяцев и журналов печатаются сокращённые.

Как уже говорилось, список литературы формируется только из затребованных именований, т. е. в списке будут только те публикации, на которые имеются в тексте ссылки (команды `\cite`).

BibTeX вводит дополнительную команду `\nocite{метка}`, которая ничего не печатает в месте применения, но помещает соответствую публикацию в список литературы. Команда `\nocite{*}`, помещает в список литературы все публикации библиографической базы (из файлов, перечисленных в команде `\bibliography`).

2. Библиографическая база данных

Файл библиографической базы данных должен иметь расширение bib и содержать последовательность записей. Каждая запись состоит из трёх основных элементов: указателя *типа* (которому предшествует знак `@`), *ключевого слова*, *данных* о публикации. Например:

```
@book{kotelnikov/chebotaev:1998,
AUTHOR   ="Котельников И. А. and Чеботаев П. З.
TITLE    ="Издательская система \LaTeXe
PUBLISHER={Сибирский хронограф},
ADDRESS  ={Новосибирск}
YEAR     = 1998          }
```

Эта запись имеет тип “book”, метку “kotelnikov/chebotaev:1998”[1], после которой следуют данные.

Данные записываются в виде названий полей и их значений, отделённых знаком ‘=’. Названия полей можно писать прописными или строчными буквами. Значения могут быть либо текстовыми, либо числовыми. Текстовое значение должно быть окружено двойными кавычками (`"`). В `строке`символом *не должно быть непарных фигурных скобок*, причем, `\{` и `\}` также учитываются. Метка и все поля разделяются запятыми. Числовое значение можно писать в кавычках или без них.

[1] Для задания меток лучше завести для себя определённые правила или воспользоваться уже существующими. Здесь использована гарвардская схема.

2.1. Записи в базе данных

BibTeX имеет достаточное количество *типов записей*, которые способны удовлетворить любые требования. Ссылки на различные публикации содержат различную информацию. Если Вы создаёте ссылку на журнальную статью, то она может включать том и номер журнала, которые, как правило, отсутствуют при на ссылке на книгу. Поэтому записи в базе данных для различных типов имеют различный набор полей. Для всех типов записей поля можно разделить на три класса:

required — обязательное поле. Если поле пропущено, то появляется сообщение об ошибке и это может плохо сказаться на форматировании записи в списке литературы. Когда требуемая информация является неосмысленной, то выбран неправильный тип записи. Однако, если требуемая информация уже содержится в других полях, то можно проигнорировать предупреждение об ошибке.

optional — необязательное поле. Если присутствует информация в этом поле, то она будет использована, но её отсутствие не вызовет проблем с форматированием. Полезно заполнять необязательные поля, если они предусмотрены типом записи, так как они могут помочь в отыскании публикаций.

ignored — игнорируемое поле. BibTeX проигнорирует любое поле, которое не входит в предыдущие классы, но есть в выбранном типе записи. Поэтому, даже если часть полезной информации о ссылке, включённой в соответствующую запись в файле, никогда не появится при составлении списков литературы, данное поле все равно будет полезно. Например, если нужно хранить аннотацию публикации в базе данных, то можно добавить поле abstract. Библиографическая база данных может хранить массу полезных комментариев на прочитанную литературу.

Далее перечислены все типы записей со всеми обязательными и необязательными полями, которые используются стандартными библиографическими стилями. Выбранный библиографический стиль может проигнорировать некоторые необязательные поля при составлении списка литературы.

Запомните: признаком типа записи в файле базы данных является знак @.

article Статья в журнале.
Обязательные поля: author, title, journal, years.
Необязательные поля: volume, numbers, pages, month, note.

book Книга с указанием издательства.
Обязательные поля: author или editor, title, publisher, years.
Необязательные поля: volume, numbers, series, month, address, edition, note.

booklet Отпечатанное и сброшюрованное издание без названия издательства или имени автора.
Обязательные поля: title.
Необязательные поля: month, address, howpublished, note.

conference Статья в трудах конференции.
Обязательные поля: author, title, booktitle, years.
Необязательные поля: month, address, pages, organization, editior,publisher, note.

inbook Часть книги, которая может быть главной или рядом страниц.
Обязательные поля: author или editor, title,chapter, pages, publisher, years.
Необязательные поля: volume, numbers, series, month, address, edition, type, note.

incollection Часть книги со своим заглавием.
Обязательные поля: author, title, publisher, booktitle, years.
Необязательные поля: volume, numbers, series, month, address, editor, edition, type, chapter, pages, note.

inproceedings То же, что и conference.

manual Техническая документация.
Обязательные поля: title.
Необязательные поля: author, organization, month, year, address, edition, note.

mastersthesis Дипломная работа.
Обязательные поля: author, title, school, years.
Необязательные поля: month, address, note.

misc Используйте этот тип, когда нет ничего более подходящего.
Обязательных полей нет.
Необязательные поля: author, title, howpublisher, month, years.

phdthesis Диссертация.
Обязательные поля: author, title, school, years.
Необязательные поля: month, address, note.

proceedings Труды конференции.
Обязательные поля: title, years. Необязательные поля: month, address, editior, publisher, organization, note.

techreport Доклады, опубликованные школами или институтами, обычно нумерованные внутри серии.
Обязательные поля: author, title, institution, years.
Необязательные поля: type, numbers, address, month, note.

unpublished Документ с автором и заглавием, но формально неопубликованный.
Обязательные поля: autor, title, note.
Необязательные поля: month, years.

В записи любого типа может содержаться необязательное поле key, которое в некоторых стилях определяет порядок расположения ссылок по алфавиту или формировании меток командой bibitem. Также необходимо включить key поле в любую запись, у которой поля author и editor опущены. Не путайте поле key с меткой.

2.2. Текст поля

Как уже говорилось, текст поля может быть заключен в фигурные скобки или в двойные кавычки.

Текст поля author (автор) или editor (редактор) — это список фамилий авторов и издателей. Библиографический стиль определяет печатать или нет сначала имя и отчество (инициалы) автора (издателя), а затем его фамилию, или наоборот. Имя и фамилию автора лучше набирать полностью, библиографический стиль сам

решит, как сократить имя. Если это не возможно, то лучше написать точно, как в оригинале. Большинство фамилий и имён можно ввести в базу данных с запятой или без нее:

```
"Иван Иванович Петров"  "Петров, Иван Иванович"
"Ludwig von Beethoven"  "von Bethoven, Ludwig"
```

Для людей с множественными фамилиями форма с запятой обязательна. Например, Per Brinch Hansen, чье имя Per, а фамилия Brinch Hansen, должен быть записан так: `"Brinch Hansen, Per"`. Если напечатать `"Per Brinch Hansen"`, то BibTeX расценит Brinch, как отчество.

BibTeX принимает текст в фигурных скобках как нечто целое. Например, скобки в `"{Barnes and Noble, Inc}"` исключает интерпретацию Inc в качестве имени. В случае, когда мы пишем `"von Bethoven, Ludwig"`, Bethoven означает фамилию, а von — вспомогательное слово, когда `"{von Bethoven}, Ludwig"`, von Bethoven — это фамилия.

Если у литературного источника несколько авторов, то их фамилии разделяются словом and. Для примера:

```
AUTHOR = "Ralf Alpher and Bethe, Hans and George Gamow"
```

Если and, является частью имени, то оно должно быть окружено фигурными скобками: `"{Barnes and Noble, Inc}"`.

В BibTeXе возможно применение символов с акцентами. Например,

```
AUTHOR = "Kurt G{\"o}del,
YEAR = 1934,
```

Чтобы обеспечить правильную обработку символа с акцентом, нужно окружить его фигурными скобками (в примере: `{\"o}` или `{\"{o}}`). При этом первым символом в скобках должен быть '\', и эти фигурные скобки могут быть окружены лишь самыми внешними скобками, которые окружают всю запись в целом.

Библиографический стиль определяет, печатать или нет всё название прописными буквами. BibTeX меняет все прописные буквы на строчные, если этого требует библиографический стиль. Если требуется, чтобы в названии прописные буквы не были заменены на строчные их необходимо заключать в фигурные скобки. Например:

```
"Путешествие в {Америку}"
"Путешествие в {А}мерику"
```

Данные записи эквивалентны.

2.3. Сокращения

Для текста поля возможны сокращения до некоторой аббревиатуры. Аббревиатурой может быть строка символов, которая начинается с буквы и не содержит пробелов или какого-нибудь из десяти следующих символов: `"` `#` `%` `(` `)` `,` `=` `{` `}`

Аббревиатура вводится без фигурных скобок и без кавычек на место текстового поля

Пусть кп аббревиатура строки *Клуб путешественников*, тогда следующие поля эквивалентны:

```
JOURNAL=кп
JOURNAL="Клуб путешественников"
```

Некоторые аббревиатуры изначально определены библиографическим стилем. Например, названия месяцев на английском языке определяются аббревиатурами вида: jan, feb, mar и т. д. Библиографические стили обычно содержат аббревиатуры названий часто цитируемой литературы. В файле базы данных можно определять дополнительные аббревиатуры с помощью команды `@string`. Например,

```
@string{кп="Клуб путешественников"}
```

определяет *кп*, как аббревиатуру данного журнала. В аргументе команды `@string` вместо внешних скобок можно использовать круглые скобки, а фигурные скобки можно использовать вместо кавычек.

Команда `@string` в файле базы данных может стоять в любом месте, но лучше ввести её до первого использования, так что лучшим местом для неё является начало файла. Команда `@string` в файле базы данных имеет старшинство перед определениями, вводимыми библиографическим стилем, так что она может переопределить сокращения, введённые стилем.

2.4. Склеивание строк

Несколько строк можно объединять в одну. Эта операция называется конкатенацией и обозначается символом #. Например, если определена строка

```
\verb|@string{кп="Клуб путешественников"}|
```

тогда легко произвести почти идентичные заголовки для нескольких различных записей:

```
@BOOK{КП-86,
 TITLE = 1986 # КП,

   ...

}

@BOOK{КП-87,
 TITLE = 1987 # КП,

   ...

}
```

и так далее. Можно соединить любое количество строк, ограниченное только общей длиной результирующей строки, определяемой конкретной реализацией BibTeXá.

2.5. Преамбула базы данных

Любой файл библиографической базы данных может иметь свою преамбулу, аналогично главному входному файлу. Она состоит из команды `@preamble`. Синтаксис данной команды сходен с командой `@string`, но не содержит имени строки и знака равенства:

```
@PREAMBLE{"\newcommand{\nosort}[1]{}# ... "}
```

В этом примере преамбула содержит определение новой команды `\nosort` (и, возможно, других, перечисленных через знак #). Команда `\nosort{arg}` имеет один аргумент, но в конечном итоге прячет его. Данную команду можно использовать для изменения

режима сортировки записей в списке литературы. Например, если сделать для двух изданий одной и той же книги поле year следующим образом:

```
YEAR="{\nosort{a}}1978"
YEAR="{\nosort{6}}1971"
```

то BibTeX (в стиле plain) первой в список литературы поместит ссылку на более позднее издание, потому что *а* в алфавите идет раньше *б*, а BibTeX посчитает, что команда `\nosort` делает акцент.

2.6. Перекрёстное цитирование

BibTeX имеет свою схему перекрёстного цитирования. В следующем примере первая запись ссылается на вторую. Например:

■

```
@INPROCEEDING{no-grants,
CROSSREF = "gg-proc
AUTHOR   = "Rocky Gneisser
TITLE    = "No Gnats Are Taken For Granite
PAGES    = "133-139"}

...

@INPROCEEDING{gg-proc,
EDITOR    = "Gerald Ford and Jimmy Carter
TITLE     = "The Gnats and Gnus 1988 Proceedings
BOOKTITLE = "The Gnats and Gnus 1988 Proceedings"}
```

Поле CROSSREF в записи no-grants показывает, что эта запись должна содержать любые пропущенные поля из записи gg-proc. В нашем примере наследуются поля EDITOR и BOOKTITLE. Кроме того, BibTeX автоматически вставит запись gg-proc в список литературы, если на неё ссылаются другие записи (не менее двух раз), даже если запись gg-proc не цитируется командами `\cite` или `\nocite`. Запись, на которую ссылаются другие записи, должна находиться в базе данных после этих записей.

Также действует обычная схема перекрёстного цитирования. Можно сделать ссылку на публикацию в примечании к другой ссылке. Например:

■

```
NOTE = "Джонсон \cite{jones:85} уточнил этот результат"
```

2.7. Записи в базе данных

Далее приводится описание полей, которые распознаются стандартными библиографическими стилями. Также в записях могут содержаться поля, игнорируемые этими стилями.

address Адрес издателя.
Для крупных издательств достаточно указать город. Для малых издательств желательно привести полный адрес.

annote Аннотация.
Стандартными стилями не используется, но может быть использована другими стилями, предназначенными для создания аннотированных библиографических указателей.

author Фамилия автора.
Если фамилий несколько, то они разделяются словом end.

booktitle Заголовок цитируемой литературы.
Для записи типа book вместо booktitle следует использовать title.

chapter Номер главы, параграфа, раздела и т.д.

crossref Ключ, указывающий на запись в библиографической базе данных, на которую имеются перекрёстные ссылки из других записей.

edition Номер издания книги.

editor Фамилия редактора.
Если еще имеется поле author, поле editor указывает редактора книги или сборника, в котором содержится цитируемая публикация.

howpublished Для публикаций, изданных необычным путём.
Буквы в первом слове переводятся в прописные.

institution Институт, поддерживающий работу, или спонсорская организация.

journal Название журнала.
В библиографической базе данных могут быть определены сокращения для названий некоторых журналов.

key Используется для алфавитного расположения и создания меток в тех случаях, когда поля author и editor опущены.

month Месяц опубликования работы или, если она не была опубликована, время написания.
Следует использовать стандартные сокращения названий месяцев.

note Любая дополнительная информация, которая может помочь пользователю.

number Номер журнала, технического доклада или выпуска в серийных изданиях.
Номер журнала обычно состоит из номера тома и номера выпуска в томе.

organization Спонсор конференции или издания.

pages Номер страниц (одной или более) или диапазон страниц.
Стандартные стили трансформируют одинарный дефис (35-36) в двойной `(35--36)`, как того требуют правила LaTeXа.

publisher Название издательства.

school Название учебного заведения, где работа была написана.

series Название серии книг.
Когда цитируется книга, поле title дает ее название, а необязательное поле series даёт
название серии, в которой книга опубликована.

title Название работы.

type Вид технического доклада, например, «Для служебного пользования».

volume Том журнала или том книги в многотомном издании.

year Год публикации или год написания для неопубликованной работы.
Последние четыре символа, отличные от знаков препинания, должны быть цифрами.

Кроме всего вышеизложенного, разработано множество библиографических стилей, пакетов и средств управления библиографическими базами данных. Также имеется документация для написания стилевых BibTeXовских файлов, так что при желании можно исправить, дополнить существующий стиль или написать полностью новый стиль.

Приложение B

Утилита MakeIndex

1. Использование утилиты MakeIndex

Утилита MakeIndex позволяет полностью автоматизировать составление *алфавитного*, *предметного* и *именного указателей*, а также — *словаря терминов*.

Для использования утилиты MakeIndex надо:

- подключить пакет makeidx;
- в преамбулу вставить декларацию `\makeindex` — для создания *алфавитного* (*предметного*, *именного*) указателя и (или) декларацию `\makeglossary` — для создания *словаря терминов*;
- вставить декларацию `\printindex` в месте печати алфавитного указателя;
- в исходном файле пометить необходимые термины.

Замечание: *Для удобства работы над составлением указателя можно использовать пакет* showidx, *который печатает на полях входы в указатель (содержимое команд* `\index`*).*

Для вставки глоссария необходимо сначала определить а затем использовать соответствующую команду или вставить содержимое глоссария с помощью команды `\input`.

Создание указателя с помощью утилиты MakeIndex состоит из нескольких этапов.

1. Компиляция LaTeX, (.idx — для индексного указателя и .glo — для словаря терминов)
2. Компиляция MakeIndex, достаточно написать в командной строке `MakeIndex файл` (без расширения), для компиляции глоссария необходимо указать соответствующие опции (.ind, для глоссария обычно используют .gls).
3. Компиляция LaTeX, содержимое .ind файла будут напечатаны в месте написания команды `\printindex`.

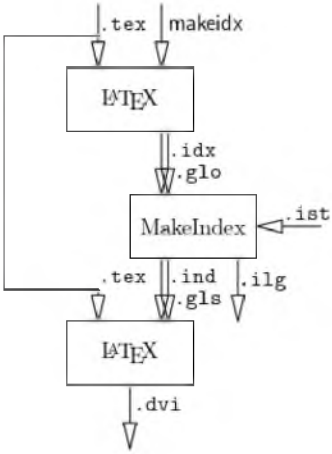


Рис. 2.1. Схема работы утилиты MakeIndex

2. Вход в указатель

Чтобы пометить термин для указателя, необходимо сразу после него вставить команду `\index{текст1!текст2!текст3}`, где *текст1* — вход в указатель первого уровня, *текст2* — вход в указатель второго уровня, *текст3* — вход в указатель третьего уровня; части *текст2* и *текст3* являются необязательными.

Например:

- Команда `альфа\index{альфа}\verb` создаёт в указателе запись 1-го уровня, состоящую из слова «альфа» и номера страницы.

Если в документе встречается несколько пометок одного и того же термина, то в указателе будет только один *текст* с перечислением всех страниц.

Утилита MakeIndex чувствительна к регистру, а пробел воспринимает как равноправный символ.

Например:

■ Команды

```
альфа\index{альфа}
альфа\index{ альфа}
альфа\index{альфа }
альфа\index{альфа  }
альфа\index{ альфа }
```

создадут пять разных входов в указатель, причём, при упорядочивании, пробел ставится раньше буквы ‘а’.

Замечание: *MakeIndex* *различает нумерацию страниц арабскими, римскими цифрами и буквами английского алфавита, причём при упорядочивании (по умолчанию) римские цифры предшествуют арабским, а буквы стоят на последнем месте.*

2.1. Диапазоны страниц и перекрёстные ссылки

MakeIndex допускает более сложные конструкции.

Для указания диапазона страниц используется пара команд `|(` и `|)` в аргументах соответствующих команд `\index`. Например:

■

```
алфавит\index{алфавит|(}
. . .
\index{алфавит|)}
```

`|see{текст}` — ссылка от одного элемента указателя на другой. Например:

■

```
алфавит\index{алфавит}
. . .
альфа\index{альфа|see{алфавит}}
```

2.2. Оформление элементов указателя

В некоторых случаях может потребоваться изменение сортировки, вставки в нужное место указателя неалфавитных символов или изменить вид элемента указателя. Для этого в аргументе команды \index используется конструкция `key@visual`, где *key* — ключ по которому будет производится сортировка, а *visual* — отображаемый элемент указателя. Например:

■
```
$\alpha$\index{альфа@$\alpha$}
алфавит\index{алфавит@\textbf{алфавит}}
```

Для изменения вида номера страницы используются обычные команды, начинающиеся с символа-ограничителя '$'.

Замечание: *Вообще, любые команды, используемые в аргументе команды* \index, *начинающиеся с символа '|' а не с '\', действуют на номер страницы.*

Например:

■
```
\newcommand{\sml}[1]{{\small #1}}
алфавит\index{алфавит|textbf}
алфавит\index{алфавит|(textit}
альфа\index{альфа|sml}
алфавит\index{алфавит|)textit}
```

Здесь номер страницы, указывающий на альфа будет записан уменьшенным шрифтом.

2.3. Печать специальных символов

MakeIndex использует специальные символы: '!', '|', '@' и '"'. Чтобы вывести на печать такой символ, необходимо перед ним поставить символ '"'. Если перед символом '"' стоит '\', то такая комбинация производит обычный акцент (умляут). Например:

■
```
\index{quote (\verb++)}
\index{quote@\texttt{} sign}
\index{maedchen@M\"adchen}
\index{atsign@\texttt{"@} sign}
\index{bar@\texttt{}|see{vertical bar}}
\index{восклицание ("!)}
\index{восклицание ("!)!Ax"!}
```

3. Параметры MakeIndex

Для генерации `.ind` файла достаточно выполнить `makeindex`, указав только имя файла или файл с расширением idx. Полный синтаксис утилиты MakeIndex:
makeindex [-ciglqr] [-o *ind*] [-p *n*] [-s *sty*] [-t *log*] [*idx0 idx1* . . .],

`c` Включить режим подавления пробелов.

`i` В качестве входного файла использовать стандартное входное устройство (stdin[1]).

`g` Использовать упорядочивание элементов указателя в соответствии с нормами немецкого языка. При этом будут правильно упорядочены акцентированные символы немецкого языка. Эта опция поддерживается не всеми версиями MakeIndex.

`l` Использовать побуквенное упорядочивание; по умолчанию используется пословное упорядочивание, при котором пробел считается символом, предшествующим всем буквам алфавита.

`q` Работать в «тихом» режиме, т. е. не будут выводится сообщения в выходной поток для ошибок (stderr).

`r` Отключить неявное формирование диапазонов страниц. По умолчанию, три и более идущих подряд номера страниц заменяются интервалом.

`o` *ind* Использовать выходной файл *ind*. По умолчанию, в качестве имени входного файла берётся указанный в командной строке файл с расширением *idx*.

`p` *n* Задать номер *n* начальной страницы выходного файла. Кроме явного числа можно задавать одно из трёх символьных значений:

- `any` — номер последней страницы основного текста +1;
- `odd` — номер первой нечётной страницы, следующей за последней страницей основного текста;

[1] В операционных семействах семейства Windows это файл `con`, чаще всего — клавиатура.

- `even` — номер первой чётной страницы, следующей за последней страницей основного текста;

`s` *`sty`* Использовать стилевой файл *sty*. Значение по умолчанию для имени стилевого файла отсутствует. Переменная окружения `INDEXSTYLE` указывает где размещены стилевые файлы.

`t` *`log`* Использовать файл протокола *log*. По умолчанию берётся указанный в командной строке файл с расширением *ilg*.

Например:

■ Чтобы откомпилировать указатель утилитой MakeIndex со стилевым файлом `book.ist`, надо написать примерно следующую команду:

```
makeindex -s book.ist -o bookind.tex -p odd book
```

При этом получится выходной файл `bookind.tex`, который можно компилировать TEXом, как самостоятельный файл. Номер первой страницы будет равен первому нечётному числу, следующему за числом страниц книги (последний номер страницы, взятый из файла `book.log`).

Список литературы

1. *Гуссенс, М.* Путеводитель по пакету LaTeX и его расширению LaTeX 2_ε [Текст]: [Пер. с англ.] / М. Гуссенс, Ф. Миттельбах, А. Самарин; под ред. И. А. Маховой. — М.: Мир, 1999. — 606 с. — ISBN 5-03-003325-4 (русск.). — ISBN 0-201-54199-8 (англ.). — РФФИ №97-01-14165.

2. *Котельников, И. А.* LaTeX по русски [Текст] / И. А. Котельников, П. З. Чеботаев. — 3-е, переработанное и дополненное изд. — Новосибирск: Сибирский хронограф, 2004. — 496 с. — ISBN 5-87550-195-2. — на обложке: «Настольная издательская система LaTeX по русски».

3. *Гуссенс, М.* Путеводитель по пакету LaTeX и его графическим расширениям. Иллюстрирование документов при помощи TeX'а и PostScript'а [Текст]: [Пер. с англ.] / М. Гуссенс, С. Ратц, Ф. Миттельбах; под ред. А. И. Лебедева. — М.: Мир, Бином ЛЗ, 2002. — 621 с. — ISBN 5-03-003388-2 («Мир»).

4. *Львовский, С. М.* Работа в системе [Электронный ресурс] / С. М. Львовский. — М.: Интернет-Университет Информационных Технологий, 2007. Режим доступа: http://biblioclub.ru/index.php?page=book&id=234150.

Учебное издание

Олег Викторович **Журенков**

Издательская система LaTeX

Подготовка оригинал-макета, дизайн обложки *О. В. Журенков*

Оригинал-макет подготовлен в пакете LaTeX 2_ε

Издательство Алтайского государственного университета
Издательская лицензия ЛР 020261 от 14.01.1997.

Подписано в печать 07.07.2021.
Дата выхода издания в свет 19.07.2021.
Формат 60x84 1/16. Усл.-печ. л. 9,6.
Тираж 150. Заказ 262.

Типография Алтайского государственного университета
656049, Барнаул, ул. Димитрова, 66