

1 Команды рубрикации

Команды рубрикации предоставляют возможность разбивать текст на разделы и подразделы различных уровней, именно:

```
\part - часть  
\chapter - глава  
\section - раздел  
\subsection - подраздел  
\subsubsection - подподраздел  
\paragraph - параграф  
\ subparagraph - подпараграф  
\appendix - приложение
```

Команды рубрикации (кроме `\appendix`, формат и действие которой зависит от базового стиля документа) имеют одинаковый формат с одним обязательным параметром, который есть заголовок рубрики. Исключением является команда `\part`, заголовок в которой не обязателен.

Образец формата:

```
\section[<заголовок для оглавления>]{<заголовок>}
```

Необязательный параметр используется редко. В нем указывается краткий заголовок, который будет использован в оглавлении и колонтитуле — «бегущем» заголовке на верху каждой страницы. Если этот параметр опущен, то обязательный аргумент команды рубрикации будет выдан и в тексте, и в оглавлении, и, если этого требует формат страницы, как колонтитул.

\LaTeX автоматически нумерует разделы и подразделы. Для статьи номера рубрик младше раздела включают номера старших рубрик вплоть до раздела, для доклада и книги старшей рубрикой, номер которой будет входить в младшие рубрики, является глава. Если необходимо иметь заголовок без номера, следует воспользоваться *-формой команды рубрикации:

```
\section*{<заголовок>}
```

Размеры шрифтов для заголовков рубрик, наличие бегущего заголовка или помещение в оглавление рубрик, заданных в *-форме, определяются в файле описания стиля. Стандартные базовые стили не выводят бегущий заголовок и автоматически вносят в оглавление только нумеруемые рубрики.

2 Форматирование абзацев

LaTeX – программное средство для создания печатных документов разных типов с развитым печатным набором. Вы вводите ваш текст и некоторое количество команд, управляющих его оформлением, и LaTeX делает остальное.

2.1 Окружение `Center` (центрирование)

```
\begin {center}  
Текст строки 1 \\  
Текст строки 2 \\  
:  
\end {center}
```

```
Текст строки 1  
Текст строки 2  
:
```

Формат центрирования позволяет вам создавать абзац состоящий из строк, которые центрируются относительно текущих левой и правой границ текста страницы. Каждая строка должна завершаться `\``.

Декларация `\centering`

Это объявление соответствует формату центрирования. Оно может использоваться внутри кавычек или скобок. Текст рисунка или таблицы может центрироваться на странице при помещении команды `\centering` в начале формата таблицы или рисунка.

```
Текст строки 1  
Текст строки 2  
⋮
```

В отличие от формата `\center`, команда `\centering` не начинает новый абзац; просто меняется формат LaTeX внутри абзаца. Чтобы воздействовать на формат абзаца, объявление `\centering` должно содержать пустую строку или команду `\end`, заканчивающие абзац.

2.2 Выравнивания

Flushleft (выравнивание влево)

```
\begin{flushleft}  
Текст строки 1 \\  
Текст строки 2 \\  
⋮  
\end{flushleft}
```

Формат `flushleft` позволяет вам создавать абзац, состоящий из строк, выровненных по левому краю. Каждая строка должна завершаться `\\"`. Пример:

```
\begin{flushleft}  
Первая строка.\\  
Еще одна (вторая) строка. Надо посмотреть, что будет, если  
выскочить за пределы строки.  
Как обработается переполнение?\\  
Третья строка.  
\end{flushleft}
```

Первая строка.
Еще одна (вторая) строка. Надо посмотреть, что будет, если
выскочить за пределы строки. Как обработается переполнение?
Третья строка.

\raggedright

Это объявление соответствует формату *flushleft* и может использоваться внутри кавычек или скобок. Точно также строки разделяются символами `\\"`. В отличие от *flushleft*, команда `\raggedright` не начинает новый абзац; просто изменяется форматирование абзаца. Чтобы воздействовать на формат абзаца, объявление `\raggedright` должно содержать пустую строку или команду `\end`, заканчивающие абзац.

Flushright

```
\begin{flushright}  
Текст строки 1 \\  
Текст строки 2 \\  
⋮  
\end{flushright}
```

Формат *flushright* позволяет вам создавать абзац, состоящий из строк, выровненных по правому краю. Каждая строка завершается `\\"`. Пример:

```
\begin{flushright}  
Первая строка.\\
```

Еще одна (вторая) строка. Надо посмотреть, что будет, если выскочить за пределы строки.

```
    Как обработается переполнение?\\  
Третья строка.  
\end{flushright}
```

Первая строка.

Еще одна (вторая) строка. Надо посмотреть, что будет, если выскочить за пределы строки. Как обработается переполнение?

Третья строка.

\raggedleft

Это объявление соответствует формату *flushright* и может использоваться внутри кавычек или скобок. В отличие от *flushright*, команда \raggedleft не начинает новый абзац. Чтобы воздействовать на формат абзаца, объявление \raggedleft должно содержать пустую строку или команду \end, заканчивающие абзац.

3 Списки

В L^AT_EX'е имеются несколько командных скобок для составления списков с перечнем пунктов. Каждый пункт в таких скобок начинается с команды \item. В пределах командных скобок должен быть по крайней мере один пункт.

Itemize (перечисление)

```
\begin {itemize}
\item первый элемент
\item второй элемент
:
\end {itemize}
```

Формат *itemize* порождает ненумерованный список. Допускается до четырех уровней вложения и вложение в абзацы иного формата. Каждый элемент списка начинается командой \item. Должна быть по крайней мере одна команда \item.

```
{\centering Уравнения Снэйфу.\par}
\begin{itemize}
\item В задаче из $N$ уравнений всегда будет $N-1$ неизвестная.
\item Самый необходимый предмет или самая необходимая доза информации
будут наименее доступными.
\item Как только вы испробуете все возможные способы решения и
не найдете подходящего, тут же найдется решение простое
```

и очевидное для всех других людей.
\end{itemize}

Уравнения Снэйфу.

- В задаче из N уравнений всегда будет $N - 1$ неизвестная.
- Самый необходимый предмет или самая необходимая доза информации будут наименее доступными.
- Как только вы испробуете все возможные способы решения и не найдете подходящего, тут же найдется решение простое и очевидное для всех других людей.

Можно использовать до четырех уровней вложения списков:

- Первая запись первого уровня.
 - Первая запись второго уровня.
 - * Третий уровень.
 - . Четвертый уровень.
 - Вторая запись второго уровня.
- Вторая запись первого уровня.

Метки, используемые процедурой itemize по умолчанию на соответствующем уровне вложенности, производятся командами

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

Переопределив их с помощью \renewcommand , можно заменить метки сразу у всех записей:

```

\begin{itemize}
\renewcommand{\labelitemi}{\tikz
\draw[ball color=blue] (0,0) circle (.5);}
\renewcommand{\labelitemii}{\tikz
\draw[ball color=red] (0,0) circle (.3);}
\item Первая запись
    \begin{itemize}
\item Первая запись второго уровня.
\begin{itemize}
\item Третий уровень.
\begin{itemize}
\item Четвертый уровень.
\end{itemize}
\end{itemize}
\item Вторая запись второго уровня.
\end{itemize}
\item Вторая запись
\end{itemize}

```



Первая запись



Первая запись второго уровня.

- * Третий уровень.

- . Четвертый уровень.



Вторая запись второго уровня.



Вторая запись

Полезно обратить внимание, что здесь команда `\labelitemi` переопределена внутри процедуры `itemize`. Если её изменить до начала процедуры, то новое определение будет использовано в последующих

примерах, а мы к этому в данном случае не стремимся. Область действия вновь определённых команд подчиняется тем же правилам, что и область действия деклараций.

Чтобы изменить одну метку, нужно описать её в необязательном аргументе команды `\item`:

```
\begin{itemize}
\item[$] Первая запись
\item[##] Вторая запись
\end{itemize}
```

\$ Первая запись

Вторая запись

Enumerate (нумерованный список)

```
\begin{enumerate}
\item первый элемент
\item второй элемент
:
\end{enumerate}
```

Формат `enumerate` создает нумерованный список. Перечисления могут быть вложенными, до четырех уровней глубины. Они могут быть также вложены в другие форматы абзаца.

1. Первая запись первого уровня.
 - (a) Первая запись второго уровня.
 - i. Третий уровень.
 - А. Четвертый уровень.
 - (b) Вторая запись второго уровня.
2. Вторая запись первого уровня.

Каждый элемент нумерованного списка начинается командой `\item`. В формате `enumerate` должна быть по крайней мере одна команда `\item`.

Соответственно четырём уровням вложенности используются четыре счётчика: `enumi`, `enumii`, `enumiii`, `enumiv`. Каждая запись увеличивает счётчик соответствующего уровня на единицу. Однако при наличии необязательного аргумента в команде `\item[mark]` счётчик не увеличивается.

«Метки по умолчанию» печатают команды

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

Они используют перечисленные выше счётчики.

```
\begin{enumerate}  
\renewcommand{\theenumi}{\Asbuk{enumi}}  
\item Первая запись  
\item Вторая запись  
\end{enumerate}
```

A. Первая запись

B. Вторая запись

1. Первая запись первого уровня.

(A) Первая запись второго уровня.

• Третий уровень.

– Четвертый уровень.

(B) Вторая запись второго уровня.

2. Вторая запись первого уровня.

Изменить формат представления `\thectr`, установленный классом печатного документа, можно при помощи следующих команд:

```
\arabic{ctr} \fnsymbol{ctr}  
\roman{ctr} \Roman{ctr}  
\alph{ctr} \Alph{ctr}
```

Пакет babel с опцией russian добавляет к ним ещё две:

```
\asbuk{ctr} \Asbuk{ctr}
```

Чтобы изменить одну метку, нужно описать её в необязательном аргументе команды `\item`.

Description (помеченный список)

```
\begin{description}  
  \item [label] Первый элемент  
  \item [label] Второй элемент  
  :  
  \end{description}
```

Этот формат используется для создания помеченных списков. Метка печатается жирным шрифтом и выравнивается по левому краю текста.

Пример:

```
\begin{description}  
  \item Первый элемент списка  
  \item [a)] Второй элемент списка  
  \item [-] Третий элемент списка  
 \end{description}
```

Первый элемент списка

a) Второй элемент списка

- Третий элемент списка

Из приведенного примера видно, что при метках различной длины текст получается неаккуратным, для его выравнивания можно пользоваться, например, командой `\makebox`, как показано ниже:

```

\begin{description}
\newcommand{\mb}[1]{\makebox[7.5em][l]%
{\it #1.}}
\item [\mb{Хеопса}] Ничто никогда не строится в срок
и в пределах
сметы.
\item [\mb{Джоунса}] Человек, который может улыбаться
во время
неудач, непременно думает о том, на кого сможет свалить
вину
за очередную неудачу.
\item [\mb{Миллера}] Нельзя ничего сказать о глубине лужи,
пока не попадешь в нее.
\item [\mb{Вейлера}] Нет невыполнимой работы для человека,
который не обязан делать ее сам.
\end{description}

```

Закон Хеопса. Ничто никогда не строится в срок и в пределах сметы.

Закон Джоунса. Человек, который может улыбаться во время неудач, непременно думает о том, на кого сможет свалить вину за очередную неудачу.

Закон Миллера. Нельзя ничего сказать о глубине лужи, пока не попадешь в нее.

Закон Вейлера. Нет невыполнимой работы для человека, который не обязан делать ее сам.

```

\def\descrlabel#1{\makebox[3.0cm][l]
{\framebox[2.7cm][c]{\rule[-2mm]{0mm}{5mm}\it#1}}}
\def\descr{\list{}{\labelwidth 3.0cm
\leftmargin+3.2cm
\let\makelabel\descrlabel}}
\let\enddescr\endlist

```

```
\begin{descr}
\item [Закон Паддера] Все, что хорошо начинается,
кончается плохо.
    Все, что начинается плохо, кончается еще хуже.
\item [Закон Букера] Даже маленькая практика стоит
большой теории.
\item [Закон самолета] Когда ваш самолет опаздывает,
самолет, на
    который вы хотели пересесть, улетает вовремя.
\item [Аксиома Кана] Если ничто другое не помогает,
прочтите,
    наконец, инструкцию!
\end{descr}
```

Закон Паддера Все, что хорошо начинается, кончается плохо. Все,
что начинается плохо, кончается еще хуже.

Закон Букера Даже маленькая практика стоит большой теории.

Закон самолета Когда ваш самолет опаздывает, самолет, на кото-
рый вы хотели пересесть, улетает вовремя.

Аксиома Кана Если ничто другое не помогает, прочтите, наконец,
инструкцию!

Quotation (цитирование)

```
\begin{quotation}
```

Текст

```
\end{quotation}
```

Края формата *quotation* оформляются с отступом от левой и пра-
вой границ текста. Текст выравнивается по обоим краям и имеется
абзацный отступ. Пустая строка, разделяющая текст, порождает но-
вый абзац.

Пример:

\begin{quotation}

То, что вообще может быть сказано, может быть сказано ясно, а о чем невозможно говорить – о том следует молчать.

{\raggedleft\it Людвиг Витгенштейн\\}\end{quotation}

То, что вообще может быть сказано, может быть сказано ясно, а о чем невозможно говорить – о том следует молчать.

Людвиг Витгенштейн

Допускается вложенность конструкций цитирования:

\begin{quotation}

Книга Пойа «Математическое открытие» начинается следующим утверждением Лейбница:

\begin{quote}

Метод решения хорош, если с самого начала мы можем предвидеть – и далее подтвердить это, – что, следуя этому методу, мы достигнем цели.

\end{quote}

В технологии программирования до сих пор нет иного метода, кроме

макетирования, который бы удовлетворял этому критерию Лейбница.

{\raggedleft\it Г.Р.Громов\\}\end{quotation}

Книга Пойа «Математическое открытие» начинается следующим утверждением Лейбница:

Метод решения хорош, если с самого начала мы можем предвидеть – и далее подтвердить это, – что, следуя этому методу, мы достигнем цели.

В технологии программирования до сих пор нет иного метода, кроме макетирования, который бы удовлетворял этому критерию Лейбница.”

Г.Р.Громов

quote (кавычки)

\begin {quote}

Текст

\end {quote}

Края формата *quote* оформляются с отступом от левого и правого края. Текст выравнивается в обеих границах. Пустая строка в тексте порождает новый абзац. Ибзацийный отступ отсутствует.

Пример:

\begin{quote}

Наука – это та часть наших знаний, которую мы сумели понять настолько

хорошо, что можем обучить этому ЭВМ. Там, где мы еще не достигли

такого уровня понимания, речь идет пока лишь о профессиональном

искусстве.

Формальная запись алгоритма или программы ЭВМ, по-существу, позволяет нам выполнить весьма полезный тест глубины наших знаний,

так как переход от искусства к науке просто означает,

что мы поняли, наконец, как автоматизировать данную предметную область.

\raggedleft{\it Дональд Кнут.} Тьюринговская лекция\\ \end{quote}

Наука – это та часть наших знаний, которую мы сумели понять настолько хорошо, что можем обучить этому ЭВМ. Там,

где мы еще не достигли такого уровня понимания, речь идет пока лишь о профессиональном искусстве. Формальная запись алгоритма или программы ЭВМ, по-существу, позволяет нам выполнить весьма полезный тест глубины наших знаний, так как переход от искусства к науке просто означает, что мы поняли, наконец, как автоматизировать данную предметную область.

Дональд Кнут. Тьюринговская лекция

Verbatim (печатать как есть)

Процедуры

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

печатают текст в точности так, как он записан во входном файле, включая пробелы, специальные символы и команды. При этом команды не исполняются, за исключением `\end{verbatim}` или `\end{verbatim*}`, которые завершают исполнение процедур. В печатном документе соответствующая часть входного файла будет напечатана прямым машинописным шрифтом:

```
\begin{verbatim}
```

Текст

```
\end {verbatim}
```

Формат *verbatim* создает абзац, который печатается шрифтом `\texttt` точно так, как набрано вами. Это превращает LaTeX в пишущую машинку с возвратами каретки и пробелами.

Текст в формате *verbatim*, размер `\large`.

Допускается изменение размера шрифта. Попытка модификации типа шрифта игнорируется.

\verb

\verb[*] char literal_text char

Вывод *literal_text* точно так, как он дан, включая специальные символы и пробелы, используя шрифт пишущей машинки (\texttt). Символ *char* выполняет роль скобок, он не может быть пробелом и не должен встречаться в составе *literal_text*. В команде \verb *char* не может быть также буквой. Не должно быть пробела между \verb или \verb* и *char* (пробел показан здесь только для ясности). *-форма отличается только тем, что пробелы явно отмечаются символом \ .

\verb+ Пример применения команды \verb.+ дает:

Пример применения команды \verb.

\verb*\$. Пример применения команды \verb*.\$ дает:

\Пример\применения\команды\verb*.

Размер шрифта можно изменить, попытка изменения типа шрифта игнорируется:

{\large\verb0 Печать размером \large.0}:

Печать размером \large.

{\itshape\verb& Попытка печати шрифтом \itshape.&}:

Попытка печати шрифтом \itshape .

Verse (стихи)

\begin {verse}

Текст

\end {verse}

Формат verse разработан для поэзии, хотя вы можете найти ему и другие применения.

```
\begin{minipage}[h]{90mm}
\begin{verse}
\item Человеческая жизнь --\\
\hspace{5ex} это краткий эпизод\\
\hspace{10ex} в книге истории, \\
И в то же время\\
```

```
\hspace{ 5ex} история -- это всего лишь фон,\\
\hspace{10ex} на котором человек\\
\hspace{15ex} пишет книгу своей жизни.
\item Так и душа мечется, не зная покоя,\\
\hspace{ 5ex} между ощущением своей малости\\
\hspace{10ex} и неисчерпаемости\ dots
\vskip 2ex
{\it 12 октября 1985 г.}\hfill И.П.Ершов
\end{verse}
\end{minipage}
\hfill
\begin{minipage}[h]{85mm}
\begin{verse} \itemsep 1ex %% расстояние между строфами (\item)
\item Если ты не дурак, поразмысли о том,\\
Хорошо ль изнурять себя долгим постом?\\
Пьющий -- смертен, но разве бессмертен непьющий?\\
Нету разницы между святым и скотом.
\item{\centering *\par}
\item Когда глину творенья Иллах замесил,\\
Он меня о желаньях моих не спросил.\\
И грешил я по мере отпущеных сил. \
Справедливо ль, чтоб в рай меня бог\\
\hfill не впустил?
\item\hfill Омар Хайям
\end{verse}
\end{minipage}
```

Человеческая жизнь –
это краткий эпизод
в книге истории,
И в то же время
история – это всего лишь фон,
на котором человек
пишет книгу своей жизни.

Так и душа мечется, не зная покоя,
между ощущением своей малости
и неисчерпаемости...

12 октября 1985 г.

И.П.Ершов

Если ты не дурак, поразмысли о том,
Хорошо ль изнурять себя долгим постом?
Пьющий – смертен, но разве бессмертен
непьющий?

Нету разницы между святым и скотом.

*

Когда глину творенья Иллах замесил,
Он меня о желаньях моих не спросил.
И грешил я по мере отпущеных сил.
Справедливо ль, чтоб в рай меня
бог не впустил?

Омар Хайям