

Введение в L^AT_EX

Занятие 4

Даниил Дрябин

Студсовет ФПМИ

осень 2022

- 1 Уточнения к прошлому занятию
- 2 Счётчики в теоремах
- 3 Создание своих символов
- 4 Подключение файлов
- 5 Пакет tikz-cd
- 6 Что дальше?

Уточнения к прошлому занятию

Разница между tabular и tabbing

Окружение `tabbing` - устаревший способ создания таблиц. Помимо того, что у него немного другой синтаксис, есть и другие отличия:

- `tabbing` не позволяет управлять выравниванием текста с помощью параметров `l/r/c`
- в `tabbing` нет способа просто отделять столбцы и строки друг от друга линиями
- `tabbing` не поддерживает продвинутое форматирования, о которых мы поговорим на следующем слайде

TLDR: всегда используйте `tabular`.

Настройки окружения `tabular`

Мы уже видели, как управлять выравниванием текста, добавляя после `\begin{tabular}` аргумент вида `{l|r|c}`. Есть и более хитрые опции:

- `{p{2cm}|m{3cm}|b{4cm}}` — в таблице будет три колонки заданной ширины, с выравниванием по ширине по горизонтали и выравниванием по верхнему краю / по середине / по нижнему краю по вертикали (чтобы это работало, потребуется пакет `array`, о котором мы еще поговорим)
- Все колонки, не помеченные ни одним символом выше, можно вместо этого пометить символом `X`, тогда все такие колонки будут равной ширины, вычисленной автоматически
- Любой из символов, описывающих колонку, можно окружить командами `>\{...\}` и `<\{...\}`, а внутри указать любой код, который будет исполняться до / после каждой ячейки в колонке

Окружение array

Окружение `array` нужно для того, чтобы делать таблицы внутри математического режима. Им можно пользоваться, например, для создания матриц специального вида:

```
\[ \begin{array}{|cc|c|}
  a_{11} & a_{12} & b_1 \\
  a_{21} & a_{22} & b_2
\end{array} \]
```

Этот код дает следующий результат:

$$\left| \begin{array}{cc|c} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{array} \right|$$

Окружение `array` можно довольно гибко настраивать, но в каждом случае, когда вам это потребуется, удобнее искать нужные вам опции отдельно.

Как сделать ссылки кликабельными

Для того, чтобы команда `\href` вообще работала, требуется подключить пакет `hyperref`. А для работы кликабельных ссылок и гиперссылок в преамбулу следует включить такую команду:

```
\hypersetup{
  unicode=true,          % Юникод в названиях разделов в PDF
  colorlinks=true,       % Цветные ссылки вместо ссылок в рамках
  linkcolor=blue,        % Внутренние ссылки
  citecolor=green,       % Ссылки на библиографию
  filecolor=magenta,     % Ссылки на файлы
  urlcolor=blue,         % Ссылки на URL
}
```

Цвета тоже можно настраивать по-разному, например, так: `black!70!blue`.
А можно создавать свои цвета, но для этого потребуется пакет `xcolor`:
`\definecolor{myblue}{rgb}{0.6, 0.6, 0.9}`.

Кириллическое значение счётчика

Если в своем документе вы использовали команду `\usepackage[russian]{babel}`, то вы можете сделать так:

```
\newcounter{cyr}  
\setcounter{cyr}{5}  
\Asbuk{cyr}  
\stepcounter{cyr}  
\asbuk{cyr}  
\Asbuk{cyr}
```

Этот код дает следующий результат:

Д е

Как пронумеровать блок формул по центру

Раньше мы упоминали окружение `split`. Использовать его вне математики нельзя, но внутри математики оно позволяет собрать любое количество строк в единый объект, который будет пронумерован по центру (если внешнее окружение предполагает нумерацию).

```
\begin{equation}
  \begin{split}
    \dot x &= 2x + y \\
    \dot y &= 3x + 5y
  \end{split}
\end{equation}
```

$$\begin{aligned} \dot x &= 2x + y \\ \dot y &= 3x + 5y \end{aligned} \tag{1}$$

Окружения figure и table

Окружения `figure` и `table` — это специальные окружения, в которых размещают изображения и таблицы соответственно. Во-первых, чтобы подписывать и грамотно размещать на странице, во-вторых — чтобы собирать их в специальный `listoffigures` / `listoftables`. Пример использования:

```
\begin{figure}[h]
  \includegraphics[width=0.5\textwidth]{path/to/image}
  \caption{Подпись}
  \label{fig:figure2}
\end{figure}
```

Окружения figure и table

Разные индентификаторы расположения:

- `h` — дает рекомендацию латеху разместить объект как можно ближе к месту объявления, но в соответствии с его внутренними правилами
- `t` — размещает объект вверху страницы
- `b` — размещает объект внизу страницы
- `p` — размещает объект на специальной странице для всех таких объектов
- `H` — размещает объект прямо в месте объявления

А если требуется разместить объект посреди текста, то помогут окружения `wrapfigure` и `wraptable` с обязательными аргументами `{r/l}{width}` (но потребуется пакет `wrapfig`).

Счётчики в теоремах

Вложенные счётчики

Вспомним, как показать, что счётчики одного типа теорем нумеруются на один уровень глубже, чем некоторый другой счётчик (да, это не обязан быть другой тип теорем!):

```
\theoremstyle{plain}
\newtheorem{theorem}{Теорема}[section]
\newtheorem{corollary}{Следствие}[theorem]
\newtheorem*{definition}{Определение}

\section{Первый раздел}
\begin{theorem}Текст.\end{theorem}
\begin{theorem}Еще текст.\end{theorem}
\begin{proof}Тривиально.\end{proof}
\begin{corollary}И еще текст.\end{corollary}
\begin{definition}Что-то новое.\end{definition}
```

Вложенные счётчики

Код с предыдущего слайда дает следующий результат:

1 Первый раздел

Теорема 1.1. *Текст.*

Теорема 1.2. *Еще текст.*

Доказательство. Тривиально.

Следствие 1.2.1. *И еще текст.*

Определение. *Что-то новое.*



Общие счётчики

Разные типы теорем могут иметь общий счётчик, инкрементирующийся при объявлении теоремы любого из этих типов.

```
\newcounter{mycount}

\theoremstyle{plain}
\newtheorem{proposition}[mycount]{Утверждение}
\newtheorem{lemma}[mycount]{Лемма}

\begin{proposition}Текст.\end{proposition}
\begin{lemma}Второй текст.\end{lemma}
\begin{proposition}Третий текст.\end{proposition}
\stepcounter{mycount}
\begin{lemma}Четвертый текст?\end{lemma}
```

Общие счётчики

Код с предыдущего слайда дает следующий результат:

Утверждение 1. *Текст.*

Лемма 2. *Второй текст.*

Утверждение 3. *Третий текст.*

Лемма 5. *Четвертый текст?*

Да, вложенные и общие счётчики можно совмещать, если того требует сложная структура вашего документа. При этом с самими счётчиками можно при этом работать с помощью методов, обсуждавшихся в прошлый раз.

Создание своих символов

Боксы

«Бокс» — это контейнер для хранения содержимого любого вида, позволяющий применять к нему некоторые модификации. Их довольно много:

- `\parbox{width}{text}` — невидимый контейнер
- `\framebox{text}` — контейнер в рамке
- `\colorbox{color}{text}` — цветной контейнер
- `\raisebox{lift}{text}` — меняет вертикальное положение содержимого
- `\rotatebox{degrees}{text}` — вращает содержимое
- `\scalebox{scale}{text}` — масштабирует содержимое
- и другие

У контейнеров также есть опциональные аргументы, которые я для простоты опускаю.

Создание символа кратости I

Попробуем создать команду для символа кратности, используя боксы. Вариантов сделать это с таким арсеналом — масса, вот один пример из интернета:

```
\newcommand{\divby}{
  \mathrel{\vbox{
    \baselineskip=0.65ex
    \lineskiplimit=0pt\hbox{.}\hbox{.}\hbox{.}
  }}
}
\[4 \divby 2\]
```

Этот код дает следующий результат:

$$4 : 2$$

Создание символа кратости II

Приведем свой способ, попроще чем тот, что был до этого.

```
\newcommand{\divby}{
  \newcommand{\divby}{\mathrel{\scalebox{0.8}{$\vdots$}}}
}
\[4 \divby 2\]
```

Этот код дает следующий результат:

$$4 \div 2$$

Отметим, что наличие математического режима вне бокса не влияет на его содержимое, поэтому нам приходится использовать «доллары».

Подключение файлов

Подключение tex-исходников

Когда кода становится слишком много, становится оправданно разделять его на отдельные файлы, и собирать документ в `main.tex` (который в идеале не должен содержать нетривиального кода). Ниже — пример типичного файла `main.tex`.

```
\input{preamble}

\begin{document}
  \input{titlepage}
  \input{chapter1}
  \input{chapter2}
  \input{chapter3}
\end{document}
```

input vs. include

Синтаксис подключения tex-файлов имеет вид `\input{filename(.tex)}`. При компиляции код из файла `filename.tex` подставляется вместо соответствующей команды. Его альтернатива — `\include{filename}`.

| input | include |
|---|--|
| <ul style="list-style-type: none"> • Подставляет текст непосредственно | <ul style="list-style-type: none"> • Начинает текст с новой страницы и производит еще некоторые манипуляции |
| <ul style="list-style-type: none"> • Может быть вложенным | <ul style="list-style-type: none"> • Не может быть вложенным |
| <ul style="list-style-type: none"> • Чтобы не компилировать часть файлов — убирать или комментировать эти строки | <ul style="list-style-type: none"> • Чтобы не компилировать часть файлов — можно добавить <code>\includeonly{name1, name2}</code> в преамбуле |

При использовании вложенного обращения к файлам (например, `\input` внутри файла, подключенного через `\input`) следует помнить, что иерархия файловой системы ведет отсчет из *корневой папки*, содержащей `main.tex`.

Подключение pdf-файлов

Подключение pdf-файлов производится с помощью пакета `pdfpages` и имеет, например, такой синтаксис:

```
\usepackage{pdfpages}

\begin{document}
  \includepdf[pages={1, 3, 5-6}]{filename.pdf}
\end{document}
```


Пакет tikz-cd

Пример использования tikz-cd

Набор пакетов `tikz` предоставляет инструменты для создания в теке разнообразной векторной графики. Для работы с коммутативными диаграммами есть специальный пакет `tikz-cd`. Ограничимся примером его использования.

```
\[
\begin{tikzcd}[row sep = huge]
  G \arrow{rr}{\phi} \arrow[swap]{dr}{\pi} & & \\
  \im\phi \arrow[dashrightarrow, swap]{dl}{\psi} & & \\
  & & \\
  & G / K \arrow[dashrightarrow, swap]{ur}{\Theta} & \\
\end{tikzcd}
\]
```

Пример использования tikz-cd

Код с предыдущего слайда дает следующий результат:

$$\begin{array}{ccc}
 G & \xrightarrow{\varphi} & \operatorname{Im} \varphi \\
 \searrow \pi & & \nearrow \psi \\
 & G/K & \nearrow \Theta
 \end{array}$$

Что дальше?

Великое множество пакетов

Есть масса полезных пакетов, о которых нет смысла рассказывать на курсе, потому что они решают свои конкретные задачи, которые могут у вас и не возникнуть (а еще потому, что их десятки и десятки). Но любой хороший пакет имеет понятную документацию! Упомянем некоторые из них:

- `multicol` — написание текста в несколько колонок
- `tocloft` — гибкая настройка страницы содержания
- `listings` — визуализация кода на языках программирования в tex-документах
- `algorithm2e` — описание алгоритмов
- `wrapfig` — размещение «плавающих» объектов (изображений, таблиц)

Целый интернет

\LaTeX — очень старая технология, люди по всему миру пользуются ей десятилетиями. Если у вас никак не получается решить некоторую проблему, не бойтесь гуглить и ходить даже по старым форумам (хотя большинство ваших проблем наверняка решит StackOverflow).

Даже если найденное вами решение устаревшее, если вы понимаете, что вам нужно использовать его всего раз, — используйте, скорее всего это ничего не испортит.

Желаю удачи!

Всё!

The image shows the Russian word 'ВСЁ!' (Everything!) in a bold, stylized, 3D font. The letters are yellow with red outlines and red shading on the sides, giving them a three-dimensional appearance. The exclamation mark is also yellow with a red outline and a red shadow. The entire text is centered on a solid blue background.