Конспект пары по LaTeX-у Всё то, что внутри звёздочек – в папке, оно есть

1 Первый день

Почему именно TeX? Tex не максимально очевиден в освоении по сравнению с тем же вордом (так как требуется хотя бы минимальный опыт в программировании иначе осваивать сложнее), но крайне упрощает жизнь в больших проектах со сложным форматированием или математикой, набирать которую в ворде – просто мука). Tex довольно старый, поэтому сейчас существует множество дополнений, которые делают одно и то же, существует легаси код, который может вызвать проблемы в будущем (у самого теха), поэтому гуглинг информации по теху – занятие не из приятных.

Первый пример. Команды имеют вид `\command[args]{args}`. `\documentclass[a4paper` стандартное начало, в дальнейшем это будет вынесено в специальный файл — преамбулу, чтобы не засорять основной док, в принципе хорошим тоном является разбиение документа на несколько файлов (логичное разбиение, а не “каждый абзац в новом файле”). Документкласс определяет тип документа, размер шрифта и тип листа (бывают и другие опции, но это стандарт). Далее следует начало окружения document, в котором и располагается весь контент, существуют и другие окружения, о которых мы поговорим в дальнейшем. (Работа с окружениями похожа чем-то на html).

*второй пример*Также изначально русского языка у нас нет, поэтому сначала нужно настроить кодировки и форматирование, для этого мы используем babel, fontenc, inputenc.

Btw есть команда `\input{...}`, которая вставит, то, что мы inuptнули (так что обычно в начале дока — `\input{preamble.tex}`)

Для использования каких-то кастомных пакетов существует команда `\usepackage{...}`, `\usepackage[...]{...}`

Математика. Для ввода математических формул существует три варианта: `$...$, \[...\], $$...$$` (последние два равнозначны, но двойные значки доллара не рекомендуются и считаются устаревшей практикой, лучше так не делать) (первый случай выражения инлайн, в строке, остальные выносят формулу в отдельную строку).

Простейшие команды используются для введения символов, которых нет на клавиатуре *третий пример*, например `\infty` – бесконечность, аналогично разные символы греческого алфавита и т.д.

Также есть разные интегралы корни и т.д., которые позволяют определять символы и внутри.

Есть полезный сайт Detexify, который позволяет по рисунку символа получить команду для него (очень полезный по опыту).

четвёртый пример Расположение индексов, одиночные индексы, индексы не из одного символа. Аргументы не из одного символа должны находиться в фигурных скобках. Далее рассказываем про наличие греческого алфавита, про то, что некоторые отсутствуют, например заглавная альфа по причине схожести. Есть также разные кванторы. Плюс некоторые греческие символы пишутся не так, как мы привыкли, исправляется это перевведением команд, об этом позже.

Дроби, скобки. Дроби могу быть написаны по-разному (dfrac, frac, (sfrac для наклонённых, но это небазовое)). В дробях можно не использовать фигурные символы, но это кринж, потому что распарсит по 1 символу. Кроме того, для разных бинарные операторов есть разные символы, кроме того, операторы вокруг себя генерят пробелы, а просто операторы пишутся другим шрифтом и генерят пробел после себя, про то, как создавать свои операторы поговорим позже. !Latex сжимает пробелы в формулах! Скобки. Есть разные виды скобок, а также разные размеры *пример пять*, рассказываем про команды, которые позволяют изменять размер скобок, а также автоматически определять нужный размер (left и right – лучшая практика). Есть фигурные, квадратные, скобки округления вниз и вверх (floor, ceil: rfloor, lfloor, rceil, lceil). Перед фигурными скобками ставим потому что иначе распарсит как какой-то аргумент, кстати, чтобы набрать нужно использовать `\backslash` (`\$`, `\backslash, \{, \}, \&, \#, \_, \%`). С помощью процента комментриуем. Не используем двойной потому что переводит строку.

Рассказанное – база, этого достаточно, чтобы писать конспекты и всё такое.

Теперь будем углубляться. Начнём с форматирования текста. Для разбиения текста на блоки существуют `\section, \subsection`. Если добавить после них звёздочку, то будет без номера(в article). В больших проектах вроде книг, существуют и другие команды для разбиения на подразделы(например в book есть chapter).

Стиль текста. Есть командный (`\textbf, \textit`) и модификаторный, использующий области видимости и модификаторы (`{\bfseries \itshape text} notb`
шестой прмер. Есть модификаторы для размера текста (`\Huge, \LARGE, \Large, \normal`
Заменять шрифт можно и модификатором, и соответствующей командой (`\rmfamily, \textrm{},` особо не нужно).

Простейшие окружения. Окружение – способ инкапсулировать много модификаторов стиля в единый блок, в который потом можно заворачи-

вать текст. Используем так: `\begin{environment}...\end{environment}`. Можно использовать окружения в окружениях.

Для конспектов очень пригодятся окружения позволяющие удобно расписывать леммы, теоремы, док-ва, примеры и т.р.

Для этого используем

```
\usepackage{amsthm}
\theoremstyle{plain}
\newtheorem{theorem}{Теорема}
\newtheorem{lemma}{Лемма}.
```

Есть даже окружения для того, чтобы в них код форматировался и очень клёво отображался, таких много, например, verbatim. Есть также куча крутых пакетов от математического общества, которые добавляют много классных вещей. Дефисы. Есть куча видов ‘‘`---` подлежащее сказуемое, `- - -` обычное тире, `- -` промежутки в тексте, `-` дефис, `$-$` минус. Вместо многоточия ставим `\dots`. Есть свои кавычки `<<, >>`.

2 Второй день

Основные разделы, по убыванию: part, chapter, section, subsection, subsubsection, paragraph, subparagraph (чему соответствует – очев просто перевод). Синтаксис имеет вид `\section_name[short_title]{full_title}` (Краткое название отображается в колонтитулах, содержаниях и т.п.). Не все разделы доступны для каждого класса документа, лучше смотреть информацию о каждом отдельно. Для нас будет актуален в основном класс article, не поддерживающий только команду `\chapter`.

- `\maketitle` — генерация заголовка, использует значения параметров задаваемых командами `\author{}, \date{}, \title{}`
- `\tableofcontents` — генерация содержания, использует все разделы с нумерацией
- `\listoffigures` — генерация списка изображений
- `\listoftables` — генерация списка таблиц
- `\appendix` — разделы ниже такой команды будут считаться разделами приложения, со своими правилами нумерации

Все эти блоки можно настраивать на своё усмотрение, будем говорить про это позднее

Задание размеров. Пакет geometry.
картиночка с описанием всех переменных для размеров
Синтаксис `\geometry{variable=value}`
Пример `\geometry{top=25mm}`
Единицы измерения:

- Относительные pt (элементарная единица измерения размера в латехе), ex (высота буквы x), em(ширина буквы M)
- Абсолютные mm, cm, in

Есть и другие параметры типа linespread (регулирует межстрочный интервал, что полезно при большом количестве межстрочной математики), parident (отступ первой строки каждого абзаца), parskip (отступ после абзаца) и т.д. К parident и parskip обращаемся через setlength. В принципе с геометрией не приходится часто иметь дело, обычно она единожды прописывается в преамбуле и всё.

Колонтитулы. Пакетов много, рассмотрим titleps. Есть setheadrule, setfootrule для размера колонтитулов, а также sethead, setfoot для надписей. Есть счётчик thepage, которым можно с помощью данного пакета отрисовывать номера страниц. pagestyle применяется до конца области видимости или использования другого pagestyle. thispagestyle применяется только к одной странице.

Есть пакет soul, добавляющий ещё модификаторов к тексту.

- so — ставит пробелы между буквами
- caps — капс
- ul — подчёркивает
- st — перечёркивает

Команды для отступов. hspace и vspace – горизонтальный и вертикальный отступы. Добавим *, получим пробелы, которые не остановить даже концом строки. Есть “распорки” hfill и vfill, которые располагают то, что до и что после по разным краям.

Пустая строка и `\par` — одно и то же, делают переход на новый абзац, начинающийся с красной строки

`\\` и `\newline` совершают насильственный переход на новую строку, для страниц аналогом является `\newpage`

`\\linebreak` совершает переход на новую строку не нарушая правил форматирования, для страниц аналогом является `\pagebreak`

Чтобы иметь красную строку в первом абзаце каждого раздела, как в русскоязычной типографии, следует использовать команду `\usepackage{indentfirst}`

Изображения и таблицы. Используем окружение tabular. *Пример с таблицей*. Используем буквы для выравнивания и неопределённое количество разделителей. Для разделения по горизонтали используем `\hline`.

Btw в TeXStudio мы имеем специальный инструмент для создания табличек Quick Tabular. Можно завернуть таблицу в center, что центрирует её. Также есть окружение table, которое генерирует специальный объект, который отображается в списке таблиц, обсуждаемом ранее. Вообще хорошо таблицу во что-то заворачивать.

Изображения. Для вставки изображений есть `\includegraphics{...}`, можно без расширения, тех распарсит сам. В квадратных скобках можно регулировать высоту/ширину, например 0.8 ширины текста: `\includegraphics[width=0.8\t` Опять же хорошо картинку центрировать или что-то подобное.

Создание команд. Нужно это при частом использовании однотипного кода (например, мы на матане и алгеме очень часто писали что-то вроде $v_1 + \ldots + v_n$, было очень удобно это оптимизировать). Для создания команд есть команда `\newcommand{name_of_command}[num_of_args]{code}`, $\#n$ — n-ый аргумент. *пример с частной производной и множеством натуральных чисел*.

!Лучше подобное, кст, прописывать в преамбуле!

Если аргументов несколько, то для каждого свои скобки

Как таковой возможности перегрузки команд в латехе нет, но есть пакет xparse, который позволяет делать что-то наподобии.

Замена команд. Если мы хотим использовать с тем же именем другую команду, то используем `\renewcommand`, это, к примеру, позволяет исправить странное написание греческих букв. Важно: не используйте команду, которую вы переопределяете при переопределении, потому что возникнет бесконечная рекурсия и ошибка + заменять команды с аргументами нужно крайне аккуратно.

Создание своих операторов. `\DeclareMathOperator{command}{text_of_operator}`. Оператор будет написан прямым шрифтом. Если добавить *, то верхний и нижний индексы станут пределами, т.е. будет писаться сверху и снизу, а не как обычно.

Создание бинарных операторов. `\newcommand{\precent}{\mathbin{\%}}` — mthbin грамотно расставляет пробелы.

Окружения. Списки создаём при помощи itemize (маркированный), enumerate (нумерованный), каждый элемент начинается с команды `\item`. Также мы можем изменять маркеры, которыми помечаем элементы списка, с помощью аргументов в квадратных скобочках: `\begin{enumerate}[label=\alph*)]`

— тут вместо цифр будут английские буквы.

Списки можно вкладывать друг в друга, маркеры и стиль нумерации будут меняться, чтобы отличаться от стиля списков выше, про управление изменением стиля будет позже. Кроме того, для каждого пункта можно отдельно выбрать символ маркер: `\item[$\Leftarrow$]` — маркером будет "следовательно" в левую сторону.

Создание теорем. `\newtheorem{command}{text}[counter]`. Команда без \. Если добавить *, то будет без номера. Counter — то, после чего будем считать с начала. Встроенные стили: plain (название жирным, текст курсивом), definition (определение, название жирное, текст обычный), remark (замечание, название курсивом, текст обычный). Стиль определяем через `\theoremstyle{style}`. (применяется ко всему объявленному после с помощью newtheorem)

Уже по умолчанию есть окружение proof, используемое для доказательств, текст прямой, в конце ставит квадратик.

Создание окружений. `\newenvironment{name}{start commands}{end commands}`. Начальные вставляются до текста, конечные – после.

Набор формул 2. Есть различные шрифты: mathbb (для красивых множеств типа рациональных, натуральных чисел и т.д.), mathbf (жирный), mathrm (прямой), mathsf (жирный прямой), mathcal ("красивый" для разных L красивых и всего такого) *пример*. Есть ещё куча разных шрифтов, но эти пригождаются чаще всего.

Можно контролировать поведение индексов у некоторых команд командой limits (например у сумм, интегралов, пределов и т.д., чтобы индексы были сверху и снизу). Если дробь объявить через dfrac, то она будет полноразмерной. limits и dfrac имеет смысл использовать только inline, потому что в `\[\]` и так будет нормально.

У большинства стрелок есть альтернатива, начинающаяся с "x", у которой нижние символы в [], а верхние в , такая стрелка автоматически растягивается, и с индексами у неё проблем нет. Текст в формулах набирается командой `\text{...}`

Записи над символами.

- `\dot` — точка
- `\ddot` — 2 точки
- `\mathring` — пустой кружочек
- `\hat` — циркумфлекс, xor, шляпка, `\widehat{}` (long variant)
- `\tilde` — тильда, волна, `\widetilde{}`

- `\bar` — черта, `\overline{}`
- `\vec` — стрелочка, `\overrightarrow{}`

Для строк и всего такого — длинный вариант

Полезные окружений. equation(*)(внутри автоматически математика, плюс уравнения нумеруются), align(*)(несколько выражений в одном блоке, строки отделяем \\, суть в том, что они выравниваются по вставленному амперсанду), multline(*)(одна большая формула, разбитая на несколько строк, строки будут располагаться по диагонали), aligned (не включена математика по умолчанию, нужно меньше переводов строк), cases (системы уравнений и всё такое прочее).

3 Третий день

Поговорим про то, где можно техать. Ну, фактически сам теховский файл – просто текстовый документ, который с помощью специального ПО мы конвертируем в красивый pdf-файл. Различные приложения лишь помогают упростить нам жизнь, заканчивая за нас команды или, например, упрощая работу с таблицами. Теперь про программы. Есть специальные программы для теха типо того же texstudio, там всё сделано по красоте, но ставить программу лишь для того, чтобы в ней техать может слегка удручать. Следующим идёт vscode с её огромным множеством плагинов, среди которых есть и плагины для теха. Так как vscode сама по себе крайне удобная, да ещё и многофункциональная, то многие люди техают именно в ней. Ну и вариант для трушных “прогеров” – техать в vim, парочка дополнений и vim с радостью будет и заканчивать команды за вас, и своевременно обновлять pdf документ, над которым вы работает, в соседнем окошечке. Все варианты хороши, поэтому стоит попробовать все и решать чисто для себя.

Набор формул 3. Месть теорем. Кроме окружений перечисленных ранее есть также split, gather, e.t.c., их мы пристально рассматривать не будем, но можно почитать на wikibooks.org: LaTeX/Adbanced Mathematics.

Чтобы пометить конкретную формулу используем `\tag{...}`, а также поставить `\label{}`

Есть окружения pmatrix (матрица с обычными скобками), bmatrix (с квадратными) и другие. Есть также пакет NiceMatrix, в котором есть множество обёрток над матрицами, можно делать расширенные матрицы, можно делать блочные матрицы. Да и мануал у них хороший.

Можно устраивать внутренние ссылки на формулы, тогда кликая на скобку перейдём к формуле, для этого используем `\eqref{}, \pageref{}`,

соответственно. Такие ссылки можно делать ко всему, что имеет нумерацию. По умолчанию для обращения к label используется ref, но существует ещё как минимум `\figref, \eqref`, также названия ссылок принято начинать с eq:, fig:, etc

Можно делать ссылки на внешние сайты с помощью команды `\href{link}{text}`.

Сноски делаются с помощью `\footnote{text}`, на них также можно ссылаться.

Счётчики. Создаём с помощью `\newcounter{name}` (начальное значение 0), можно ещё `\newcounter{name}[other name]` второй обнуляется каждый раз, когда инкрементируется первый.

- `\stepcounter{name}` — ++name
- `\addtocounter{name}{number}` — $name+ = number$
- `\setcounter{name}{number}` — $name = number$

К счётчикам можно обращаться по-разному:
`\thename, \arabic{name}(\thename), \roman{name}`(маленькие римские), `\Roman{name}`(большие римские) (только положительные), `\alph{name}` (маленькие буквы, $0 \leq name \leq 26$), `\Alph{name}`. Есть также встроенные счётчики: part, chapter, etc., page, figure, footnote, enumi, enumii, enumiii, enumiv (это были номера внутри вложенного списка, на 4-х уровнях вложенности), equation (нумерация из align)

В списках можем использовать счётчики, в квадратных скобках после фигурных.

Создание окружений. Часть уже была.

BTW Создание окружений чем-то похоже на создание контекстных менеджеров в питоне :)

Создание стилей теорем

```
\newtheoremstyle{stylename} % название стиля
{spaceabove} % отступ до теоремы
{spacebelow} % отступ после теоремы
{bodyfont} % шрифт
{indent} % отступ
{headfont} % шрифт заголо вка
{headpunctuation} % отделение заголовка от теоремы
{headspace} % отступ от заголовка от теоремы
{headspec} % фиксация самого заголовка
```

4 Четвёртый день

Счётчики в теоремах. Кроме того, что уже было рассказано в плане следствия за другими объектами (леммы нумеруются начиная с теоремы и всё такое). Можно руками в квадратных скобках поставить имя счётчика и теоремы/леммы будут его использовать и инкрементировать.

Подключение файлов. 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). Т.е. в main.tex мы в основном выполняем input-ы и используем окружение document.

При компиляции кода код из input-го файла подставляется вместо соответствующей команды. Альтернатива input-у – include. Основным отличием является то, что include предполагает хранить в себе цельные куски кода, а потому будет начинать текст с новой страницы и производить ещё некоторые манипуляции, кроме того он не может быть вложенным, а для того, чтобы не компилировать часть файлов, нужно всего лишь добавить `\includeonly{name1, name2}` в преамбуле.

!При использовании вложенных input-ов, include-ов и других команд, работающих с файлами, важно помнить, что ищем мы относительно файла main.tex, поэтому нужно быть аккуратными с относительными путями.

```
Подключение pdf-файлов. Есть разные пакеты, поговорим про pdfpages. Синтаксис: \
Графика в texe и пакет tikz-cd. Вообще tikz - это инструмент в texe для создания
Tikz. Рисунки в texe. Простые примеры:
\tikz \draw (0pt, 0pt) -- (1in, 8pt); - линия
\tikz \fill[orange] (1ex, 1ex) circle (1ex); - оранжевый кружок
Путь - основной блок всех рисунков в TikZ. Он состоит из точек (x, y) и --. Весь
\begin{tikzpicture}
\draw (-1.5, 0) -- (1.5, 0) -- (0, -1.5) -- (0, 1.5);
\end{tikzpicture} - Получится отзеркаленная относительно (0,0) четвёрка
\tikz - аналогичен способу с окружением
Для круга -- заменяем на circle(10pt);, для эллипса на ellipse (20pt and 10pt);,
Для сетки используем grid, кстати, в начале можем явно указывать откуда начинать
\draw [xstep=0.4, ystep=0.5] (0,0)  grid (2,2); (есть и просто аргумент step). *
Сетка - вспомогательные линии, для них можно сделать свой стиль:
\tikzset{help lines/.style={very thin, gray} }
\tikz [step=.5cm, help lines] (-1.4, -1.4) grid (1.4, 1.4);
Можно выстраивать иерархию стилей и передавать свои параметры, например:
\tikzset{help lines/.style={very thin, color=#1!50},
```

help lines/.default={black} }
\tikzset{help grid/.style={step=#1, help lines=black},
help grid/.default={0.5cm} }
Можно передавать и несколько параметров (но это путь для самураев): \tikzset{nam
Опции рисунка:
- Толщина:
  - ultra/very thin
  - thin
  - semithick
  - thick
  - ultra/very thick
- Цвет:
  - gray, red, blue
  - {rgb,255:red,21; green,66; blue,128}
  - микс: red!10!blue (10% красного)
  - прозрачность: green!50
- Заполнение линии:
  - loosely/densely (интенсивность) dashed (пунктиром)
  - loosely/densely dotted (точечками)

Можно задавать полярные координаты (angle:radius), можно в пути указывать коорди
cycle - зациклить путь, указываем вместо точки в пути, возвращает в начало
arc(start angle, end angle, step) - дуга
Обычные стрелочки выглядят некрасиво, поэтому можно передать аргументом для всег
Для расположения точки используем команду \coordinate (name) at (pos);
\node - полезная конструкция для узлов, пример \node[draw, circle, through=(A)]
Пакет tikzcalc позволяет проводить вычисления над точками: складывать их и всё т
(A | - B) - точка на пересечении вертикали через A и горизонтали через B
($(point)!length!(point1)$) - точка на расстоянии length (мб отрицательном) от p
$(p)!(p1)!(p2)$ - проекция p1 на pp2
Пакет tkz-euclide позволяет работать с отрезками, углами, etc.
\tkzMarkAngle[mark=,arc=ll,size=10pt](C,D,B); - отмечает угол CDB, количество l
\tkzMarkRightAngle - прямой угол
\tkzLabelAngle[pos=0.6](...) {text} - помечаем угол текстом
\tkzMarkSegment[mark=|](E,B); - отрезок помечаем чертой
В tikz есть циклы, но не только там, поэтому можно мутить такое:

```
\begin{equation*}
r =
\sqrt{\foreach \x in {a,...,g} {
\ifthenelse{\equal{\x}{a}}{}{+}
\x^2
}}
```

\end{equation*}
вывод этой штуки
объявлять переменные можно с помощью \def\name{value}
Можно управлять тем, где будет название у точки: [label=above:name]
Пути и их пересечения. \node (E) [name intersections={of D and E, by={[label=abo
[label=below:C']C'}}]; - назвали пересечение окружностей D и E - C и C'. Также
Btw (A) ! .5 ! (B) - середина AB
Btw Класс документа standalone предназначен для выполнения как раз рисуночков и
Btw для использования пакетов именно tikz-a используем \usetikzlibrary{name} (на
Btw Есть прекраснейшая книга по tikz, называется TikZ & PGF.

5 Пятый день

Презентации в LaTeX-e. Основы вёрстки презентаций. Теперь мы используем класс до
Слайды делаем с помощью окружения frame: \begin{frame}...\end{frame}(но в одном
Аналог \maketitle - \titlepage
В латехе уже есть множество шаблонов, полезных в презентациях:
- usetheme - меняет геометрию презентации
- usecolortheme - меняет основной цвет оформления презентации
- usefonttheme - меняет шрифт в презентации
- useinnertheme - меняет тему текста внутри презентации

Разделы остаются как и в обычных документах
В некоторых темах они отображаются, а в некоторых - нет
Можно использовать AtBeginSection или похожие команды для работы с секциями
пример с этой командой и tableofcontents
Новые окружения. Во-первых у самого frame есть разные опции:
- выравнивание текста [b],[c],[t]
- [fragile] часто используется, если вы добавляете код в текст
- [plain] для того, чтобы убрать какое-либо оформление
- {name}

Для теорем и выделения блоков текста используем окружение \begin{block}{name}, т
alertblock - блок со специальным выделением
Когда разбиваем текст на несколько столбцов используем окружение columns, при эт
Оверлеи. Это основное отличие презентаций от документов. Мы можем объединять нес
Есть встроенные команды, помогающие работать с оверлеями в презентациях:
- pause: весь текст после этой команды пишется только со следующего слайда
- uncover: текст, который передаётся в качестве аргумента, будет показан на слай
- only: текст будет показан на одном слайде

У textbf, textit, ... тоже есть спецификации для выделения только на выбранных с
Можно также делать ссылки и на фреймы

Список полезных пакетов на всякий случай:

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