Latex学习笔记

语法

文档类型

`\documentclass[...]{...}`每个latex文档的起点。

{ }内参数
告诉编译器创建文档类型，不同的文档类型有不同的默认格式，例如页边距，字体大小和编号方式

- `article` 最常用的文档，适合写短篇论文、期刊文章、简短报告和信件。
- `report`：适用于较长的报告、学位论文或技术文档
- `book`：专为书籍编写设计，提供了额外的功能，如双面打印的页边距、封面和章节的开始页
- `letter`：专为写信设计，提供了一些特定的命令来格式化收件人地址、信件日期和落款
- `beamer`：用于制作幻灯片，它提供了各种预设的主题和功能，可以轻松制作出美观的演示文稿

[ ]内参数
可以设置文档 **基础字体大小，纸张大小，单双面打印，单双列排版**等功能，这些参数可以组合使用，用`,`分隔

- `10pt`, `11pt`, `12pt`：设置基础字体大小
- `a4paper`, `letterpaper`：设置纸张大小，a4为亚欧常用，letter北美常用
- `oneside`, `twoside`：设置单双面打印
- `onecolum`, `twocolum`：设置单双列排版

```
\documentclass{article}
\documentclass[12pt, a4paper]{report}
```

宏包

`\usepackage{...}`写在导言区，即`\documentclass`和`\begin{document}`之间，用于调取宏包，宏包（package）是事先写好的代码库，用于拓展功能。

常用的宏包有：

- `amsmath`：提供复杂数学公式的支持
- `graphicx`：用于在文档中插入图片
- `hyperref`：为文档中的交叉引用、目录和网址添加超链接,通常是最后一个被加载的宏包
- `babel`或`ctex`：支持多语言排版，`babel`提供了多种语言支持，而`ctex`是专门为中文排版设计的宏集
- `inputenc`：处理输入编码，确保LaTeX编译器能正确读取

标题和作者

这几个命令用来定义文档的元数据。`\maketitle`命令用于生成标题页，使用前面定义的标题、作者和日期信息， **必须放在\begin{document}之后，希望标题出现的位置**

```
\title{my article}
\author{kekdou}
\date{\today}     % \today会自动插入今天的日期

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

如果需要副标题
`\title`命令并不支持直接添加副标题，可以通过手动换行或使用`titlesec`宏包的`\subtitle`命令实现

```
%手动换行
\title{主标题 \\ \large 副标题}     % \\用来强制换行，\large命令调整副标题字体大小
```

如果有多个作者，并且表示出机构

```
\author{
    kekdou\thanks{THU,Peking}\and     % \and用于并列作者
    kekdou2\thanks{PKU,Peking}        % \thanks{...}命令会为作者添加注脚
}
```

摘要和关键词

摘要需使用`abstraction`环境，会自动生成“摘要”的标题并对文本进行格式化，取决于文档类型。`latex`核心系统没有内置的关键词环境，因此需要手动创建关键词部分

```
\begin{document}
\maketitle
\begin{abstraction}
...                %摘要内容
\begin{abstraction}

\noindent\textbf{关键词：} ...      %填写需要的关键词
...          %\noindent用于取消首行缩进
\end{document}
```

章节命令

用于组织文档的层次结构，会自动添加章节编号，并在目录中生成对应的条目
不同文档类型有不同级别的章节命令，以最常用的article举例：

```
\section{...}   %一级标题
...
\subsection{...}   %二级标题
...
\subsubsection{...}    %三级标题
...
\paragraph{...}   %四级标题
...
\subparagraph{...}    %五级标题
...
```

如果使用的是article，则\section是一级标题。如果是report或book，一级标题是\part或\chapter

此外，如果想创建无编号章节，只需在章节命令后面添加一个*。这对于“引言”或“致谢”这样的章节非常有用

```
\section*{...}   %无编号章节
```

注意： *无编号的章节不会被自动添加到目录中，如果想出现在目录中，但又不想编号，可以使用* `\addcontentsline` *命令手动添加*

文本样式

- `\textbf{}`：将文本变为粗体
- `\textit{}`：将文本变为斜体
- `\underline{}`：给文本添加下划线

此外，也可以调整字体大小，从`\tiny`到`\Huge`

- `\tiny`
- `\scriptsize`
- `\footnotesize`：以此类推
- `\small`：比normal小1号
- `\normalsize`：设置正常字体大小
- `\large`：比normal大1号
- `\Large`：以此类推
- `\LARGE`
- `\huge`
- `\Huge`

空格

latex会根据上下文自动调整空格大小，并且将连续的空格或换行视为一个空格

- **单个空格** 敲一个空格或换行
- **强制空格**
  - \：一个单词的词距
  - \ ：一个普通的空格
  - \quad：一个很大的空格

换行

在latex换行与换段容易混淆

- \\或\newline（强制换行）
  这两个命令的作用相同，用于在当前位置强制换到新的一行
- 空行（开始新段落）
  - 在两行文本之间留一个空行，即开始一个新的段落
  - 新段落会在开头自动缩进，并且与前一段落之间有一个垂直的间距

缩进

默认情况下，latex会在每个段落的开头自动进行缩进

取消缩进： 使用\noindent命令取消缩进

列表

分为有序列表和无序列表，可以进行互相嵌套处理

无序列表：

```
\begin{itemize}
\item apple
\item banana
\item orange
\end{itemize}

\begin{itemize}
\item apple
\item banana
\begin{itemize}
\item tomato
\item potato
\end{itemize}
\item orange
\end{itemize}
```

有序列表：

```
\begin{enumerate}
\item first step
\item second step
\item third step
\end{enumerate}

\begin{enumerate}
\item first step
\item second step
\begin{enumerate}
\item first step
\item second step
\end{enumerate}
\item third step
\end{enumerate}
```

\item的缩进是可选的，为了可读性是可以缩进的，不影响编译结果

数学公式

分为独立公式和行内公式
行内公式 $...$或\(...\),用来在文本中插入简短的公式

```
牛顿第二定律是 $F=ma$
```

独立公式 \[...\]或\begin{equation}...\end{equation}
\[...\],没有编号
\begin{equation}...\end{equation}会自动编号

```
%这是一个没有编号的独立公式:
\[
x = \frac{-b \pm \sqrt{b^2-4ac}}{2a}
\]

%这是一个有编号的独立公式:
\begin{equation}
y = ax^2 + bx + c
\end{equation}
```

关于多行公式的对齐问题 ，需要使用amsmath宏包提供的环境

使用align环境，并在想要对其的位置使用符号&，通常放在等号=前面

```
\usepackage{amsmath}     %确保在导言区加载此宏包
...
```

```
\begin{align}
y&=(x+1)^2 \\      % \\用于换行
 &=x^2 + 2x + 1    % &指定对齐点
\end{align}
```

latex提供了丰富的数学符号，详细见另一个笔记

表格

`tabular`环境是表格的基础。`\begin{tabular}[<垂直对齐>]{<列格式>}`开始一个表格环境。第二个{}里的内容定义了每一列的格式和分隔符。

- 在`{}`中定义表格每一列的格式与分隔符
  - `l`：左对齐
  - `c`：居中对齐
  - `r`：右对齐
  - `|`：画一条竖直线
- 在`[]`中可选用垂直对齐的参数，表格嵌入段落时的基线位置
  - `t`：顶端对齐
  - `c`：居中对齐
  - `b`：底端对齐
- `&`用来分隔不同单元格
- `\\`用来换行
- `\hline`画一条完整水平线

```
\begin{document}

这是一个段落，紧接着一个默认对齐的表格：
\begin{tabular}{|l|l|}
\hline
第一行&文本 \\
\hline
第二行&文本 \\
\hline
\end{tabular}
表格后的文本。

\vspace{1cm}  %添加一些垂直间距

这是一个段落，紧接着一个顶端对齐的表格：
\begin{tabular}[t]{|l|l|}
\hline
第一行&文本 \\
\hline
第二行&文本 \\
\hline
\end{tabular}
表格后的文本。
```

```
\vspace{1cm}    %添加一些垂直间距

这是一个段落，紧接着一个底端对齐的表格：
\begin{tabular}[b]{|l|l|}
\hline
第一行&文本 \\
\hline
第二行&文本 \\
\hline
\end{tabular}
表格后的文本。

\end{document}
```

同时，也可以对固定列的宽度，利用`p{ }`的表格项

```
\begin{tabular}{|c|lll|p{4em}|}
\hline
姓名&地理&生物&化学&备注 \\
\hline
小红&90&85&92&优秀 \\
\hline
张华&87&75&79&良好 \\
\hline
大明&91&75&97&存在偏科现象 \\
\hline
\end{tabular}
```

还可以利用`@{ }`来插入自定义内容或消除列间距， 插入的内容不会算作一个新列

利用这个特性可以实现按小数点对齐等等

```
\begin{tabular}{l @{--} r}
  起始时间 & 结束时间 \\
  10:00 & 11:00 \\
  12:30 & 13:00 \\
\end{tabular}
```

实现单元格的合并

合并单元格需要`multirow`宏包，使用`\multicolum`命令能够合并列，使用`\multirow`命令能够合并行

合并列

`\multicolumn{<合并的列数>}{<列格式>}{<单元格内容>}`

- <合并的列数>：需要合并的列数
- <列格式>：合并后的单元格的对齐方式,可以是l、c、r，或者用|来定义边框
- <单元格内容>：合并后的单元格中要显示的内容

```
\usepackage{multirow}   %导言区加载
...
\begin{tabular}{|c|c|c|}
\hline
\multicolumn{2}{|c|}{合并两列}&列3 \\
\hline
A&B&C \\
\hline
\end{tabular}
```

需要注意的是\multicolumn命令后的列格式里只能有一个c或l或r或p{宽}

合并行

\multirow{<合并的行数>}{<单元格宽度>}{<单元格内容>}

- <合并的行数>：需要合并的行数
- <单元格宽度>：合并后的单元格的宽度。使用 * 表示自动计算宽度
- 合并后的单元格中要显示的内容

在tabular环境中,\hline命令会画一条横贯整个表格的完整横线，当合并了单元格时，需要用到\cline，它只在指定的列范围内画一条横线

\cline{<起始列号>-<结束列号>}

- <起始列号>：画线的第一列
- <结束列号>：画线的最后一列

```
\usepackage{multirow}   %导言区加载
...
\begin{tabular}{|c|c|c|}
\hline
\multirow{2}{*}{合并两行}&A&B \\ \cline{2-3}  %只画部分横线
& C & D \\
\hline
\end{tabular}
```

一个综合运用：

```
\usepackage{multirow}   %导言区加载
...
\begin{tabular}{|c|c|c|}
\hline
\multicolumn{2}{|c|}{\multirow{2}{*}{合并行和列}}&列3  \\ \cline{3-3}
\multicolumn{2}{|c|}{}&列3 \\
\hline
A&B&C \\
\hline
\end{tabular}
```

下面解释一下这个表格：

1. `\multicolumn{2}{|c|}{\multirow{2}{*}{合并行和列}}`
   - `\multicolumn{2}{...}`：表示合并 2 列
   - `{|c|}`：左右有边框|，内容居中
   - `{\multirow{2}{*}{合并行和列}}`：表示这个单元格要跨越2行，内容是“合并行和列”
2. `cline{3-3}`：画第一行从第3列到第3列的横线
3. `\multicolumn{2}{|c|}{}`：再次合并了2列，但是内容是空的，作用是为`\multirow`单元格的第二行占位，让表格结构正确
4. `cline{3-3}`：画第二行从第3列到第3列的横线

单元格的拆分

`\vline`命令，可以在某一行内画一条只占一行高度的分割线，可以用来拆分某一行已有的表项

```
\begin{tabular}{|c|}
\hline
红 \\ \hline
红 \vline 黄 \\ \hline
红 \vline 黄 \vline 蓝 \\ \hline
\end{tabular}
```

但是使用`\vline`命令拆分表项不易掌握间距，因此对于一些简单的图表可以使用嵌套的表格

这里注意@{}的用法

```
\begin{tabular}{|c|}
\hline
红 \\
\hline
\begin{tabular}{@{}c|c@{}}
红&蓝
\end{tabular} \\
\hline
\begin{tabular}{@{}c|c|c@{}}
红&蓝&黄
\end{tabular} \\
\hline
\end{tabular}
```

浮动表格

`tabular`环境只是创建了表格内容本身，如果希望表格可以浮动到页面的最佳位置，需要将`tabular`放在`table`环境中

`\begin{table}[<位置>]`

- `h`：尽量放在你代码所在的位置（here）

- t：放在页面的顶部（top）
- b：放在页面的底部（bottom）
- p：放在一个单独的浮动页上（page）

也可以组合使用这些参数，例如\begin{table}[htbp]，这告诉LaTeX按照h、t、b、p的顺序尝试放置表格

同时配合其他命令：

- \centerin：将表格居中
- \caption{}：为表格添加标题，它会自动编号
- \label{}：为表格打上标签，方便在正文中引用

```
\begin{table}[h]
\centering
\caption{水果价格表}
\label{tab:price}
\begin{tabular}{|l|c|r|}
\hline
水果&数量&价格 \\
\hline
苹果&5&10 \\
\hline
香蕉&3&6 \\
\hline
\end{tabular}
\end{table}
```

三线表也是一种常用的表格
三线表（只用横线，不用竖线）是一种更常见、更美观的表格形式，需要用到booktabs宏包

- \toprule：画顶线
- \midrule：画中线
- \bottomrule：画底线

booktabs宏包通常与tabular或table环境一起使用，在tabular列格式中不用画竖线

```
\documentclass{article}
\usepackage{booktabs}   %导入宏包
...
\begin{table}[htbp]
\centering
\caption{三线表示例}
\label{tab:example}
\begin{tabular}{lcr}  %列中没有竖线
\toprule   %顶线
项目&数量&价格 \\
\midrule   %中线
苹果&5&10元 \\
香蕉&3&6元 \\
橙子&7&14元 \\
```

```
\bottomrule % 底线
\end{tabular}
\end{table}
```

一般来说，更加推荐画三线表

图片

要插入图片，需要`\usepackage{graphicx}`宏包

`\includegraphics[]{}`
可以在方括号内添加参数，用于控制图片的大小和样式

- `width=<宽度>`：设置图片宽度，通常使用相对长度如`0.8\textwidth`（文本宽度的 80%，这样可以确保图片在不同页面大小下都能自适应
- `height=<高度>`：设置图片高度
- `scale=<比例>`：按比例缩放图片，例如`scale=0.5`会将图片缩小一半
- `angle=<角度>`：旋转图片，例如`angle=90`

花括号里填写图片的 **文件名**，如果不在同一个文件夹里，需要提供 **完整路径**

但`\includegraphics`命令只会把图片放在代码所在的位置，为了让图片能“浮动”到最合适的位置，并且能添加标题、标签和在目录中显示，你需要将它放在`figure`环境中

`\begin{figure}[<位置>]`
与`\begin(table)`用法相同，均提供浮动环境

```
\usepackage{graphicx}    %记得在引言区加载宏包
...
\begin{figure}[htbp]
\centering
\includegraphics[width=0.8\textwidth]{example-image}
\caption{示例图片}
\label{fig:flowchart}
\end{figure}
```

引用

可以给任何章节、图标或公式打上`\label{<标签名>}`标签，然后在文档的其他地方用`\ref{<标签名>}`来引用它。文档内容变化时，latex会自动更新引用编号。

`\label`命令必须放在你想要引用的对象之后，对于章节，通常放在 ***`\section`命令之后****，对于浮动体，则* ***放在`\caption`之后***

{ }内的参数可以填写任何想要的文本标签，但为了方便管理和阅读，一般遵循以下规则：

- **唯一性：** 标签在整个文档中必须是唯一的
- **语义化：** 标签应清楚的表明它所标记的内容

- **前缀：** 使用前缀来区分不同类型的对象是一个好习惯

推荐的命名格式：

- `sec:`：用于章节（section），如`\section{引言}\label{sec:intro}`
- `fig:`：用于图片或图（figure），如`\caption{流程图}\label{fig:flowchart}`
- `tab:`：用于表格（table），如`\caption{实验数据}\label{tab:data}`
- `eq:`：用于公式（equation），如`\begin{equation}\label{eq:einstain}`

```
\section{简介}
\label{sec:intro}
...
...
正如第\ref{sec:intro}节所述...
```