FREE eBook

LEARNING latex

Free unaffiliated eBook created from
Stack Overflow contributors.

#latex

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: latex

It is an unofficial and free latex ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official latex.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with latex

Remarks

LaTeX

What is LaTeX?

LaTeX (pronounced *lay-tech* or *lah-tekh*) is a markup language for typesetting documents similar to how HTML is one for web sites.

LaTeX has advantages over What-You-See-Is-What-You-Get (WYSIWYG) editors such as Microsoft Word because with LaTeX you provide the content, and LaTeX takes care of the layout. Separation of content from typesetting results in documents that are consistently and beautifully formatted. Furthermore, because LaTeX markup is of plain text format (unlike more complex file formats produced by WYSIWYG editors, like `.docx`), LaTeX files are lightweight and can be easily kept under version control.

LaTeX documents are typically compiled to PDF files so that consistency in layout is retained across different viewers, and for printing.

LaTeX is especially popular in academic writing due to its rich support for typesetting equations, cross-referencing figures and tables, and citations and bibliographies.

Versions

Version	Release Date
LaTeX 2.09	1985-09-01
LaTeX 2e	1994-06-01

Examples

Installation and Setup

You can choose between major distributions of LaTeX:

- TeX Live (Windows, Linux, and OS X), the standard, cross-platform distribution.
- MacTeX (Mac) A packaged version of TeX Live made for OS X with some Mac-specific tools
- MiKTeX (Windows) A separate distribution entirely that

All distributions are more or less equivalent in an ideal world. TeX Live has the advantage of being

available on all platforms and thus has much better community support. MiKTeX can take advantage of Windows-specific features. For licensing reasons, MiKTeX will also distribute a few packages that TeX Live will not.

In all cases, the full install is recommended. Specifically, using MiKTeX's download-on-command feature will hang/crash many editors.

Installation

Windows (TeXLive)

1. Download the most recent TeXLive `install-tl-windows.exe` from their website.
2. Run `install-tl-windows.exe` and follow the instructions.

Windows (MiKTeX)

1. Download the most recent MiKTeX installer from their website.
2. Run the installer and follow the instructions.

Mac OS X (TeXLive)

1. Download the most recent MacTeX from their website.
2. Run `MacTeX.pkg` and follow the instructions.

Linux (TeXLive)

Linux users have two options:

1. Install via your distribution's package manager (usually several releases behind)
2. Install from upstream (released yearly, updated often)

Using Package Managers

- Arch Linux: `pacman -S texlive-most`
- Debian/Ubuntu/Mint: `apt-get install texlive-full`
- Fedora: `yum install texlive`

Note that using this method means that you will be dependent on that package's maintainer for the distribution for updates. These packages will often be several releases behind the most recent distribution, often meaning critical updates will be missing. It's almost always best to install from upstream. Also note that the distribution's package manager will probably not recognize the direct installation and could try to install it when one installs other related support packages.

Installing from Upstream

1. Download the most recent TeXLive `install-tl-unx.tar.gz` from their website.
2. Extract the files from the archive with `tar -zxvf install-tl-unx.tar.gz`.
3. Change into the downloaded folder with `cd install-tl-unx`.
4. Run `./install-tl` and follow the instructions.

   TeXLive should now be installed under `/usr/local/texlive/YEAR/`, where `YEAR` is the four digit year (e.g. `2016`). In this way, it is possible to have multiple TeXLive versions alongside each other and switch between them by changing your PATH variable.

   Open this folder and check the `bin` folder. It should contain a subfolder, which (depending on your platform) will be something like `i386-linux` or `x86_64-linux`.
5. Add the TeX Live binary folder to your path with

   ```
   EXPORT PATH=/usr/local/texlive/YEAR/bin/PLATFORM:$PATH
   ```

   where `YEAR` is the four digit year (e.g. `2016`), and `PLATFORM` is your platform (e.g. `x86_64-linux`).

Test Installation

The LaTeX installation is now complete. To test it, create a new file with your favorite text editor, name it `test.tex` and add the following content:

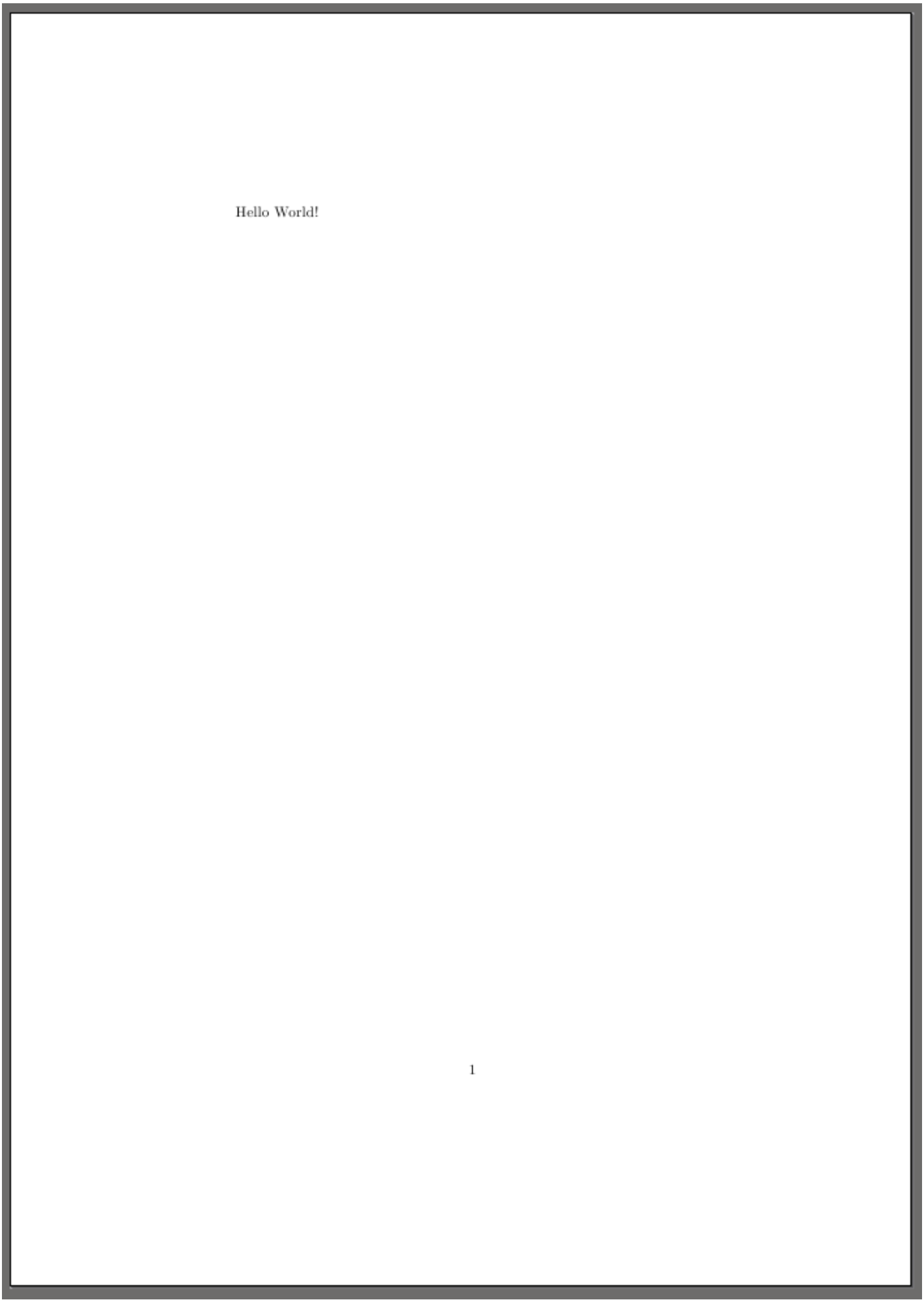
```
\documentclass{article}
\begin{document}
  Hello World!
\end{document}
```

Now, open the console or terminal, navigate to the folder where you saved `test.tex` and run

```
pdflatex test
```

(Note that your editor may have facilities to run this for you.)

This creates several new files, including `test.pdf`. This is the output document, and looks like this:



Congratulations, you have successfully installed LaTeX, and created your first LaTeX document!

- The Vim editor with the LaTeX-suite plugin.
- Texmaker – a specialized LaTeX IDE.
- TeXstudio – another LaTeX IDE.
- TeXworks – one more LaTeX IDE.

While experienced users of Emacs or Vim may want to stick to their editor (whose plugins provide a host of functionality unavailable elsewhere), a specialized IDE might be easier to install/use for beginners. The last three on the list have a preview function where one can see the results of the compilation of the document.

Additionally, there are online LaTeX tools that can be of use to beginners or people that must collaborate, e.g. ShareLaTeX and Overleaf.

Read Getting started with latex online: https://riptutorial.com/latex/topic/2269/getting-started-with-latex

Chapter 2: Accessing documentation of LaTeX packages

Examples

CTAN

The Comprehensive TeX Archive Network (CTAN) is indeed that, *the* comprehensive repository of LaTeX packages. Most if not all quality packages (and more) are on there, and all the good ones include documentation.

1. Enter the package name into the search bar.

2. Select the package from the list.

3. Access the documentation documents.

BibLaTeX – Sophisticated Bibliographies in LaTeX

BibLaTeX is a complete reimplementation of the bibliographic facilities provided by LaTeX. Formatting of the bibliography is entirely controlled by LaTeX macros, and a working knowledge of LaTeX should be sufficient to design new bibliography and citation styles. BibLaTeX uses its own data backend program called "biber" to read and process the bibliographic data. With biber, BibLaTeX has many features rivalling or surpassing other bibliography systems. To mention a few:

- Full Unicode support
- Highly customisable sorting using the Unicode Collation Algorithm + CLDR tailoring
- Highly customisable bibliography labels
- Complex macro-based on-the-fly data modification without changing your data sources
- A tool mode for transforming bibliographic data sources
- Multiple bibliographies and lists of bibliographic information in the same document with different sorting
- Highly customisable data source inheritance rules
- Polyglossia and babel suppport for automatic language switching for bibliographic entries and citations
- Automatic bibliography data recoding (UTF-8 -> latin1, LaTeX macros -> UTF-8 etc)
- Remote data sources
- Highly sophisticated automatic name and name list disambiguation system
- Highly customisable data model so users can define their own bibliographic data types
- Validation of bibliographic data against a data model
- Subdivided and/or filtered bibligraphies, bibliographies per chapter, section etc.

Apart from the features unique to BibLaTeX, the package also incorporates core features of the following packages: babelbib, bibtopic, bibunits, chapterbib, cite, inlinebib, mcite and mciteplus, mlbib, multibib, splitbib.

Sources	/macros/latex/contrib/biblatex
Documentation	Readme Package documentation (English) Release notes for current version
Version	3.4
License	The LaTeX Project Public License 1.3
Copyright	2012–2016 Philipp Lehman, Joseph Wright, Audrey Boruvka, Philip Kime 2006–2012 Philipp Lehman
Maintainer	Philipp Lehman (inactive) Philip Kime
Contained in	TeX Live as biblatex MiKTeX as biblatex
Topics	BibLaTeX bibliography support bibliography processor

Download the contents of this package in one zip archive (17.2M).

Important: CTAN holds the most recent versions. If your installation is outdated the documentation won't match! In that case, refer to the documentation documents shipped with your

LaTeX distribution.

TeX Live -- texdoc

If you use the TeX Live distribution you can use the command-line program `texdoc`. For instance,

```
texdoc biblatex
```

will open the documentation of package `biblatex`.

Or if you are not command-line-savvy, the same can be found online at http://www.texdoc.net/

Read Accessing documentation of LaTeX packages online:
https://riptutorial.com/latex/topic/5820/accessing-documentation-of-latex-packages

Chapter 3: Add Citation

Examples

Add citation to already existing LaTeX document

At the end of the document add the following:

\bibliographystyle{*style*}

\bibliography{*file location*}

Create a file with extension *.bib* and save the citation as follows:

```
@inproceedings{citation_name,
  title={Paper Title},
  author={List Authors},
  pages={45--48},
  year={2013},
  organization={organization name}
}
```

To cite use the following: **\citet{*citation_name*}**

Read Add Citation online: https://riptutorial.com/latex/topic/8357/add-citation

Chapter 4: Build Tools

Examples

Arara

Arara is a cross-platform automation tool that's specially designed for TeX. It's included in a standard distribution, so there's no need to install anything additional. It's most effectively understood as a means to record the compilation instructions in the TeX file itself:

```
% arara: pdflatex
\documentclass{article}
\begin{document}
  Hello, world
\end{document}
```

These can be much more complicated, though:

```
% arara: pdflatex
% arara: biber
% arara: pdflatex

% To support a self-contained example, this builds a BibTeX file on-the-fly
\begin{filecontents}{references.bib}
@article{dijkstra,
 author = {Dijkstra, Edsger W.},
 title = {Self-stabilizing Systems in Spite of Distributed Control},
 journal = {Commun. ACM},
 issue_date = {Nov. 1974},
 volume = {17},
 number = {11},
 month = nov,
 year = {1974},
 issn = {0001-0782},
 pages = {643--644},
 numpages = {2},
 url = {http://doi.acm.org/10.1145/361179.361202},
 doi = {10.1145/361179.361202},
 acmid = {361202},
 publisher = {ACM},
 address = {New York, NY, USA},
 keywords = {distributed control, error recovery, harmonious cooperation, multiprocessing,
mutual exclusion, networks, robustness, self-repair, self-stabilization, sharing,
synchronization},
}
\end{filecontents}


\documentclass{article}
\usepackage[backend=biber]{biblatex}
\addbibresource{references.bib}

\begin{document}
Hello, World! \cite{dijkstra}.
```

```
\printbibliography
\end{document}
```

Read Build Tools online: https://riptutorial.com/latex/topic/5015/build-tools

Chapter 5: Counters, if statements and loops with latex

Examples

Counter declaration, initialization and printing to pdf

It is possible to use integer variables with latex. To create a new variable we need the `\newcounter{name}` command, where `name` is the name of the new counter. The `name` must contain only letters. This command creates a new one with name `\thename`. With this command we can print `name` variable onto the paper. The initial value of `name` is 0. To give value to "name" we can use `\setcounter{name}{n}` where n is an integer. `\value{name}` is a function which returns with the value of `name`.

```
\documentclass{article}
\begin{document}
\newcounter{num}                    %new counter, initial value is 0
\thenum                             %print 0
\setcounter{num}{3}                 %set num to 3
\thenum                             %print 3
\newcounter{number}
\setcounter{number}{\value{num}}    %set number to value of num
\thenumber                          %print 3

Latex provides some other formats to print a number.

Other types of printing:

\arabic{num}\\
\Roman{num}\\ %→ I, II, III, IV, . . . (num = 1, 2, 3, . . . )
\roman{num}\\ %→ i, ii, iii, iv, . . . (num = 1, 2, 3, . . . )
\Alph{num}\\  %→ A, B, C, D, . . . (num = 1, 2, 3, . . . , 26)
\alph{num}\\  %→ a, b, c, d, . . . (num = 1, 2, 3, . . . , 26)
\fnsymbol{num}\\ %→ *, †, ‡, §, ¶, k, **, ††, ‡‡ (num = 1, 2, 3, . . . , 9)
\end{document}
```

0
3
3
Latex provides some other formats to print a number.
Other types of printing:
3
III
iii
C
c
‡

Operations with counters

This example shows how to use mathematical operations with counters. It may be useful for loops in latex.

Addition: `\addtocounter{num}{n}`

this command adds `n` to `num`, where `num` is a counter and `n` is a positive integer.

Subtraction: `\addtocounter{num}{-n}`

this command subtracts `n` from `num`, where `num` is a counter and `n` is a positive integer.

Multiplication: `\multiply\value{num} by n`

this command multiply `num` by `n`, where `num` is a counter and `n` is an integer.

Division `\divide\value{num} by n`

this command divides `num` by `n` and gets the integer part of the quotient (`num` is a counter and `n` is an integer)

```
\documentclass{article}
    \begin{document}
    \newcounter{num}
    \setcounter{num}{3}
    \addtocounter{num}{10}
    \thenum\\%prints 13
    \addtocounter{num}{-3}
    \thenum\\%prints 10
    \stepcounter{num}
    \thenum\\%prints 11
    \multiply\value{num} by \value{num}
    \thenum\\%prints 121
    \multiply\value{num} by 2
    \thenum\\%prints 242
    \divide\value{num} by 60
    \thenum%prints 4
\end{document}
```

`\newcommand{num}` declares counter. `\setcounter{num}{3}` sets num value to 3.

`\addtocounter{num}{10}` adds 10 to num.

`\addtocounter{num}{-3}` subtract 3 from num.

`\stepcounter{num}` adds 1 to num

`\multiply\value{num} by \value{num}` squares num.

`\multiply\value{num} by 2` doubles num.

`\divide\value{num} by 60` divides num by 60 and gets the integer part.

The result of the code: 13\\10\\11\\121\\242\\4

(\\ symbolizes new line)

intcalc package adds some other integer operations e.g. mod, pow, sng, abs, inv ...

intcalc_package.pdf

If statements

In latex we can use built-in commands to execute code whether the conditions are true or not.

Comparing two integers: `\ifnum\value{num}>n {A} \else {B}\fi`

This code executes A if num>n else B. We can substitute > with < and =.

If a number is odd: `\ifodd\value{num} {A}\else {B}\fi`

If num is odd then it executes A else B.

If with condition: `\ifthenelse{condition}{A}{B}`

We have to load ifthen package to use this command. If condition are true then it executes A else B.

It is possible to create complex condition with `\( \)`, `\AND`, `\OR`, `\NOT`.

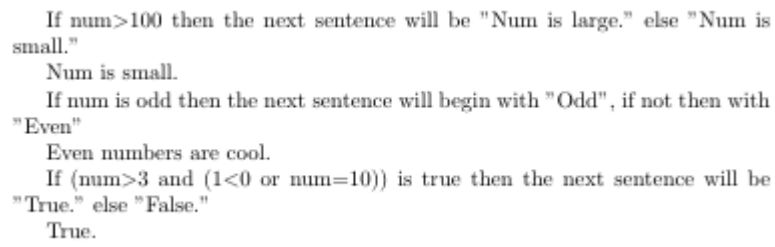
For example: `\ifthenelse{\(\NOT 4<2 \OR 4>11\)\AND\isodd{4}}{A}{B}`

This piece of code writes down "B" on the page. `\NOT 4<2` is true and `4>11` is false. If we connect a false and a true statement with "OR" then the result is true. So `\(\NOT 4<2 \OR 4>11\)` is true. `\isodd{4}` is false because 4 is even. A false and a true statement connected with "AND" is false, so the output is B.

An example code:

```
\documentclass{article}
\usepackage{ifthen}
\begin{document}
    \newcounter{num}
    \setcounter{num}{10}

    If num$>$100 then the next sentence will be "Num is large." else "Num is small."

    Num is \ifnum \value{num}>100 {large} \else {small}.

    If num is odd then the next sentence will begin with "Odd" if not then with "Even"

    \ifodd \value{num} {Odd} \else {Even} numbers are cool.

    If (num$>$3 and (1$<$0 or num$=$10)) is true then the next sentence will be "True." else
"False."

    \ifthenelse{\value{num}>3\AND\(1<0 \OR \value{num}=10\)}{True.}{False.}
```

```
\end{document}
```


If num>100 then the next sentence will be "Num is large." else "Num is small."
Num is small.
If num is odd then the next sentence will begin with "Odd", if not then with "Even"
Even numbers are cool.
If (num>3 and (1<0 or num=10)) is true then the next sentence will be "True." else "False."
True.

Loops - repeating things

We can create loops in latex. They are similar but not as customizable as loops in other programming languages. One alternative to use loops are @loops. If we use a command which includes "@" in its name, we must be put it between `\makeatletter` and `\makeatother`. It is not allowed to use them in a macro which describes a new definition.

Wrong:

```
\def\is#1#2{\makeatletter\@ifstar{#1}{#2}\makeatother
```

Right:

```
\makeatletter\def\is#1#2{\@ifstar{#1}{#2}}\makeatother
```

@for loop: `\@for\command:={list}\do{commands}`

Example:

```
\makeatletter
\@for\sun:={rising,setting}\do{The sun is \sun.}
\makeatother
```

It creates the following text: The sun is rising. The sun is setting.

@whilenum loop: `\@whilenum condition\do{commands}`

Example:

```
\makeatletter
\newcounter{int}
\@whilenum\value{int}<10\do
{\stepcounter{int}\ifthenelse{\isodd{\value{int}}}{\theint}{}}
\makeatother
```

This code writes odd numbers from 1 to 9.

"loop repeat" loop: `\loop {commands} \ifnum condition \repeat`

Executes commands till condition is true.

Example

```
\setcounter{int}{1}
\loop
\theint
\addtocounter{int}{2}
\ifnum \value{int}<10
\repeat
```

This code does the same as @whilenum loop.

An example code:

```
\documentclass{article}
\usepackage{ifthen}
\usepackage{amsmath} %\text{} command needs this package
\begin{document}
    Demonstration of @for loop:

    \makeatletter
    \@for\sun:={rising,setting}\do{The sun is \sun. }
    \makeatother

    \newcounter{int}

    @whilenum loop:

    \setcounter{int}{0}
    \makeatletter
    \@whilenum\value{int}<20\do
    {\stepcounter{int}\ifthenelse{\isodd{\value{int}}}{\theint\text{ }}{}}
    \makeatother

    "loop repeat" loop:

    \setcounter{int}{1}
    \loop
    \theint
    \text{ }\addtocounter{int}{2}\ifnum\value{int}<20
    \repeat
\end{document}
```

Demonstration of @for loop:
The sun is rising. The sun is setting.
@whilenum loop:
1 3 5 7 9 11 13 15 17 19
"loop repeat" loop:
1 3 5 7 9 11 13 15 17 19

Using loops in Tikz

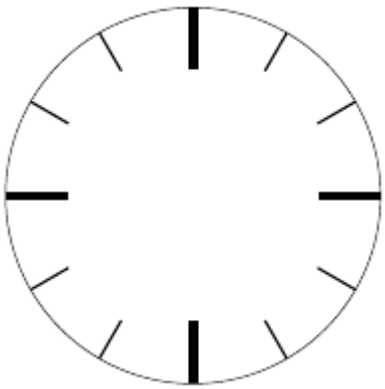
Loops are useful in Tikz.

The following code draws a clock without numbers:

```
\documentclass{article}
\usepackage{ifthen}
```

```
\usepackage{intcalc}
\usepackage{tikz}
\newcounter{num}

\begin{document}
\begin{tikzpicture}
    \makeatletter
    \setcounter{num}{1}
    \newcounter{angle}
    \draw (0,0) circle (3cm);
    \@whilenum\value{num}<13\do{
    \setcounter{angle}{360}
    \multiply\value{angle} by \value{num}
    \divide\value{angle} by 12
    \ifnum \intcalcMod{\value{num}}{3}=0{
    \draw[line width=4pt] (\theangle:2cm) -- (\theangle:3cm);    }\else
    {
    \draw[line width=1pt] (\theangle:2.3cm) -- (\theangle:3cm);
    }\fi
    \addtocounter{num}{1}
    }
    \makeatother
\end{tikzpicture}
\end{document}
```

The result:



Read Counters, if statements and loops with latex online:
https://riptutorial.com/latex/topic/9224/counters--if-statements-and-loops-with-latex

Chapter 6: Creating a Bibliography

Syntax

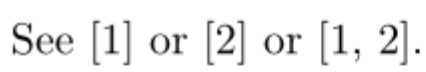
- For a manually-formatted bibliography, there is no need to have citations - `\cite` - within the document.

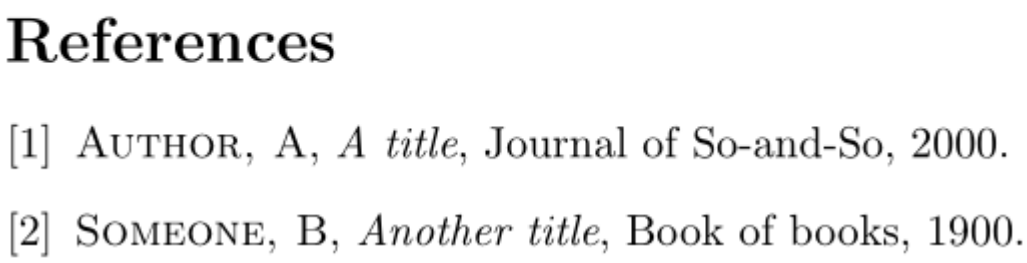
Parameters

Parameter	Detail
`thebibliography`	This environment sets the scope for the actual bibliography. It defines a list-like environment within which you can use `\bibitem` to set a bibliography item.
`{x}`	The `thebibliography` environment takes a single argument that represents the widest element to be expected in the enumeration of the `\bibitem`s. For less than 10 entries, use a single character/digit; for less than 100 entries, use two characters/digits, ...
`\bibitem{<a>} <b>`	Set the bibliography item `<b>` and make it available to `\cite` within the document using the label `<a>`.

Examples

Basic bibliography without packages (manual formatting)






```
\documentclass{article}% or book, report, ...

\begin{document}

See \cite{citeA} or \cite{citeB} or \cite{citeA, citeB}.

\begin{thebibliography}{x}
  % \bibitem{<biblabel>} <citation>
```

```
  \bibitem{citeA}
    {\scshape Author, A}, {\itshape A title}, Journal of So-and-So, 2000.
  \bibitem{citeB}
    {\scshape Someone, B}, {\itshape Another title}, Book of books, 1900.
\end{thebibliography}

\end{document}
```

Note that unless you really know *why*, you should probably not do this. Using designated packages (see other examples) is preferable.

Basic bibliography with biber

To start a bibliography you need to define your sources. Create a database file (like `sources.bib`) and include some content:

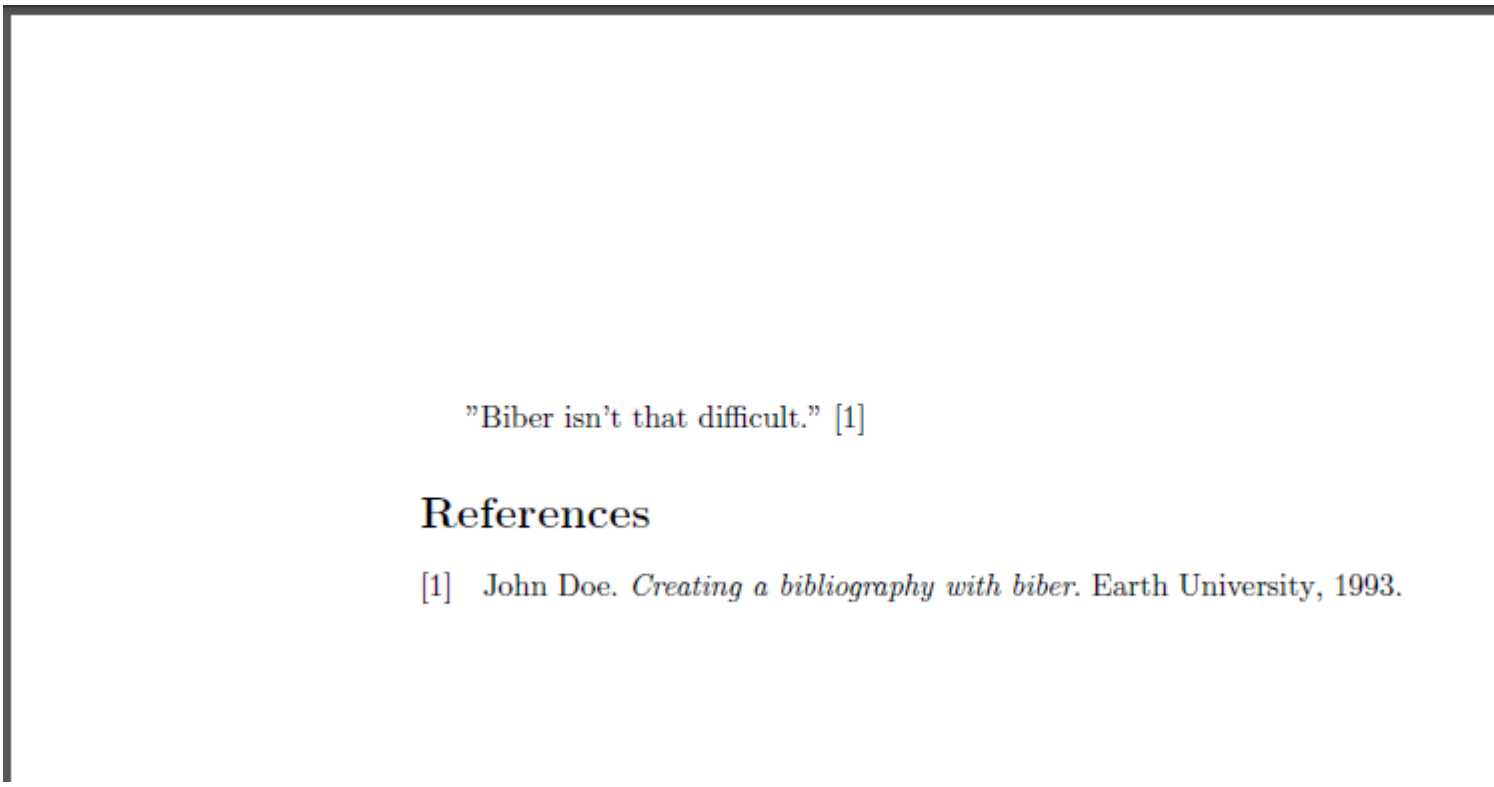
```
@book{Doe1993,
    Author = {John Doe},
    Publisher = {Earth University},
    Title = {Creating a bibliography with biber},
    Year = {1993}}
```

You can then include your database file in your main document and cite the new source (`Doe1993`).

```
\documentclass{article}

% Include the biblatex package and tell it to use biber as a backend.
% Without specifying the backend, it assumes biber.
\usepackage[backend=biber]{biblatex}

% Define where biber can find your sources
\addbibresource{sources.bib}

\begin{document}
"Biber isn't that difficult." \cite{Doe1993}
% Use \cite{source-ID} to generate a citation

% Print the bibliography
\printbibliography

\end{document}
```

To compile the document, you will need to run 3 commands in sequence:

1. `pdflatex` to create an auxiliary file which tells biber what sources are needed
2. `biber` to create an auxiliary file with all the sources which can be used by `pdflatex`
3. `pdflatex` to include the auxiliary file and create the PDF



Find many more options and additional fields for bib files in the package documentation on CTAN.

Read Creating a Bibliography online: https://riptutorial.com/latex/topic/3488/creating-a-bibliography

Chapter 7: Creating posters using beamer

Introduction

Creating a poster using beamerposter package is very similar to creating a single frame. Put the content in columns. Within each column, separate the content using blocks.

Examples

Orientation and size

While adding the beamerposter package, provide the required parameters.

```
\usepackage[orientation=landscape,size=a1]{beamerposter}
```

You can also customize the size of the poster.

```
\usepackage[orientation=portrait,size=custom,height=110,width=80,scale=1.4]{beamerposter}
```

The height and width dimensions here, are in cms. The `scale` is used for the font size.

Basic outline of a beamer poster

In landscape orientation

```
\documentclass[final,t]{beamer}
\mode<presentation>
  {
  \usetheme{Berlin}
  }

\usepackage[orientation=landscape,size=a1,scale=1,debug]{beamerposter}
\usepackage{lipsum} % for dummy text


\title[]{\huge Awesome title}
\author[]{\large \textbf{Author Name1} \and Author Name2 \and Author Name3}
\institute[]{\Large Dept of XYZ, ABC Institute}
\date{}

\begin{document}

\begin{frame}
\maketitle
\begin{columns}[t]
  \begin{column}{.32\linewidth}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
```

```
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}


  \end{column}

 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


  \begin{column}{.32\linewidth}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}


  \end{column}


  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

  \begin{column}{.32\linewidth}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}

  \begin{block}{Some heading}
  \lipsum[1]
  \end{block}


  \end{column}
  \end{columns}

\end{frame}

\end{document}
```

Some heading

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Some heading

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Some heading

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

https://riptutorial.com/latex/topic/10714/creating-posters-using-beamer

Chapter 8: Defining macros

Syntax

- \newcommand{\macro}{replacement text}
- \newcommand{\macro}[argcount]{replacement text}
- \renewcommand{\macro}{replacement text}
- \renewcommand{\macro}[argcount]{replacement text}

Parameters

Parameter	Details
`\macro`	The macro to define
`argcount`	The number of arguments the macro expects (optional)
`replacement text`	The replacement text for the macro. Inside that text `#1`, `#2` etc. are replaced with the macro arguments.

Examples

Basic definition of macros

Define a new basic command

A macro can be defined using `\newcommand`. For example:

```
\newcommand{\foo}{Just foo, you see?}
```

defines a macro `\foo` that expands to `Just foo, you see?`. It can then be used like any built-in command, for example after that definition:

```
He said: ``\foo''
```

expands to

```
He said: ``Just foo, you see?''
```

Define a new command with arguments

Macros can also have arguments. The number of arguments is given as optional argument

between the command name and the replacement text. In the replacement text, the arguments are accessed with `#1`, `#2` etc. For example:

```
\newcommand{\better}[2]{A #1 is better than a #2.}
\better{solution}{problem} % gives: A solution is better than a problem
```

Redefining an existing command

If a macro has already been defined, `\newcommand` gives an error. To give a new definition for an existing command, `\renewcommand` is used instead. Other than the different name, the syntax is exactly the same. For example, after the definition of `\foo` above, one could use:

```
\renewcommand{\foo}{Another foo, please.}
```

After that redefinition, the macro `\foo` no longer expands to `Just foo, you see?` but to `Another foo, please.`

Read Defining macros online: https://riptutorial.com/latex/topic/7658/defining-macros

Chapter 9: Document classes

Syntax

- \documentclass{...}

Remarks

This topic aims to explain the different types of document and their specificities.

A good way to organize it would be 1 example per type

Examples

Article

```
\documentclass{article}
```

When to use the article class ?

For articles in scientific journals, presentations, short reports, program documentation, invitations, ... [1]

What are the specificities of this class ?

An article doesn't contain chapters or parts. It can be divided in sections, subsections and paragraphs etc.

By default, the title is shown at the top of the first page, and not on a separate title page.

Simple example

```
\documentclass{article}

\title{Hello world}
\author{Me    }
\date{\today}

\begin{document}

\maketitle

Hello, World!
```

```
\end{document}
```

Beamer

```
\documentclass{beamer}
```

When to use the beamer class ?

For presentation slides.

What are the specificities of this class ?

The output is landscape-oriented. The document is separated in "frames" (slides).

Simple example

Following example was adapted from : sharelatex.com/learn/Beamer

```
\documentclass{beamer}

\usepackage[utf8]{inputenc}

\title{Sample title}
\author{Me}
\date{\today}

\begin{document}

\frame{\titlepage}

\begin{frame}
\frametitle{Sample frame title}
This is a text in first frame. This is a text in first frame. This is a text in first frame.
\end{frame}

\end{document}
```

Defining the document class

The very first line in each of your LaTeX programs should do this. It should follow the form `\documentclass{...}`. What you put within the curly braces is very important. Some document classes give you extra commands to use, others use a different format, and all have specific parameters you can input (described in the parameters section).

Read Document classes online: https://riptutorial.com/latex/topic/7458/document-classes

Chapter 10: Drawing graphs

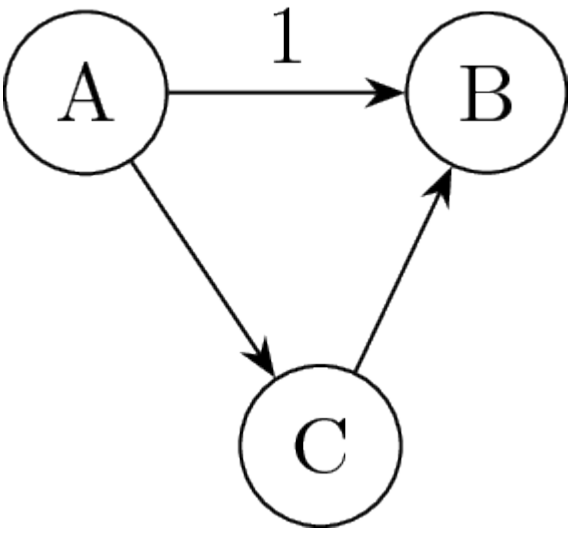
Examples

TikZ -- Manual layout

Package TikZ lends itself very well to drawing graphs.

This is a small example (requires TikZ 3.0+):

```
\documentclass{standalone}

\usepackage{tikz}
\usetikzlibrary{positioning,arrows.meta}

\begin{document}
  \begin{tikzpicture}[auto,vertex/.style={draw,circle}]
    \node[vertex] (a) {A};
    \node[vertex,right=1cm of a] (b) {B};
    \node[vertex,below right=1cm and 0.5cm of a] (c) {C};

    \path[-{Stealth[]}]
      (a) edge node {1} (b)
      (a) edge (c)
      (c) edge (b);
  \end{tikzpicture}
\end{document}
```

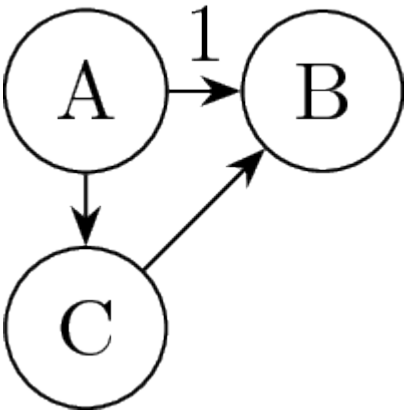


You can create arbitrarily complex graphs; beware lengthy code, though. Recall that there is `\foreach` and take note of all the positioning and styling options (cf. TikZ manual, section 13 to 17).

TikZ -- Graph specifications

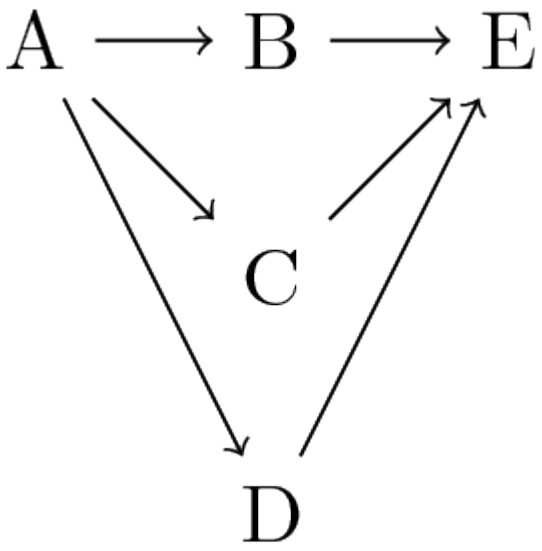
TikZ provides syntax similar to DOT which you can use to tighten up your graph drawing code considerably.

```
\documentclass{standalone}

\usepackage{tikz}
\usetikzlibrary{graphs,quotes,arrows.meta}

\begin{document}
  \begin{tikzpicture}
    \graph[nodes={draw,circle},edges={-{Stealth[]}}] {
      A -> ["1"] B,
      A -> C,
      C -> B
    };
  \end{tikzpicture}
\end{document}
```



As you can see, you trade fine-grained control for easier syntax. The `graphs` library really shines when you specify more complicated graphs:

```
\documentclass{standalone}

\usepackage{tikz}
\usetikzlibrary{graphs,graphs.standard}

\begin{document}
  \begin{tikzpicture}
    \graph {
      A -> { subgraph I_n [V= {B,C,D}] } -> E
    };
  \end{tikzpicture}
\end{document}
```

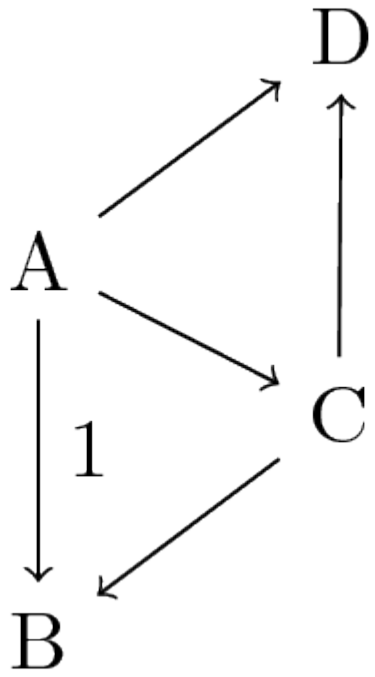


There are many more options and pre-defined graphs; see section 19 of the TikZ manual.

TikZ -- Algorithmic graph drawing

TikZ implements several algorithms for *automatic* graph layouts (requires LuaLaTeX).

```
\documentclass{article}

\usepackage{tikz}
\usetikzlibrary{graphs,graphdrawing,quotes}
\usegdlibrary{force}

\begin{document}
  \begin{tikzpicture}
    \graph[spring layout] {
      A -> ["1"] B,
      A -> {C, D},
      C -> {B, D},
    };
  \end{tikzpicture}
\end{document}
```



There are several algorithms and many options to influence them. See part IV of the TikZ manual for details.

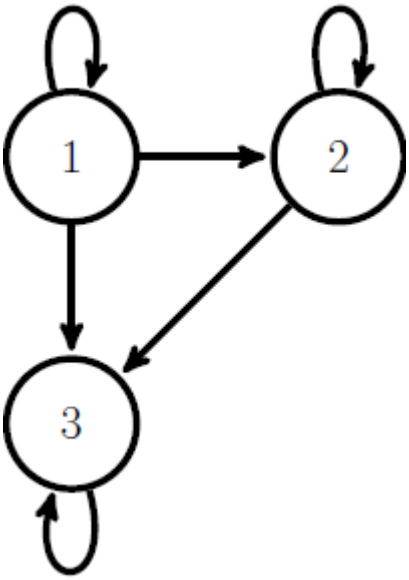
State Transition Diagram of a Markov Chain

Suppose the following matrix is the transition probability matrix associated with a Markov chain.

```
    0.5   0.2  0.3
P=  0.0   0.1  0.9
    0.0   0.0  1.0
```

In order to study the nature of the states of a Markov chain, a state transition diagram of the Markov chain is drawn.

```
\documentclass[12pt,a4paper]{article}
\usepackage{tikz}
\usetikzlibrary{shapes,arrows,positioning}
\begin{tikzpicture}[->,>=stealth',shorten >=2pt, line width=3pt,
                                node distance=2cm, style ={minimum size=20mm}]
\tikzstyle{every node}=[font=\huge]
\node [circle, draw] (a) {1};
\path  (a) edge [loop above] (a);
\node [circle, draw] (b) [right=of a] {2};
\path  (b) edge [loop above] (b);
\draw[->] (a) -- (b);
\node [circle, draw] (c) [below=of a] {3};
\path  (c) edge [loop below] (c);
\draw[->] (a) -- (c);
\draw[->] (b) -- (c);
\end{tikzpicture}
```



Read Drawing graphs online: https://riptutorial.com/latex/topic/5955/drawing-graphs

Chapter 11: Engraving Sheet Music

Examples

LilyPond

The LilyPond notation engraver can be used with LaTeX via the `lilypond-book` command. First lets create a LaTeX document (with the file extension `.lytex`) to embed our music in:

```
\documentclass[letterpaper,12pt]{article}

\begin{document}

\begin{center}
    {\fontsize{24pt}{24pt}\textbf{Twa Corbies}}\\
\end{center}

\begin{flushright}
    \textsc{Your Name}
\end{flushright}

% We don't need to require anything for this because lilypond-book will process it.
\lilypondfile{TwaCorbies.ly}
\end{document}
```

Then we create our LilyPond file (`.ly`), including the `lilypond-book-preamble.ly` file (which LilyPond will know how to find):

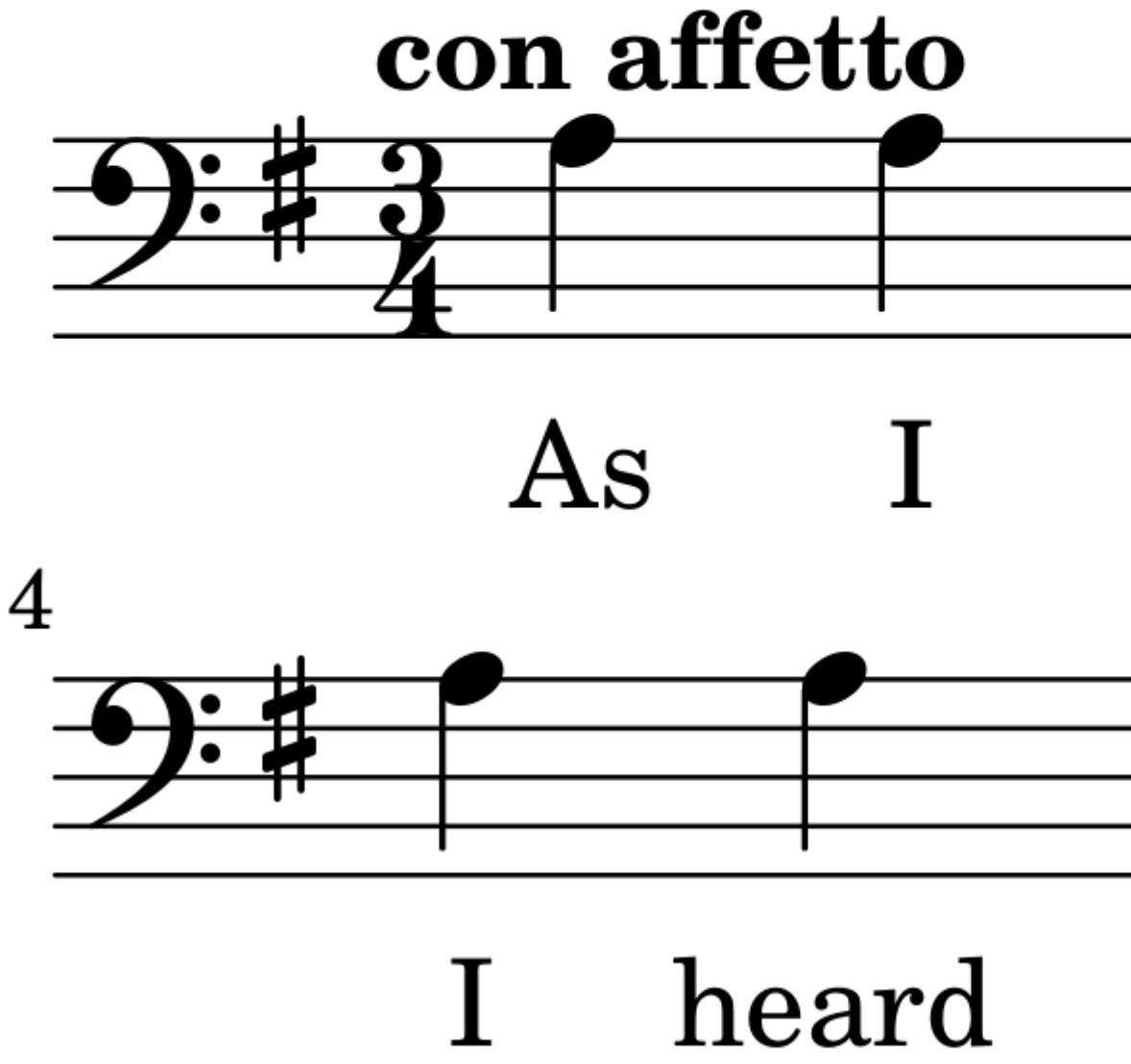
```
\version "2.16.2"

\include "lilypond-book-preamble.ly"

voice = <<
    \relative c' {
        \tempo "con affetto"
        \clef bass
        \key e \minor
        \time 3/4

        a a b | c a a | g a2 |
        a4 a b | c2 ~ c8 a8 | a8 g a2 |
        \bar "|."
    }
    \addlyrics{
        As I was wal -- king all a -- lane
        I heard twa cor -- bies make a mane.
    }
>>

\score {
    <<
        \new Staff = "voice" {
            \voice
        }
```

```
    >>
    \layout { }
    \midi {
        \context {
            \Score
            tempoWholesPerMinute = #(ly:make-moment 90 4)
        }
    }
}
```

to build, we then run the `lilypond-book` command:

```
lilypond-book --include=mymusicsourcedirectory/ --pdf TwaCorbies.lytex
```

which will output a PDF containing your LilyPond engraved music:

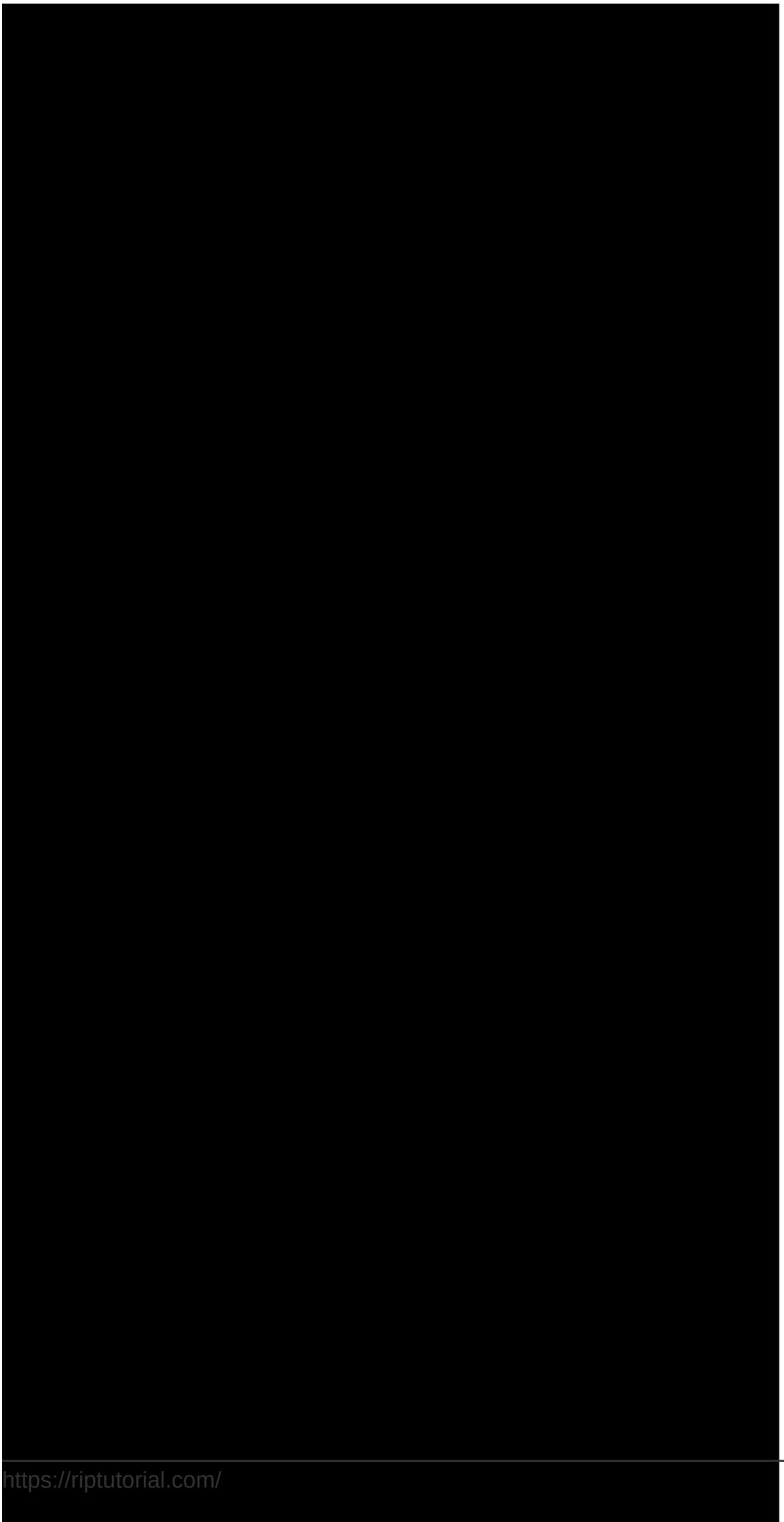

con affetto
As I
4
I heard

Chapter 12: Header and Footer

Examples

Using fancyhdr and titleps packages

```
\documentclass[12pt]{article}

\usepackage{titleps}
\usepackage{fancyhdr}
\usepackage{graphicx}
\usepackage{lipsum} % for dummy text

\pagestyle{myheadings}
\pagestyle{fancy}
\fancyhf{}

\setlength{\headheight}{30pt}

\renewcommand{\headrulewidth}{4pt}
\renewcommand{\footrulewidth}{2pt}

\fancyhead[L]{\includegraphics[width=1cm]{example-image-a}}
\fancyhead[C]{}
\fancyhead[R]{\rightmark}
\fancyfoot[L]{ABC}
\fancyfoot[C]{\textcopyright xyz}
\fancyfoot[R]{\thepage}



\begin{document}

\section{First section}
\subsection{One}
 \lipsum[1-3]
\subsection{Two}
 \lipsum[4-6]

\end{document}
```



A

1 F

1.1

Lorem
vestibul
mauris.
magna.
tique se
Cras vi
ultrices
est, iacu
ces bibe
at, moll
risus. D
eget orc

Nam
auctor l
et, tellu
magna,
Suspend
natoque
Aliquam
cursus l

Null
at, tinci
ummy p
massa a
leo. Ma
cipit a,
lorem.
Integer
in tellus
eu enim

ABC

Chapter 13: Presentation with beamer package

Parameters

theme	AnnArbor
color theme	seahoarse

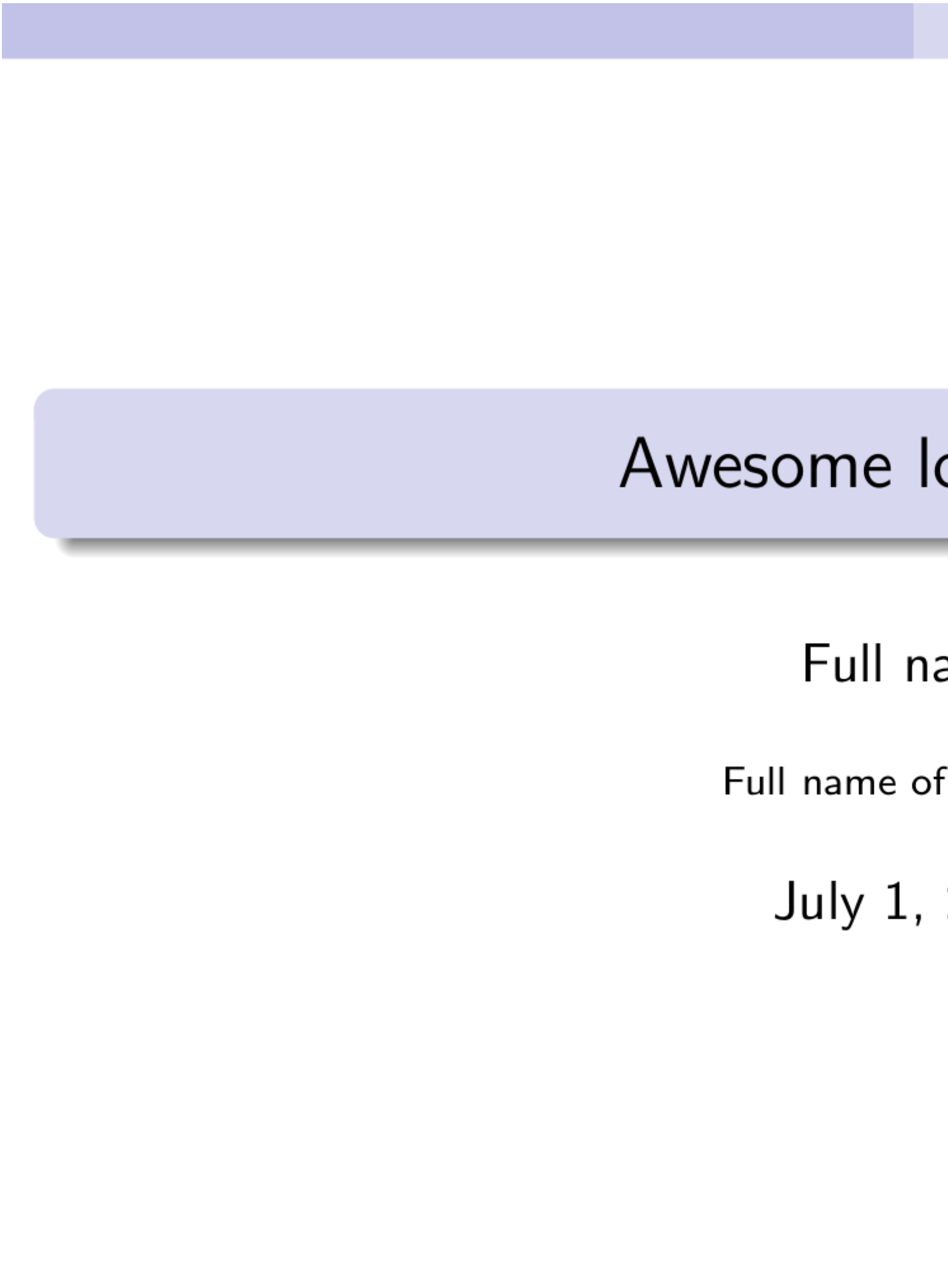
Remarks

For other themes and colors you can visit here

Examples

Simple one author title slide

```
\documentclass{beamer}

\mode<presentation>

\usetheme{AnnArbor}

\usecolortheme{seahorse}

\title[Short topic]{Awesome long topic}

\author[Name]{Full name}

\institute[Institute short form]{Full name of institute}

\date{\today}

\begin{document}

\maketitle

\end{document}
```


Awesome lo
Full na
Full name of
July 1,

https://riptutorial.com/latex/topic/10677/presentation-with-beamer-package

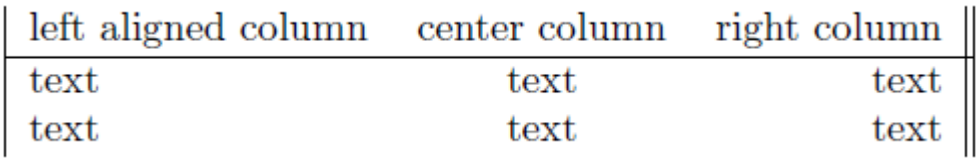
Chapter 14: Tables

Examples

The tabular environment

The `tabular` environment is the most basic way to create a table in LaTeX and doesn't require any other packages.

```
\begin{tabular}{|lcr||}
  left aligned column & center column & right column \\
  \hline
  text & text & text \\
  text & text & text \\
\end{tabular}
```



The parameter (`|lcr||` in the example) is called the **table specification** and tells LaTeX how many columns there are and how they are supposed to be formatted. Each letter represents a single column. Possible values are:

Character	Meaning
l	left aligned column
c	centered column
r	right aligned column
p{'width'} e.g. `p{5cm}`	paragraph column with defined width
| (pipe character)	vertical line
|| (2 pipes)	2 vertical lines

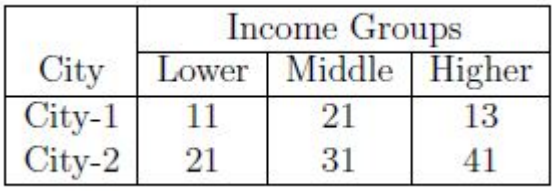
Cells are seperated by the `&` character. A row is ended by 2 back slashes `\\`.

Horizontal lines can be inserted by using the `\hline` command.

Tables are always formatted to be wide enough to include all the content. If a table is to big, LaTeX will print `overfull hbox` warnings. Possible solutions include using the `p{'width'}` specifier or other packages like `tabularx`.

A table with column headings spanning over several columns can be created using the command `\multicolumn{cols}{pos}{text}`.

```
\begin{center}
\begin{tabular}{|c|c|c|c|}
\hline
&\multicolumn{3}{|c|}{Income Groups}\\
\cline{2-4}
City&Lower&Middle&Higher\\
\hline
City-1& 11 & 21 & 13\\
City-2& 21 & 31 &41\\
\hline
\end{tabular}
\end{center}
```



City	Income Groups		
	Lower	Middle	Higher
City-1	11	21	13
City-2	21	31	41

Note that the command `\multicolumn` has three mandatory arguments: the first argument specifies the number of columns over which the heading spans; the second argument specifies the position of the heading `(l,c,r)`; and the third argument is the text for heading. The command `\cline{2-4}` specifies the the starting column(here, 2) and ending column(here, 4) over which a line is to be drawn.

Coloring Table

To make the table more readable, following are the ways to color it:

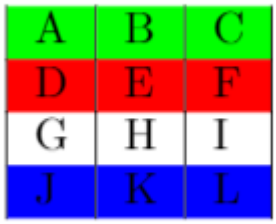
1. Rows
2. Columns
3. Lines
4. Cells

Coloring Rows

Use `\rowcolor` (provided by `colortbl`; also loaded by `xcolor` under the `[table]` package option). Example:

```
\documentclass{article}
\usepackage[table]{xcolor}

\begin{document}

\begin{tabular}{ | l | l | l | }
  \rowcolor{green}
  A & B & C \\
  \rowcolor{red}
  D & E & F \\
```

```
  G & H & I \\
  \rowcolor{blue}
  J & K & L
\end{tabular}

\end{document}
```



Coloring Columns

Columns can be colored using following ways:

- Defining column color property outside the table tag using `\newcolumntype`:

```
\newcolumntype{a}{ >{\columncolor{yellow}} c }
```

- Defining column color property inside the table parameters

```
\begin{tabular}{ | >{\columncolor{red}} c | l | l }
```

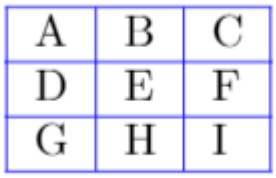
Example:

```
\documentclass{article}
\usepackage[table]{xcolor}

\newcolumntype{a}{>{\columncolor{yellow}}c}
\newcolumntype{b}{>{\columncolor{green}}c}

\begin{document}

\begin{tabular}{ a | >{\columncolor{red}}c | l | b }
  \hline
  A & B & C & D \\
  E & F & G & H \\
  \hline
\end{tabular}

\end{document}
```

Coloring Lines

Use `\arrayrulecolor`. Example:

```
\documentclass{article}
\usepackage[table]{xcolor}

\arrayrulecolor{blue}

\begin{document}

\begin{tabular}{ | l | l | l | }
  \hline
  A & B & C \\
  \hline
  D & E & F\\
  \hline
  G & H & I \\
  \hline
\end{tabular}

\end{document}
```

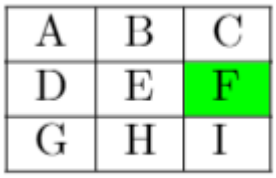


Coloring Cells

Use `\cellcolor`. Example:

```
\documentclass{article}
\usepackage[table]{xcolor}

\begin{document}

\begin{tabular}{ | l | l | l | }
  \hline
  A & B & C \\
  \hline
  D & E & \cellcolor{green}F \\
  \hline
```

```
  G & H & I \\
  \hline
\end{tabular}

\end{document}
```



We can define our own colors too using package `colortbl`. Following are the tags examples:

```
    \definecolor{Gray}{gray}{0.85}
    \columncolor[RGB]{230, 242, 255}}
    \columncolor[HTML]{AAACED}
```

Read Tables online: https://riptutorial.com/latex/topic/4956/tables

Chapter 15: Text Formatting

Examples

Emphazise Text

In order to emphasize text the command `\emph` can be used which usually displays the text in an italics font:

```
This is some text with \emph{emphasized words}.
```

Strike through text

The command `\sout` of the package `ulem` strikes through a text:

```
\sout{This text is striked through}
```

The package `ulem` redefines the command `\emph`. When you do not want to have this behavior you can use the package `ulem` with the option `normalem`:

```
\usepackage[normalem]{ulem}
```

Bold text

In order to typeset text in bold, use `\textbf`:

```
\textbf{This text is typeset in bold.}
```

Read Text Formatting online: https://riptutorial.com/latex/topic/7245/text-formatting

Chapter 16: Title Pages

Remarks

`\title{<title>}`, `\author{<author>}` and `\date{<date}` internally store the content.

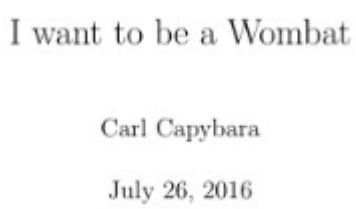
`\maketitle` produces a standard title page with the previously defined values.

Examples

Standard report titlepage

```
\documentclass{report}

\begin{document}

\title{I want to be a Wombat}
\author{Carl Capybara}
\maketitle

\end{document}
```

This will create a title page with no other content:



Read Title Pages online: https://riptutorial.com/latex/topic/3010/title-pages

Chapter 17: Typesetting mathematics

Introduction

One of the biggest advantages of LaTeX is its skill in typesetting equations. Here, the fundamentals of typesetting equations, some of the various packages that can be used, as well as common symbols, are described.

Syntax

- \begin{equation} ... \end{equation}
- text $ ... $ text
- \usepackage{amsmath} ... \begin{equation*} ... \end{equation*}

Remarks

Here are some basic ideas to make sure your code doesn't break on you and your equations look better:

1. Make sure all brackets, curly braces, dollar signs, and `\begin{} \end{}` commands are matching. This is something where one small mistake can mess your whole piece of code up in a big way.
2. If you get errors, make sure you have the proper package loaded (for example, don't use the `\begin{equation*}` command without the `amsmath` package).
3. Never, ever, *ever* use double dollar signs (`$$an equation here$$`) instead of `\begin{equation}`.
4. Never use math mode as a way to make your text italic.
5. Completely stuck? Try TeX.SX, a site for answering questions about TeX, LaTeX, and related languages.

Good luck!

Examples

Basic Equations

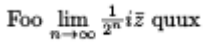
Simple, Inline Equations

You can do a simple inline equation by using `$an equation here$`.

For example, you might do

```
$\lim\limits_{n \to \infty} \frac{1}{2^n} i\bar z$
```

which, if we put a little fake text around it, gives



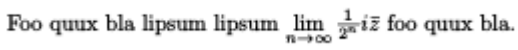
Numbered, Centered Equations

When writing papers or other documents, it is sometimes preferable to have your equations centered and numbered, as opposed to in-line. Then, use the `\begin{equation}` and `\end{equation}` commands.

For example, if we use the code

```
\begin{equation}
\lim\limits_{n \to \infty} \frac{1}{2^n} i\bar z
\end{equation}
```

And add a little text around it, we get

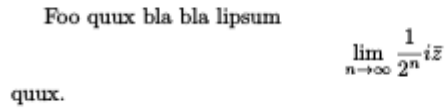


You can remove the numbering of the equation by using `\begin{equation*}` and `\end{equation*}`.

For example, if we use the code

```
\begin{equation*}
\lim\limits_{n \to \infty} \frac{1}{2^n} i\bar z
\end{equation*}
```

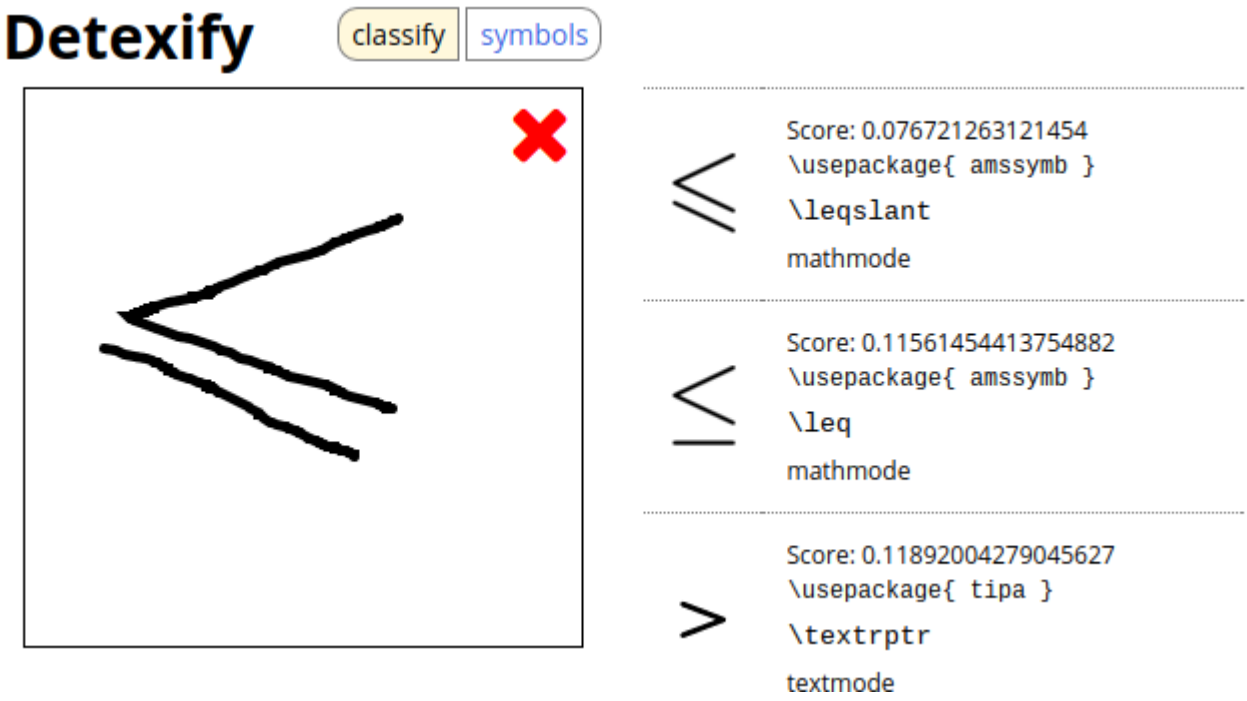
and add a little text around it, we get



(though it should be noted you have to use the `amsmath` package for this).

Finding Symbols

Sometimes, it can be difficult to find the mathematical symbol you need. There are several options here. The first (and quickest) is to use Detexify, where you draw the symbol you'd like, and it tries to find what you want, like as shown below:



Another option is to use the comprehensive LaTeX symbols list, which can be found here. If you are using the package `unicode-math` this list of all supported symbols can be helpful. Another option is this website, which has common math symbols.

Packages available for use

While standard LaTeX is all that is needed for most simple mathematical formulae and equations, sometimes more symbols and tools are needed. There are multiple packages available that will enhance your equations and provide you with more to work with. Three of the main packages are described below. Remember, to load a package, type `\usepackage{package}` in your document preamble.

amsmath

The `amsmath` package is an incredibly useful package. It is used to allow your equations to be centered but not numbered, as in `\begin{equation*}`, it is used to create matrices (as described below) and it introduces many other useful commands, such as `\overset` and `\underset`, described below. The `amsmath` package documentation can be found here.

mathtools

The `mathtools` package builds off of the `amsmath` package, adding further useful symbols and tools. It automatically loads the `amsmath` package, so you do not need to load both in your document preamble. The `mathtools` documentation can be found here.

amssymb

The `amssymb` package provides many extra symbols that can be very handy for more complex equations. The `amssymb` documentation can be found here.

Font packages

There are also various fonts you can use for your equations, as described on this question on the TeX stack exchange, for TeX, LaTeX, and friends.

This paper is a concise explanation of the different features provided by some packages as well as standard LaTeX; it is very helpful.

Good Commands to Know

Some of the most common commands include:

- **Fractions and Square Roots:** For fractions, use `\frac {numerator}{denominator}`. For square roots, use `\sqrt[root]{number}`.
- **Greek letters:** use the commands given in the table below:

αA	`\alpha A`	νN	`\nu N`
βB	`\beta B`	$\xi \Xi$	`\xi\Xi`
$\gamma \Gamma$	`\gamma \Gamma`	$o O$	`o O`
$\delta \Delta$	`\delta \Delta`	$\pi \Pi$	`\pi \Pi`
$\epsilon \varepsilon E$	`\epsilon \varepsilon E`	$\rho \varrho P$	`\rho\varrho P`
ζZ	`\zeta Z`	$\sigma \Sigma$	`\sigma \Sigma`
ηH	`\eta H`	τT	`\tau T`
$\theta \vartheta \Theta$	`\theta \vartheta \Theta`	$\upsilon \Upsilon$	`\upsilon \Upsilon`
ιI	`\iota I`	$\phi \varphi \Phi$	`\phi \varphi \Phi`
κK	`\kappa K`	χX	`\chi X`
$\lambda \Lambda$	`\lambda \Lambda`	$\psi \Psi$	`\psi \Psi`
μM	`\mu M`	$\omega \Omega$	`\omega \Omega`

- **Operators:** `\leq` gives the less than or equal to symbol, `\geq` gives the greater than or equal to symbol, `\neq` gives the not equal symbol, `\sum` gives the summation symbol, `\partial` gives the partial derivative symbol, `\nabla` gives the Laplacian operator, `\times` gives the cross product or multiplication symbol, `\cdot` gives the dot product or multiplication symbol, and `\int` gives the integral symbol.
- **Arrows:** `\rightarrow` and `\leftarrow` give right and left arrows, respectively.
- **Percents:** If typing % in LaTeX, it is important to include a backslash, `\%` as the percent symbol is normally used for comments.
- **Superscripts and Subscripts:** To do a superscript, you can type `x^2`, or, for longer

superscripts, `x^{2x}`. To do a subscript, you can type `x_a`, or, for longer subscripts, `x_{ab}`.

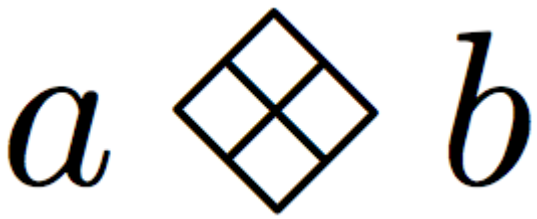
- **Bold:** Use `\boldmath{...}` to make your math symbols bold. Other options are given at this TeX.SX question. Math symbols are automatically italicized; if you don't want this to be true, make your equation text as described below.
- **Infinity:** To write infinity, use the command `\infty`.
- **Moving items over or under another:** First, for math operators only, there is an alternate method. You can type the math operator, say `\int`, and then use the `\limits` command. An example is `\int\limits_{\infty}` or `\int\limits^{\infty}`. Then, for normal cases, you can do `\overset{top}{normal}` or `\underset{bottom}{normal}`. This can be very useful for doing vectors. For example, you might do `\overset{\rightarrow}{x}` The `amsmath` package is need for `overset` and `underset`.
- **Curly Braces:** Because curly braces are used in commands, it is necessary to type `\{` or `\}` to get curly braces.
- **Text:** To include text in equations, type `\usepackage{amsmath}` in the preamble, and then type `\text{...}`.
- **Space:** To add space in your equations, type `\quad` between the two items you want to separate (for example, you might have `$2x \quad cos`).

Creating New Symbols

Let's say you cannot find the symbol you need anywhere. You can create a custom symbol. For example, the code

```
\documentclass{article}
\usepackage{graphicx,amsmath,amssymb}
\DeclareRobustCommand{\diamondtimes}{%
  \mathbin{\text{\rotatebox[origin=c]{45}{$\boxplus$}}}%
}

\begin{document}
$a\diamondtimes b$
\end{document}
```

creates and calls a symbol, giving



This is a simpler example; it merely has to rotate an already existent symbol. However, you can create more complex symbols.

This section is in the process of being expanded.

Matrices

Matrices

You must always use the `amsmath` package if you are going to use the following commands. There are four main types of matrix, as shown in the code below:

```
\begin{matrix}
    a & b \\
    c & d
\end{matrix}
\quad
\begin{pmatrix}
   a & b \\
   c & d
\end{pmatrix}
\quad
\begin{bmatrix}
    a & b \\
    c & d
\end{bmatrix}
\quad
\begin{vmatrix}
    a & b \\
    c & d
\end{vmatrix}
\quad
\begin{Vmatrix}
    a & b \\
    c & d
\end{Vmatrix}
```

This code produces

$$\begin{matrix} a & b \\ c & d \end{matrix} \quad \begin{pmatrix} a & b \\ c & d \end{pmatrix} \quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad \begin{vmatrix} a & b \\ c & d \end{vmatrix} \quad \begin{Vmatrix} a & b \\ c & d \end{Vmatrix}$$

There are a couple important things to note about this:

1. It is important you put your matrix within the `equation`, `equation*`, or `$...$` environment - the `bmatrix` command is not a math environment on its own.
2. The construction of the matrix is actually fairly simple. For each row, you create each element (say `x_{11}`), then put a `&`, and then write the next element. For multiple rows, at the end of each row put `\\` (you do not have to do this for the last row). It is fairly similar to a table in this.

Read Typesetting mathematics online: https://riptutorial.com/latex/topic/5950/typesetting-mathematics

Credits

S. No	Chapters	Contributors
1	Getting started with latex	Community, eyqs, Harry, hbaderts, jani, Louis, Nijin22, Sean Allred, SnoringFrog, Spacedman, Sumner Evans, TuringTux, tversteeg
2	Accessing documentation of LaTeX packages	Raphael, samcarter
3	Add Citation	Hamzawey
4	Build Tools	Sean Allred
5	Counters, if statements and loops with latex	csekri
6	Creating a Bibliography	Nijin22, Raphael, Werner
7	Creating posters using beamer	Sukanya B
8	Defining macros	celtschk, strpeter
9	Document classes	Community, heather
10	Drawing graphs	L.V.Rao, Raphael
11	Engraving Sheet Music	Sam Whited
12	Header and Footer	Sukanya B
13	Presentation with beamer package	Sukanya B
14	Tables	L.V.Rao, Nijin22, Nikita Jain, Werner
15	Text Formatting	celtschk, Stephan Kulla
16	Title Pages	adn, Johannes_B, Nijin22, Sam Whited
17	Typesetting	heather

	mathematics	