LaTeX Learning

Getting started with LaTeX

Soroush Omidvartehrani

Ferdowsi University of Mashhad

April 2022

Outline:

1 Introduction

2 Document Structure

3 Page Layout and Style

4 Including Images

5 Creating Tables

6 Equations

7 Reference

1 Introduction
- What is LaTeX?
- A LaTeX Editor
- Why LaTeX?
- TeX vs. LaTeX
- Engines and Compilers
- TeX Distributions
- LaTeX Commands
- Hello World!

2 Document Structure

3 Page Layout and Style

4 Including Images

5 Creating Tables

6 Equations

7 Reference

LaTeX

- Is a document preparation system
- Pronounced as 'lay-tech' or 'lah-tech'
- Often stylized as LaTeX
- Includes features designed for the production of technical and scientific documentation
- Is the de facto standard for the communication and publication of scientific documents
- Uses plain text (as opposed to the formatted text found in "What You See Is What You Get" word processors)

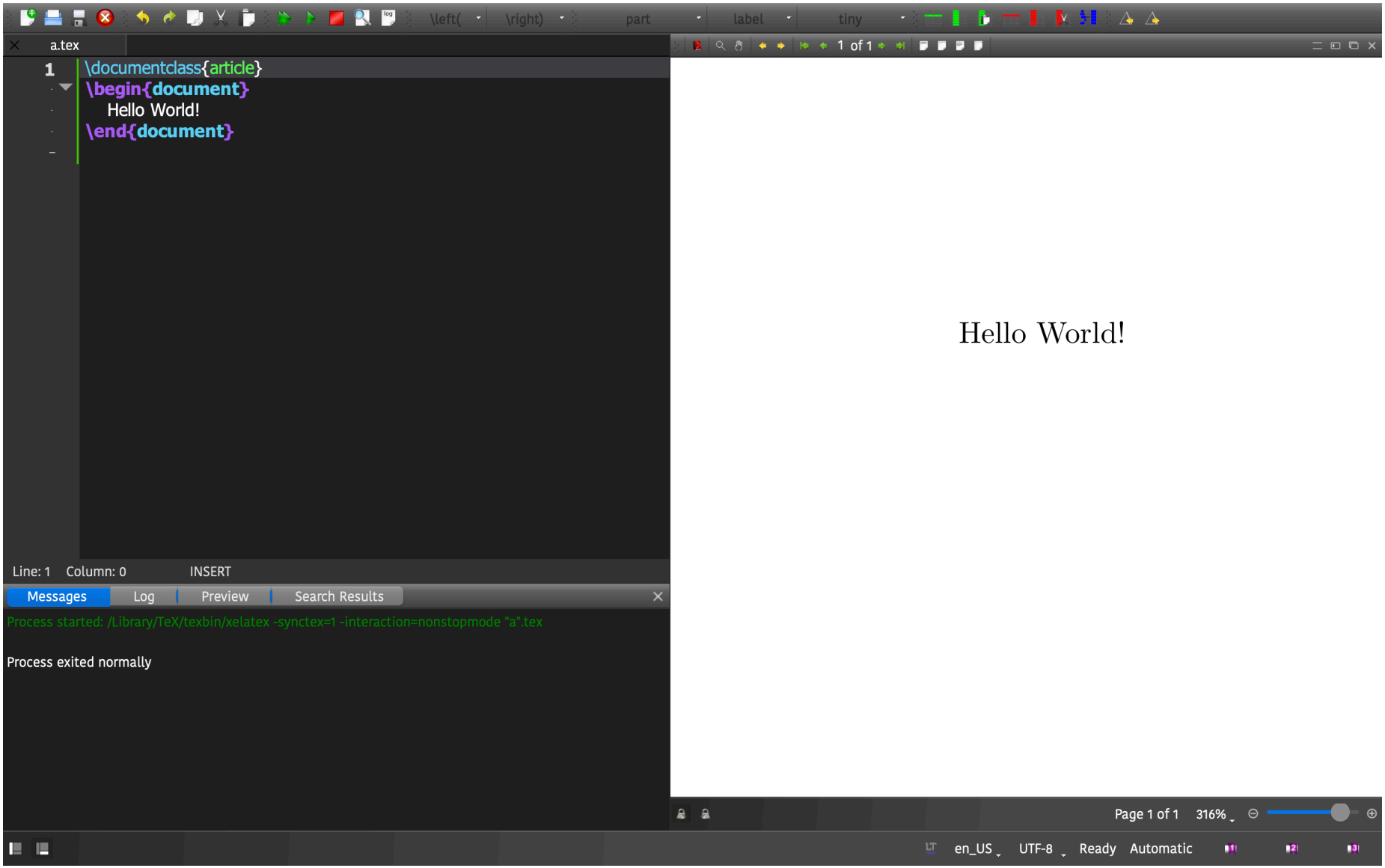


Figure 1: A LaTeX editor (TeXstudio)

"Focus on content, NOT on formatting"

- Free
- High quality ducuments
- Consistent fonts, tables, figures etc.
- Generating indexes, footnotes and references easily.

TeX

- Low-level markup and programming language
- By Donald Knuth (between 1977 and 1989)
- Stable and powerful but time-consuming and difficult to learn

LaTeX

- Based on TeX
- By Leslie Lamport (still actively maintained)
- Easier to code

- Leslie Lamport created LaTeX in the early 1980s to provide a higher level language to work in than TeX.
- LaTeX is a set of commands defined in terms of the underlying TeX commands, often at many many layers of abstraction.
- All of the commands you use in a LaTeX document are actually just complicated sets of TeX commands underneath, unless of course you use a TeX command directly!
- Concepts like packages[1], environments[2], and document classes[3] were all introduced by Leslie Lamport in LaTeX.

[1] `\usepackage{<package name>}`
[2] `\begin{<environment>}` ... `\end{<environment>}`
[3] `\documentclass{<class name>}`

- TeX-based programs (often referred to as TeX engines) receive their name by adding a prefix to the word "TeX", such as pdfTeX, XeTeX and LuaTeX.
- These programs are derived from Knuth's original TeX software they contain features and functionality which aren't available in Knuth's original version.
- XeLaTeX, pdfLaTeX or LuaLaTeX are not actually the names of TeX engines, all they signify is which TeX engine is being used to run the LaTeX macro collection:
  - pdfLaTeX means using the LaTeX macro package with the pdfTeX engine
  - XeLaTeX means using the LaTeX macro package with the XeTeX engine (for multilingual work)
  - LuaLaTeX means using the LaTeX macro package with the LuaTeX engine (perhaps for advanced customized document production)

- Choosing a compiler depends on each project's needs.
  - LaTeX supports only `.eps` and `.ps` image formats for use with `\includegraphics`.
  - pdfLaTeX supports `.png`, `.jpg`, `.pdf` image formats. It will convert `.eps` images to `.pdf` on-the-fly during compilation.
  - XeLaTeX and LuaLaTeX both supports `UTF-8` robustly out of the box, as well as Truetype and OpenType.

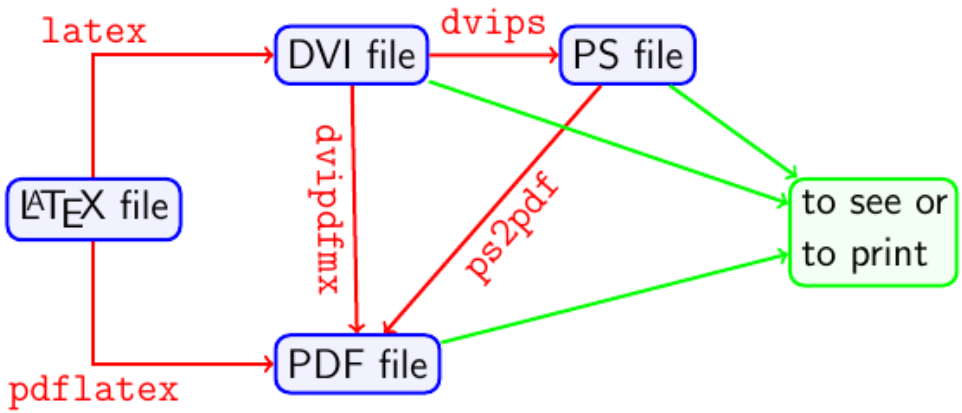


Figure 2: LaTeX compilation file flow

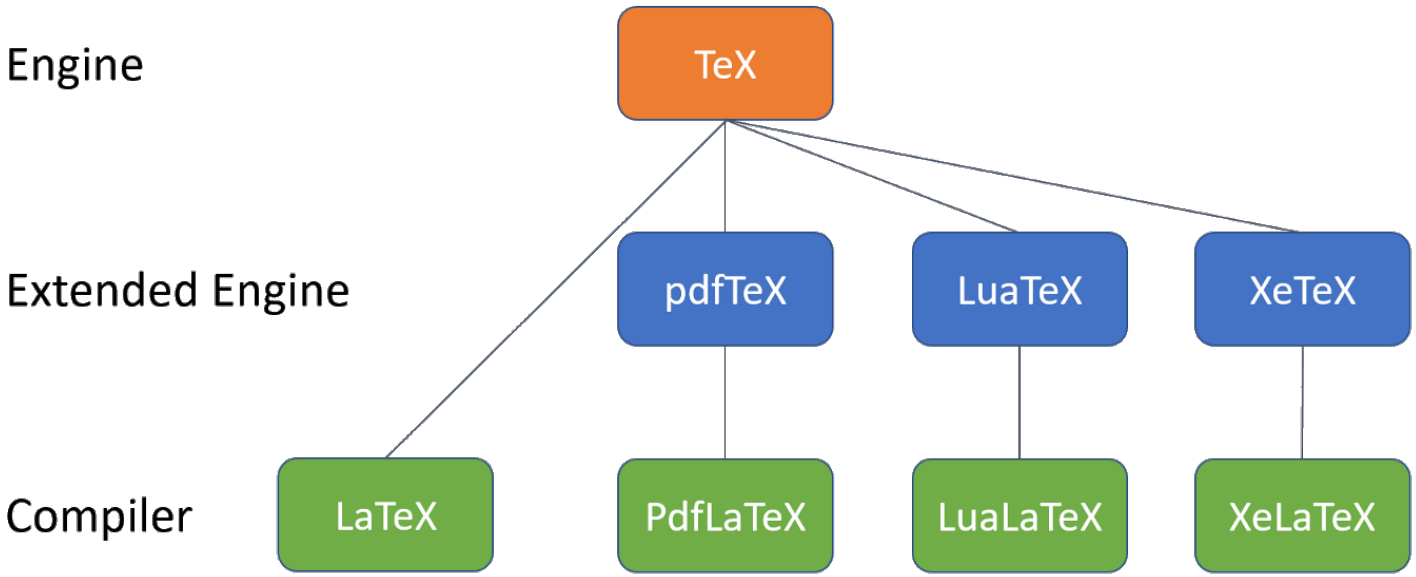


Figure 3: TeX family

- The set of programs that make it possible to compile TeX and LaTeX documents is called a TeX typesetting or a TeX distribution.
- A TeX distribution contains the latest stable releases of TeX engines, LaTeX packages, TeX-related tools, etc.
- There are many TeX distributions available for different operating systems:
  - TeX Live for Linux and other UNIX-like systems
  - MacTeX redistribution of TeX Live for macOS
  - MiKTeX for Windows
  - teTeX for Linux and other UNIX-like systems; it is no longer actively maintained now
  - proTeXt is based on MiKTeX

- Commands are special words that determine LaTeX behaviour.
- They begin with a backslash, followed by (big or small) letters, and are usually named descriptively.
- Commands can have parameters given in curly braces (mandatory: at least one has to be provided) or square brackets (optional: default values can be used).
- Calling a command can look like these:

Code:

```
\command
\command{argument}
\command[optional_arguments]{argument}
```

Code:

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

Output:

Hello World!

Let's try!
(hello_world.tex)

You can access to all the codes in the presentation from attached files.

Assignment:

Create your first document.
What keywords have you used? Is there anything in the output that you have not explicitly written?

1 Introduction

2 Document Structure
- documentclass Command
- Top Matter
- Packages
- Sectioning Commands

3 Page Layout and Style

4 Including Images

5 Creating Tables

6 Equations

7 Reference

Code:

```
\documentclass{<class name>}
```

Document Classes:

- article: For articles in scientific journals, presentations, short reports, program documentation, invitations, ...
- report: For longer reports containing several chapters, small books, thesis, ...
- book: For books.
- letter: For writing letters.
- slides: For slides. The class uses big sans serif letters.

- beamer: For writing presentations (more powerful than slides).
- IEEEtran: For articles with the IEEE Transactions format.
- scrlttr2: As a replacement for the standard letter class.

Code:

```
\documentclass[<option 1>, <option 2>, etc.]{<class name>}
```

Options:

- Font size (10pt: default, 11pt, 12pt)
- Paper size (a4paper: default, letterpaper, a5paper, etc.)
- Multiple columns (onecolumn: default, twocolumn)
- Landscape print mode (landscape)
- Single- and double-sided documents (oneside, twoside)
- Draft mode (final: default, draft: figures are not loaded and indicated by a frame)

- Formula-specific options
  - fleqn: left-alignment of formulas
  - leqno: labels formulas on the left-hand side instead of right

$$f(x) = ax + b \qquad (1)$$

(1) $f(x) = ax + b$

- Titlepage behavior (notitlepage: default for article, titlepage: default for report and book)
- Chapter opening page (openany: default for report, openright: default for book)

- You can put the title, author name, and date in curly braces after `\title`, `\author`, and `\date` commands.
- If you omit the `\date` command, LaTeX uses today's date by default.
- You should finish the top matter with the `\maketitle` command, which tells LaTeX to print the title, author, and date in a nicely formatted manner.

Code:

```
\documentclass[12pt,a4paper]{report}
\begin{document}
  \title{LaTeX Learning}
  \author{Soroush Omidvar}
  \date{Nov 2020}
  \maketitle
\end{document}
```

- It's similar to adding a library to a Java program so that your code can utilize functions predefined by yourself or other people.

Code:

```
\usepackage[<option 1>, <option 2>, etc.]{<package name>}
```

- Some important packages are:

Code:

```
\usepackage{amsmath}
% for mathematical symbols

\usepackage{graphicx}
% to manage external pictures

\usepackage{geometry}
% for management of document margins
```

- Do not need to specify section numbers.
- Do not need to use `\begin{}` and `\end{}` commands.
- Certain commands are appropriate to different document classes. (like: `\chapter{}`, which is only available in books and reports)

Code:

```
\chapter{chapter_name}
% chapter's content...

\section{section_name}
% section's content...

\subsection{subsection_name}
% subsection's content...

\subsubsection{subsubsection_name}
%subsubsection's content...
```

- All auto-numbered headings get entered in the Table of Contents (ToC) automatically, by using the `\tableofcontents` command. (we will discuss it in detail in section 7)
- To get an unnumbered section heading, follow the command name with *.
- The `\tableofcontents` command normally shows only numbered section headings, and only down to the level defined by the tocdepth counter.

Code:

```
\section*{sec_name}
```

- in book structure **\appendix** can be used to indicate that following sections or chapters are to be numbered as appendices. Use it once for all appendices.

Code:

```
\appendix
\section{appen_name}
```

Let's try!
(structure.tex)

Assignment:

Create a Persian book document with multiple chapters, sections, subsections, subsubsections, and an appendix. Generate the ToC, and try to change the depth of that.

1 Introduction

2 Document Structure

3 Page Layout and Style
- Line Breaking
- Page Breaking
- Blank Spaces
- Multi-column Pages
- Orientation
- Font Size
- Font Styles
- Font Families
- Alternative Fonts
- List Structures
- Margins
- Paragraph Alignment
- Paragraph Indent
- Special Paragraphs
- Colors

4 Including Images

5 Creating Tables

6 Equations

7 Reference

- The most standard way how to break lines is to create a new paragraph. This is done by leaving an empty line in the code.
- There are other commands (with the same effect) that break the line, which are different from a paragraph break as we are still using the same paragraph:
  - `\\` (two backslashes)
  - `\newline`
- Another command, called `\linebreak`, tells LaTeX to end the line but keep the full justification. By this command, the space between the words would be stretched to reach the right margin, which causes unpleasant gaps.
  - `\linebreak` can have an optional argument: `\linebreak[number]`
  - The optional argument, a number, converts the `\linebreak` command from a demand to a request. The number must range from 0 to 4. The higher the number, the stronger the request. Number 4 is the default behaviour if no value is given.

- LaTeX itself take care of the page breaking.
- It also offers several commands to insert a page break: `\pagebreak`, `\newpage`, `\clearpage`, etc.
- The `\pagebreak[number]` command tells LaTeXto break the current page at the point of the command.
  - The text has been stretched to fill the page down to the bottom, which is desirable for having the same text height on all pages.
  - This command can take an optional argument that behaves like `\linebreak[number]` (for pages instead of lines).
- The `\newpage` command ends the current page.
  - It doesn't stretch the text.
- If you use the two-column format, both `\pagebreak` and `\newpage` will begin on a new column instead of a new page.
- `\clearpage` works like `\newpage`, except that it will start on a new page, even in two-column mode.

- Horizontal spaces of arbitrary length may be inserted with `\hspace`.
- `\hfill` inserts a blank space that will stretch accordingly to fill the space available.
- The commands `\hrulefill` and `\dotfill` do the same as `\hfill` but instead of blank spaces they insert a horizontal ruler and a string of dots, respectively.
- Vertical blank spaces have the same syntax as horizontal ones.
- `\vspace` inserts a vertical spaces.
- `\vfill` inserts a blank space that will stretch accordingly to fill the vertical space available.

Code:

```
Horizontal \hspace{1cm} spaces can be inserted manually.

Left Side \hfill Right Side.

\vspace{1cm}
After 1 cm vertical space. \dotfill
\vfill
Text at the bottom of the area. \hrulefill
```

Output:

Horizontal spaces can be inserted manually.

Left Side Right Side.

After 1 cm vertical space. .....

Text at the bottom of the area. _____

- You can simply pass the optional argument twocolumn to the document class, which will give the desired effect, but it has limitations.
- Using **multicol** package provides the following advantages:
  - Can support up to ten columns.
  - Implements a multicols environment, therefore, it is possible to mix the number of columns within a document.
  - Column environments can be easily customised locally or globally.

Code:

```
\begin{multicols}{2}
% your  text
\end{multicols}
```

- In a portrait document, some contents, like a large diagram or table that would be displayed better on a landscape page.

Code:

```
\usepackage{lscape}
% use pdflscape insttead of lscape, if you are using pdfLaTeX

\begin{landscape}
% your landscape contents
\end{landscape}
```

- **lscape** is more applicable to books or reports than to typical academic publications.
- Using **pdflscape** will rotate the page inside the PDF too, so it looks better.

- To scale text relative to the default body text size (built-in sizes), use the following commands, which change the size within a given scope.

Code and Output:

Command	Output	Command	Output
`\tiny`	sample text	`\large`	sample text
`\scriptsize`	sample text	`\Large`	sample text
`\footnotesize`	sample text	`\LARGE`	sample text
`\small`	sample text	`\huge`	sample text
`\normalsize`	sample text	`\Huge`	sample text

- To make a text bold use `\textbf` command.
- Use the `\underline` command for Underlining text.
- To make a text italic use the `\emph` or `\textit` command.
  - What the `\emph` command actually does with its argument depends on the context. inside normal text the emphasized text is italicized, but this behaviour is reversed inside an italicized text.

Code:

```
Turing's housekeeper found him \textbf{dead}; \textit{he had died the
    previous day.} \underline{Cyanide poisoning} was established as
   the cause of death.
```

Output:

Turing's housekeeper found him **dead**; *he had died the previous day.* Cyanide poisoning was established as the cause of death.

- The most common font styles in LaTeX are bold, italics and underlined, but there are a few more.

Code and Output:

Style	Command	Output
medium	`\textmd{Sample Text 0123}`	Sample Text 0123
slanted	`\textsl{Sample Text 0123}`	*Sample Text 0123*
uppercase	`\uppercase{Sample Text 0123}`	SAMPLE TEXT 0123
lowercase	`\lowercase{Sample Text 0123}`	sample text 0123
small caps	`\textsc{Sample Text 0123}`	Sample Text 0123
upright	`\textup{Sample Text 0123}`	Sample Text 0123

- `ulem` package has six more different ways to differentiate a piece of text with strikes and lines.
- This package also solves the problem of `\underline` command that can not break a long piece of text properly to the next line.

Code:

```
\usepackage{ulem}
\begin{document}
  \uline{Underlined}
  \uuline{Double-Underlined}
  \uwave{Wavy-Underlined}
  \sout{Strikethrough}
  \xout{Struck with Hatching}
  \dashuline{Dashed Underline}
  \dotuline{Dotted Underline}
\end{document}
```

Output:

Underlined
Double-Underlined
Wavy-Underlined
~~Strikethrough~~
Struck with Hatching
Dashed Underline
Dotted Underline

- Font families (collections of fonts that share similar design elements) can generally be grouped into three main categories:
  1. Serif (default)
     - They are determined by serifs (small line or taper regularly added to the end of a character's stem)
     - Serif fonts are typically used in printed books, newspapers, and magazines.
     - Examples: Times New Roman, Georgia, etc.
  2. Sans serif
     - A sans serif font doesn't have any serif in its characters.
     - Sans serif font will typically be the default font in digital word processing programs.
     - Examples: Arial, Helvetica, etc.
  3. Monospaced
     - A monospaced font has the same fixed-width characters.
     - They are mainly used for source code listings or well-aligned content.
     - Examples: Monaco, Everson Mono, etc.

Figure 4: Font family classification

- LaTeX typesets documents using the Computer Modern typeface family.
- Computer Modern contains serif, sans serif, and monospaced fonts in several weights and optical sizes.
- LaTeX commands generally refer to serif, sans serif, and monospaced font families with the shorthand `rm`, `sf`, and `tt`, respectively.
- Document font family can be changed by setting the family default.

Code:

```
\renewcommand{\familydefault}{<family>}
% <family> options are: \rmdefault, \sfdefault, and \ttdefault.
```

- To set a font family only for a part of our document, we can use different commands and switches.
  - `\textrm` command for serif family (switch: `\rmfamily`)
  - `\textsf` command for sans serif family (switch: `\sffamily`)
  - `\texttt` command for monospaced family (switch: `\ttfamily`)

Code:

```
\textrm{Serif Font Family}
\textsf{Sans Serif Font Family}
\texttt{Monospaced Font Family}
```

Output:

Serif Font Family
Sans Serif Font Family
Monospaced Font Family

Code:

```
\rmfamily
Serif Font Family
\sffamily
Sans Serif Font Family
\ttfamily
Monospaced Font Family
```

Output:

Serif Font Family
Sans Serif Font Family
Monospaced Font Family

- Vector-based fonts didn't exist when TeX was initially designed.
- These days, LaTeX engines also support the TrueType (TTF) and OpenType (OTF) fonts.
- If you are using lualatex or xelatex, you can use TTF and OTF fonts with the **fontspec** package.

Code:

```
\usepackage{fontspec}
\setmainfont{Times New Roman} % to set document font
\setromanfont{Times New Roman} % to set a sans font
\setsansfont{Arial} % to set a sans serif font
\setmonofont{Courier New} % to set a monospaced font
```

- List structures in LaTeX are simply environments which essentially come in three types:
  - **itemize** for a bullet list
  - **enumerate** for an enumerated list
  - **description** for a descriptive list
- All lists follow the basic format:

Code:

```
\begin{list_type}
  \item first item
  \item second item
  \item third item % ...
\end{list_type}
```

- **itemize**:

Code:

```
\begin{itemize}
  \item first item
  \item second item
  \item third item
\end{itemize}
```

Output:

- first item
- second item
- third item

- **enumerate**:

Code:

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

Output:

1. first item
2. second item
3. third item

- Unlike **itemize** and **enumerate**, a descriptive list, or **description**, does not have a label.
- A word or phrase is used, which is passed to item as an optional argument
- **description**:

Code:

```
\begin{description}
  \item [Sadra] detail
  \item [Amir] detail
  \item [Kourosh] detail
\end{description}
```

Output:

Sadra detail

Amir detail

Kourosh detail

- You can insert a list inside another list.
- The above lists may be included within one another, either mixed or of one type.

Code:

```
\begin{enumerate}
  \item first enumerate item
  \begin{itemize}
    \item first itemize item
    \begin{description}
      \item [Name]  first description item
    \end{description}
  \end{itemize}
  \item second enumerate item
\end{enumerate}
```

Output:

1. first enumerate item
   - first itemize item

     Name first description item
2. second enumerate item

- Margins improve readability.
- A more modern and flexible approach is to use the **geometry** package.
- It allows you to specify the 4 margins without needing to remember the particular page dimensions.
- You can also change the paper size and orientation using the geometry package (it will inherit them from the document class if you already specified).
- The geometry package understands "key=value" options separated by commas.

Code:

```
\usepackage[paper=a4paper, top=1in, bottom=1.5in, left=1.5cm, right
    =15mm]{geometry}
```

- A page layout in the geometry package contains the following definitions:
  - **paper**: **total body** and **margins**
  - **total body**: **body** (text area) (optional **head**, **foot** and **marginpar**)
  - **margins**: **left** (**inner**), **right** (**outer**), **top** and **bottom**

- This package also has an auto-completion mechanism, in which unspecified dimensions are automatically determined using default margin ratios:
  - top:bottom = 2:3
  - left:right = 1:1 (for one-sided documents)
  - inner:outer = 2:3 (for two-sided documents)

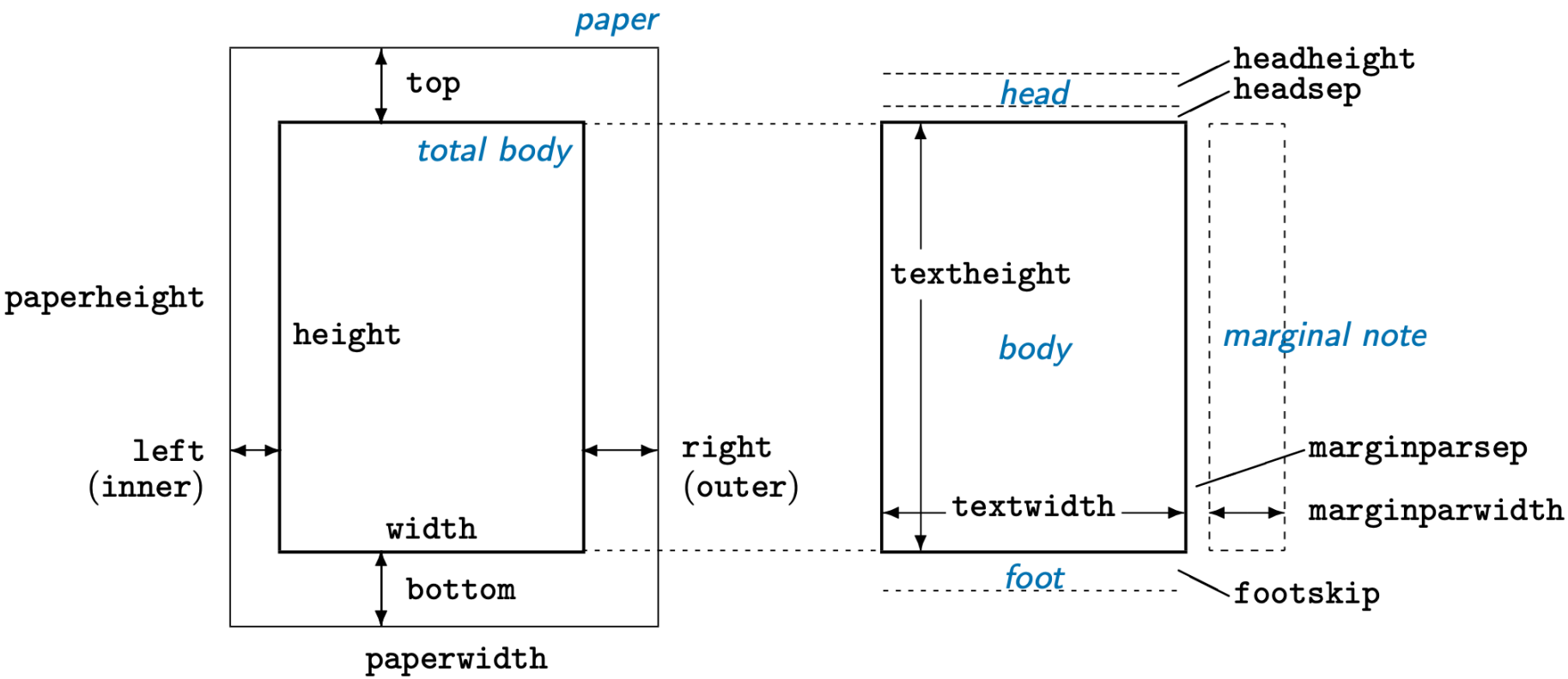


Figure 5: Dimension names in the geometry package

Some useful options that can be used to set the margins are as follow.

- **left** and **right**: width of the left and the right margin (for one-sided documents), like left=3cm.
- **inner** and **outer**: width of the inner and the outer margin (for two-sided documents), such as inner=1in.
- **top** and **bottom**: height of the top and the bottom margin, like top=22mm.
- **twoside**: the left and right margins will be swapped on left-hand pages.
- **bindingoffset**: reserve width to compensate for the part of the inner margin that's hidden in the binding.

Code:

```
\usepackage{geometry}
\geometry{inner=2cm, outer=3cm, bindingoffset=1cm, twoside}
```

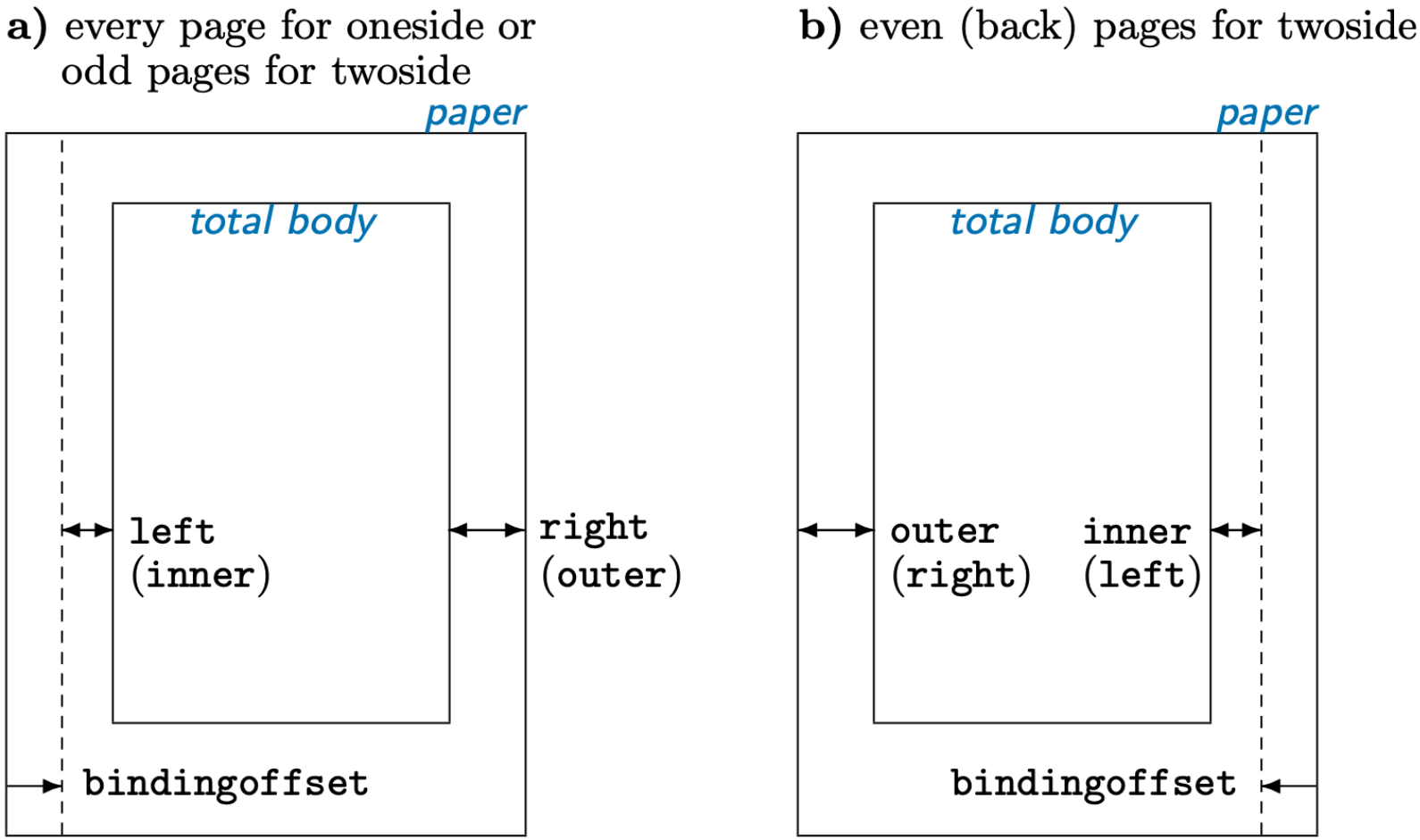


Figure 6: bindingoffset option

- Paragraphs in LaTeX are usually fully justified.
- For whatever reason, should you wish to alter the justification of a paragraph, there are three environments at hand, and also LaTeX command equivalents.

Code:

Alignment	Environment	Command
Left justified	flushleft	`\raggedright`
Right justified	flushright	`\raggedleft`
Center	center	`\centering`

- All text between the `\begin` and `\end` of the specified environment will be justified appropriately.

- By default, the first paragraph after a heading follows no indentation.
- The size of subsequent paragraph indents is determined by a parameter called **\parindent**.
- Also, **\parskip** determines space between paragraph and preceeding text
- It is possible to override **\parindent** and **\parskip** by using the **\setlength** command.

Code:

```
\setlength{\parindent}{1cm} % Default is 15pt.
```

- If you want to indent a paragraph that is not indented, you can use **\indent**
- To create a non-indented paragraph, you can use **\noindent**

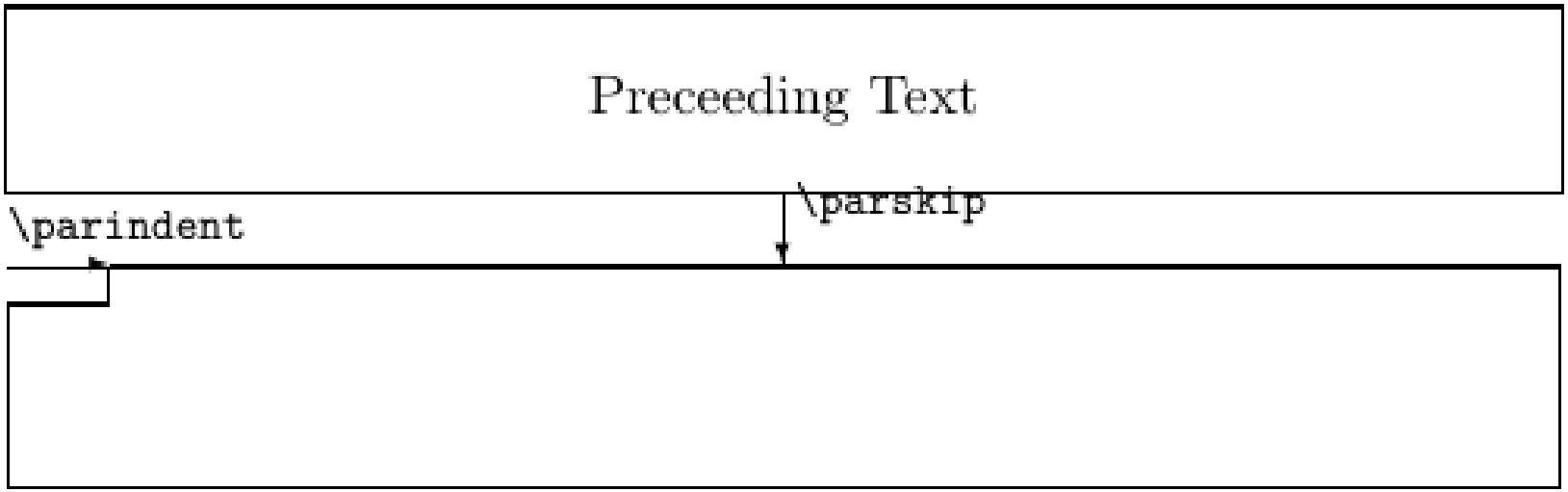


Figure 7: Schematic presentation of lengths in a paragraph

- If you use the verbatim environment, everything input between the begin and end commands are processed as if by a typewriter.
- All spaces and new lines are reproduced as given, and the text is displayed in an appropriate fixed-width font.
- Any LaTeX command will be ignored and handled as plain text. This is ideal for typesetting program source code.

Code:

```
\begin{verbatim}
  The verbatim environment
  simply reproduces every
  character you input,
  including all  s p a c e s!
\end{verbatim}
```

Output:

```
The verbatim environment
simply reproduces every
character you input,
including all  s p a c e s!
```

- If you just want to introduce a short verbatim phrase, you don't need to use the whole environment, but you have the `\verb` command.
- The first character following `\verb` is the delimiter: here we have used "+", but you can use any character you like except "*".

Code:

```
\verb+my text+
```

- Adding colors to your text is supported by the **xcolor** package.
- you can set the font color, text background, page background, or Mathematical formula.
- Two ways to type colored text is by `\textcolor{color}{text}` and `{\color{color} text}`
- The `\color` environment allows the text to run over multiple lines and other text environments whereas the text in `\textcolor` must all be one paragraph.

Code:

```
some black text, \textcolor{red}{followed by a red fragment}, going
    black again.
```

Output:

some black text, followed by a red fragment, going black again.

- You can choose from predefined colors (black, blue, brown, cyan, darkgray, gray, green, lightgray, lime, magenta, olive, orange, pink, purple, red, teal, violet, white, yellow) or define your own colors using RGB, Hex, or cmyk.
- the colors are defined by:
  `\definecolor{name}{model}{color-spec}`
  - name is the name of the color; you can call it as you like
  - model is the way you describe the color, and is one of gray, rgb, RGB, HTML, and cmyk.
  - color-spec is the description of the color.

Code:

```
\definecolor{cl0}{gray}{0.95} % from 0 (black) to 1 (white)
\definecolor{cl1}{rgb}{1,0.5,0} % red, green, blue
\definecolor{cl2}{RGB}{255,127,0} % numbers are between 0 and 255
\definecolor{cl3}{HTML}{FF7F00} % similar to HTML (RRGGBB)
\definecolor{cl4}{cmyk}{0,0.5,1,0} % cyan, magenta, yellow, black
```

- It is possible to directly use a color without naming it first:

Code:

```
{\color[rgb]{1,0,0} text}
\textcolor[rgb]{0,1,0}{text}
```

Output:

{\color[rgb]{1,0,0} text}
\textcolor[rgb]{0,1,0}{text}

- You can change the background color of the whole page by `\pagecolor{color}`
- Changing background color for the text is possible by `\colorbox{color}{text}`

Code:

```
\colorbox{yellow}{hahaha!}
```

Output:

\colorbox{yellow}{hahaha!}

Let's try!
(layout.tex)

Assignment:

Typeset a document that has the following properties:

- Different margins for odd and even pages
- Different font families
- A two-column format on a horizontal page
- A dotted underline paragraph
- Nested ordered and unordered lists
- A left-justified paragraph including a LaTeX command
- A paragraph with blue texts and yellow background

1 Introduction

2 Document Structure

3 Page Layout and Style

4 Including Images
- Floats
- figure Environment
- Image Positioning
- Multiple Images (Subfigures)

5 Creating Tables

6 Equations

7 Reference

- Floats are containers for things in a document that cannot be broken over a page.
- LaTeX by default recognizes table and figure floats.
- LaTeX automatically floats tables and figures, depending on how much space is left on the page.
- The figure environment displays pictures as floating elements within the document.
- If you include the picture inside the figure environment, LaTeX will position it appropriately.
- You can have more control over how the figures are displayed by the additional parameter(s) that can be passed to determine the figure positioning.

- Including images can be possible by using the standard package called `graphicx`.
- The figure environment takes care of the numbering and positioning of the images.
- To include a figure, you must use the `\includegraphics` command, which takes the path to your image file, and also image width as an optional parameter.
- If an image is stored in the same directory as the .tex file, you can use the image name, like `Alan.jpeg`, to include it, and if you keep image files in a different folder, like a folder named graphics, then just write `graphics/Alan.jpeg` into the braces.
- Another useful command in the figure environment is `\caption`, which produces the text shown below the image.

Code:

```
\usepackage{graphicx}

\begin{document}
  \begin{figure}
    \includegraphics[width=0.5\
        linewidth]{Turing.jpeg}
    \caption{Alan Turing}
  \end{figure}
\end{document}
```

Output:

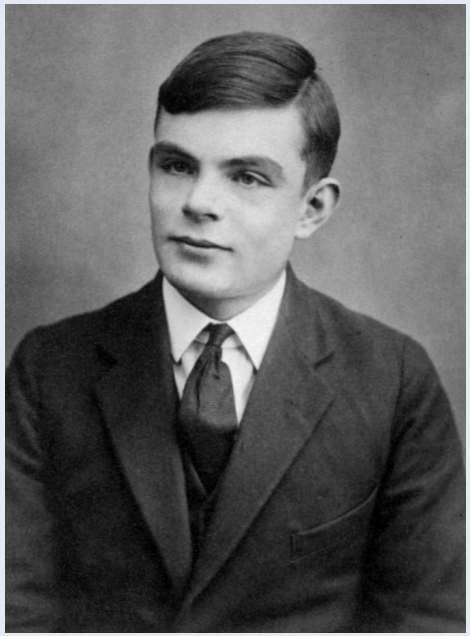


Figure 8: Alan Turing

- The figure doesn't necessarily show up in the exact place as you put your code in the .tex file.
- To change the positioning of an image, you should treat it as an object inside the LaTeX document.
- The placement specifier, as an optional parameter of the figure environment, allows us to have more control over where a figure is placed.

Code:

```
\begin{figure}[<placement specifier>]
  % figure contents
\end{figure}
```

- Placement Specifiers:
  - **h**: Place the float here, approximately (not exactly) at the same point it occurs in the source text.
  - **t**: Position at the top of the page.
  - **b**: Position at the bottom of the page.
  - **p**: Put on a special page for floats only.
  - **!**: will force the specified location.
  - **H**: Places the float at precisely the location in the LaTeX code, which requires the float package (`\usepackage{float}`).
- You can use multiple placements, like `[!htbp]`, allowing the figure to go to different positions.

Code:

```
\begin{figure}[htp]
```

- We usually need to center figures. This can be achieved by using `\centering` inside the figure environment.

- In some situations, adding a single image is not optimal, especially when the reader is supposed to compare several results.
- One way to overcome this condition is to use a different environment, called subfigure.
- The subfigure environment (inside a figure environment) allows you to place multiple images at a certain location next to each other.
- You should load up the subcaption package to use the subfigure environment.

Code:

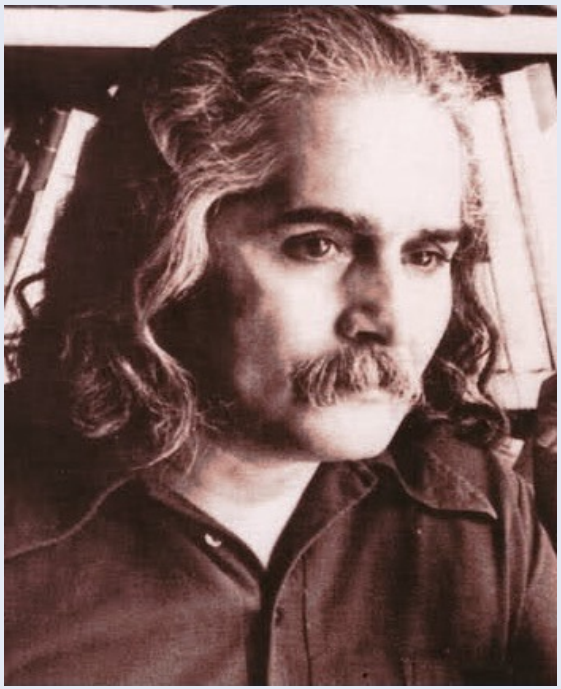
```
\begin{subfigure}[pos][height][inner-pos]{width}
  % subfigure contents
\end{subfigure}
```

- The subfigure options (**pos**, **height**, and **inner-pos**) are as follows.
  - **pos** governs how the subfigure vertically aligns with the surrounding text. It can be **c** (default, means vertical centre), **t** (matches the top line of the subfigure with the baseline of the surrounding text), and **b** (matches the bottom line of the subfigure with the baseline)
  - **height** stretches the height of the subfigure.
  - **inner-pos** controls the placement of contents inside the subfigure environment. It can be **t** (places the contents at the top of the box), **c** (default, places the contents in the vertical centre), **b** (places the contents at the bottom), and **s** (stretches the contents out vertically).
  - inner-pos only makes sense if the default height of the environment is modified (to have a place for changing the position of contents).

Code:

```
\begin{figure}
  \centering
  \begin{subfigure}[b]{0.3\textwidth}
    \centering
    \includegraphics[width=\textwidth]{Akhavan}
    \caption{Mehdi Akhavan}
  \end{subfigure}
  \hfill
  \begin{subfigure}[b]{0.3\textwidth}
    \centering
    \includegraphics[width=\textwidth]{Hedayat}
    \caption{Sadegh Hedayat}
  \end{subfigure}
  \hfill
  \begin{subfigure}[b]{0.3\textwidth}
    \centering
    \includegraphics[width=\textwidth]{Irajmirza}
    \caption{Iraj Mirza}
  \end{subfigure}
  \caption{Iranian poets and writers}
\end{figure}
```

Output:



(a) Mehdi Akhavan

(b) Sadegh Hedayat

(c) Iraj Mirza

Figure 9: Iranian poets and writers

Code:

```
\begin{figure}
  \centering
  \begin{subfigure}[b]{0.2\textwidth}
    \includegraphics[width=\textwidth]{Kierkegaard}
    \caption{Kierkegaard}
  \end{subfigure}
  \begin{subfigure}[b]{0.2\textwidth}
    \includegraphics[width=\textwidth]{Nietzsche}
    \caption{Nietzsche}
  \end{subfigure}
  \\
  \begin{subfigure}[b]{0.2\textwidth}
    \includegraphics[width=\textwidth]{Dostoyevsky}
    \caption{Dostoyevsky}
  \end{subfigure}
  \begin{subfigure}[b]{0.2\textwidth}
    \includegraphics[width=\textwidth]{Kafka}
    \caption{Kafka}
  \end{subfigure}
  \caption{Existentialist philosophers}
\end{figure}
```

Output:

(a) Kierkegaard

(b) Nietzsche

(c) Dostoyevsky

(d) Kafka

Figure 10: Existentialist philosophers

Let's try!
(images.tex)

Assignment:

- Create a document including multiple figures surrounded by random text (lorem ipsum).
- Modify the placements of the figures by using and changing the order of the specifiers.
- Use the subfigure environment and create subfigures with different inner positions.

1 Introduction

2 Document Structure

3 Page Layout and Style

4 Including Images

5 Creating Tables

- tabbing Environment
- Typesetting Tables (tabular)
- tabularx Environment
- Combining Rows and Columns
- table Environment
- Appearance of a Table

6 Equations

7 Reference

- The tabbing environment can align text in columns.
- This environment lets you create tab stops, and you can tab to a particular distance from the left margin.
- This environment is less often used than the tabular.
- Different practical commands that you can use in the tabbing environment are as follow.
  - `\=`: set a tab stop
  - `\>`: jump to the next tab stop
  - `\<`: go back a tab stop
  - `\+`: shift the left border by one tab stop to the right
  - `\-`: shift the left border by one tab stop to the left
  - `\\`: start a new line
  - `\kill`: set any tabs stops defined in that line, but will not typeset the line itself.

Code:

```
\begin{tabbing}
  \textbf{Anathema:} \=  Also known as \= : \= Pagan Angel  \\
  \> Origin \> : \> Leslie Liverpool, Merseyside, England \\
  \> Website \> : \>  \url{www.anathema.ws}\\
  \> Years \> : \> 1990 \textendash\ 2020
\end{tabbing}
```

Output:

Anathema: Also known as : Pagan Angel
Origin : Leslie Liverpool, Merseyside, England
Website : `www.anathema.ws`
Years active : 1990 – 2020

- The default LATEX method to create tables is the tabular environment.
- This environment requires a parameter to specify the number of columns and their alignments.
  - **c** means that the contents of the column will be centred.
  - **r** means that the contents of the column will be right-aligned.
  - **l** means that the contents of the column will be left-aligned.
  - You can put separator lines (|) between each column.
- `\hline` inserts a horizontal line on the table's top (or at the bottom).
- `&` is used as a cell separator.
- `\\` sets the end of the row.

- This example (shown in a `center` environment) displays a 3 by 3 table with different column alignments.
- You can use more number of separator lines (|) or `\hline`, which is not recommended.

Code:

```
\begin{tabular}{|c|l|r|}
  \hline
  A & B & C \\
  \hline \hline
  Cell 1 & Cell 2 & Cell 3 \\
  Cell 4 & Cell 5 & Cell 6 \\
  Cell 7 & Cell 8 & Cell 9 \\
  \hline
\end{tabular}
```

Output:

A	B	C
Cell 1	Cell 2	Cell 3
Cell 4	Cell 5	Cell 6
Cell 7	Cell 8	Cell 9

- If you require a fixed length either for each column or for the entire table, you can add the `array` package to the document preamble, and use parameter `m{<size>}` to set a length.
- The aligning options are `m` for middle, `p` for top and `b` for bottom.

Code:

```
\begin{tabular}{|m{15mm}|m{1cm}|m
    {1cm}|}
  \hline
  A &  B &  C \\
  \hline \hline
  A long dummy text in Cell 1 &
     Cell 2 & Cell 3 \\
  \hline
  Cell 4 & Cell 5 & Cell 6 \\
  \hline
  Cell 7 & Cell 8 & Cell 9 \\
  \hline
\end{tabular}
```

Output:

A	B	C
A long dummy text in Cell 1	Cell 2	Cell 3
Cell 4	Cell 5	Cell 6
Cell 7	Cell 8	Cell 9

- The environment tabularx is very similar to tabular, but the opening statement is different.

Code:

```
\begin{tabularx}{<width>}{<preamble>}
```

- The columns affected by the tabularx environment should be denoted with the letter X in the preamble argument.
- In the first argument, you should specify the table width, like `0.7\textwidth`.
- In the second argument,
  - `>{\raggedright\arraybackslash}X`: sets the alignment of the column to the left.
  - `>{\centering\arraybackslash}X`: sets the alignment of the column to the center.
  - `>{\raggedleft\arraybackslash}X`: sets the alignment of the column to the right.

Code:

```
\begin{tabularx}{0.9\textwidth}
  {
  | >{\raggedright\arraybackslash}X
  | >{\centering\arraybackslash}X
  | >{\raggedleft\arraybackslash}X |
  }
  \hline
  A &  B &  C \\
  \hline
  \lipsum[1][1] & Cell 2 & Cell 3 \\
  \hline
  Cell 4 & \lipsum[1][1] & Cell 6 \\
  \hline
  Cell 7 & Cell 8 & \lipsum[1][1] \\
  \hline
\end{tabularx}
```

Output:

A	B	C
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Cell 2	Cell 3
Cell 4	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Cell 6
Cell 7	Cell 8	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.

- Larger table cells can be created by merging rows and columns.
- `\multicolumn{<num_cols>}{<alignment>}{<content>}` command is used to merge several columns.
  - `num_cols` is the number of subsequent columns to merge.
  - `alignment` is either l, c, r, or to have text wrapping specify a width, like `p{25mm}`.
  - `content` is the data you want to be contained within the merged cell.
- You need to add the `multirow` package to combine rows.
- `\multirow{<num_rows>}{<width>}{<content>}` command is used to merge several columns.
  - using `*` for the width argumment specify the content's natural width.

Code:

```
\begin{tabular}{|c|c|c|}
  \hline
  \multicolumn{3}{|c|}{Multiple
      Column} \\
  \hline
  A &  B & C \\
  \hline
  \multirow{3}{*}{Multiple row} &
      Cell 1 & Cell 2 \\
  & Cell 3 & Cell 4 \\
  & Cell 5 & Cell 6 \\
  & Cell 7 & Cell 8 \\
  \hline
\end{tabular}
```

Output:

Multiple Column		
A	B	C
Multiple row	Cell 1	Cell 2
	Cell 3	Cell 4
	Cell 5	Cell 6
	Cell 7	Cell 8

- Similar to the figure environment, if you include the table inside a table environment, LaTeX will position it appropriately, and you can have more control over it by the additional parameter(s), like that of the figure, which can be passed to determine the table position.
- The table environment takes care of the numbering of the tables.
- You can also use `\caption` command, which produces the text shown below the table.

Code:

```
\begin{table}[h!]
  \begin{tabular}{|c|c|c|}
  \hline
  A & B & C \\ \hline
  D & E & F \\ \hline
  \end{tabular}
  \caption{This is the caption.}
\end{table}
```

Output:

A	B	C
D	E	F

Table 1: This is the caption.

- `\setlength{\arrayrulewidth}{<width>}` sets the thickness of the borders of the table.
- `\setlength{\tabcolsep}{<size>}` sets the space between the text and the left and right borders of the cell.
- `\renewcommand{\arraystretch}{<height>}` sets the height of each row relative to its default height.

Code:

```
\setlength{\arrayrulewidth}{1pt}
\setlength{\tabcolsep}{15pt}
\renewcommand{\arraystretch}{1.5}
\begin{table}[h!]
  \begin{tabular}{|c|c|c|}
  \hline
  A &  B & C \\ \hline
  D & E  & F  \\ \hline
  \end{tabular}
  \caption{This is the caption.}
\end{table}
```

Output:

A	B	C
D	E	F

Table 2: This is the caption.

- You can apply alternating colours to the rows of your table by using the xcolor package with the table option.
- It is possible by adding `\usepackage[table]{xcolor}` to the preamble.
- The command `\rowcolors{<start>}{<odd>}{<even>}`, which applies the alternating colours, takes three parameters.
  - `<start>` specifies the row to start.
  - `<odd>` specifies the colour for odd rows.
  - `<even>` specifies the colour for even rows.
- Choosing the available colours and different ways of creating a new one is based on the xcolor package.

Code:

```
\begin{table}
  \rowcolors{2}{yellow}{blue}
  \begin{tabular}
  {|m{15mm}|m{15mm}|m{15mm}|}
  \hline
  A & B & C \\
  \hline
  Cell 1 & Cell 2 & Cell 3 \\
  \hline
  Cell 4 & Cell 5 & Cell 6 \\
  \hline
  Cell 7 & Cell 8 & Cell 9 \\
  \hline
  Cell 10 & Cell 11 & Cell 12 \\
  \hline
  \end{tabular}
  \caption{A table with
      alternating colours.}
\end{table}
```

Output:

A	B	C
Cell 1	Cell 2	Cell 3
Cell 4	Cell 5	Cell 6
Cell 7	Cell 8	Cell 9
Cell 10	Cell 11	Cell 12

Table 3: A table with alternating colours.

- `\arrayrulecolor` command is used for changing the colour of lines.
- `\cellcolor` command is used for changing the background colour of a cell.
- `\rowcolor` command is used for changing the background colour of a row.
- To change the background colour of a column, you should define a new column type. The below command defines a column type called `<name>` whose alignment is p with the specified column width, and the colour is set with HTML format. This new column type can be used in the tabular environment.

Code:

```
\newcolumntype{<name>}
{>{\columncolor[HTML]{<HTML-color-code>}} p{<width>}}
```

Code:

```
\newcolumntype{s}{>{\columncolor[HTML]{304FFE}} >{\raggedright\
    arraybackslash}X }
\arrayrulecolor[HTML]{00C853}
\begin{table}
  \begin{tabularx}{0.9\textwidth} {
    | s
    | >{\raggedright\arraybackslash}X
    | >{\raggedright\arraybackslash}X | }
  \hline
  \rowcolor{lightgray} A &  B &  C \\ \hline
  \lipsum[1][1] & \lipsum[1][1] & \lipsum[1][1] \\ \hline
  \lipsum[1][1] & \lipsum[1][1] & \lipsum[1][1] \\ \hline
  \lipsum[1][1] & \lipsum[1][1] & \cellcolor[HTML]{F57C00} \lipsum
      [1][1] \\
  \hline
  \end{tabularx}
\end{table}
```

Output:

A	B	C
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.	Lorem ipsum dolor sit amet, consectetuer adipiscing elit.

Let's try!
(tables.tex)

Assignment:

- Create a document including a table with different merged cells (rows and columns).
- Change the appearance of the table and use the specifiers to modify its placement.

1 Introduction

2 Document Structure

3 Page Layout and Style

4 Including Images

5 Creating Tables

6 Equations
- Mathematical Modes
- amsmath Package
- Math Functions
- Math Operators

7 Reference

- LaTeX allows you to use two writing modes for mathematical expressions: the `inline` math mode and `display` math mode.
  1. `inline` math mode is used to write formulas in the middle of the text. Different delimiters can be used to typeset the math in `inline` mode. They are as follows.
     - `\(...\)`
     - `$...$`
     - `\begin{math}...\end{math}`
  2. `display` math mode is used to write important expressions that deserve to be highlighted. They are not part of a paragraph and are placed on separate lines in the centre of the page. To typeset maths in `display` mode, you can use the following constructions.
     - `\[...\]`
     - `\begin{displaymath}...\end{displaymath}`
     - `\begin{equation}...\end{equation}`

Code:

```
\documentclass{article}
\usepackage{lipsum}
\begin{document}
  \lipsum[1][1-3]
  Albert Einstein's famous formula is \(E=mc^2\).
  Albert Einstein's famous formula is $E=mc^2$.
  Albert Einstein's famous formula is \begin{math}E=mc^2\end{math}.
\end{document}
```

Output:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Albert Einstein's famous formula is $E = mc^2$. Albert Einstein's famous formula is $E = mc^2$. Albert Einstein's famous formula is $E = mc^2$.

Code:

```
\documentclass{article}
\usepackage{lipsum}
\begin{document}
  \lipsum[1][1-3] Albert Einstein's famous formula is
  \begin{equation}
    E=mc^2
  \end{equation}
\end{document}
```

Output:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Albert Einstein's famous formula is

$$E = mc^2 \tag{1}$$

- The standard LaTeX tools for equations may lack some flexibility, like overlapping or trimming part of the equation when it's too long, including several equations in the same line, etc.
- The `amsmath` package provides more options for displaying equations. It can be added to the preamble of the document using `\usepackage{amsmath}`.
- If you want unnumbered equations, use `equation*` environment.
- You can also display long equations[4] and split[5], align[6], group and center[7] them by using different environments provided by the `amsmath` package.

[4] `multline` environment
[5] `Split` environment
[6] `align` environment
[7] `gather` environment

Formula	LaTeX Code
x^2	`x^2`
x^{a+b}	`x^{a+b}`
x_2	`x_2`
x_{k+1}	`x_{k+1}`
x^2_{k+1}	`x_{k+1}^2`
$e^{e^{e^x}}$	`e^{e^{e^{x}}}`
$n \times m$	`n\times m`
± 1	`\pm 1`
$1 \div 2$	`1\div 2`
$\frac{1}{2}$	`\frac{1}{2}`

Table : Math Functions

Formula	LaTeX Code
$\sqrt{2}$	`\sqrt{2}`
$\sqrt[3]{8}$	`\sqrt[3]{8}`
$\ln e^x = x$	`\ln e^x = x`
$\{1,2,3,\ldots,n\}$	`\{1,2,3,\ldots ,n\}`
$\sum_{i=1}^{n} i$	`\sum_{i=1}^{n}i`
$\prod_{i=1}^{n} i$	`\prod_{i=1}^n i`
$\int_0^1 x^2 dx$	`\int_0^1x^2dx`
$\int\limits_{x\in C} dx$	`\int\limits_{x\in C}dx`
$\lim_{h\to 0}\frac{f(x+h)}{h}$	`\lim_{h\to 0}\frac{f(x+h)}{h}`
$\frac{1}{x}\Big\vert_1^\infty$	`\left.\frac{1}{x}\right|_1^\infty`

Table : Math Functions

$\amalg$ \amalg	$\circ$ \circ	$\ominus$ \ominus
$*$ \ast	$\cup$ \cup	$\oplus$ \oplus
$\bigcirc$ \bigcirc	$\dagger$ \dagger	$\oslash$ \oslash
$\bigtriangledown$ \bigtriangledown	$\ddagger$ \ddagger	$\otimes$ \otimes
$\bigtriangleup$ \bigtriangleup	$\diamond$ \diamond	$\pm$ \pm
$\bullet$ \bullet	$\div$ \div	$\setminus$ \setminus
$\cap$ \cap	$\mp$ \mp	$\sqcap$ \sqcap
$\cdot$ \cdot	$\odot$ \odot	$\sqcup$ \sqcup
$\star$ \star	$\times$ \times	$\triangleleft$ \triangleleft
$\triangleright$ \triangleright	$\uplus$ \uplus	$\vee$ \vee
$\wedge$ \wedge	$\wr$ \wr	

Table : Binary operation symbols

$\approx$ \approx	$\equiv$ \equiv	$\prec$ \prec	$\succ$ \succ
$\asymp$ \asymp	$\frown$ \frown	$\preceq$ \preceq	$\succeq$ \succeq
$\bowtie$ \bowtie	$\mid$ \mid	$\propto$ \propto	$\vdash$ \vdash
$\cong$ \cong	$\models$ \models	$\sim$ \sim	
$\dashv$ \dashv	$\parallel$ \parallel	$\simeq$ \simeq	
$\doteq$ \doteq	$\perp$ \perp	$\smile$ \smile	

Table : Binary relation symbols

$\geq$ \geq	$\gg$ \gg	$\leq$ \leq	$\ll$ \ll	$\neq$ \neq

Table : Inequality relation symbols

$\downarrow$	\downarrow	$\Longleftarrow$	\Longleftarrow
$\Downarrow$	\Downarrow	$\longleftrightarrow$	\longleftrightarrow
$\hookleftarrow$	\hookleftarrow	$\Longleftrightarrow$	\Longleftrightarrow
$\hookrightarrow$	\hookrightarrow	$\longmapsto$	\longmapsto
$\leftarrow$	\leftarrow	$\longrightarrow$	\longrightarrow
$\Leftarrow$	\Leftarrow	$\Longrightarrow$	\Longrightarrow
$\leftrightarrow$	\leftrightarrow	$\mapsto$	\mapsto
$\Leftrightarrow$	\Leftrightarrow	$\nwarrow$	\nwarrow
$\longleftarrow$	\longleftarrow	$\rightarrow$	\rightarrow
$\Rightarrow$	\Rightarrow	$\searrow$	\searrow
$\swarrow$	\swarrow	$\uparrow$	\uparrow
$\Uparrow$	\Uparrow	$\updownarrow$	\updownarrow
$\Updownarrow$	\Updownarrow	$\updownarrow$	\updownarrow

Table : Arrows

α \alpha	ζ \zeta	λ \lambda	π \pi	ϕ \phi
β \beta	η \eta	μ \mu	ρ \rho	χ \chi
γ \gamma	θ \theta	ν \nu	σ \sigma	ψ \psi
δ \delta	ι \iota	ξ \xi	τ \tau	ω \omega
ϵ \epsilon	κ \kappa	o o	υ \upsilon	

Table : Lowercase Greek letters

Γ \Gamma	Λ \Lambda	Σ \Sigma	Ψ \Psi
Δ \Delta	Ξ \Xi	Υ \Upsilon	Ω \Omega
Θ \Theta	Π \Pi	Φ \Phi	

Table : Uppercase Greek letters

$\acute{a}$ \acute{a}	$\check{a}$ \check{a}	$\grave{a}$ \grave{a}	$\tilde{a}$ \tilde{a}
$\bar{a}$ \bar{a}	$\ddot{a}$ \ddot{a}	$\hat{a}$ \hat{a}	$\vec{a}$ \vec{a}
$\breve{a}$ \breve{a}	$\dot{a}$ \dot{a}	$\mathring{a}$ \mathring{a}	

Table : Accents symbols

$\bot$ \bot	$\forall$ \forall	$\imath$ \imath	$\ni$ \ni	$\top$ \top
ℓ \ell	$\hbar$ \hbar	$\in$ \in	∂ \partial	$\wp$ \wp
$\exists$ \exists	$\Im$ \Im	$\jmath$ \jmath	$\Re$ \Re	

Table : Symbols dereived from letters

$\sqsubseteq$ \sqsubseteq	$\subset$ \subset	$\supset$ \supset
$\sqsupseteq$ \sqsupseteq	$\subseteq$ \subseteq	$\supseteq$ \supseteq

Table : Subset and superset symbols

$\aleph$ \aleph	$\emptyset$ \emptyset	∇ \nabla	$\sharp$ \sharp
$\angle$ \angle	$\flat$ \flat	$\natural$ \natural	$\spadesuit$ \spadesuit
$\clubsuit$ \clubsuit	$\heartsuit$ \heartsuit	$\neg$ \neg	$\surd$ \surd
$\diamondsuit$ \diamondsuit	∞ \infty	$\prime$ \prime	$\triangle$ \triangle

Table : Misc symbols

$\ldots$ \ldots	$\ddots$ \ddots	$\cdots$ \cdots	$\vdots$ \vdots

Table : Ellipsis

Code:

```
\begin{equation}
  x = a_0 + \frac{1}{\displaystyle a_1
    + \frac{1}{\displaystyle a_2
    + \frac{1}{\displaystyle a_3 + a_4}}}
\end{equation}
```

Output:

$$x = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + a_4}}} \tag{2}$$

Code:

```
\begin{equation}
  z = \overbrace{
    \underbrace{x}_\text{real} + i
    \underbrace{y}_\text{imaginary}
    }^\text{complex number}
\end{equation}
```

Output:

$$z = \overbrace{\underbrace{x}_{\text{real}} + i \underbrace{y}_{\text{imaginary}}}^{\text{complex number}} \tag{3}$$

Code:

```
\begin{equation}
  u(x) =
  \begin{cases}
    \exp{x} & \text{if } x \geq 0 \\
    1       & \text{if } x < 0
  \end{cases}
\end{equation}
```

Output:

$$u(x) = \begin{cases} \exp x & \text{if } x \geq 0 \\ 1 & \text{if } x < 0 \end{cases} \tag{4}$$

Code:

```
\begin{align*}
  S(\omega)
  &= \frac{\alpha g^2}{\omega^5} e^{[ -0.74\bigl\{\frac{\omega U_\
      omega 19.5}{g}\bigr\}^{\!-4}\,]} \\
  &= \frac{\alpha g^2}{\omega^5} \exp\Bigl[ -0.74\Bigl\{\frac{\omega
      U_\omega 19.5}{g}\Bigr\}^{\!-4}\,\Bigr]
\end{align*}
```

Output:

$$
\begin{aligned}
S(\omega) &= \frac{\alpha g^2}{\omega^5} e^{[-0.74\left\{\frac{\omega U_\omega 19.5}{g}\right\}^{-4}]} \\
&= \frac{\alpha g^2}{\omega^5} \exp\left[-0.74\left\{\frac{\omega U_\omega 19.5}{g}\right\}^{-4}\right]
\end{aligned}
$$

Let's try!
(equations.tex)

Assignment:

Typeset the following equations:

- $$\int_{-\infty}^{\infty} e^{-x^2}\,dx = \sqrt{\pi} \tag{5}$$

- $$\begin{bmatrix} P \\ Q \end{bmatrix} = \gamma \begin{bmatrix} \omega_1/N + \Lambda \\ \omega_2/N + \Lambda \end{bmatrix} \qquad \text{for some } \gamma \in \mathrm{M}_2(\mathbb{Z}/N\mathbb{Z})$$

1 Introduction

2 Document Structure

3 Page Layout and Style

4 Including Images

5 Creating Tables

6 Equations

7 Reference
- Cross-references
- hyperref Package
- Table of Contents
- List of Figures and Tables
- Bibliography (BibTeX)
- Indexing (makeidx)

- In LaTeX, you can easily reference almost anything that can be numbered, and LaTeX automatically updates the numbering elements (like equations, tables and figures) in the document.
- You can reference chapters, sections, subsections, footnotes, theorems, equations, figures and tables.
- `\label{<name>}` give the object you want to reference a name that can be used to refer to that later.
- `\ref{<name>}` reference an object with the specified name, and print the number assigned to the object.
- `\pageref{<name>}` print the page number where the object with the specified name is found.
- If a label is declared within a float environment, the `\ref{}` command will return the respective float (figure, table, etc.) number, but it must occur after or within the `\caption{}` command.

- LaTeX users usually add a prefix to the label to describe it. Following this convention, the label of what you are referencing will look like `\label{<prefix>:<name>}`. These prefixes can become increasingly helpful in large documents.

prefix	What you are referencing
ch:	chapter
sec:	section
subsec:	subsection
fig:	figure
tab:	table
eq:	equation
lst:	code listing
itm:	enumerated list item
alg:	algorithm
app:	appendix subsection

- You must compile your document twice to see the results of cross-referencing. In the first step, the compiler stores the labels with the right number for referencing, and then it replaces the references with these numbers.
- The label can be referenced with a tilde (~) that indicates a non-breaking space.
- If you reference something that does not exist, the compilation of the document will be successful but will return a warning, and replace the reference with "??".

Output:

```
LaTeX Warning: There were undefined references.
```

Code:

```
As you can see in Table~\ref{tab:A
    -to-F}, \lipsum[1][1-3]

\begin{table}[h!]
  \begin{tabular}{|c|c|c|}
  \hline
  A &  B & C \\ \hline
  D & E  & F  \\ \hline
  \end{tabular}
  \caption{A caption.}
  \label{tab:A-to-F}
\end{table}
```

Output:

As you can see in Table 17, Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris.

A	B	C
D	E	F

Table 17: A caption.

- The package `hyperref` provides LaTeX with the ability to create hyperlinks within the document.
- Loading the package in the preamble will automatically turn all your internal references into hyperlinks.
- By using hyperref, the connections of labels and references will become links, and you will be able to click on them to be redirected to the right place. Also, the table of contents and list of figures and tables will be made of hyperlinks, too.
- This package also provides the `\url{<URL address>}` and `\href{<URL address>}{<description>}` commands that show the URL (or description in the latter), and if you click on it, your browser opens it.
- You can insert email links by `\href{mailto:<email>}{<email>}` that shows your email address, and if the (pdf) readers click on it, they can send an email.

- There are many options that can be passed as arguments of the `hyperref` package when it is loaded or by using the `\hypersetup` command.
- Some of the possible variables that can be changed are as follows.

Variable	Comment
pdfborder	set the style of the border around a link.
colorlinks	surround the links by color frames (false) or colors the text of the links (true).
hidelinks	hide links (removing color and border)
linkcolor	color of internal links (sections, pages, etc.)
citecolor	color of citation links (bibliography)
urlcolor	color of URL links (mail, web)
linkbordercolor	color of frame around internal links (if colorlinks=false)
citebordercolor	color of frame around citations
urlbordercolor	color of frame around URL links

- The table of contents can be automatically generated and modified in a LaTeX document.
- You can use the command `\tableofcontents` to create it.
- You must compile the document twice to see the effects in the table of contents.
- You can change the default title of the table of contents, which is "Contents", into whatever you need by using `\renewcommand*\contentsname{<new name>}`.
- To manually add entries (like an unnumbered section), use the command `\addcontentsline`.

Code:

```
\addcontentsline{toc}{section}{<section name>}
\section*{<section name>}
```

- For changing the depth of the table of contents, you can set the tocdepth by using the command `\setcounter{tocdepth}{X}`, where X is the desired depth, and `X = {0,1,2,3,4,5}`.
  - A value of 0 means that your table of contents will show nothing, and 5 means that sections, subsections, subsubsections, paragraphs, and subparagraphs will be shown.
- The value should be set in the preamble of the document.
- The output below shows the result of setting tocdepth to 2.

Output:

Contents

1 Section **2**
1.1 Subsection 2

- The generation of a list of figures and tables is the same. You can use `\listoffigures` and `\listoftables` commands.
- You can change the default titles, "List of Tables" and "List of Figures", to any other text by using the following commands.
  - `\renewcommand{\listfigurename}{<new name>}`
  - `\renewcommand{\listtablename}{<new name>}`
- The sectioning and caption commands provide an optional argument that can take a different (usually a shorter) title for the ToC, LoF, or LoT.

Output:

```
\begin{figure}
  % figure contents
  \caption[Short version for LoF]{Long version to appear next to the
      figure}
\end{figure}
```

- A BibTeX database is a plain text file that must be given the extension .bib.
- You can convert references in other formats using online tools, export references from Google Scholar into BibTeX, or type the database from scratch.
- The order of the fields in each item type is unimportant.
- Your bib file can contain references you don't cite. BibTeX will put in the list of references at the end of your paper only the ones you cite.
- After creating a database (.bib) file, you should specify the style and location of the bibliography in your LaTeX document.

- To have a LaTeX file with references to your bibliography database, you should do the following steps:

1. Create a database (.bib) file, including your references.
2. Refer to a key in your BibTeX file whenever you want to cite an item in the file by using `\cite{<key>}` command
3. Use the `\bibliographystyle` command to set the style (like acm, alpha, ieeetr, plain, abbrv, etc.) you want to use. It determines how the references are formatted in the bibliography of the document.
4. Use the `\bibliography` command to generate the bibliography at the point in your document where you want it to appear.

Code: (bibliography.bib)

```
@book{lamport1985i1,
  title={I1 ($\backslash$LaTeX)---A Document},
  author={Lamport, Leslie},
  volume={410},
  year={1985},
  publisher={pub-AW}}
```

Code: (LaTeX file.tex)

```
\documentclass[11pt]{article}
\usepackage{lipsum}
\begin{document}
  \LaTeX\ is a special version of Donald Knuth's TEX program for
      computer typesetting~\cite{lamport1985i1}, \lipsum[1][1-8]
  \bibliography{bibliography}{}
  \bibliographystyle{plain}
\end{document}
```

Output:

LaTeX is a special version of Donald Knuth's TEX program for computer typesetting [1], 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.

References

[1] Leslie Lamport. *I1 (\LaTeX)—A Document*, volume 410. pub-AW, 1985.

- You should decide the entry type before entering a reference in the BibTeX database. Although there is no fixed classification scheme, there are enough entry types to handle almost any reference.
- References to different types of publications contain different fields of information.
  - For example, a reference to a journal article might include the number of the journal, which is not a meaningful field for a book.
- Different fields of database entries are divided into three classes:
  - **required:** Omitting the field will produce a warning message and sometimes a badly formatted bibliography entry.
  - **optional:** If present, the field's information will be used but can be omitted without causing any problems.
  - **ignored:** The field is ignored.

- The following are some of the standard entry types.

Entry Type	Description
article	An article from a journal or magazine.
book	A book with an explicit publisher.
booklet	A work that is printed and bound, but without a named publisher or sponsoring institution.
inproceedings	An article in a conference proceedings.
manual	Technical documentation.
mastersthesis	A Master's thesis.
phdthesis	A PhD thesis.
proceedings	The proceedings of a conference.
techreport	A report published by a school or other institution, usually numbered within a series.
unpublished	A document having an author and title, but not formally published.
misc	Use this type when nothing else fits.

- Below is a description of some of the standard fields recognized by bibliography styles.

Field	Description
author	The name(s) of the author(s).
booktitle	Title of a book, part of which is being cited.
chapter	A chapter (or section or whatever) number.
edition	The edition of a book.
editor	Name(s) of editor(s).
organization	The organization that sponsors a conference or that publishes a manual.
journal	A journal name.
publisher	The publisher's name.
title	The work's title.
volume	The volume of a journal or multivolume book.
year	The year of publication or, for an unpublished work, the year it was written.

- The `makeidx` package can be used to enable the indexing feature of the document. After loading this package, the `\makeindex` command (located in the preamble) tells LaTeX to do the necessary steps for indexing.
- Command `\index{<key>}` tells LaTeX what to index, where `<key>` is the index entry and does not appear in the output.
- You should enter the index commands at the points in the text that you want to be referenced in the index.
- Command `\printindex` is used to show the index (usually at the end of the document).
- If some entry has subsections, you can use `!` to categorize an index under another one. For example, `\index{science!physics}` would create an index entry with 'physics' categorized under 'science'.

- To perform multi-page indexing, add a `|(` and `|)` to the end of the `\index` command, like `\index{math|(}` and `\index{math|)}`.
- You can use cross-referencing by `\index{<key>| see{<key>}}`, where the latter is the main key.
- When the `.tex` file is processed, each `\index` command writes an index entry and a page number to a `.idx` file (with the same name as the input). This file should be processed with the `makeindex` program by running `makeindex <file name>` command in the command line. The `makeindex` program generates a sorted index in another file with the extension `.ind` and the same name. After that, If the LaTeX input file is processed again, this sorted index gets included.

Code:

```
\documentclass[11pt]{article}
\usepackage{makeidx} \makeindex
\usepackage{lipsum}
\begin{document}
  \lipsum \index{data}
  \lipsum \index{algorithm}
  \lipsum \index{algorithm!random
      forests}
  \lipsum \index{algorithm!
      gradient boosting}
  \lipsum \index{algorithm!
      decision trees}
  \lipsum \index{information|see {
      data}}
  \lipsum \index{model building|(}
  \lipsum \index{model building|)}
  \printindex
\end{document}
```

Output: (Index Part)

Index

algorithm, 4
    decision trees, 8
    gradient boosting, 7
    random forests, 5

data, 2

information, *see* data

model building, 12–13

Let's try!
(references.tex)

Assignment:

- Create a document with multiple images and tables, and mention them in the text.
- Change the title of tables only in the LoT.
- Find a way to mention the relative page references of float environments (figures and tables) more naturally, like "in the following page"
- Use a bibliography stored as a .bib file to show the references in the last part of the document

References:

[1] A simple guide to latex - step by step.
`https://latex-tutorial.com/tutorials/`.

[2] Overleaf (latex documentation).
`https://www.overleaf.com/learn`.

[3] Wikibooks (latex documentation).
`https://en.wikibooks.org/wiki/LaTeX`.

[4] S. Kottwitz. LaTeX beginner's guide. Packt Publishing Ltd, 2011.

Contact:


Soroush Omidvartehrani


omidvar@mail.um.ac.ir


sesh.ir