Introduction to LaTeX[1]

Yu Ouyang

QIPSR Fall Software Festival

October 18, 2013

Contents

[1]Materials in this workshop borrow substantially from Andrew Roberts's excellent set of LaTeX tutorials, available online.

List of Tables

List of Figures

1 Introduction

This document provides an introduction to LaTeX, a popular document preparation system and document markup language. LaTeX refers not to a particular editing program or word-processor, but to a different way to write and construct good-looking documents. Before you begin, keep in mind that becoming proficient in LaTeX requires a certain amount of start-up costs. The desire to quit will be strong. Nonetheless, your efforts will be well-worthed. Soon, you will be able to utilize the powers of LaTeX to create professionally typeset documents and presentations slides.

2 What is LaTeX?

In 1978, Donald Knuth - arguably one of the most famous and well respected computer scientists - embarked on a project to create a typesetting system, called TeX (pronounced "tech"), after being disappointed with the quality of his acclaimed The Art of Programming series. Around 10 years later, he froze the language after originally anticipating spending a single year! TeX gave extremely fine-grained control of document layout. However, the vast flexibility meant it was complex, so by the mid-80s Leslie Lamport created a set of macros that abstracted away many of the complexities. This allowed for a simpler approach for creating documents, where content and style were separate. This extension became LaTeX (pronounced "lay-tech" or "la-tech").

LaTeX is essentially a markup language. Content is written in plain text and can be annotated with various "commands" that describe how certain elements should be displayed. The LaTeX interpreter reads in a LaTeX marked-up file, renders the content into a document and dumps it a new file.

3 Why Use LaTeX?

I've been using MS Word (or comparable program) to create documents for years, why should I change now? The answer is: you do not have to. However, LaTeX does offer a number of advantages compared to What-You-See-Is-What-You-Get (WYSIWYG) word-processing programs such as MS Word.

3.1 Separation of Content and Style

Not the most obvious advantagem LaTeX allows you to concentrate on the content as you write. You introduce structure explicitly by telling LaTeX when a new section begins, for example, but you don't then mess around trying to decide how the section headers should *look*.

Compared this to what you may typically do in a word-processing program. Usually, you would begin by highlighting a given section header and applying formatting to it: maybe a larger font, maybe underline, etc. The point is that this will then have to be applied to every header manually. LaTeX is better as it uses a document style. This defines how different elements within your document should look. If you fancy a change, you only change the style definitions once, then the presentation of the *entire* document will be updated automatically. This also ensures a consistent looking document

3.2 Portability

An actual LaTeX file is merely a text file, which is about the most portable format in computing. Further, since the LaTeX system that can process the text file and produce the finished document has been implemented on just about every mainstream platform you care to mention. In short, you do not have to worry whether your current computer will allow you to edit your LaTeX file. If you want, some apps will even allow you to create entire documents in LaTeX using your tablet or phone.

3.3 Flexibility

You can get LaTeX to do just about anything you can think of! Over the years, an overwhelming selection of packages to extend its potential and macros that can simplify complex tasks have come into being, most of which are freely available on th Comprehensive TeX Archive Network (CTAN). For example, LaTeX's main users are within academia and research institutions and they benefit hugely thanks to the Bibtex package that provides bibliography management. There are other crazy packages that you can install which allow you to typeset music scores, chessboards and cross-words! CTAN is the main repository of these resources. Most are well documented and as you can imagine, with LaTeX being around for so long, the

number of extensions is vast. The chances are, if you're struggling to do a task, someone will have undoubtedly written a package to solve it easily!

3.4 Control

Even with simple documents, you can quickly become frustrated by most word-processors rather unintelligent interference. The hours that are wasted trying to position that image which you *know* will fit at the bottom of the page, but Word refuses to put it there! How many can relate to this experience? You have your 30 page document with text, tables and images. You just spent the evening getting it formatted nicely - all your figures in the right place and then you notice that one of your paragraphs isn't clear enough. You add **one** sentence, which then pushes an image on to the next page, leaving a massive gap at the bottom of that page where your image once was. This then daisy-chains down, knocking other tables and images out of place all the way to the end of your document! It's a real laugh. Fortunately, LaTeX is much more clever in this respect and positions your images and tables with a lot of common sense. So, if you want your image to appear at the bottom of a given page, it'll stay there!

Whilst LaTeX makes decent typesetting decisions for you, if you want to, you can have total control over the presentation of your document. This extends to when you have large document (20+ pages). Usually, with word processors, writing and formatting large documents is not a pleasant experience. With LaTeX, such problems do not exist. Additionally, you are free to split up large documents into smaller chunks and then let LaTeX combine them altogether later (like one chapter per file). It can also create tables of content, indexes and bibliographies easily (usually with just one command), even on multi-file projects.

3.5 Cost

This is where LaTeX really excels. It is Free! Moreover, unlike most open source software, the phrase "*you get what you pay for*" does not apply.

4 Your First LaTeX Document

The primary reason why people isn't using LaTeX is because you can't just load up and go, like you can with a word processor. The first thing you need to be aware of is that LaTeX uses a markup language in order to describe document structure and presentation. What LaTeX does is to convert your source text, combined with the markup, into a high quality document. For the purpose of analogy, web pages work in a similar way: the HTML is used to describe the document, but it is your browser that presents it in its full glory - with different colors, fonts, sizes, etc.

Let's begin by creating a simple LaTeX document. Open TeXstudio[2] and type (or copy-and-paste) the following. Compile the document (F1) to see the PDF output.

```
%Our first LaTeX example!

\documentclass{article}

\begin{document}

Hello World!

\end{document}
```

What does it all mean?

`%Our first LaTeX example!` is a *comment*. This is because it begins with the percent symbol (%), which when LaTeX sees, simply ignores the rest of the line.

`\documentclass{article}` tells LaTeX to use the article document class. A document class file defines the formatting, which in this case is a generic article format. The handy thing is that if you want to change the appearance of your document, substitute article for another class file that exists.

`\begin{document}` alerts LaTeX that content of the document is about to commence. Anything above this command is known generally to belong in the *preamble*.

[2] TeXstudio is a popular graphical-user interface to write LaTeX documents for Windows users.

`Hello World!` This is the only actual line containing real content - the text that we wanted displayed on the page.

`\end{document}` tells LaTeX that the document source is complete.

5 Document Structure

As we saw in the simplistic example, LaTeX practically forces you to declare structure within your documents. This is a good thing though. Because once LaTeXunderstands how you want your document organized, it will take care of all the tedious business of the layout and presentation for you. The separation of content and layout allows you to concentrate on the job at hand, i.e., communicating your research.

Since we rarely create a document with only "Hello World!" in the content, we will examine the basics that will allow us to create an entire article or report in LaTeX.

5.1 Preamble

The preamble consists of everything from the start of the LaTeX source file until the `\begin{document}` command. It normally contains commands that affect the entire document. Open a new file in TeXstudio, and type the following:

```
% Our second LaTeX example - document structure

\documentclass{article}

\begin{document}


\end{document}
```

The first line is a comment (as denoted by the % sign). The `\documentclass` command takes an argument, which is this case is article, because that's the type of document we want to produce. Other classes that exist are book, report, thesis etc. It is also possible to create your own, as is often done by journal publishers, who simply provide you with their own class file, which tells LaTeX how to format your

content. But we'll be happy with the standard article class for now! And finally, the `\begin{document}`. This strictly isn't part of the preamble, but I'll put it here anyway, as it implies the end of the preamble by nature of stating that the document is now starting.

5.2 Top Matter

At the beginning of most documents will be information about the document itself, such as the title and date, and also information about the authors, such as name, address, email etc. All of this type of information within LaTeX is collectively referred to as top matter. Although never explicitly specified, that is, there is no such `\topmatter` command, you are likely to encounter the term within LaTeX documentation.

Let's add some information to the top matter section part of our new file. Add the following to our file and put it after the `\begin{document}` command:

```
\title{Title of Your Paper}
\author{Your Name \\
  Name of Your Department\\
  University of Kentucky \\
  \texttt{email address}}
\date{\today}
\maketitle
```

The `\title` command is fairly obvious. Simply put the title you want between the curly braces. `\author` would also seems easy, until you notice that we've crammed in all sorts of other information along with the name. This is merely a common, albeit ungraceful, hack due to the basic article class. If you are provided with a class file from a publisher, or if you use the AMS article class (amsart), then you have a more logical approach to entering author information. In the meantime, you can see how we use the new line command (\\) to start writing on a new line. Your email address is at the end, and the `\texttt` commands formats the email address using a mono-spaced font. The `\date` command takes an argument to signify the date the document was written. I've used a built-in command called `\today` which, when processed by LaTeX, will be replaced with the current date. But you are free to put whatever you want as a date, in no set order. If braces are left empty, then

the date is then omitted. Without `\maketitle`, the top matter would not appear in the document. So it is needed to commit your article attributes to paper.

5.3 Abstract

Let's now add an abstract to our paper. As most research papers have an abstract, then there is a predefined commands for telling LaTeX which part of the content makes up the abstract. This should appear in its logical order, therefore, after the top matter, but before the main sections of the body.

```
\begin{abstract}
Your abstract goes here...
...
\end{abstract}
```

5.4 Sectioning Commands

In most research paper, the main body of the article follows the article abstract. Usually, the article is divided into sections. The commands for inserting sections are fairly intuitive. Of course, certain commands are appropriate to different document classes. For example, a book has chapters but a article doesn't.

Let's add some sections to our article.

```
\section{Introduction}

This section's content...

\section{Theory}

This section's content...

\section{Data and Analysis}

This subsection's content...

\subsection{Data}

This subsection's content...

\subsubsection{Variables}

This subsection's content...

\section{Conclusion}

This subsubsection's content...
```

6 Everything Else You Need to Know to Write in LaTeX (Mostly)

When we want to make simple changes to our text, this is known as formatting. The term formatting is rather broad, but in this case it needs to be as this section will guide you various text, paragraph and page formatting techniques. Formatting tends to refer to most things to do with appearance, it makes the list of possible topics quite eclectic: text style, font, size; paragraph alignment, interline spacing, indents; special paragraph types; list structures; footnotes, margin notes, etc.

A lot of the formatting techniques are required to differentiate certain elements from the rest of the text. It is often necessary to add emphasis to key words or phrases. A numbered or bulleted list is also commonly used as a clear and concise way of communicating an important issue. Footnotes are useful for providing extra information or clarification without interrupting the main flow of text. So, for these reasons, formatting is very important. However, it is also very easy to abuse, and a document that has been over-done can look and read worse than one with none at all.

To keep things simple, we will cover only the most common techniques.

6.1 Page Layout

LaTeX and the document class will normally take care of page layout issues for you. For submission to an academic publication, this entire topic will be out of your hands, as the publishers want to control the presentation. However, for your own documents, there are some obvious settings that you may wish to change: page dimensions, size of content on page, page orientation, etc. The purpose of this tutorial is to show you how to configure your pages.

6.1.1 Page Dimensions

There are many ways to specify page dimensions in LaTeX. To keep things simple, we will only examine how to set the default page size. Consider the following code:

```
\documentclass[letterpaper]{article}
```

The above example illustrates how to pass the optional argument to the `\documentclass`, which will then modify the page dimensions accordingly. The standard document classes that are a part of LaTeX are built to be fairly generic, which is why you have the flexibility of specifying the page size. Other classes may have different options (or none at all). Normally, 3rd party classes come with some documentation to let you know.

Additionally, there are several packages designed to solve the problem of varying pages sizes, which override any defaults setup by the document class. Examples such as a4 are rather specialized in preparing only one type of page. One of the most versatile packages for page layout needs is `geometry`. It will feature a number of times throughout this tutorial a it has many tricks up its sleeve! Anyway, to set the page size, add the following to your preamble:

```
\usepackage[letterpaper]{geometry}
```

`letterpaper` is just one of the many pre-defined page sizes built-in, other include: `a0paper`, `a1paper`, `a6paper`, `a4paper`, `b0paper`, `b1paper`, `b6paper`, `letterpaper`, `legalpaper`, `executivepaper`.

6.1.2 Page Orientations

When you talk about changing page orientation, it usually means changing to landscape mode, since portrait is the default. We will examine two slightly different styles of changing orientation.

The first is for when you want all of your document to be in landscape from the very beginning. There are various packages available to achieve this, but the one I prefer is the `geometry` package. All you need to do is call the package, with `landscape` as an option:

```
\usepackage[landscape]{geometry}
```

Although, if you intend to use `geometry` to set your paper size, don't add the `\usepackage` commands twice, simply string all the options together, separating with a comma:

```
\usepackage[landscape, letterpaper]{geometry}
```

The second method is for when you are writing a document in portrait, but you have some contents, like a large diagram or table that would be displayed better on a landscape page. However, you still want the consistency of your headers and footers appearing the same place as the other pages.

The `lscape` package is for this very purpose. It supplies a `landscape` environment, and anything inside is basically rotated. No actual page dimensions are changed. This approach is applicable to books or reports than to typical academic publications.

To see an example, add `\usepackage{lscape}` and `\usepackage{threeparttable}` to the preamble and the following to your LaTeX file:

```
\begin{landscape}
\begin{table}[h]
\caption{Title of this Table}
\label{exampleTable}
\centering
    \begin{threeparttable}
        \begin{tabular}{ l . . }
        \toprule
        {Variables} & \multicolumn{1}{c}{Column Two} & \multicolumn{1}{c}{Column Thre
        \midrule
        Var1 & 2    & 3     \\
        Var2 & 5.0  & 6.0   \\
        Var3 & 8.00 & 9.00  \\
        \bottomrule
        \end{tabular}
        \begin{tablenotes}[para, flushleft]
            Note: This is the notes for this table. This is really important.
        \end{tablenotes}
     \end{threeparttable}
\end{table}
\end{landscape}
```

The output is:

Table 1: A Very Large Table That Will Only Fit In A Landscape Orientation

Variables	Column Two	Column Three
Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Note: This is the notes for this table. This is really important.

6.1.3 Full Page

By default, LaTeX leaves a fairly large margin on the top, bottom, left, and right sides. The `fullpage` package tells LaTeX to set all margins to 1 inch or 1.5 cm. To set all margins to 1 inch, add the following to the preamble:

```
\usepackage{fullpage}
```

6.2 Quotes

LaTeX treats left and right quotes as different entities. For single quotes, ‘ (on the keyboard, this symbol is found on the key adjacent to the number 1) gives a left quote mark, and ’ is the right. For double quotes, simply double the symbols, and LaTeX will interpret them accordingly.

6.3 Emphasizing Text

In order to add some emphasis to a word or phrase. the simplest way is to use the `\textit{text}` command. In TeXstudio, the shortcut is CTRL-I, just like most word processor.

6.4 Bold Text

To bold text, use the `\textbf{text}` command. In TeXstudio, the shortcut is CTRL-B, just like most word processor.

6.5 Line Spacing

Sometimes, it's necessary to require something other than single line spacing. First, add `\usepackage{setspace}` to the document preamble.

To double-space the entire document, add `\doublespacing` immediately after the `\begin{document}` command.

The `setspace` package also provides the following environments for use within your document:

```
doublespace    -  all lines are double spaced
onehalfspace   -  line spacing set to one-and-half spacing
singlespace    -  normal linespacing
```

6.6 Verbatim Environment

This environment was used in an example in the previous tutorial. 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 monospace font. Ideal for typesetting program source code, for example.

6.7 Quote and Quotation Environments

There are two environments available for quoting passages within your own documents, with only a subtle difference between them. The `quote` environment is designed for a short quotation, or a series of small quotes, separated by blank lines. `quotation` on the other-hand, is for use with longer quotes, of more than one paragraph. Both are indented on either margin, and you will need to add your own quotation marks if you want them.

6.8 List Structures

Lists often appear in documents, especially academic, as their purpose is often to present information in a clear and concise fashion. We will discuss the two most common: `itemize` and `enumerate`.

All lists follow the basic format:

```
\begin{list_type}
  \item The first item
  \item The second item
  \item The third, etc
\end{list_type}
```

6.8.1 Itemize Environment

This environment is for your standard bulleted list of items:

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

The above codes result in:

- The first item
- The second item
- The third etc

6.8.2 Enumerate Environment

The `enumerate` environment is for ordered lists, where by default, each item is numbered sequentially.

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

The above codes produce:

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

6.9 Nested Lists

LaTeX will happily allow you to insert a list environment into an existing one (up to a depth of four). Simply begin the appropriate environment at the desired point within the current list. LaTeX will sort out the layout and any numbering for you. For example, the codes below:

```
\begin{enumerate}
  \item The first item
  \begin{enumerate}
    \item Nested item 1
    \item Nested item 2
  \end{enumerate}
  \item The second item
  \item The third etc
\end{enumerate}
```

produce:

- The first item
  - Nested item 1
  - Nested item 2
- The second item
- The third etc

6.10 Footnotes

Footnotes are a very useful way of providing extra information to the reader. It is normally non-essential, and so can be placed at the bottom of the page, which means the main body remains concise.

The footnote facility is easy to use. The command you need is: `\footnote{text}`. Do not leave a space between the command and the word where you wish the footnote marker to appear, otherwise LaTeX will process that space and will leave the output not looking as intended.

To see an example, add the following to your LaTeX file:

```
This is a sentence.\footnote{This is our footnote.}
```

The above codes produce:

This is a sentence.[3]

[3] This is our footnote.

7 Bibliography

In the previous section (Document Structure), we examined how to create a basic research paper using LaTeX. For any academic/research writing, incorporating your references into your document is an important task. Fortunately, as LaTeX was aimed for this sort of work, it has a variety of features that make dealing with your references much simpler. LaTeX has built in support for citing references. However, a much more powerful and flexible solution is achieved thanks to an auxiliary tool called BibTeX (which comes bundled as standard with LaTeX.)

BibTeX allows you to store all your references in an external, flat-file database. You can then easily link this database to any LaTeX document, and cite any reference that is contained within the file.A BibTeX database is stored as a .bib file. It is a plain text file, and so can be viewed and edited easily. The structure of the file is also quite simple. Here is an example of a BibTeX entry[4]:

```
@article{Dovi2002APSR,
    author = {Dovi, Suzanne},
    journal = {American Political Science Review},
    number = {December},
    pages = {729--743},
    title = {{Preferable Descriptive Representatives}},
    volume = {96},
    year = {2002}
}
```

The information in the sample BibTeX entry should be fairly straightforward, containing all of the usual information that we would include in a citation, e.g., title of article, author, journal, etc. The only piece of information that may be "new" is `Dovi2002APSR`. This is known as the citation key, or the shorthand that you would use to cite this particular article.

To take full advantage of BibTeX, we will use the Natbib package. Add the following to the preamble, i.e., after `\documentclass{article}` and before `\begin{document}`.

[4]No need to worry to create BibTeX files manually. If you use a citation manager such as Mendeley (free) or Endnote (commercial), the citation manager software can generate a BibTeX file automatically for you.

```
\usepackage{natbib}
```

Further, add the following to the end of your LaTeX file, right before the `\end{document}` command.

```
\newpage
\bibliographystyle{apsr2}
\bibliography{library}
```

The commands above tell LaTeX that: (1) we want to start on a new page; (2) to specify the bibliography style; and (3) and the name of our bibliography file. In this particular case, the name of our BibTeX file is `library`. The bibliography style file is a .bst file that specifies how exact formats of the citations (both in-text and the references section). Note that both the BibTeX and the bibliography style files must be in the same location (directory) on your computer the LaTeX file that you're writing.

Two of the most common LaTeX commands to cite a reference are: `\citet{Dovi2002APSR}` and `\citep{Dovi2002APSR}`. To see examples, add the following to the main body of the LaTeX file:

```
This is true about descriptive representation \citep{Dovi2002APSR}.
\citet{Dovi2002APSR} says this about descriptive representation.
```

8 Table

Tables are common in academic writing, often for summarizing results. Constructing tables in LaTeX may look intimidating at first, so we will start slowly and build up. For tables, we will use the `dcolumn` package since it is one of the easiest ways to make tables in LaTeX. Add the following to the preamble:

```
\usepackage{dcolumn}
\newcolumntype{.}{D{.}{.}{-1}}
```

8.1 The Tabular Environment

To begin, we shall first get familiar with some LaTeX terminology. An environment in LaTeX is a special declaration for formatting specific types of text. We have already used environments before. For example, we previously introduced the abstract environment, so that the abstract of the paper could be formatted differently that the main body of text. All environments begin and end in the same fashion:

```
\begin{environment-name}
...
...
\end{environment-name}
```

The *tabular* is another such environment, designed for formatting your data into nicely arranged tables. Arguments are required after the environment declaration to describe the alignment of each column. The number of columns does not need to be specified as it is inferred by looking at the number of arguments provided. It is also possible to add vertical lines between the columns here. The following symbols are available to describe the table columns:

```
l   left-justified column
c   centered column
r   right-justified column
.   aligned by decimal point
```

Once in the environment,

```
&   column separator
\\  start new row
```

Let's look at an example. Add the following to the body of your LaTeX file:

```
\begin{tabular}{ l c r }
  Var1 & 2    & 3    \\
  Var2 & 5.0  & 6.0  \\
  Var3 & 8.00 & 9.00 \\
\end{tabular}
```

The above codes produce the following table.

Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

The above table has three columns. From left to right, the first column is left-justified, the second column is centered, and the third column is right-justified.

Let's add a vertical line between each column. Add the following to the body of your LaTeX file:

```
\begin{tabular}{ l | c | r }
  Var1 & 2    & 3    \\
  Var2 & 5.0  & 6.0  \\
  Var3 & 8.00 & 9.00 \\
\end{tabular}
```

The above codes produce the following table.

Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Let's now add top rule and bottom rule. Add the following to the body of your LaTeX file:

```
\begin{tabular}{ l | c | r }
\toprule
  Var1 & 2    & 3    \\
  Var2 & 5.0  & 6.0  \\
  Var3 & 8.00 & 9.00 \\
\bottomrule
\end{tabular}
```

The above codes produce the following table.

Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Let's name each column. Add the following to the body of your LaTeX file:

```
\begin{tabular}{ l | c | r }
\toprule
Variables & Column Two & Column Three \\
  Var1 & 2    & 3    \\
  Var2 & 5.0  & 6.0  \\
  Var3 & 8.00 & 9.00 \\
\bottomrule
\end{tabular}
```

The above codes produce the following table.

Variables	Column Two	Column Three
Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Now, let's add a midrule and table notes. Add the following to the body of your LaTeX file:

```
\begin{tabular}{ l | c | r }
\toprule
Variables & Column Two & Column Three \\
\midrule
  Var1 & 2    & 3    \\
  Var2 & 5.0  & 6.0  \\
  Var3 & 8.00 & 9.00 \\
\bottomrule
```

```
\end{tabular}
\begin{tablenotes}
    \item Note: This is the notes for this table.
\end{tablenotes}
```

The above codes produce the following table.

Variables	Column Two	Column Three
Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Note: This is the notes for this table.

Typically, if we have a table like this, we would want it to look more like this, i.e., add title; remove the vertical lines; align columns by decimals, etc.:

Table 2: Title of this Table

Variables	Column Two	Column Three
Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Note: This is the notes for this table. This is really important.

Below are the codes to produce this table. The codes are more complex, but the table looks much better. The codes are indented to better display the structure of codes for the table.

```
\begin{table}[h]
\caption{Title of this Table}
\label{exampleTable}
\centering
    \begin{threeparttable}
        \begin{tabular}{ l . . }
        \toprule
        {Variables} & \multicolumn{1}{c}{Column Two} & \multicolumn{1}{c}{Column Thre
        \midrule
        Var1 & 2    & 3     \\
        Var2 & 5.0  & 6.0   \\
        Var3 & 8.00 & 9.00  \\
        \bottomrule
        \end{tabular}
        \begin{tablenotes}[para, flushleft]
            Note: This is the notes for this table. This is really important.
        \end{tablenotes}
     \end{threeparttable}
\end{table}
```

Eventually, we will cover what the many components in the above codes mean and do. For the moment, however, we will transition to importing images into LaTeX.

9 Important Images

Importing images into LaTeX is fairly simple. The hard part is getting your images into the right format. The discussions in this section will focus on using the `graphicx` package to assist in importing images. Combined with the default settings in LaTeX, the `graphicx` package will allow us to import images saved in the four most commonly used formats for images: (1) JEPG (.jpg); (2) PNG (.png); EPS (.eps); and (4) PDF (.pdf). Add the following to the preamble:

```
\usepackage{graphicx}
```

The syntax for using `graphicx` is:

```
\includegraphics[optional attributes]{image name}
```

The arguments in square brackets are optional, whereas curly braces are compulsory. The variety of possible attributes can be fairly large, so we will cover the most useful:

```
width=xx        Specifiy the preferred width of the imported image to xx
height=xx       Specify the preferred height of the imported image to xx
                    *NOTE: Only specifying either width or height will scale the image
                     whilst maintaining the aspect ratio.
scale=xx        Scales the image by the desired scale factor,
                e.g., 0.5 to reduce by half, or 2 to double
angle=xx        This option can rotate the image by xx degrees (counter-clockwise)
trim=l b r t    This option will crop the imported image by:
                    l from the left,
                    b from the bottom,
                    r from the right, and
                    t from the top.
                        Where l, b, r, and t are lengths.
clip            For the trim option to work, you must set clip=true
```

Note: To use more than one option at a time, simply separate with a comma.

Now, let's see the graphics in action. Type the following in your LaTeX file and compile (F1) to see the results.

```
\includegraphics{chick.jpg}
```

The above command imports the image, however, it is very large. So, let's scale it down.

Type the following into the LaTeX file and compile.

```
\includegraphics[scale=.5]{chick.jpg}
```

This has now reduced by half. What if we want to be more specific and give the actual lengths of the image dimensions? Type the following into the LaTeX file and compile.

```
\includegraphics[width=2.5cm]{chick.jpg}
```

We could also rotate the image. Type the following into the LaTeX file and compile.

```
\includegraphics[scale=.5, angle=180]{chick.jpg}
```

Last, here's an example of how to crop an image should you wish to focus in one particular area of interest. Type the following into the LaTeX file and compile.

```
\includegraphics[trim= 10mm 80mm 20mm 5mm, clip, scale=.5]{chick.jpg}
```

Note the presence of clip, as the trim operation will not work without it. This concludes just about all you need to know about how to import images in a LaTeX document. As mentioned before, the hardest part, if necessary, is to convert your images into a format that LaTeX likes.

There is one major topic missing that we have not cover, and that has to do with making that image a figure. In addition to the image itself, you will want a caption, and maybe some way to cross-reference the image. This is what we will turn to next.

10 Floats, Figures, and Captions

Previously, we learned how to import graphics. However, just having a picture stuck in-between paragraphs does not look professional. Minimally, we would want to add captions, and possibly the ability to cross-reference. What we need is a way to define *figures*. It would also be good if LaTeX could apply similar principles to when it arranges text to look its best, to arranging pictures too. This is where *floats* come into play.

10.1 Floats

Floating elements refer to anything within a document that cannot be broken over a page. This basically boils down to tables and figures.[5] They require special treatment and the concept of *floating* was the solution for a robust method for dealing with such elements, whilst keeping the document presentation as "nice" as possible.

The most common problem is that there is not enough space on the remainder of a given page to fit the specified figure. To overcome this, LaTeX will float this over to the next page, while filling the current page with body text. Contrast this to what happens in a word processor, for example. If an image is added but is too large to fit on the current page, it will position it on the next page, but, will leave a large gap, instead of rearranging subsequent text to fill the space. It requires a lot of manual tweaking to rectify, whereas LaTeX takes care of all of this automatically.

10.2 Figures

To create a figure, you must use the *figure* environment.

```
\begin{figure}[placement specifier]
...figure content...
\end{figure}
```

[5] We could have tables that stretches over multiple pages; however, this is slightly too complex for this workshop.

In the previous section, we discussed how floats are used to allow LaTeX to handle figures, while maintaining the best possible presentation. However, there may be times when you disagree, and a typical example is with its positioning of figures. The *placement specifier* parameter exists as a compromise, and its purpose is to give the author a greater degree of control over where certain floats are placed.

```
Specifier              Permission
h          Place the float here, i.e., 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
!          Override internal parameters LaTeX uses for determing good float positions
```

What you do with these *placement permissions* is to list which of the options that you wish to make available to LaTeX. These are simply possibilities, and LaTeX will decide when typesetting your document which of your supplied specifiers it thinks is best.

10.3 Tables

Although tables have already been covered, it was only the internal syntax that was discussed. The `tabular` environment that was used to construct the tables is not a float by default. Therefore, for tables you wish to float, wrap the `tabular` environment within a `table` environment, like this:

```
\begin{table}
\begin{tabular}{...}
... table data ...
\end{tabular}
\end{table}
```

You may not see the reasons here, but such distinctions are necessary, because you may not want all tables to be treated as a float.

10.4 Captions

It is always good practice to add a caption to any figure or table. Fortunately, this is very simple in LaTeX. All you need to do is use the `\caption{text}` command within the float environment. Because of how LaTeX deals sensibly with logical structure, it will automatically keep track of the numbering of figures, so you do not need to include this within the caption text.

The location of the caption is traditionally underneath the float. However, it is up to you to therefore insert the caption command after the actual contents of the float (but still within the environment). If you place it before, then the caption will appear above the float. To see examples, add the following to your LaTeX file and compile to see the results:

```
\begin{figure}
  \caption{A picture of a chick - title at the top}
  \begin{center}
    \includegraphics[scale=.5]{chick.jpg}
  \end{center}
\end{figure}

\begin{figure}
  \begin{center}
    \includegraphics[scale=.5]{chick.jpg}
  \end{center}
  \caption{A picture of a chick - title on the bottom}
\end{figure}
```

Figure 1: A picture of a chick - title at the top

A figure with the title at the top.

Figure 2: A picture of a chick - title on the bottom

Now the title of the figure is on the bottom.

10.5 Labels and Cross Reference

Now is a good opportunity to talk about labels and cross-referencing. Their purpose within LaTeX is to act as a marker, which can then be referenced to at any point within your document. It is very common to refer to each of your figures within the body of text. However, you don't want to be keeping track of the numbers, so you can use a label instead, and let LaTeX replace it with the correct figure number.

To add a label, you must embed the following command: `\label{marker}`. To then reference a label: `\ref{marker}`. Alternatively, to get a page reference: `\pageref{marker}`. See an example, consider the following codes:

```
\begin{figure}[h]
  \caption{A picture of a chick - title at the top}
  \label{chickImage}
  \begin{center}
    \includegraphics[scale=.3]{chick.jpg}
  \end{center}
\end{figure}


Figure~\ref{chickImage} shows a picture of a chick.
```

When a label is declared within a float environment, the `\ref` will return the respective fig/table number (although, it must occur after the caption). When declared outside, it will give the section number.

The tilde (`~`) in the above example is a special symbol within LaTeX. It represents a non-breaking space. It is useful here because it keeps 'Figure' and what ever number it `\ref` refers to as a single unit, and won't get split over lines or pages.

Figure 3: A picture of a chick - title at the top

Figure 3 shows a picture of a chick.

11 Mathematics

Typesetting mathematics is one of LaTeX's greatest strengths. However, since this is only an introductory workshop, we will focus on just the basics. At the end of this section, you should be able to do almost everything you typically would need, though this would be far, far short of LaTeX's full mathematical capabilities.

11.1 Math Environments

LaTeX needs to know beforehand that the subsequent text does in fact contain mathematical elements. This is because LaTeX typesets maths notation differently than normal text. Therefore, special environments have been declared for this purpose. They can be distinguished into two categories depending on how they are presented:

- *text* - inline math, that is, within the body of text
- *displayed* - math that are separate from the main text

As maths require special environments, there are naturally the appropriate environment names you can use in the standard way.

11.1.1 Inline Math

As for inline math, use `$` sign to begin and another `$` sign to end the inline math environment.

11.1.2 Align Environment

In addition to inline math, LaTeX also allows you to offset equations in special environments. One of the best is the `align` environment from the `amsmath` package. To use, add `\usepackage{amsmath}` to the preamble.

An example of an equation using the `align` environment is:

```
\begin{align}
LHS = RHS
```

```
\end{align}
```

which produces

$$LHS = RHS \tag{1}$$

We can ask LaTeX to suppress the equation number. To see an example, add the following and compile:

```
\begin{align}
LHS = RHS \nonumber
\end{align}
```

which produces:

$$LHS = RHS$$

Note that this is the exact same equation as before, except now LaTeX does not number the equation.

If we have multiple equations, we can ask LaTeX to align the equations by the equal sign:

```
\begin{align}
f(x) &= (x+a)(c+b) \nonumber \\
     &= xc + xb + ac + ab
\end{align}
```

which produces:

$$\begin{aligned} f(x) &= (x+a)(c+b) \\ &= xc + xb + ac + ab \end{aligned} \tag{2}$$

Note that we used the `&` symbol to tell LaTeX where to align, and `\\` for start writing on a new line. We also suppressed the equation number for the first equation.

11.2 Symbols

As you know, mathematics has lots and lots of symbols. If there is one aspect of maths that is difficult in LaTeX, it is trying to remember how to produce them, though LaTeX's original creator tried to keep the symbol codes as straightforward as possible.

For example, the code `\alpha` will create the lower case Greek symbol α, and the code `\beta` will create the lower case Greek symbol β. Re-producing here a complete list of math symbols in LaTeX is simply not feasible. However, a web search for the correct LaTeX code for the particular symbol you need will take all of one second.

11.3 Fractions

To create a fraction, you must use the `\frac{numerator}{denominator}` command. You can also embed fractions within fractions, as shown in the examples below:

```
\frac{x+y}{y-z}
```

$$\frac{x+y}{y-z}$$

```
\frac{\frac{1}{x}+\frac{1}{y}}{y-z}
```

$$\frac{\frac{1}{x}+\frac{1}{y}}{y-z}$$

11.4 Powers and Indices

Powers and indices are mathematically equivalent to superscripts and subscripts in normal text mode. The carat (`^`) character is used to raise something, and the underscore (`_`) is for lowering. How to use them is best shown by example:

```
x^n
```

x^n

```
x^{2n}
```

x^{2n}

```
n_i
```

n_i

```
n_{ij}
```

n_{ij}

Note: if more than one character is to be raised (or lowered) then you must group them using the curly braces .

Also, if you need to assign both a power and an index to the same entity, then that is achieved like this: `x^{2i}_{3j}` (or `x_{3j}^{2i}`, order is not significant).

11.5 Adding Text to Equations

What if we want to include some text in, say, an equation? For instance, how can write the following in LaTeX?

$$50 \text{ apples} \times 100 \text{ apples} = \text{lots of apples}$$

Let's first try what we already know and see if it works. Add the following into your LaTeX file and compile.

```
\begin{align}
50 apples X 100 apples = lots of apples
\end{align}
```

The above codes produce:

$$50 apples X 100 apples = lots of apples \tag{3}$$

Doesn't look pretty, does it? How about this?

$$50 \text{ apples} \times 100 \text{ apples} = \text{lots of apples} \tag{4}$$

The second version is definitely better. Let's look at the codes that produced this equation:

```
50 \textnormal{ apples} \times 100 \textnormal{ apples}
     = \textnormal{lots of apples}
```

Note that we used the `\textnormal{}` command to input text inside the math environment, and the `\times` command to tell LaTeX to display the multiplication symbol.

12 Making Presentation Slides

One advantage to learning LaTeX is that making presentation slides is easy, especially if we're already using LaTeX for writing. We will use Beamer; Beamer is a LaTeX document class for creating slides for presentations. There are just a few things to remember. Consider the following codes:

```
\documentclass{beamer}
\usetheme{default}
\begin{document}

\begin{frame}{A sample slide}

A displayed formula:

\[
  \int_{-\infty}^\infty e^{-x^2} \, dx = \sqrt{\pi}
\]

An itemized list:

\begin{itemize}
  \item itemized item 1
  \item itemized item 2
  \item itemized item 3
\end{itemize}

\begin{theorem}
  In a right triangle, the square of hypotenuse equals
  the sum of squares of two other sides.
\end{theorem}

\end{frame}

\end{document}
```

and the output slide.

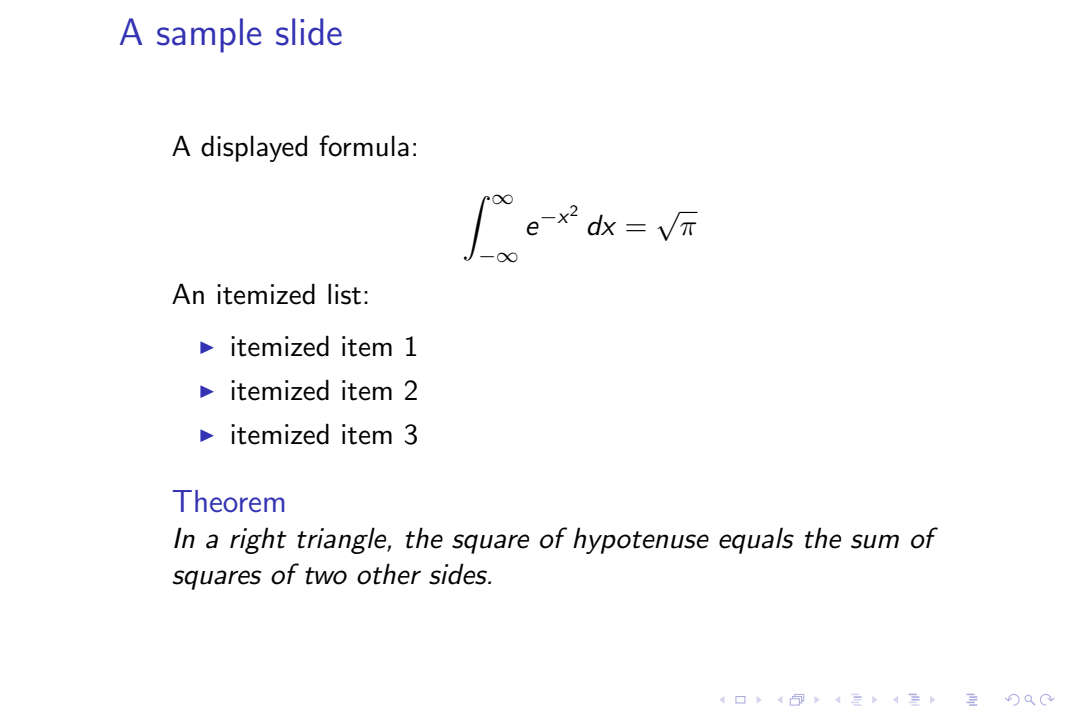

Figure 4: Example Beamer Slide

A lot is going one; let's take it line by line. In Line 1, we set the document class as `beamer`. In Line 2, we tell LaTeX to use the default theme. There are a variety of themes that you can choose from; some research institutes, departments, or universities even have their own theme. In Line 3, we begin the document. Lines 5-26 is one slide. Each beamer slide is called a "frame." Note that in Line 5 we began a new frame and tell LaTeX that the slide's title is: "A sample slide." Lines 7-24 are the content of this particular slide, which includes: (1) a formula; (2) an itemized list; and (3) a theorem. Last, we end the frame in Line 26, and the document in Line 28.

12.1 Overlays

Beamer has a very rich selection of methods for creating overlays. The simplest way to create a set of overlaid slides is to use `\pause` command everywhere you want a break between overlays. Try adding a few `\pause` command to the previous example.

```
\begin{frame}{A sample slide}

\pause
A displayed formula:

\[
  \int_{-\infty}^\infty e^{-x^2} \, dx = \sqrt{\pi}
\]

\pause
An itemized list:

\begin{itemize}
  \item itemized item 1
  \item itemized item 2
  \item itemized item 3
\end{itemize}

\pause
\begin{theorem}
  In a right triangle, the square of hypotenuse equals
  the sum of squares of two other sides.
\end{theorem}

\end{frame}
```

12.2 Transparent Overlays

By default, future overlays are invisible. We can set Beamer settings to let us see a transparent or dynamic form of overlays yet to come. To set setting to either transparent or dynamic, add the following to the preamble:
`\setbeamercovered{transparent}` (transparent), or
`\setbeamercovered{dynamic}` (dynamic). Let's try them one at a time.

12.2.1 Transparent (default = 15% opaqueness)

```
\documentclass{beamer}
\usetheme{default}

\setbeamercovered{transparent}


\begin{document}

\begin{frame}{A sample slide}

\pause
A displayed formula:

\[
  \int_{-\infty}^\infty e^{-x^2} \, dx = \sqrt{\pi}
\]

\pause
An itemized list:

\begin{itemize}
  \item itemized item 1
  \item itemized item 2
  \item itemized item 3
\end{itemize}

\pause
\begin{theorem}
  In a right triangle, the square of hypotenuse equals
  the sum of squares of two other sides.
\end{theorem}

\end{frame}
```

```
\end{document}
```

12.2.2 Dynamic

```
\documentclass{beamer}
\usetheme{default}

\setbeamercovered{dynamic}


\begin{document}

\begin{frame}{A sample slide}

\pause
A displayed formula:

\[
  \int_{-\infty}^\infty e^{-x^2} \, dx = \sqrt{\pi}
\]

\pause
An itemized list:

\begin{itemize}
  \item itemized item 1
  \item itemized item 2
  \item itemized item 3
\end{itemize}

\pause
\begin{theorem}
  In a right triangle, the square of hypotenuse equals
  the sum of squares of two other sides.
\end{theorem}

\end{frame}

\end{document}
```

12.3 Why LaTeX and Beamer

Perhaps you already realized why making presentation slides using LaTeX and beamer is great if you're already writing using LaTeX. For instance, suppose we have the following table from our research paper that we would like to include in a research presentation. This is the same table from earlier.

Table 3: Title of this Table

Variables	Column Two	Column Three
Var1	2	3
Var2	5.0	6.0
Var3	8.00	9.00

Note: This is the notes for this table. This is really important.

To put this table in a beamer slide, all we have to do is to copy and paste the codes into a frame in beamer. For example:

```
\documentclass{beamer}
\usetheme{default}
\begin{document}

\begin{frame}{Our Results Table}

\begin{table}[h]
\caption{Title of this Table}
\label{exampleTable}
\centering
    \begin{threeparttable}
     \begin{tabular}{ l . . }
     \toprule
     {Variables} & \multicolumn{1}{c}{Column Two} & \multicolumn{1}{c}{Column Three}
     \midrule
     Var1 & 2    & 3     \\
     Var2 & 5.0  & 6.0   \\
```

```
      Var3 & 8.00 & 9.00  \\
      \bottomrule
      \end{tabular}
      \begin{tablenotes}[para, flushleft]
        Note: This is the notes for this table. This is really important.
      \end{tablenotes}
      \end{threeparttable}
\end{table}

\end{frame}

\end{document}
```

which gives us:

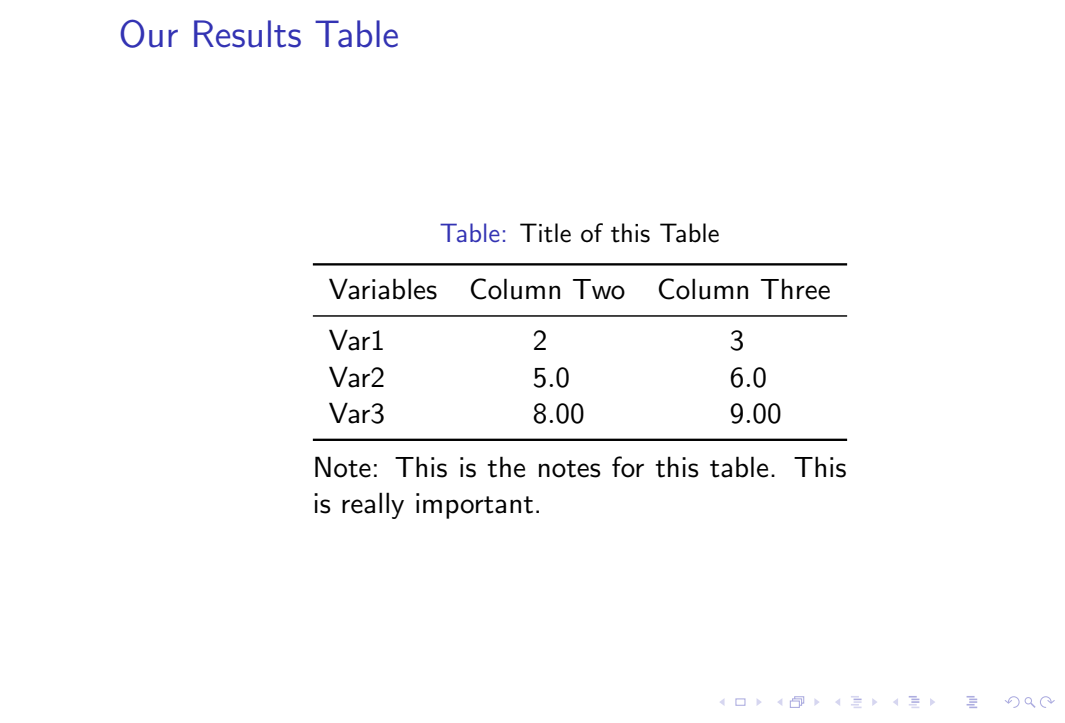

Figure 5: Example of Table in Beamer

12.4 Add Title Slide

The following codes:

```
\documentclass{beamer}
\usetheme{default}


\title{Title of Presentation}
\subtitle{If You Want One}
\author[shortName]{author 1\inst{1}}
\institute[shortInst]{\inst{1} University of the North}
\date[shortDate]{Date}


\begin{document}

\begin{frame}
\titlepage
\end{frame}

\end{document}
```

produces:

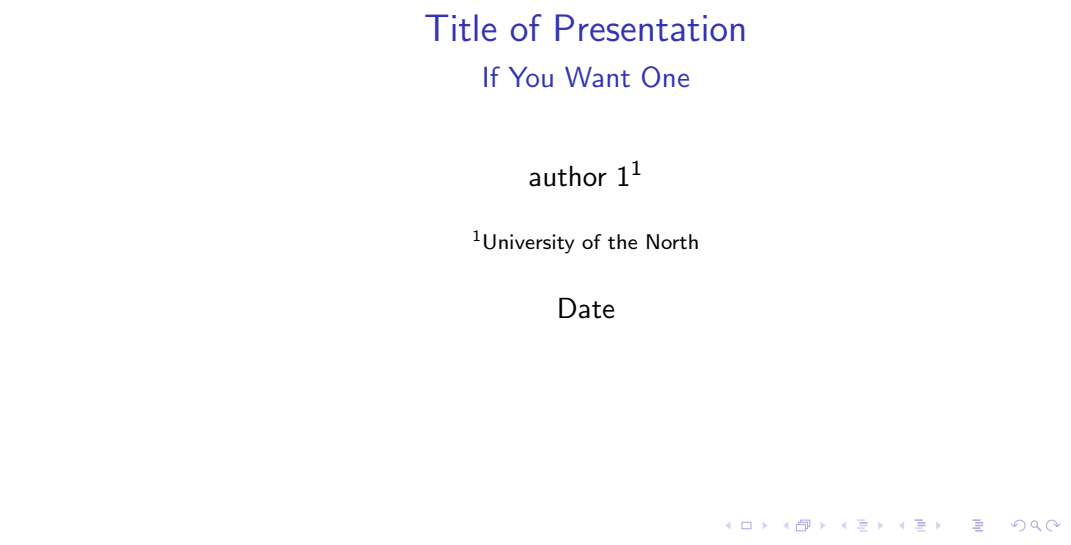

Figure 6: Example Title Page

12.4.1 Multiple Authors

What if you're working with a co-author who's at another institution? We can easily adapt the previous codes to include the second author's information.

```
\documentclass{beamer}
\usetheme{default}


\title{Title of Presentation}
\subtitle{If You Want One}
\author[shortName]{author 1\inst{1} \and
                   author 2\inst{2}}
\institute[shortInst]{\inst{1} University of the North \and
                      \inst{2} University of the East}
```

```
\date[shortDate]{Date}


\begin{document}

\begin{frame}
\titlepage
\end{frame}

\end{document}
```

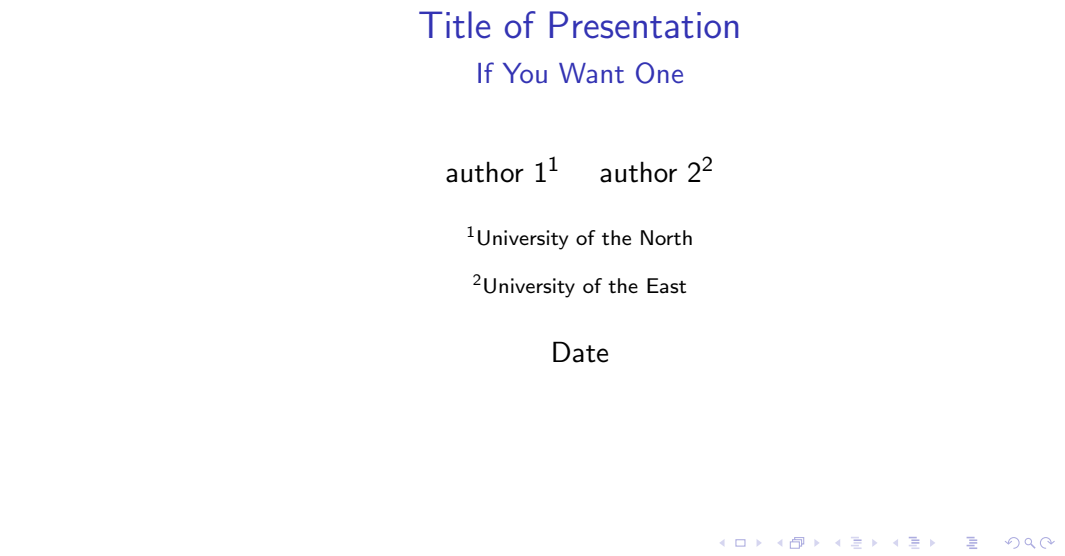

Figure 7: Example Title Page - Multiple Authors