Getting Started

LaTeX, which is pronounced Lah-tech or Lay-tech, is a document preparation system for high-quality typesetting. It is most often used for medium-to-large technical or scientific documents but it can be used for almost any form of publishing. Some of the perks of using LaTeXare:

- Typesetting journal articles, technical reports, books, and slide presentations
- Control over large documents containing sectioning, cross-references, tables and figures
- Typesetting of complex mathematical formulas
- Advanced typesetting of mathematics with AMS-LaTeX
- Automatic generation of bibliographies and indexes
- Multi-lingual typesetting
- Inclusion of artwork, and process or spot color
- Using PostScript or Metafont fonts

We will be making use of the **free** LaTeX software *ShareLateX*. You have already created an account, so enter your username (or email) and password to log-in. This brings us to the projects page where all your LaTeXprojects will be stored. To create a new project, click the red new project button, you now have a number of options:

- you can create a blank project,
- see an example project,
- upload an existing LaTeXproject or
- start building from one of their helpful templates.

We are going to select a **blank project**. Now name your project and hit **create**. This brings us to the editor.

- The **left-hand panel** is where you manage all the files associated with this project.
- The **center panel** is the code editor where you write the document.
- The **right-hand panel** is the preview area.

Basic Layout: Preamble and Document

You will notice that *ShareLaTeX* has already given you some code by default. Let's go through each of these lines and discuss what they do. Each statement starting with a backslash is a LaTeXcommand, *ShareLaTeX* has nicely color coded them in blue to make them easy to recognize.

Preamble

- The `documentclass` command tells LaTeXwhat type of document you want to produce, popular options include article, book, report, slides and letter.
- The command (`usepackage[utf8]inputenc`) on the second line sets LaTeXup to be able to use accented characters.
- The `title` command tells LaTeXthe title of your document. **Change this by placing a new title in the curly brackets.**
- The next two commands give LaTeXinformation regarding the document's author and writing date **Update these as well**.

Document

- The `begin` command on line 8 tells LaTeXwe want to start writing the document.
- The `maketitle` command puts a title into the document using the information we entered in the preamble.
- The `section` command is used to help you break up your document into logical parts. This first section has been named Introduction and it was labeled with a number 1 (since it's the first section).
- Line 14 is the `end` command which tells LaTeXwe have finished writing the document. **Every time you use a `begin` command you must also remember to use an `end` command later!**

We can now see what our document looks like by clicking the blue recompile button. Hovering your mouse over the left hand upper corner of the preview panel gives you options to zoom in and out on your document to enable you to check everything is as expected. Double clicking on specific lines in the document will highlight the line(s) in the LaTeXcode that generated that line in the document.

So there you have it, your first LaTeXdocument!

Packages

LaTeXhas many built in functions, but often times it is handy to use additional functions housed within LaTeX*packages*. To import a package, you simple add the `usepackage` directive to the preamble of your document.

Three packages I often find myself using are, `amsmath, graphicx,` and `multicol`. Let's go ahead and source these into our LaTeXdocument, and we will see how to use them shortly.

Typesetting Math

There are two main modes of typesetting math in LaTeX, embedding the math directly by *encapsulating* it in *dollar signs*, or using a predefined *math environment*.

Inline Math

The use of `$` for inline math environments is common, but it is difficult for the computer to track a missing `$`. Therefore, I advocate for the use of the *displaymath* environment, using

```
\[\]
```

for inline math environments.

Example: This formula

```
$f(x) = x^2$
```

is an example. Or we could use,

```
\[f(x) = x^2\]
```

instead.

Equations and Align

Arguably the most useful *math environments* are the *equation environment* for single equations and the *align environment* for multiple equations.

```
\begin{equation}
  1 + 2 = 3
\end{equation}

\begin{equation*}
  1 = 3 - 2
\end{equation*}

\begin{align*}
  1 + 2 &= 3\\
  1 &= 3 - 2
\end{align*}

\begin{align*}
  f(x) &= x^2\\
  g(x) &= \frac{1}{x}\\
  F(x) &= \int^a_b \frac{1}{3}x^3
\end{align*}

\begin{equation*}
\frac{1}{\sqrt{x}}
\end{equation*}
```

Tables and Matrices

To insert tables in LaTeXwe use the *tabular environment*. Here are the steps to assembling a table:

- add `begin{tabular}` and `end{tabular}` commands to your document
- specify the column format by inserting `{}` to the right of the `begin{tabular}` command,
  - a lower case r is right justified,
  - a lower case c is center justified, and
  - a lower case l is left justified.
- add lines between columns using the `|` (bar) symbol between the letters
- when entering the elements of the table,
  - start each row on a new line,
  - separate column entries with an ampersand `&`, and
  - finish each row (except the last) with a double backslash (\\).

```
\begin{tabular}{l | c | c | C | c | l}
Participant & Swim & Cycle & Run & Total & Additional Comments \\
John & 13:00 & 24:15 & 18:34 & 55:53 \\
Norman & 8:00 & 22:45 & 23:02 & 53:47 & Winner \\
Alex & 14:00 & 28:00 & n/a & n/a & Injured \\
Sarah & 9:22 & 21:00 & 24:03 & 54:35
\end{tabular}
```

You have assembled your first LaTeXtable! Often, just like with images, we want more control over positioning and we may want to add a caption and label. To do this we use the *table environment* (which is similar to the environment we will see for images). To put our existing table in this environment, we simply enclose the above code in `begin{table}` and `end{table}` commands.

```
\begin{table}[h]
\begin{tabular}{l | c | c | C | c | l}
Participant & Swim & Cycle & Run & Total & Additional Comments \\
\hline
John & 13:00 & 24:15 & 18:34 & 55:53 \\
Norman & 8:00 & 22:45 & 23:02 & 53:47 & Winner \\
Alex & 14:00 & 28:00 & n/a & n/a & Injured \\
Sarah & 9:22 & 21:00 & 24:03 & 54:35
\end{tabular}
\caption{Triathlon Results}
\label{tab:tri}
\end{table}
```

Matrices

Matrices use syntax similar to tables. This is an example of using functions within the `amsmath` package. A matrix needs to be in a math environment, where we could use the equation (or inline) environments from before. Let's practice with the inline *displaymath* environment. With matrices we **don't** need to declare how many columns we use, we can simply start adding entries.

Like tables, we enter each row on a new line, separating column entries with an ampersand, and separate rows using a double backslash.

To change the brackets surrounding the matrix, we change the environment. The *pmatrix* environment uses parenthesis, the *bmatrix* uses square brackets, *Bmatrix* with a capital B uses curly brackets, *vmatrix* uses vertical lines and *Vmatrix* with a capital V uses double vertical lines.

Let's create the matrix displayed below.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Got it? Here's a challenge!

$$\begin{pmatrix} a & b & & \\ c & d & & 0 \\ & & a & b \\ 0 & & c & d \end{pmatrix}$$

Figures

Often (for me) or from time to time, it is necessary to add pictures to your documents. Using LaTeXall pictures will be automatically indexed and tagged with successive numbers. To import images we will use function (environment) from the `graphicx` package.

```
\begin{figure}
  \includegraphics[width = \linewidth]{boat.jpg}
  \caption{A boat.}
  \label{fig:boat1}
\end{figure}

Figure \ref{fig:boat1} shows a boat.
```

The *figure environment* takes care of the numbering and positioning of the image for you. To include a figure you need to

- use the `includegraphics{}` command,
- insert the name (or path) to your image in the parenthesis
- (optional) insert the image size in the brackets ([scale = 0.5] will shrink it in half)
- (optional) a caption for the figure
- (optional) a label for the image (if you are to reference it later).

As I mentioned before the brackets contain the path to the image. In this case the image is stored in the **same** directory as my .tex file, so I simply put boat.jpg here to include it. By default LaTeXlooks for .jpg files, so omitting the suffix would also work. For large documents, you probably want to store image files in a different folder, say we created a folder images, then we would simply write images/boat.jpg into the braces.

Setting the float by adding [h!] behind the figure environment `begin` tag will force the figure to be shown at the location in the document. Possible values are:

- h (here) - same location
- t (top) - top of page
- b (bottom) - bottom of page
- p (page) - on an extra page
- ! (override) - will force the specified location

I typically use the [h!] option. The float package (`usepackage{float}`) allows to set the option to [H], which is even stricter than [h!].

Footnotes

One of the benifits of using LaTeXis how easy footnotes can be added and referred to. How to use footnotes? LaTeXoffers the `footnote` command and referencing works using the `label` and `ref` commands.

```
This is some example text \footnote{\label{myfootnote} Hello footnote}.
```

After compilation you will see the footnote appearing on the bottom of your page. It's imperative, that the label is *contained* within the footnote itself, otherwise the label will refer to the section (or subsection). Footnotes are numbered automatically. If you want to refer to them later on, you can use the `ref` command as follows:

```
I'm referring to footnote \ref{myfootnote}.
```

Table of Contents

Generating a table of contents can be done with a few simple commands. LaTeXwill use the section headings to create the table of contents and there are commands to create a list of figures and a list of tables as well. To create a table of contents, you add a `tableofcontents` command after `begin{document}`. The table of contents will automatically beplaced on its own page, at the beginning of the document.

You can also add a table of content of the list of figures and the tables that you have in your document. This I typically place in the `appendix` environment, like so

```
\begin{appendix}
  \listoffigures
  \listoftables
\end{appendix}
```

Bibliography

To make a bibliography in LaTeXyou have many different options. You can use the `bibliography` environment **in** the document, or you can use either bibtex or biblatex (both external programs) to create the bibliography.

For my first years using LaTeXI exclusively made my bibliography within my LaTeXdocument. Here's an example of a bibliography of mine:

```
\begin{thebibliography}{99}
\bibitem{rosenrub1}
{\footnotesize
Rosenbaum, P. R. \& Rubin, D. B. (1983).
The central role of the propensity score in observational studies for causal effects.
\textit{Biometrika}, 70(1), 41-55.}

\bibitem{rosenrub2}
{\footnotesize Rosenbaum, P. R. \& Rubin, D. B. (1984).
Reducing Bias in Observational Studies Using Sub-classification the Propensity Score.
\textit{Journal of the American Statistical Association}, 79(387), 516-
524.}

\bibitem{gelman}
{\footnotesize
Gelman, A., Hill, J.(2007). Data Analysis Using Regression and Multilevel/Hierarchical
Models. Cambridge University Press. }

\end{thebibliography}
```

- To start this type of bibliography you use the `begin{thebibliography}` command.
- The argument directly next to this command (`{99}`) loosely tells LaTeXhow much space it should allocate for your bibliography.
- I typically default to using `{99}`, unless I know my bibliography is going to be short and then I'll use `{9}`.
- The `bibitem` command begins a new bibliography entry.
  - In the `{}` after `bibitem` you place the name that you will use when referencing this bibliography item in the document,
  - Following the `bibitem` command, you place the reference, where you are required to format it yourself.
- Once you have completed your bibliography, you are required to close with an `end{thebibliography}` command.

Using a *.bib* file requires you to save your bibliography entries externally and then import them into LaTeX.

Much literature exists on how to detect plagiarism in the CS classroom [1, 2, 3, 4, 5, 6, 7] and plagiarism education that teach about, and to deter, certain forms of plagiarism [8, 9, 10, 11]. However, largely missing from the literature is how one defines plagiarism, and how to convey that definition clearly across the CS curriculum (i.e., in all classes, not just CS ethics classes).

References

[1] S. Niezgoda and T. P. Way, "Snitch: A software tool for detecting cut and paste plagiarism," in *ACM SIGCSE Bulletin*, vol. 38, pp. 51–55, ACM, 2006.

[2] S. Burrows, S. M. Tahaghoghi, and J. Zobel, "Efficient plagiarism detection for large code repositories," *Software-Practice and Experience*, vol. 37, no. 2, pp. 151–176, 2007.

[3] C. Arwin and S. M. Tahaghoghi, "Plagiarism detection across programming languages," in *Proceedings of the 29th Australasian Computer Science Conference-Volume 48*, pp. 277–286, Australian Computer Society, Inc., 2006.

[4] Z. Mei and L. Dongsheng, "An xml plagiarism detection model for c program," in *2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, vol. 1, pp. 460–464, IEEE, 2010.

[5] C. Zhao, H. Yan, and C. Song, "A system for automatic assessment and plagiarism detection of student programs," in *E-Business and E-Government (ICEE), 2010 International Conference on*, pp. 3619–3624, IEEE, 2010.

[6] F. Hattingh, A. A. Buitendag, and J. S. Van Der Walt, "Presenting an alternative source code plagiarism detection framework for improving the teaching and learning of programming," *Journal of Information Technology Education*, vol. 12, pp. 45–58, 2013.

[7] L. Zhang, D. Liu, Y. Li, and M. Zhong, "Ast-based plagiarism detection method," in *Internet of Things*, pp. 611–618, Springer, 2012.

[8] R. Barrett and J. Malcolm, "Embedding plagiarism education in the assessment process," *International Journal for Educational Integrity*, vol. 2, no. 1, pp. 38–45, 2006.

[9] C. L. Aasheim, P. S. Rutner, L. Li, and S. R. Williams, "Plagiarism and programming: A survey of student attitudes," *Journal of Information Systems Education*, vol. 23, no. 3, pp. 297–313, 2012.

[10] H. Rudolph and P. Radcliffe, "Anti-plagiarism: A weapon of mass self delusion?," in *Proceedings of the 2007 AAEE Conference*, 2007.

[11] T. Le, A. Carbone, J. Sheard, M. Schuhmacher, M. de Raath, and C. Johnson, "Educating computer programming students about plagiarism through use of a code similarity detection tool," in *Learning and Teaching in Computing and Engineering (LaTiCE), 2013*, pp. 98–105, IEEE, 2013.