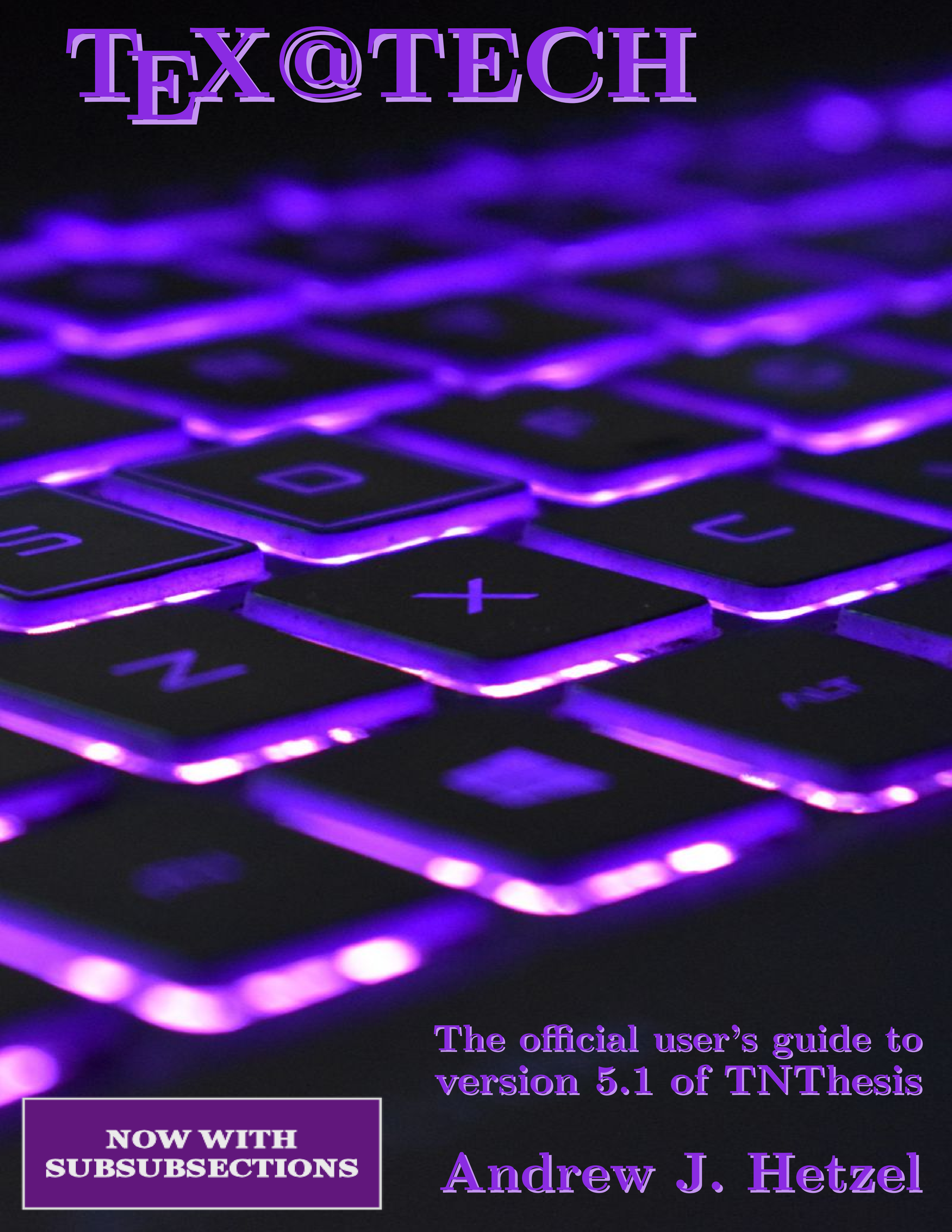
TeX@TECH



The official user's guide to version 5.1 of TNThesis

NOW WITH SUBSUBSUBSECTIONS

Andrew J. Hetzel

TEX@TECH

The official user's guide to version 5.1 of TNThesis

Andrew J. Hetzel, PhD

Tennessee Tech University

 Published by Aleph Naught Publishing House

ISBN 2-718-28182-8

To all my students, past and present, who push me to be my best every day.

Preface

Version 5

THIS past year has brought some wonderful opportunities to interact with students at Tech who were using LaTeX to produce their theses and dissertations. I truly enjoy the process of developing solutions to their problems (although I will admit that it created a bit more stress than I wished for during my vacation in Charleston). Needs are as unique as both the students themselves and the departments under which they are receiving their degrees. Thankfully, LaTeX is always up for the task.

More than any other version before it, version 5 of the class file is chiefly created from the ideas that these discussions engendered. I would have never even considered the changes that I did had it not been for these students, and so to them I am ever grateful. Since I am a person who believes so strongly in the power of great typographical design, it is important to me that the class file remains both relevant and responsive. Furthermore, I want this user's manual to be a valuable resource—a task that seems to be almost Sisyphean in the day in age of online crowd-sourced solutions. However, I know that the complexities of LaTeX do not always lend themselves to mere "copy and paste" answers, and so I want to believe that this manual (and others like it in the TeX community) will continue to have some measure of worth.

One issue of which I have become aware is the exclusive use of the standard `figure` and `table` environments, in spite of the fact that for several versions now the class file has featured far more user-friendly macros that automatically create the proper placement for the floats and their captions. Since I do not want to completely remove the ability to use these standard environment calls—I can foresee times when they may be necessary—I have built into this version several warnings to encourage the use of the corresponding macros. This way, the compilation is not halted, but hopefully the message is still delivered.

I realize now that I am in a losing battle with engineering professors who love their subsubsections almost as much as their outdated `eps` filetypes. So, in this version I have (begrudgingly) allowed for the creation of subsubsections with the promise that I will call it quits the moment I come across a serious request for an even finer level than this[1] . However, once again, I would strongly caution their use, particularly in the presence of figures and tables, as this can create undesirable page breaks. *Videte quid optes.*

A question this summer by Wu Yun inspired the novel ability to create as many additional lists as one would like. So, if you would like a list of acronyms, a list of abbreviations, *and* a list of symbols, version 5 has you covered.

Given how immense the `tikz` package is, I have moved away from autoloading it to loading it by means of a class option, so that works that don't require it will compile faster. In addition, the

[1] If you're curious, the next level down from a subsubsection is a paragraph...seriously.

class file will no longer autoload the `listings` package (used for the printing of computer code) to the arguably more popular `minted` package. Such a move has further allowed me to streamline the syntax for displaying computer code in a float.

Finally, version 5 represents a significant name change to both the class file and corresponding compilation file. The class file is now called `TNThesis.cls` and the file from which the work is compiled is now called `TNTmain.tex`. These name changes not only reflect recommended practice for LaTeX package and class filenames but also reflect the usage of the class file beyond just the mathematical community.

I wish each of you all the best in creating stunning theses and dissertations that you can be proud of for years to come. As always, if you have any questions or concerns, please do not hesitate to contact me.

Happy typesetting!

Andrew J. Hetzel
`ahetzel@tntech.edu`

August 11, 2022

Table of Contents

Lists

Table of Contents *vii*

Bangs, Bolts, & Bombs *xi*

Cumulus of Commands *xiii*

List of Figures *xvi*

List of Tables *xix*

Quick Information

I Basic Procedure *1*

II Examples of Math Into LaTeX *2*

III Legend for the User's Manual *7*

Becoming Familiar With LaTeX

1 In the Beginning... *9*
- 1.1 From Where Did It All Come? *9*
- 1.2 How Does It All Work? *11*

2 ...There Was TeX *15*
- 2.1 Setting Up MikTeX *15*
- 2.2 Taking It for a Test Spin *16*
- 2.3 When Good Code Goes Bad *21*
- 2.4 Resources *24*

3 Some Basic Tools *27*
- 3.1 Put Your Best Font Forward *27*
- 3.2 When You Just Need a Little Space *31*
- 3.3 Of Subscripts and Superscripts *34*
- 3.4 Going Greek (and a Little Hebrew) *36*

4 On Display *39*
- 4.1 One Singular Expression *39*
- 4.2 The Equation Environment *42*

4.3 References and Labels *43*
4.4 Size Matters *47*
4.4.1 `\left`, `\left`, `\left`, `\right`, `\left` *47*
4.4.2 `\Biggie` Size It *48*
4.5 Alignment *52*
4.5.1 The `align` Environment *52*
4.5.2 The `alignat` Environment *53*
4.5.3 The `aligned` Environment *53*
4.5.4 The `gather` Environment *54*
4.5.5 The `multline` Environment *55*
4.5.6 The `split` Environment *55*
5 Diagrams *57*
5.1 Simple Triangular Diagrams *58*
5.2 Arrow Options *60*
5.3 More Exotic Diagrams *64*
5.4 Graphs *67*
6 Theorems and Friends *73*
6.1 The Theorem Environment *73*
6.2 Attribution of a Theorem *76*
6.3 Make Your Own Macros at Home *77*
6.3.1 Operators *77*
6.3.2 General Macros *78*
6.3.3 Macro Functions *80*
7 Making a List, Checking It Twice *83*
7.1 Itemize *85*
7.2 Enumerate *91*
7.3 Description *95*
8 It Figures! *101*
8.1 Captivating Captions *101*
8.2 The `\Graphic` Macro *102*
8.3 More Complicated Figures *112*
8.3.1 Framed Figures *114*
8.3.2 Side-by-side Figures *114*
9 Tabling Issues *117*
9.1 The `Table` and `Tabular` Environments *117*
9.2 The `\Table` Macro *120*

9.3 A Few Basic Table Styles *124*
9.4 Taking Your Table to the Next Row *127*
10 Listing Your Listings *135*
10.1 Putting Down Roots *135*
10.2 The Listing Environment *137*
10.3 Very Small and Very Big Listings *141*

How to use `TNThesis`

11 Do It With Style *145*
11.1 Typographic Style in LaTeX *145*
11.2 Common Issues in Mathematical Writing *147*
11.3 Special Conventions in Mathematical Writing *151*
11.4 Considerations for Math Environments *153*
12 Front Matter Matters *155*
12.1 Packages Loaded by `TNThesis` *155*
12.2 Overview of `TNTmain` *156*
12.3 Front Matter: The Preliminary Pages *157*
12.3.1 The Title Page *158*
12.3.2 The Abstract Page *160*
12.3.3 The Copyright Page *162*
12.3.4 The Table of Contents *163*
12.3.5 The Approval Page *163*
12.3.6 The Dedication Page *165*
12.3.7 The Acknowledgments Page *166*
12.3.8 The List of Figures, List of Tables, and List of Listings *166*
12.3.9 An Additional List (or Three) *167*
13 We're Way Past Introductions *169*
13.1 Chapters *170*
13.2 Sections *171*
13.3 Subsections *173*
13.4 Subsubsections (If You Must) *174*
13.5 Putting It All Together *175*
14 Pulling Up the Rear *177*
14.1 Appendices *177*
14.2 The Bibliography *178*
14.2.1 Making a `bib` *178*

14.2.2 Oh, the Possibilities! *182*
14.2.3 Citations *188*
14.3 The Vita Page *190*
15 The Terror of Errors *191*
15.1 Error Messages *191*
15.2 Other Options for Dealing With Errors *194*

Appendices

A Which Editor Is Right for Me? *197*
A.1 LyX *199*
A.2 TeXnicCenter *200*
A.3 TeXstudio *201*
A.4 TeXworks *202*
A.5 WinEdt *203*
B APA 7 Style for Bibliographies *205*
C The `TNTmain` Program *209*
D The `TNThesis.cls` Code *211*
E History *249*

Reference Material

Glossary *257*
Analphabetic Index *325*
Alphabetic Index *325*
Epigraphs *343*

List
Bangs, Bolts, & Bombs

1. 💣 Blowing the dust off MikTeX *16*
2. ⚡ Have packages delivered straight to your computer *16*
3. ❗ Be careful loading packages *18*
4. ⚡ A comment about % *24*
5. ❗ Boldly going where no man has gone before *29*
6. ❗ Avoiding the almighty double dollar *41*
7. ⚡ An alternative to the dollar (sign) *41*
8. 💣 Whatever happens, don't blank out! *43*
9. ❗ Twice as nice just before the end *44*
10. ⚡ How to make your references clever *46*
11. ❗ Dancing with the stars *46*
12. ⚡ Take it to "de limit" *48*
13. ❗ Don't poke the bear! *51*
14. 💣 One is the loneliest delimiter *51*
15. 💣 Seeing the interval as both half-open and half-closed *51*
16. ❗ If number, reference you must *56*
17. ❗ Abbreviating too muc *59*
18. 💣 Know your options first *59*
19. 💣 Drawing requires semicolons *68*
20. ⚡ I want my QED! *75*
21. 💣 Brigadier general with a **`\newcommand`** *79*
22. 💣 You can give your macro any name but... *81*
23. ⚡ A very flexible item *84*
24. ❗ When *not* to use a list environment *97*

25. What happens in an environment stays in an environment *112*

26. The caption is on top of the table. Figure that! *113*

27. I was referring to that table, not your figure *113*

28. What the H! *114*

29. The case of the missing data *126*

30. Multi-multi *129*

31. Make it a happy row *132*

32. The evils of `hyperref` *156*

33. Doing the bare minimum *157*

34. It's okay to blank out *160*

35. The file is always in the last place you look *160*

36. Before washing, separate your files *161*

37. It's mine and you can't have it *162*

38. The seal of approval for your approval page *164*

39. Who stole my list! *167*

40. Follow the rules *170*

41. Get your title back with title case *172*

42. Get in line, chapter! *176*

43. Stop trying to compile me! *176*

44. What year is it? *180*

45. To heck with BibTeX! *182*

46. Let `biblatex` do its job! *182*

47. Making a mad dash *187*

48. Talking with a foreign accent *187*

49. The value of issuing a citation *189*

50. It's not your fault *194*

Cumulus of Commands

1 \documentclass{⟨name⟩} 17

2 \begin{document}
⟨code⟩
\end{document} 18

3 \substack{⟨expression 1⟩\\...\\⟨expression n⟩} 34

4 \begin{⟨env name⟩}
⟨text⟩
\end{⟨env name⟩} 42

5 \label{⟨name⟩} 43

6 \ref{⟨name⟩} 44

7 \begin{equation*}
⟨equation⟩
\end{equation*} 46

8 \SuperSize{⟨mag factor⟩}⟨delimiter⟩ 50

9 \ointerval{⟨start⟩}{⟨end⟩}
\rinterval{⟨start⟩}{⟨end⟩}
\linterval{⟨start⟩}{⟨end⟩} 51

10 \begin{tikzcd}
⟨code⟩
\end{tikzcd} 58

11 \arrow{⟨direction⟩}⟨label⟩ 58

12 \begin{tikzpicture}
⟨code⟩
\end{tikzpicture} 67

13 \draw ⟨point 1⟩-- ⟨point 2⟩; 67

14 \coordinate ⟨label⟩ at ⟨ordered pair⟩; 67

15 \fill ⟨center⟩ circle (⟨radius⟩); 68

16 \begin{theorem}
⟨theorem text⟩
\end{theorem} 73

17 \begin{⟨theorem type⟩}
⟨theorem text⟩
\end{⟨theorem type⟩} 74

18 \DeclareMathOperator{⟨macro⟩}{⟨operator⟩}
\DeclareMathOperator*{⟨macro⟩}{⟨operator⟩} 78

19 \newcommand*{⟨name⟩}{⟨code⟩} 78

20 \newcommand*[⟨number⟩]{⟨macro⟩}{⟨code⟩} 80

21 \begin{⟨list name⟩}
\item ⟨item 1⟩
⋮
\item ⟨item n⟩
\end{⟨list name⟩} 83

22
```
\Graphic{⟨filename⟩}
```
103

23
```
\GlobalFloatParameters{⟨key=value options⟩}
```
112

24
```
\caption{⟨caption text⟩}
```
112

25
```
\begin{figure}
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \includegraphics{⟨filename⟩}
\end{figure}
```
112

26
```
\begin{figure}
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \begin{framed}
        \includegraphics{⟨filename⟩}
    \end{framed}
\end{figure}
```
114

27
```
\begin{figure}
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \hspace*{fill}
    {\includegraphics{⟨filename⟩} \hfill
        \includegraphics{⟨filename⟩}}
    \hspace*{fill}
\end{figure}
```
114

28
```
\begin{table}
    \caption{⟨text⟩}
    \label{⟨name⟩}
\begin{tabular}{⟨column specs⟩}
    ⟨(1,1) entry⟩& ··· & ⟨(1,n) entry⟩\\
    ⟨(2,1) entry⟩& ··· & ⟨(2,n) entry⟩\\
        ⋮
    ⟨(m,1) entry⟩& ··· & ⟨(m,n) entry⟩\\
\end{tabular}
\end{table}
```
117

29
```
\Table{%
    ⟨(1,1) entry⟩& ··· & ⟨(1,n) entry⟩\\
    ⟨(2,1) entry⟩& ··· & ⟨(2,n) entry⟩\\
        ⋮
    ⟨(m,1) entry⟩& ··· & ⟨(m,n) entry⟩\\
}
```
120

30
```
\renewcommand{\arraystretch}{⟨factor⟩}
```
127

31
```
\setlength{\extrarowheight}{⟨length⟩}
```
127

32
```
\addlinespace[⟨length⟩]
```
127

33
```
\multicolumn{⟨number⟩}{⟨alignment⟩}{⟨text⟩}
```
127

34
```
\cmidrule{⟨range⟩}
```
128

35
```
\multirow{⟨numrows⟩}{⟨width⟩}{⟨text⟩}
```
128

36
```
\cellcolor{⟨color⟩}
\columncolor{⟨color⟩}
\rowcolor{⟨color⟩}
```
130

37
```
\ldelim⟨delimiter⟩{⟨numrows⟩}{⟨width⟩}
\rdelim⟨delimiter⟩{⟨numrows⟩}{⟨width⟩}
```
130

38
```
\begin{Listing}{⟨language⟩}
    ⟨first line of code⟩
    ⟨second line of code⟩
        ⋮
    ⟨last line of code⟩
\end{Listing}
```
137

39
```
\mintinline{⟨lang⟩}⟨character⟩⟨code⟩⟨character⟩
```
141

40
```
\inputminted{⟨language⟩}{⟨filename⟩}
```
142

41
```
\DisableLigatures{⟨encoding=T1⟩}
```
146

42 \ThesisTitle{⟨title⟩} 159

43 \ThesisAuthor{⟨author⟩} 159

44 \DocumentType{⟨type⟩} 159

45 \DegreeType{⟨degree⟩} 159

46 \Major{⟨major⟩} 159

47 \GraduationMonth{⟨month⟩} 159

48 \GraduationYear{⟨year⟩} 159

49 \ThesisAbstract{⟨filename⟩} 161

50 \CopyrightOwner{⟨name⟩} 162

51 \CommitteeMember{⟨number⟩}{⟨name⟩} 163

52 \Dedication{⟨text⟩} 165

53 \Acknowledgments{⟨filename⟩} 166

54 \SpecialListItem{⟨macro name⟩}{⟨file extension⟩}{⟨type⟩} 167

55 \⟨name⟩{⟨item⟩} 167

56 \⟨name⟩width{⟨width⟩} 168

57 \chapter{⟨title⟩} 170

58 \section{⟨title⟩} 171

59 \subsection{⟨title⟩} 173

60 \subsubsection{⟨title⟩} 174

61 \MyChapter{⟨number⟩}{⟨filename⟩} 175

62 \MyAppendix{⟨number⟩}{⟨filename⟩} 178

63 \MyBibReferences{⟨filename⟩} 180

64 \cite{⟨label⟩} 188

65 \citep{⟨label⟩} 188

66 \citet{⟨label⟩} 188

67 \MyVita{⟨filename⟩} 190

List of Figures

1.1 Donald Knuth, 2011 and Leslie Lamport, 2020 *10*
2.1 Example of the beginning of a LaTeX source code file *18*
2.2 Preamble material versus document material in a LaTeX source code file *20*
2.3 Example of LaTeX output *20*
2.4 Effect of removing the backslash from a LaTeX command *21*
2.5 Failed attempt to create two separate paragraphs in LaTeX source code file *22*
2.6 LaTeX output showing the successful creation of two separate paragraphs *23*
2.7 Example of a LaTeX error message *23*
3.1 Length units in LaTeX *32*
3.2 LaTeX code showing the use of _ and ^ on `\sum` *35*
3.3 LaTeX code showing the use of _ and ^ on`\sum` *35*
3.4 LaTeX code showing the use of _ and ^ on `\prod` *35*
3.5 LaTeX code showing the use of _ and ^ on `\cap` *35*
3.6 LaTeX code showing the use of _ on `\lim` *35*
4.1 Example of a displayed formula *40*
4.2 Example of displayed text *41*
4.3 Example of some output of the `equation` environment *43*
4.4 Agreement of reference and source number after second compilation *45*
5.1 Square diagram using the `tikzcd` package *64*
5.2 Diagram illustrating the use of the `swap` option for an arrow *65*
5.3 Diagram illustrating a dashed arrow *65*
5.4 An example of an infinite diagram using the `tikzcd` package *66*
5.5 Diagram illustrating a bent arrow *66*
5.6 Code for drawing the complete graph on five vertices using the `tikz` package *69*
5.7 Code for drawing a red triangle in the complete graph on five vertices using the `tikz` package *70*
6.1 Example of `theorem` and `proof` environments in LaTeX output *74*
7.1 Example of an `itemize` environment using • *85*
7.2 Example of an `itemize` environment using $*$ *86*
7.3 Example of an `itemize` environment using $>$ *86*
7.4 Example of an `itemize` environment using † *86*

7.5 Example of an `itemize` environment using ‡ *87*

7.6 Example of an `itemize` environment using ✓ *87*

7.7 Example of an `itemize` environment using $\circ$ *87*

7.8 Example of an `itemize` environment using ~ *88*

7.9 Example of an `itemize` environment using ※ *88*

7.10 Example of an `itemize` environment using $\neg$ *88*

7.11 Example of an `itemize` environment using $\rightarrow$ *89*

7.12 Example of an `itemize` environment using § *89*

7.13 Example of an `itemize` environment using — *89*

7.14 Example of an `itemize` environment using $\cdot$ *90*

7.15 Example of an `itemize` environment using $\star$ *90*

7.16 Example of an `itemize` environment using $\triangleright$ *90*

7.17 Example of an `itemize` environment using ✠ *91*

7.18 Example of an `enumerate` environment using lowercase letters encased in parentheses *91*

7.19 Example of an `enumerate` environment using boldfaced capital letters *92*

7.20 Example of an `enumerate` environment using lowercase Roman numerals *92*

7.21 Example of an `enumerate` environment using Arabic numerals starting at 13 *92*

7.22 Example of an `enumerate` environment using uppercase Roman numerals *93*

7.23 Example of an `enumerate` environment using Arabic numerals prefaced by the octothorp symbol # *93*

7.24 Example of an `enumerate` environment using circled Arabic numerals *93*

7.25 Example of an `enumerate` environment using Arabic numerals in square brackets *94*

7.26 Example of an `enumerate` environment using capital letters in angled brackets *94*

7.27 Example of an `enumerate` environment using uppercase Roman numerals offset by forward slashes *94*

7.28 Example of an `enumerate` environment using italicized Arabic numerals *95*

7.29 Example of a `description` environment using normal font *95*

7.30 Example of a `description` environment using a next line format *96*

7.31 Example of a `description` environment using standard style *96*

7.32 Example of a `description` environment using slanted font *96*

7.33 Example of a `description` environment using small caps type *98*

7.34 Example of a `description` environment using slanted boldfaced type *98*

7.35 Example of an environment using each of `itemize`, `enumerate`, and `description` *99*

8.1 An example of a caption using APA 7 style *102*

8.2 An example of the `note` key for a figure *107*
8.3 An example of output using the `\Graphic` command *111*
9.1 An example of output using the `table` environment *120*
9.2 An example of the `callout` key in a table *124*
9.3 Example of a simple table with different fonts in each column *125*
9.4 Example of a simple table using `\toprule`, `\midrule`, and `\bottomrule` *125*
9.5 Example of a simple table using a double `\midrule` *125*
9.6 Example of a simple table using column dividers *126*
9.7 Example of a simple table with column and row dividers *126*
9.8 Example of a table using `\multicolumn` *128*
9.9 Example of a table using `\multirow` *129*
9.10 Example of a table using `\multirow` and without center midrule lines *130*
9.11 Example of a table using `\cellcolor` *131*
9.12 Example of a table using `\columncolor` *131*
9.13 Example of a table using `\rowcolor` *132*
9.14 Example of a table using a big delimiter *133*
9.15 Example of a matrix using big delimiters and the `array` environment *133*
10.1 An example of output using the `Listing` environment *141*
10.2 An example of output using the `\mintinline` command *142*
12.1 An example of a title page *158*
12.2 The inverted Mayan pyramid look of the title on the title page *160*
12.3 An example of an abstract page *161*
12.4 An example of a copyright statement for an individual *162*
12.5 An example of an approval page *164*
12.6 An example of a dedication *165*
12.7 An example of an acknowledgment *166*
13.1 An example of the beginning of a chapter *171*
13.2 An example of the beginning of a section *172*
13.3 An example of the beginning of a subsection *174*
13.4 An example of the beginning of a subsubsection *175*
14.1 An example of an entry in a `bib` file *179*
14.2 Graphic showing the procedure for compiling a `bib` file in TeXworks *181*
14.3 An example of the `\cite` command in the source code *188*
14.4 An example of a vita *190*

List of Tables

3.1 Standard LaTeX text font options *28*

3.2 Variations on LaTeX text font commands *29*

3.3 Standard LaTeX math font options *30*

3.4 Horizontal space commands in LaTeX *33*

3.5 Commands for lowercase Greek letters *36*

3.6 Commands for some Hebrew letters *36*

3.7 Commands for uppercase Greek letters *37*

4.1 Additional LaTeX delimiters *48*

4.2 Relative sizes of enlargement macros applied to the token | *49*

4.3 Sizing commands and their corresponding font sizes assuming a standard of 12 pt *50*

5.1 Standard left and right arrows in the `tikz` package *61*

5.2 Mapping arrows in the `tikz` package *61*

5.3 Hook arrows in the `tikz` package *61*

5.4 Harpoon arrows in the `tikz` package *61*

5.5 Squiggly arrows in the `tikz` package *61*

5.6 Headless/tailless arrows in the `tikz` package *62*

5.7 Dashed arrows in the `tikz` package *62*

5.8 Two-headed arrows in the `tikz` package *62*

5.9 Tailed arrows in the `tikz` package *62*

5.10 Standard colors in the `tikz` package *64*

5.11 Line thickness options in the `tikz` package *71*

5.12 Line styles in the `tikz` package *71*

6.1 Standard operator macros provided by the `amsmath` package *77*

7.1 General keys for a list environment *85*

7.2 Commands for the **label** key in the `enumerate` environment *91*

9.1 Column parameters for a table *119*

14.2 LaTeX commands for foreign language symbols *187*

14.3 Examples of the use of `\citep` and `\citet` *188*

14.4 Variations on the `\cite` command *189*

Resource I

Basic Procedure

Step 0

If you are not using an online editor, **download** and install the latest version of MikTeX (currently version 2.9). If MikTeX is already on your system, update all associated files through the MikTeX Console.

Step 1

Download `TNThesis.cls` and `TNTmain.tex`.

Step 2

Use a LaTeX editor to create separate `tex` files for your abstract, acknowledgements, each chapter, each appendix, and your vita. Create a `bib` file for your bibliography.

Step 3

Compile your work. Fix any errors and recompile, as necessary.

Step 4

After successfully defending your work, **upload** the `pdf` file of your work. **Indicate in the comments section that your work was created using LaTeX.**

Step 5

Make the changes to your work recommended by your editor at the Graduate School and resubmit.

Step 6

Do your happy dance.

Examples of Math Into LaTeX

1. `\[\lim_{\theta\to\infty} \sin\left(\frac{1}{\theta}\right) = 0\]`

$$\lim_{\theta\to\infty} \sin\left(\frac{1}{\theta}\right) = 0$$

2. `\[\int_a^b f(x)\,\mathrm{d}x = F(x)\,\bigg|_a^b\]`[1]

$$\int_a^b f(x)\,\mathrm{d}x = F(x)\,\bigg|_a^b$$

3. `\[\iint_D f(x,y)\,\mathrm{d}A =`
`\int_a^b \int_{g_1(x)}^{g_2(x)} f(x,y) \,\mathrm{d}y\,\mathrm{d}x\]`

$$\iint_D f(x,y)\,\mathrm{d}A = \int_a^b \int_{g_1(x)}^{g_2(x)} f(x,y) \,\mathrm{d}y\,\mathrm{d}x$$

4. `\[\iint_{\partial E} \mathbf{F} \cdot \mathrm{d}\mathbf{S} =`
`\iiint_E \Div \mathbf{F} \, \mathrm{d}V\]`[2]

$$\iint_{\partial E} \mathbf{F} \cdot \mathrm{d}\mathbf{S} = \iiint_E \operatorname{div} \mathbf{F} \, \mathrm{d}V$$

5. `\[\oint_\mathscr{C} \mathbf{F} \cdot \mathrm{d}\mathbf{r}\]`

$$\oint_\mathscr{C} \mathbf{F} \cdot \mathrm{d}\mathbf{r}$$

---

[1] Technically, the “d” in differentials should be typeset upright while the corresponding variable should be italicized. As well, Euler’s constant e and imaginary unit i should be typeset upright. See ISO 80000-2:2019.

[2] The operator `\Div` was initially set up using the `\DeclareMathOperator` command (see page 78).

6. `\[\nabla^2 \mathbf{F}(x,y) = \frac{\partial^2}{\partial x^2}\mathbf{F}+ \frac{\partial^2}{\partial y^2}\mathbf{F}\]`

$$\nabla^2 \mathbf{F}(x,y) = \frac{\partial^2}{\partial x^2}\mathbf{F}+ \frac{\partial^2}{\partial y^2}\mathbf{F}$$

7. `\[f^{\prime\prime}(x) = \ddot{y} = \frac{\mathrm{d}^2y}{\mathrm{d}x^2}\]`

$$f^{\prime\prime}(x) = \ddot{y} = \frac{\mathrm{d}^2y}{\mathrm{d}x^2}$$

8. `\[f  \colon A \to B\]`[3]

$$f \colon A \to B$$

9. `\[0 \rightarrow A \xrightarrow[\text{1-1}]{f} B \xrightarrow[\text{onto}]{g} C \rightarrow 0\]`

$$0 \rightarrow A \xrightarrow[\text{1-1}]{f} B \xrightarrow[\text{onto}]{g} C \rightarrow 0$$

10. `\[f(n) = \begin{cases} \frac{n}{2} &\text{if }n\text{ is even,}\\`[4] `-\frac{n-1}{2} & \text{if }n\text{ is odd}\end{cases}\]`

$$f(n) = \begin{cases} \frac{n}{2} &\text{if }n\text{ is even,}\\ -\frac{n-1}{2} & \text{if }n\text{ is odd}\end{cases}$$

11. `\[\mathbf{a} = \langle \alpha_1,\alpha_2, \ldots ,\alpha_n \rangle\]`[5]

$$\mathbf{a} = \langle \alpha_1,\alpha_2, \ldots ,\alpha_n \rangle$$

---

[3] The use of the macro `\colon` is preferred to : as it results in better spacing.

[4] The American Mathematical Society recommends placing a comma after each case of a piecewise-defined function, save the final case, which should have the appropriate punctuation following it (e.g., a period if it is the end of a sentence).

[5] The use of `\langle` and `\rangle` is preferred in this context to `<` and `>`.

12. `\[\|z\|^2 = z\bar{z}\]`[6]

$$\|z\|^2 = z\bar{z}$$

13. `\[\overline{a_nz^n + a_{n-1}z^{n-1} + \cdots + a_1z + a_0} = \overline{a_n}\bar{z}^n + \overline{a_{n-1}}\bar{z}^{n-1} + \cdots + \overline{a_1}\bar{z} + \overline{a_0}\]`

$$\overline{a_nz^n + a_{n-1}z^{n-1} + \cdots + a_1z + a_0} = \overline{a_n}\bar{z}^n + \overline{a_{n-1}}\bar{z}^{n-1} + \cdots + \overline{a_1}\bar{z} + \overline{a_0}$$

14. `\[\sqrt[n]{x_1^n + x_2^n + x_3^n + \cdots + x_m^n}\]`[7]

$$\sqrt[n]{x_1^n + x_2^n + x_3^n + \cdots + x_m^n}$$

15. `\[M = \begin{pmatrix*}[r] -1 &2 \\ 3 & -4 \end{pmatrix*}\]`[8]

$$M = \begin{pmatrix*}[r] -1 & 2 \\ 3 & -4 \end{pmatrix*}$$

16. `\[M = \begin{bmatrix*}[r] -1 & 2 \\ 3 & -4 \end{bmatrix*}\]`

$$M = \begin{bmatrix*}[r] -1 & 2 \\ 3 & -4 \end{bmatrix*}$$

17. `\[M = \begin{bmatrix*}[l] a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots && \ddots & \\ a_{n1} & a_{n2} & \ldots & a_{nn}\end{bmatrix*}\]`

$$M = \begin{bmatrix*}[l] a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots && \ddots & \\ a_{n1} & a_{n2} & \ldots & a_{nn}\end{bmatrix*}$$

---

[6] Instead of `\|`, you can also use the macro `\Vert`.

[7] If a simple square root $\sqrt{}$ is needed, just use `\sqrt{...}`.

[8] The optional argument `r` right-aligns each entry in its column, `l` left-aligns, and `c` centers.

18. `\[\det M = \begin{vmatrix*}[c] 13 & 2\\ 3 & 41 \end{vmatrix*}\]`

$$\det M = \begin{vmatrix} 13 & 2 \\ 3 & 41 \end{vmatrix}$$

19. `\[x \notin \mathbb{Q} \implies x \notin \mathbb{N}`
`\centernot\implies x \in \mathbb{C} \backslash \mathbb{R}\]`[9] [,10]

$$x \notin \mathbb{Q} \implies x \notin \mathbb{N} \centernot\implies x \in \mathbb{C} \backslash \mathbb{R}$$

20. `\[\binom{n}{k} = \frac{n!}{k!\,(n-k)!}\]`

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

21. `\[x \equiv a \negthickspace\negthickspace \pmod{b}\]`[11]

$$x \equiv a \pmod{b}$$

22. `\[A \nsubseteq \bigcap B_i \subsetneq \bigoplus C_j`
`\subset \bigotimes D_k \subseteq \coprod E_\ell\]`

$$A \nsubseteq \bigcap B_i \subsetneq \bigoplus C_j \subset \bigotimes D_k \subseteq \coprod E_\ell$$

23. `\[x^2 = 1 \iff x = \pm 1 \iff x =\mp 1\]`

$$x^2 = 1 \iff x = \pm 1 \iff x = \mp 1$$

24. `\[bigcup_{n=0}^\infty \left\{\frac{-n}{1},\frac{n}{1}\right\} = \mathbb{Z}\]`

$$\bigcup_{n=0}^\infty \left\{-\frac{n}{1},\frac{n}{1}\right\} = \mathbb{Z}$$

---

[10] The macro `\centernot` should be used in front of the command for a mathematical symbol one wishes to negate, unless that symbol already has an established command for its negation, such as`\notin`.

[10] Instead of `\backslash`, one can use `\setminus`. However, the spacing will be slightly different.

[11] The macro `\negthickspace` is designed to close up excessive spacing here.

25. `\[\rinterval{0}{1} = \{x \mid x \geq 0\ \text{and}\ x<1\}\]`[12]

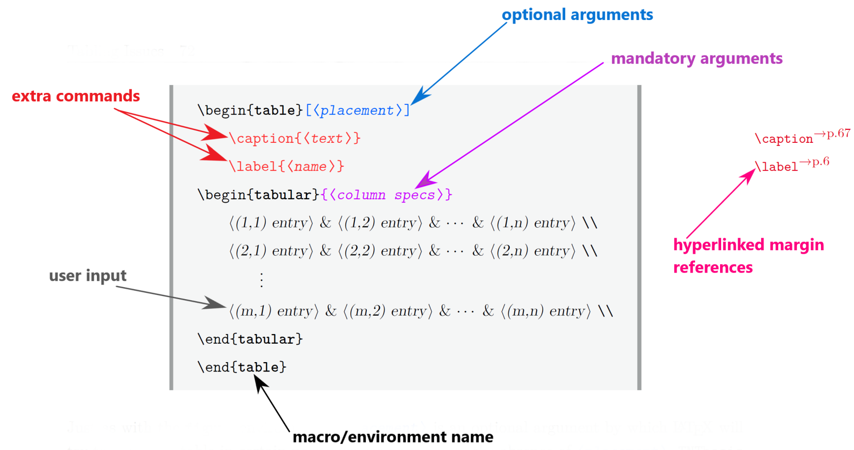
$$[0,1) = \{x \mid x \geq 0 \text{ and } x < 1\}$$

[12] Knuth recommends using the thin space command `\,` immediately after the opening curly braces `\{` and just before the closing curly braces `\}` to balance the spacing created by `\mid`.

Legend for the User's Manual

macro syntax, displayed	`\CommitteeMember`
special remark	**A comment about %** — If you are...
LaTeX code with output	`$\sum_{i=1}^3 i=6$` → $\sum_{i=1}^{3} i = 6$
command referenced in text	`\hspace*{3in}`
emphasis in LaTeX code	`\arrow{dd}{\Phi}`
mandatory macro arguments	{⟨*column specs*⟩}
optional macro arguments	[⟨*placement*⟩]
secondary commands	\caption{⟨*text*⟩}
hyperlinked margin reference	\label→p.43
hyperlinked URL	**miktex.org**
hyperlinked in-text reference	Appendix B
hyperlinked glossary reference	glue
user input	⟨*(1,1) entry*⟩
LaTeX commands; definitions	**`\documentclass`**
TeX code; program names	`minted` package
LaTeX editor menu options	FILE, TOOLS
emphasis; titles	do you *really* think so?

Command Syntax



∗ Concerning items offset by angled brackets ⟨...⟩, the input should be given *without* the angled brackets. For example, if you were creating a table whose first column indicates types of fruit in a particular region, you would replace ⟨*(2,1) entry*⟩ with, say, the word *Kiwis*, **not** ⟨*Kiwis*⟩.

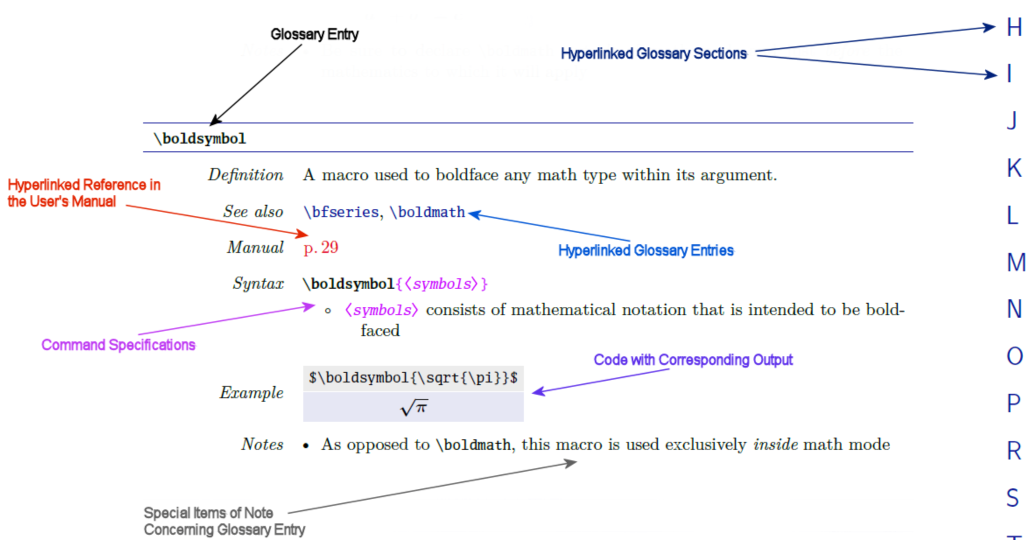
Bangs, Bolts, and Bombs

! indicates a common pitfall that you should avoid

ϟ indicates a tip that will allow you to work more efficiently

💣 indicates an issue that will prevent LaTeX from compiling correctly or at all

Glossary Entry



Chapter 1

In the Beginning...

The only true wisdom is in knowing that you know nothing.

Socrates

1.1 From Where Did It All Come?

First of all, let me congratulate you on making the bold decision to type up your thesis or dissertation using LaTeX. I'm sure your friends, parents—heck, even your major advisor—tried to stop you. "Just type it up in Microsoft Word," they said. "It will be so much easier!"

But I know you. You're the kind of person that when your math teacher told the class to take out their graphing calculators, you reached into your backpack for your slide rule...just because you enjoyed the challenge. You're the kind of person that when you last went camping and your friends asked if your brought the fire starter, you proudly replied, "Where's the fun in that!?" And, yes, you're the kind of person that taught yourself Russian just so you could read the original text of *War & Peace*...only to find out that the English translation was spot on. Sound familiar?

All kidding aside, LaTeX offers some serious advantages over software like Word, and I don't just mean with respect to typesetting complex mathematical formulae (over which I would rather drink molten lava than use Word). LaTeX is a dynamically different system than its more commonplace counterparts, in that LaTeX is software for type*setting* documents, whereas Word is merely for typing documents. "What's the difference?" you ask. I'm glad you asked. Word displays what you type letter by letter in real time, so that if you would suddenly stop typing, then your document would look exactly as Word has it displayed on the screen in that moment. Keep typing and Word simply adds to what is already there. On the other hand, LaTeX can't show you what your output will truly look like until you have finished the *entire* document. That's because LaTeX assesses all that you have typed up and uses some pretty sophisticated algorithms to determine things like end-line hyphenation, line breaks, and page breaks to produce the most aesthetically pleasing and balanced spacing possible to your text. This is exactly why LaTeX is the software of choice for all major mathematics journals out there! Furthermore, on the flip side of the same coin, LaTeX is so powerful that, with the appropriate commands, you can control the placement of every single typographical object on a page to within *five millionths of an inch.*[1] Try doing that with Word!

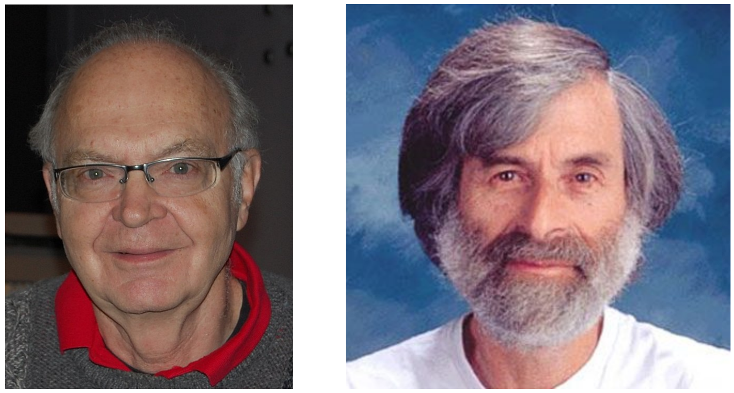


FIGURE 1.1: Donald Knuth, 2011 and Leslie Lamport, 2020

But I'm preaching to the choir here. After all, you're the kind of person who has already decided to use LaTeX for your work (remember, I know you). So then, let me tell you something you probably didn't know and regale you with a story about the origins of it all. The year was 1977. Jimmy Carter becomes president of the United States, *Star Wars* debuts in theaters, and Stanford University computer science professor Donald Knuth is seriously peeved by the lack of typographical consistency in his multi-volume work *The Art of Computer Programming*. It was out of his frustration that the software dubbed TeX by Knuth was born.[2] Indeed, when Knuth debuted TeX to the world in 1984, it represented the very first software capable of consistently typesetting documents containing mathematics!

Nevertheless, TeX suffered from severe limitations, most notably the complexity in syntax required to typeset certain mathematical expressions. So, it wasn't too long before computer scientists realized that to take TeX to the next level, a more user-friendly interface was going to be necessary. A few years after TeX's world premiere, mathematician-turned-computer scientist Leslie Lamport thus introduced LaTeX, a system of commands (more specifically, "macros") that provided simple expressions for more complicated sequences of commands (more specifically, "primitives") that TeX could understand. And while there have been several competitors to LaTeX, no other system has eclipsed it in popularity.

"Is there an easy way to understand the real difference between TeX and LaTeX?," you ask. Man! you have some great questions. At the risk of sounding overly simplistic, LaTeX tells the system *what* to print while TeX tells the system *where* to put it. So, really the two systems work in tandem with one another. For the sake of simplicity (which this user's manual has as its aim), I hope you'll forgive

[1] Based upon the fact that LaTeX can place objects within 1 sp, where there are 65,536 sp/pt and 72.27 pt/in.

[2] The term TeX is derived from a Greek word meaning "skill".

me for referring to the combined system as just LaTeX. Where necessary, I'll draw a distinction between the two.

The "modern" version of LaTeX, referred to as LaTeX 2_ε, is built on top of an extended version of the original TeX system called e-TeX. LaTeX 2_ε supplanted LaTeX 2.09 and was released in 1994. At that same time, work began on LaTeX3, which is intended to be a more comprehensive, coding-friendly system than LaTeX 2_ε. However, the LaTeX3 Project faces a daunting, uphill battle in trying to design something that will meet the requisite goals while not rendering the pantheon of packages produced over the past two and half decades totally obsolete.

Just like its predecessors, LaTeX 2_εuses a type of command called a **macro** to dictate what special symbols to print. The syntax of these macros consists of a backslash `\` followed by a group of alphabetic characters (typically a suggestive word or at least an abbreviation of a word). Furthermore, often a macro or group of macros is surrounded by dollar signs `$...$` to indicate that the respective group of symbols is to be typeset in the special positioning and italicized font indicative of mathematical notation (although this is not the only way to produce "math mode"; see Chapter 4). For example, the expression `S = \sum x_i` contains the macro `\sum` and typesets as $S = \sum x_i$. (If you wish to typeset an actual dollar sign in your text, use the command `\$`.)

LaTeX $2_\varepsilon$is included in all modern distributions of LaTeX. A **distribution** not only includes the base software of LaTeX and TeX but a uniquely curated selection of **packages**, which are special add-on programs designed to enhance LaTeX's capabilities through additional macros. Some of the most popular current distributions include

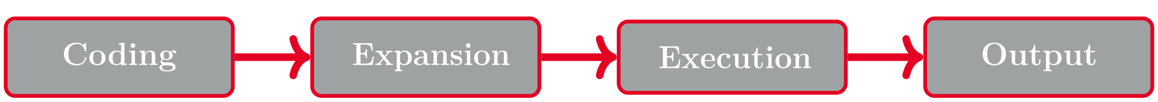
- proTeXt (**tug.org/protext/**)
- TeX Live (**tug.org/texlive/**)
- MacTeX (**tug.org/mactex/**)
- MikTeX (**miktex.org**)

For this manual, we will focus only the MikTeX distribution, although most of what is discussed will apply no matter which distribution you have.

1.2 How Does It All Work?

My purpose here is not to overwhelm you with all the details about how LaTeX takes your source code and transforms it into a stunning `pdf` document (there have in fact been entire books written in this area). However, I would like to give you a broad idea of how it all works.

The actual process of transforming your code into a pdf document is called **compiling**. Compiling in LaTeX effectively involves four distinct stages:



Think of it this way: if compiling is like digestion, then

- coding is like seeing the food that you want to eat
- expansion is like chewing the food in your mouth
- execution is like processing the food in your stomach and intestines
- output is like...well, you know

(This analogy is due to Knuth himself; I guarantee that you will never forget these four stages again.) Specifically, in expansion LaTeX takes the code that you wrote, identifies "expandable" commands and macros, and replaces these commands and macros with their corresponding definitions. In particular, if a command or macro requires an argument, this is the stage at which LaTeX will track down the value of the argument. In execution LaTeX takes all of the expanded commands and macros (in conjunction with those that did not require expansion) and actually processes the associated code. In output LaTeX uses its internal algorithms to translate this code into an actual document.

Simple enough, right? Well, actually what goes on behind the scenes can be considerably more complicated than things appear. An almost constant thorn in package writers' and class writers' side is to either stave off premature execution or to kick-start the execution process with certain macros. In fact, commands such as `\expandafter`, `\protect`, and `\noexpand` were designed for just such a purpose.

To facilitate the document creation process, LaTeX utilizes a myriad different types of files along the way, some of which are created only by compilation (indicated below in green) and some of which are executed only when you compile a *second time* or *third time*.

`aux` (auxiliary file) contains information that allows LaTeX to compile faster after the first time; also contains the sectioning commands that will eventually be written to the table of contents

`bbl` (BibTeX bibliography file) contains information for insertion into the bibliography of the work when the work is compiled a second time

`bcf` (BibTeX bibliography file) final bibliography file containing corresponding citation information for each bibliographic entry

`bib` (bibliographic data file) contains information that will be processed and placed in the bibliography of the work; useful if one has numerous sources in the bibliography and/or you are able to copy and paste bibliographic information from a large database

`blg` (BibTeX log file) records any errors created in the process

`bst` (BibTeX style file) determines the style for the bibliography of the work (e.g., `TNThesis5.1` sets the bibliography in APA style, the standard for thesis and dissertations at Tennessee Tech University)

`bsx` (BibTeX style file) determines the style for the bibliography of the work `cls` (document class file) determines the default parameters for the type of document that you intend to create (e.g., `TNThesis5.1` is the class file for your thesis or dissertation); when one declares `\documentclass{`⟨*name*⟩`}` in the preamble of a `tex` file, ⟨*name*⟩ must be associated with a ⟨*name*⟩`cls` file accessible by the `tex` file

`csx` (cite style file) determines the style for the citations

`dtx` (document source file) a precursor file to the creation of a `sty` file

`fd` (font file) contains information on the creation of a particular font

`glo` (glossary file) contains information that is written into a glossary, if one exists in the work

`idx` (index source file) contains the raw, unsorted information on the entries of the index

`ilg` (index log file) records any errors that occur in the process of creating an index

`ind` (index file) this is the file where the sorted (alphabetized) information on the entries of the index is stored and from which LaTeX creates the index

`ins` (installation file) extracts files out of a `dtx` file to create a `sty` file

`ist` (index style file) contains information to format the index in a particular style

`lof` (list of figures file) contains the information used in the creation of your list of figures

`log` (main log file) contains a record of which special packages LaTeX implements in compiling the given .tex file, as well as a listing of all errors encountered and when they occurred in the compilation process

`lol` (list of listings file) contains the information used in the creation of your list of listings

`lot` (list of tables file) contains the information used in the creation of your list of tables

`sty` (style file) creates a set of default parameters and macros for a document; used in conjunction with a given `cls` file

`synctex` (synchronization file) for those using certain editors (e.g., *TeXstudio*) that allow you to automatically go to the corresponding location in the pdf output when a change is made in the source code, the `synctex` file stores the information that TeX needs to make such a connection possible

`tex` (LaTeX file) this is a file containing your source code from which you will compile to ultimately create your document (e.g., `TNTmain` is a `tex` file)

`toc` (table of contents file) stores information that LaTeX places into the table of contents of the work

To view any of these files, you need only open the file in your LaTeX editor. Of course, after compiling a second time, this is generally when the all-important `pdf` file is created with your output.

As I mentioned at the beginning of the chapter, TeX has some pretty sophisticated algorithms to determine where objects get placed on a page. In broad terms, TeX has two modes for handling information: horizontal mode and vertical mode. TeX essentially organizes your document paragraph by paragraph. Within a paragraph, TeX places each line in an invisible horizontal box, using its algorithms to decide how much information to pack into any given horizontal box. It then stacks the collection of these horizontal boxes vertically to create, voilà, the paragraph. Once enough paragraphs are created, TeX will then stack these paragraphs vertically to create, what else, a page (modulo any necessary paragraph breaks at the end of the page, of course.) And so it is with each page that you eventually see in your output. Ultimately, the way to understand what TeX is doing behind the scenes is that it is storing the data that you give it in a bunch of invisible horizontal and vertical boxes which, when put together, are designed to convince the reader that he or she is really looking at a professional-looking publication. Pretty clever, huh.

Let's begin creating your very first LaTeX document. The first line of *any* compilable LaTeX document declares the document class of the corresponding work. A **document class** is a corresponding program that effectively creates the default appearance of your work. It is what tells TeX what the default margins will be, what the default title placement will be, what the default pagination will look like, etc. You will generally choose a class based upon the *type* of work you intend to create.

In the subsequent chapters of this manual, you will learn how to harness the incredible power of LaTeX to produce a thesis or dissertation that you will be proud to show off! (Well, at least appearance-wise; you're entirely on your own for the content.) So strap into your NTSB-approved three-point harness and turn the page.

Chapter 2

...There Was TeX

Sometimes it's necessary to go a long distance out of the way in order to come back a short distance correctly.

Edward Albee

The first section of this chapter describes how to set up MikTeX on your personal computer. If you intend to use an online editor such as Overleaf to create and compile your work, you may skip this section and start with Section 2.2. Moreover, you will need to make the appropriate adjustments to the instructions to suit the editor that you are using.

2.1 Setting Up MikTeX

The first step in using LaTeX is to get a distribution installed on your computer. While you are certainly free to use whatever distribution you like, I will tell you that `TNThesis` was built with MikTeX in mind, and so that is what I will focus on here. MikTeX, available for Windows, Mac, and Linux, can be downloaded at **miktex.org**. Its wizard will gently guide you through the installation process.

Once installed, you will find that there are two subprograms of MikTeX that serve as a means for the user to interface with the system, the MikTeX Console for maintaining your MikTeX installation and TeXworks for typing up and editing LaTeX documents (any such software designed to allow you to edit LaTeX documents is referred to as an **editor**). You certainly aren't bound to use TeXworks as your editor if you would prefer something else. Some other options include LyX (**lyx.org**), TeXnicCenter (**texniccenter.org**), and TeXstudio (**texstudio.sourceforge.net**). If you prefer to have something that is completely web-based, you can also consider Overleaf (**overleaf.com**), although I will have a few things to say about it later. I myself use WinEdt (**winedt.com**) for its versatility of tools. Each editor has its pros and cons, which I discuss in detail in Appendix A.

However, since TEXworks comes with the MikTEX distribution itself and is frankly the best editor for beginners, I will couch the following discussion assuming that you are using TEXworks as your editor (although many of the basic ideas apply no matter which editor you use).

Blowing the dust off MikTEX

If you already have MikTEX installed on your computer, it may be necessary to update the packages, especially if you are encountering compilation errors referencing the `etex` package. In the MikTEX Console, under OVERVIEW, click on CHECK FOR UPDATES. Then click on GO TO UPDATES PAGE, and tell MikTEX to UPDATE NOW.

Have packages delivered straight to your computer

Unless you want to spend an inordinate amount of time telling LATEX to download each package used in `TNThesis` one by one on your first run through, I *strongly* suggest that you start by going to the GENERAL submenu under SETTINGS in the MikTEX Console and select the option ALWAYS INSTALL MISSING PACKAGES ON-THE-FLY. This way, when you declare the use of a special package in your preamble, LATEX will automatically download it for you when you compile your document for the first time. Why doesn't MikTEX just come equipped with all of the necessary packages at the point you install it on your computer? Because the MikTEX repository contains 5,848 separate packages![a] There would be so much data in the MikTEX system on your computer, you could never efficiently compile anything. So, the system is purposefully designed so that you have to declare in the preamble of your document any special packages that you will need for compiling your document. But don't worry, most documents require only 5–10 packages and for the sake of your thesis or dissertation, I have already taken the liberty of having MikTEX call all essential packages through the `TNThesis` class file.

[a]As of August 15, 2022

2.2 Taking It for a Test Spin

So, now that you've got MikTEX successfully installed on your computer, it's time to see what this baby can do! Remember that I am assuming you are using TEXworks to write your document. If you are using some other editor, the procedure is generally the same.

To start with, you will notice that next to a small green right-pointing triangle ▶ at the top is the option PDFLATEX-MAKEINDEX-BIBTEX. Click on the corresponding drop-down menu and select just PDFLATEX instead (but do *not* click on the small green right-pointing triangle...yet). Later on, if you use TEXworks to type up your thesis or dissertation, you will want to set the preferences so that PDFLATEX is the default. The options of PDFTEX, LUATEX, and XETEX represent different modern engines for creating documents. The engine pdfTeX is the simplest one of this group and is designed to write the output to a `pdf` file (as opposed to the "old days" when TEX would write the output to a `dvi` file and then you would separately have that file written to a postscript file so that you could actually view your output). The option of MAKEINDEX is there if you wish to create an index that will appear in the back matter of your work. However, this tends to be a rather (and by "rather", I mean "really") labor-intensive process and is not required by the Graduate School for your thesis or dissertation. (If you wish to have an index in your thesis or dissertation, please contact me and I would be happy to provide you with information on a way to accomplish this.)

The option of BIBTEX is there for those who wish to use this special software for setting up the bibliography to their work. However, the `TNThesis` class file uses another piece of software called Biber (see the second-to-last option in the drop-down menu) to create the bibliography. For now, though, don't worry about the details of creating a bibliography (this is covered in Section 14.2). You first need to become comfortable with the basics of typesetting in LATEX.

Let's begin creating your very first LATEX document. The first line of *any* compilable LATEX document declares the document class of the corresponding work. A **document class** is a corresponding program that effectively creates the default appearance of your work. It is what tells TEX what the default margins will be, what the default title placement will be, what the default pagination will look like, etc. You will generally choose a class based upon the *type* of work you intend to create. Some of the choices for document classes include the following:

`amsart`, `amsbook`, `article`, `beamer`, `book`, `exam`, `leaflet`, `letter`, `memoir`, `minimal`, `moderncv`, `novel`, `paper`, `proc`, `report`, `scrbook`, `slides`, `standalone`

There are a myriad other special classes, too, including the one that you will eventually use to type your thesis or dissertation. To declare a document class, use the command

```
\documentclass{⟨name⟩}
```

where ⟨*name*⟩ is the name of the class (e.g., `book`, `leaflet`). To get you started, let's go with the `article` document class, so that your first line should look like `\documentclass{article}`.[1]

The default font size for the article class is 10 pt. Say you prefer 12 pt. This can be easily changed by way of the same command. In general, when a command in LATEX has an associated optional parameter, it usually can be activated by way of inserting the syntax `[⟨option⟩]` between the command name and the first curly brace (or in some rare instances after a set of curly braces). In our case, we can change the syntax of our first line to now look like `\documentclass[12pt]{article}`. Other parameters are not as easily changed. Say, for instance, you want 1 inch left and right margins. To achieve this, you can use the following syntax:

[1] As indicated in the section *Legend to the User's Manual*, remember that the use of the **dark orange** coloring here is merely to emphasize important aspects of the LATEX code.

```
\setlength{\leftmargin}{1in}
\setlength{\rightmargin}{1in}
```

Be careful loading packages

! You should only load packages that you know for sure you will use in the process of typing up your work. Some packages automatically change certain parameters of the document class. Others will change certain definitions of standard commands. Even the order in which you load packages can make a difference in the appearance of your document.

At this point, you might want to consider special packages that you wish to load with your document class. Remember that a package is an add-on program that provides additional macros for you to use in your work. You will not have to worry about loading packages when you type up your thesis or dissertation using the `TNThesis` class file, but for the sake of general practice, go ahead and load a few standard packages using the following syntax:

```
\usepackage{amsmath}
\usepackage{amsthm}
\usepackage{amssymb}
\usepackage{amsfonts}
```

So far, your code should look like that in Figure 2.1.

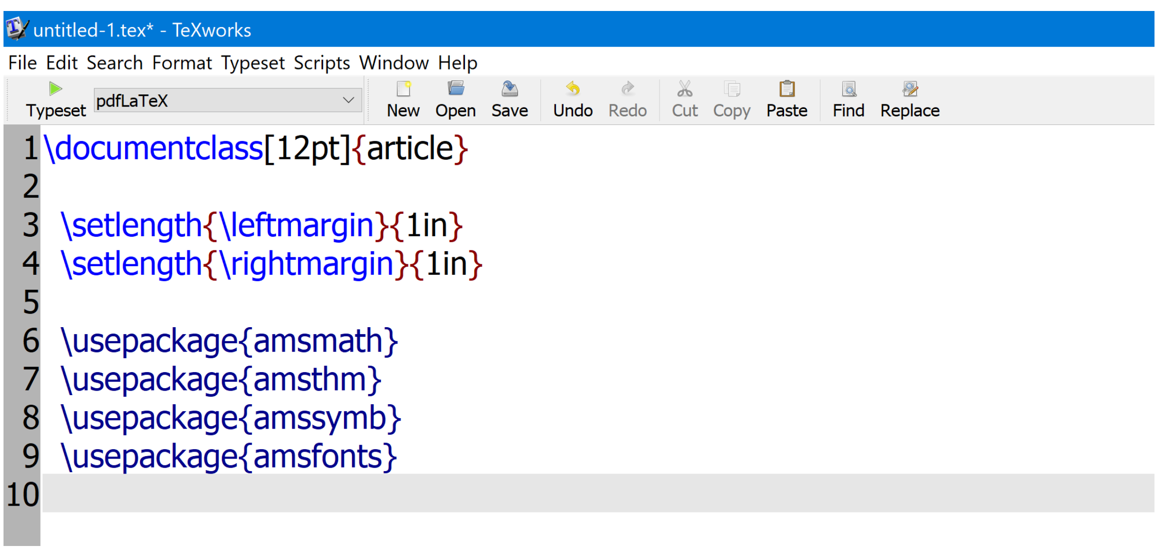


FIGURE 2.1: Example of the beginning of a LATEX source code file

Let's begin the actual writing of the document then. The body of your entire work is sandwiched between two commands (technically, it's referred to as an *environment*; see page 42):

```
\begin{document}
    ⟨code⟩
\end{document}
```

Everything that came before `\begin{document}` is referred to as the **preamble**of your code (see Figure 2.2). Certain commands such as `\documentclass` and `\usepackage` must appear in the preamble. Other commands can appear in either the preamble or the document environment. Now, starting on a line after `\begin{document}`, type (uh, copy and paste) the paragraph that follows.

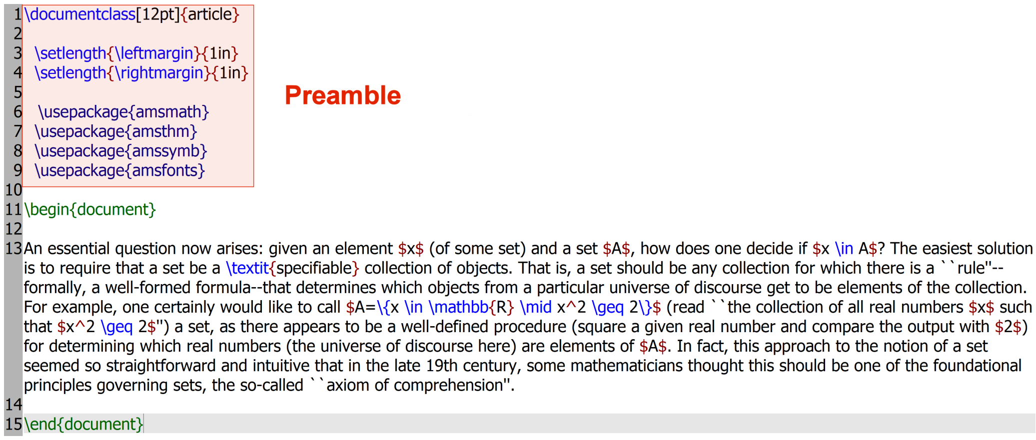
```
An essential question now arises: given an element $x$ (of some set) and a set $A$,
how does one decide if $x \in A$? The easiest solution is to require that a set be a
\textit{specifiable} collection of objects. That is, a set should be any collection for
which there is a ``rule''---formally, a well-formed formula---that determines which
objects from a particular universe of discourse get to be elements of the collection.
For example, one certainly would like to call $A=\{x \in \mathbb{R} \mid x^2 \geq 2\}$
(read ``the collection of all real numbers $x$ such that $x^2 \geq 2$'') a set, as there
appears to be a well-defined procedure (square a given real number and compare
the output with $2$) for determining which real numbers (the universe of discourse
here) are elements of $A$. In fact, this approach to the notion of a set seemed so
straightforward and intuitive that in the late 19th century, some mathematicians
thought this should be one of the foundational principles governing sets, the so-called
``axiom of comprehension''.
```

Once you have finished doing this, drop a line and type `\end{document}`. (Technically, there is no absolute need to give `\begin{document}` and `\end{document}` their own separate lines, it just makes things more readable.)

Before compiling the document, we will need to save it. Under the drop-down menu for FILE at the top, select the SAVE option and give your file a name. I will call mine `test`, which is then automatically saved as a `tex` file. Now, for the moment of truth. Remember that little green right-pointing triangle up top? Remember how I told you don't push the little green right-pointing triangle yet? Push the little green right-pointing triangle. Go ahead, it will be okay. Push the little green right-pointing triangle. What happened? Suddenly something resembling a scene out of *The Matrix* went flying by at the bottom of the screen and in less than a second or two a pdf file like that shown in Figure 2.3 popped up on the screen with the name that you gave the `tex` file except with a `pdf` extension instead. Congratulations! You just created your first document in LaTeX. How do you feel? Excited? Powerful? A little nauseous? Yes, we've all been there.

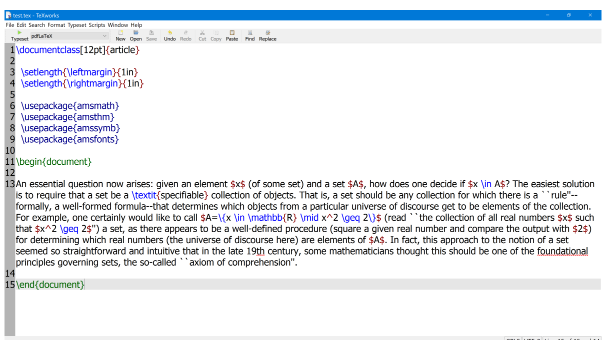
But I want to ask you a few questions. Every time something was offset by dollar signs in the code, do you see what effect that had on the typesetting of the characters in between the dollar signs? Do you see what effect the command `\textit` had on the word "specifiable"? Do you see what effect the command `\mathbb` had on "R"? It really is a brave new world, isn't it?

But we're not done yet. Not by a long shot.




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

 \setlength{\leftmargin}{1in}
 \setlength{\rightmargin}{1in}

  \usepackage{amsmath}
 \usepackage{amsthm}
 \usepackage{amssymb}
 \usepackage{amsfonts}

\begin{document}

An essential question now arises: given an element $x$ (of some set) and a set $A$, how does one decide if $x \in A$? The easiest solution is to require that a set be a \textit{specifiable} collection of objects. That is, a set should be any collection for which there is a ``rule''--formally, a well-formed formula--that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call $A=\{x \in \mathbb{R} \mid x^2 \geq 2\}$ (read ``the collection of all real numbers $x$ such that $x^2 \geq 2$'') a set, as there appears to be a well-defined procedure (square a given real number and compare the output with $2$) for determining which real numbers (the universe of discourse here) are elements of $A$. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called ``axiom of comprehension''.

\end{document}
```

FIGURE 2.2: Preamble material versus document material in a LATEX source code file



An essential question now arises: given an element x (of some set) and a set A, how does one decide if $x \in A$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call $A = \{x \in \mathbb{R} \mid x^2 \geq 2\}$ (read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 2.3: Example of LATEX output

2.3 When Good Code Goes Bad

Before you fancy yourself too big and bad, I want to give you a small taste of the reaction of LATEX to certain issues that arise. Don't worry, I'll start gently.

Suppose you made a mistake and forgot the backslash in front of the command `\in` on the first line. Go ahead and remove it and recompile your work (uh, push the little green right-pointing triangle). You should see the output given in Figure 2.4. LATEX successfully compiled, but the effect is obviously not what you intended.

The point is that LATEX does not assume what your intentions are here—after all, it might have been a case in which you truly wanted to express x times i times n times A. In addition, don't miss the fact that LATEX does not respect spaces you leave in your code. If you want spaces between the x and i and the n and A, you actually have to put them in the code by way of a space macro (see Table 3.4 for some of the options along these lines).

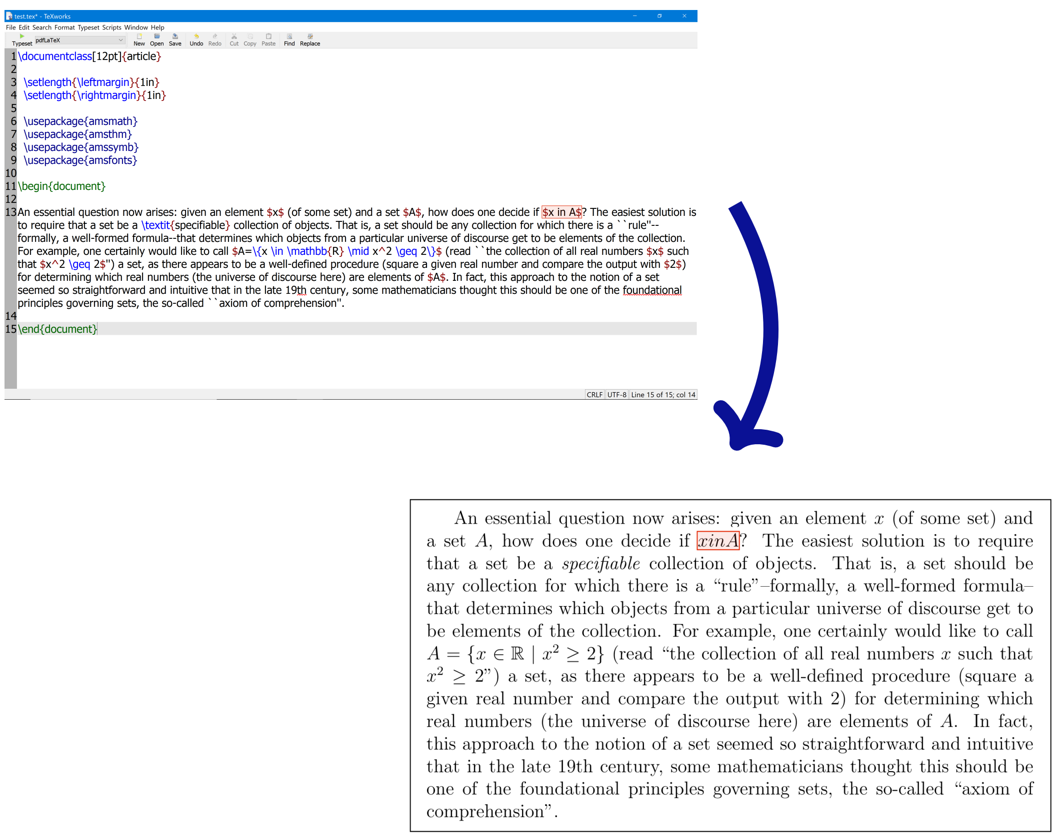


FIGURE 2.4: Effect of removing the backslash from a LATEX command

Suppose now that you really intended for this one paragraph to be two, say starting at the words "In fact" in the last sentence. Go ahead and hit return immediately before these two words, so that the last sentence begins on the line immediately below the second-to-last sentence. That is, your code should now look like Figure 2.5.

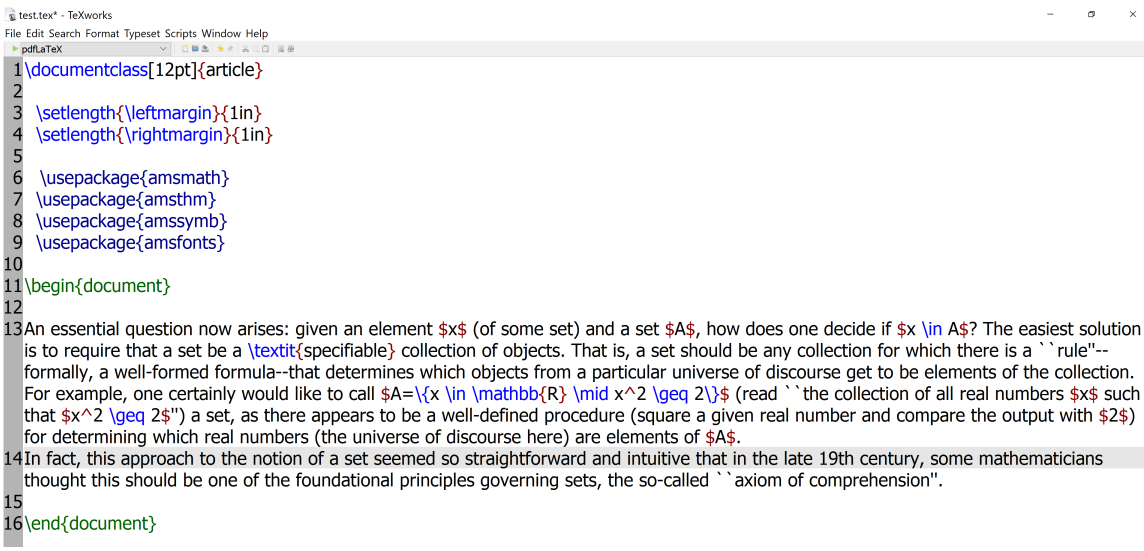


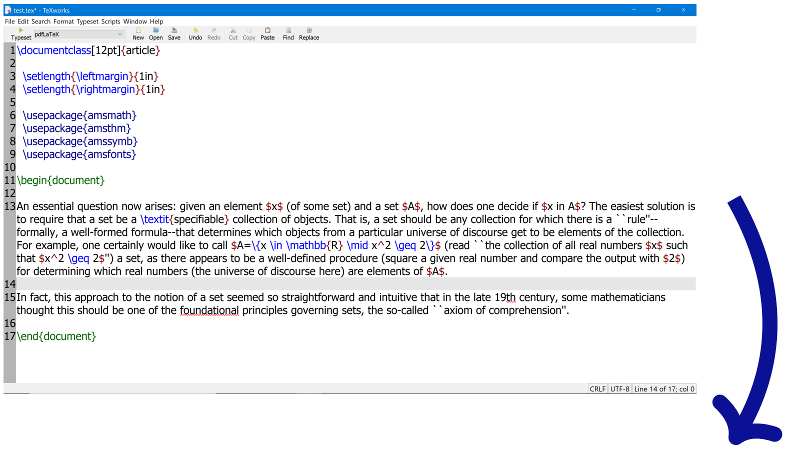
FIGURE 2.5: Failed attempt to create two separate paragraphs in LATEX source code file

You know the drill. Push the little green right-pointing triangle and recompile your document. Fantastic, wasn't it?! What!, you didn't see any change to your document? Exactly. Remember above when I said that LATEX does not respect spaces that you leave in your code. Well, that applies here, as well. When you put a line directly below another line in LATEX, it still assumes that you intend for these two lines to be associated to each other. So, how then does one create a separate paragraph? The answer is found in a small qualification I should make to what I have been telling you about LATEX and spaces. You see, I should have said that LATEX does not respect *small* spaces you leave in your code. However, it will acknowledge *large* spaces. So, go ahead and hit return again just before the words "In fact" so that there is now a whole line gap between the second-to-last and last sentences. Recompile and you should see the desired output, which is given in Figure 2.6.

So far, not too bad, right? Well, now the gloves comes off! I want you to go to your code and *remove* the first backslash in the math environment that starts with `$A=`, so that your code there now looks like

```
$A={x \in \mathbb{R} \mid x^2 \geq 2\}$.
```

You should already anticipate a major problem, as there is a lack of symmetry between the first type of curly brace `{` and the last type of curly brace `\}` in this statement. Go ahead and recompile and see what happens. You should find that LATEX stopped compiling at this line and threw up an error message, as given in Figure 2.7.



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

 \setlength{\leftmargin}{1in}
 \setlength{\rightmargin}{1in}

 \usepackage{amsmath}
 \usepackage{amsthm}
 \usepackage{amssymb}
 \usepackage{amsfonts}

\begin{document}

An essential question now arises: given an element $x$ (of some set) and a set $A$, how does one decide if $x in A$? The easiest solution is
to require that a set be a \textit{specifiable} collection of objects. That is, a set should be any collection for which there is a ``rule''--
formally, a well-formed formula--that determines which objects from a particular universe of discourse get to be elements of the collection.
For example, one certainly would like to call $A=\{x \in \mathbb{R} \mid x^2 \geq 2\}$ (read ``the collection of all real numbers $x$ such
that $x^2 \geq 2$'') a set, as there appears to be a well-defined procedure (square a given real number and compare the output with $2$)
for determining which real numbers (the universe of discourse here) are elements of $A$.

In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians
thought this should be one of the foundational principles governing sets, the so-called ``axiom of comprehension''.

\end{document}
```

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $x \in A$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call $A = \{x \in \mathbb{R} \mid x^2 \geq 2\}$ (read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A.

In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 2.6: LATEX output showing the successful creation of two separate paragraphs

```
! Missing } inserted.
<inserted text>
                }
l.15 ...ll $A={x \in \mathbb{R} \mid x^2 \geq 2\}$
                                                    (read ``the collection of...

?
```

FIGURE 2.7: Example of a LATEX error message

In other words, this is an issue that LATEX does not want to automatically try to work around. Thankfully, in this case, LATEX told you what was at issue (namely, a curly brace) and where to find it (l.15 refers to line 15 in the code). Now, if your document is pretty large and the error occurs on, say, line 347, don't fret; TEXworks will not make you count from line 1 to line 347 to finally find the error. If you access the drop-down menu under FORMAT you will find an option to show line numbers, which should allow you to quickly find the line of your error. Now, to fix your mistake, you must first tell LATEX to stop the compiling process by clicking on the red octagon with the white X in the middle that has now appeared in place of the little green right-pointing triangle. You can then go to the appropriate line number and repair the problem.

Of course, I say all of this rather glibly. The truth of the matter is that it is often times quite a challenge to figure out exactly what the problem is that LATEX has encountered and where it is. For instance, if you have an error somewhere in a math environment that spans multiple lines of code, LATEX will only tell you that there is an issue on the last line of the code since it regards the entire math environment as effectively one command. I can tell you from personal experience that it is quite the exercise in anger management when you have an environment spanning more than 200 lines of code and LATEX tells you that you are missing a curly brace somewhere amongst more than 100. Calm blue ocean, calm blue ocean.

A comment about %

If you are unsure about how to fix a problem in your code, one really good practice is to "comment out" parts of your code around where you believe the problem is. In LATEX, when you place the percentage symbol `%` in front of a part of code, LATEX ignores the rest of that line when it compiles again. By doing this, you need not delete something that may, in fact, turn out to be perfectly fine code. It also gives you the capability of easily making changes and seeing what impact those changes have. (By the way, since the percentage symbol is reserved for this purpose, if you need to type the percentage symbol in your text somewhere, use `\%`.

In Chapter 15, I have collected some of the most common error messages that LATEX gives and how you can generally deal with them.

Congratulations! You have now conquered (and yes, I will use that word) some of the most basic features of using LATEX. The next chapter will take you a little further into other aspects of using LATEX that come up in thesis or dissertation writing.

2.4 Resources

You might find it a bit funny that I have placed a section on websites and books to which you can turn for additional help in a chapter on getting started. But I wish to communicate to you from the very beginning that LATEX is so deep that all of us need some assistance from time to time. Hopefully by putting this section here, it will be easy enough for you to find it again *when* you hit the proverbial brick wall and need some place to turn to ask questions (other than this manual, of course).

Let me also say that this manual is not designed to be a full-blown exposition on all the LaTeX macros that you may need as you develop your thesis or dissertation (see the first item in the recommended resources below for that purpose). However, I have created a very small sampling starting on page 2 that demonstrates some of the most frequently used pieces of mathematical notation in such works. And if that does not suffice to answer your questions, here are a few resources that I would recommend:

- **symbols-a4.pdf** Maintained by Scott Pakin, this document lists (I kid you not) 18,150 symbols and the corresponding LaTeX commands. It is the most current list of its type right now by far. Just be aware that some of these symbols require you to load special packages, although the exact packages needed are clearly indicated above the table in which the desired symbol is given.
- **tug.org** TUG stands for TeX Users Group. It is a nice place to turn to for short articles concerning general information and procedures.
- *Practical LaTeX* by George Grätzer (ISBN 978-3319064246) Grätzer is one of the biggest names in LaTeX manuals. He has published numerous texts on the topic, including an extensive treatment entitled *Math Into LaTeX* and *More Math Into LaTeX* (ISBN 978-3319237954). In *Practical LaTeX*, Grätzer attempts to just hit some of the most common issues when creating documents in LaTeX.
- **tex.stackexchange.com** So, admittedly this question-and-answer forum is one of my favorite places to turn to find answers to my LaTeX questions. However, you may find more here in the area of "advanced" questions concerning LaTeX, such as coding using LaTeX. But you often do come across some basic queries that may be helpful to novice users. Be careful, though, as many responses are far more complicated than necessary (although its ranking system for answers almost always puts the best answers first).
- **en.wikibooks.org/wiki/LaTeX** This is a great place to turn for basic information about LaTeX. It's very easy to navigate and is well-organized.
- **ctan.org** CTAN stands for Comprehensive TeX Archive Network and represents the largest repository of packages for LaTeX in the world (at the writing of this manual, it contained 6,273 packages). If you need a package not included in the MikTeX distribution, this would be the first place to look for it. In addition, for each of the packages on this site, there is an associated user's document with it to illustrate how to use the corresponding macros of the package, which can be valuable in its own right.

And here are a few sites to proceed into with a little caution:

- **latex.org** This is a question-and-answer forum, but it seems like only one or two people do all of the answering. Once in a blue moon, I'll find a useful answer at this site, but I tend to have greater success with other sites.

- **stackoverflow.com** Stack Overflow has a format that is very similar to that of Stack Exchange, except there is much more in the way of traffic on Stack Exchange, and so there are far more answers and categories there. In fact, it is not uncommon to see some who repsond to LATEX questions on Stack Overflow by telling the individual that they should go to Stack Exchange to ask his or her question.

- **overleaf.com** Overleaf sells itself as a free online LATEX editor (well, free unless you want your advisor, for example, to be able to edit your work from Overleaf; then it's $15 a month). On one hand, a completely online LATEX editor is obviously the next natural evolution of editor. Moreover, if you are a student who is beholden to using a university-owned computer to type up your work, then such an administrator-free system is the way to go (unless you particularly enjoy contacting ITS every time you want to download a new package to your computer).

  However, as someone who has used LATEX for more than 20 years, I would advise you to be greatly wary of anything that claims it allows you to effortlessly learn LATEX in 30 minutes. While it has tools that facilitate certain aspects of the process, you should not be seduced into thinking that Overleaf will provide you with everything that you want. The power of LATEX stems from the fact that it can put on the page whatever you dream up. And power like that often requires a little leg work to be done on the part of the user.

Chapter 3

Some Basic Tools

You know that there is a problem with the education system when you realize that out of the 3 R's, only one actually begins with an R.

Dennis Miller

IN this chapter, I cover a few special groups of commands that will allow you to start crafting some impressive looking pieces of both text and mathematics. These sections are designed to give you the tools necessary to start really individualizing your work.

3.1 Put Your Best Font Forward

In writing, there is nothing that catches the eye quite like the font style itself. In fact, an entire discipline called typography is dedicated to the study of how font and letter placement affects how a reader interprets what he or she reads. LaTeX has a plethora of commands to create variations on font appearance. The standard options that are available to you are provided in Table 3.1.

At least for the Computer Modern[1] font style, there is no difference amongst normal, upright, middle weight, and roman. However, Computer Modern is in the serif class of font, which means that the sans serif version created by using `\textsf` produces a different appearance to the respective words (compare the top of the letter "f" in the word "fun" for both normal and sans serif).

With respect to italicization, LaTeX offers you three options. While the command `\textit` produces what you might consider a "standard" italicization of the input[2] , `\textsl` yields a variation that is a little less pretentious. On the other hand, the `\emph` command is more of a switch. If the surrounding environment is already italicized, then `\emph` will typeset its input as normal text; otherwise, it will typeset its input in italics. This action can be particularly useful for ensuring that certain words stand out even if the environment changes (say by a cranky editor of your work).

[1] Computer Modern is a variant of Latin Modern, in which your thesis or dissertation is typeset.

[2] In `TNThesis`, all theorem environments are typeset in this type of italics.

Table 3.1: Standard LaTeX text font options

Name	Example of Syntax	Output
normal	`\textnormal{Math is fun!}`	Math is fun!
upright	`\textup{Math is fun!}`	Math is fun!
middle weight	`\textmd{Math is fun!}`	Math is fun!
roman	`\textrm{Math is fun!}`	Math is fun!
boldface	`\textbf{Math is fun!}`	**Math is fun!**
italics	`\textit{Math is fun!}`	*Math is fun!*
emphasis	`\emph{Math is fun!}`	*Math is fun!*
slant	`\textsl{Math is fun!}`	*Math is fun!*
small caps	`\textsc{Math is fun!}`	MATH IS FUN!
teletype	`\texttt{Math is fun!}`	`Math is fun!`
sans serif	`\textsf{Math is fun!}`	Math is fun!
underline	`\underline{Math is fun!}`	Math is fun!

The teletype option typesets its input in a simple, monospaced variant of the font. The namesake comes from the fact that the appearance resembles that which was once produced by typewriters (if you don't know what a "typewriter" is, I'm certain you will find one in the Smithsonian). This style is the standard for typesetting programming code in a document.

While `TNThesis` sets the font of your thesis or dissertation for you, you should not be afraid to experiment with other fonts for other documents. A wonderful place to begin your journey is at the website tug.org/FontCatalogue. There you will find a wealth of options, including examples of the fonts, whether or not a particular font has a compatible math font, and the basic procedure on how to apply each font to your work (usually effected by using the `\usepackage` command).

But what if you need to use one of the font options in Table 3.1 on an entire *section* of text instead of just a few words? For such a purpose, LaTeX offers variations on these commands that can apply to an entire environment (and so care has to be taken to encapsulate the affected section of text with curly braces `{...}` if there are not natural starting and stopping points to the section, such as given by `\begin{environment name}`...`\end{environment name}`; see Section 4.2 for more details). These variations are given in Table 3.2.

TABLE 3.2: Variations on LaTeX text font commands

Standard Command	Variation
`\textnormal`	`{\normalfont Math is fun!}`
`\textup`	`{\upshape Math is fun!}`
`\textmd`	`{\mdseries Math is fun!}`
`\textrm`	`{\rmfamily Math is fun!}`
`\textbf`	`{\bfseries Math is fun!}`
`\textit`	`{\itshape Math is fun!}`
`\textsl`	`{\slshape Math is fun!}`
`\textsc`	`{\scshape Math is fun!}`
`\texttt`	`{\ttfamily Math is fun!}`
`\textsf`	`{\sffamily Math is fun!}`

LaTeX also has a very nice cache of commands to create variations on *math* font appearance, as well. These are given in Table 3.3.

You can even put together combinations of these, as in

```
\underline{\textbf{\textsl{boldfaced slanted underlined}}}
```

if you want ***boldfaced slanted underlined***. Perhaps though it is best not to get too crazy with this.

Boldly going where no man has gone before

! The use of the `\mathbf` command is regrettably limited to letters and numbers. In particular, the syntax `$\mathbf{\pi}$` will simply print the non-boldfaced symbol π. To get around this issue, use instead the command `\boldsymbol`, which will boldface not only letters and numbers, but mathematical symbols, as well. For example, `$\boldsymbol{\sqrt{2\pi}}$` will produce $\boldsymbol{\sqrt{2\pi}}$.

[3] One of my favorite variants of blackboard bold is called “double stroke”. Place the command `\usepackage{dsfont}` in the preamble of your document and then use the command `\mathds` in place of `\mathbb`. This will produce the following style: $\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$.

Table 3.3: Standard LaTeX math font options

Name	Example of Syntax	Output
math boldface	`$\mathbf{ABC...}$`	$\mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathbf{abcdefghijklmnopqrstuvwxyz}$ $\mathbf{0123456789}$
math blackboard bold[3]	`$\mathbb{ABC...}$`	$\mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
math calligraphic	`$\mathcal{ABC...}$`	$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$
math script	`$\mathscr{ABC...}$`	$\mathscr{ABCDEFGHIJKLM}$ $\mathscr{NOPQRSTUVWXYZ}$
math fraktur	`$\mathfrak{ABC...}$`	$\mathfrak{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathfrak{abcdefghijklmnopqrstuvwxyz}$ $\mathfrak{0123456789}$
math roman	`$\mathrm{ABC...}$`	$\mathrm{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathrm{abcdefghijklmnopqrstuvwxyz}$ $\mathrm{0123456789}$
math teletype	`$\mathtt{ABC...}$`	$\mathtt{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathtt{abcdefghijklmnopqrstuvwxyz}$ $\mathtt{0123456789}$
math sans serif	`$\mathsf{ABC...}$`	$\mathsf{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathsf{abcdefghijklmnopqrstuvwxyz}$ $\mathsf{0123456789}$
math italics	`$\mathit{ABC...}$`	$\mathit{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$ $\mathit{abcdefghijklmnopqrstuvwxyz}$ $\mathit{0123456789}$

3.2 When You Just Need a Little Space

Sometimes you need to compel LaTeX to force a line break. For this task, you can use either the double backslash `\\` or the command `\newline` where you want the break to occur. I offer both since there are times one will work whereas the other will not; it just depends upon the environment.

If you need to create a special type of spacing in your document, then you can use `\hspace*{⟨length⟩}` or `\vspace*{⟨length⟩}` if you want a horizontal spacing or a vertical spacing, respectively. Since you asked, ⟨*length*⟩ may be expressed in any of the units described in Figure 3.1.[4]

For example, `\hspace*{1.5in}` will create a horizontal space of 1.5 inches to the right precisely where the command is invoked in your code and `\hspace*{-2mm}` will create a horizontal space of 2 millimeters to the *left* precisely where the code is invoked in your code.

There are also "unstarred" versions of `\hspace*` and `\vspace*`. The difference is that the unstarred versions of these spacing macros allow TeX to override the request if its internal algorithms would not have generally allowed the space where the macro is called. The starred versions, on the other hand, tell TeX to give priority instead to the request for additional space (which, by the way, is what I assume you want by using the macro in the first place). For example, since TeX's algorithm naturally clears out all available space at the beginning of a new line of text, if you type

```
\hspace{3em} I want this new line indented by a length of 3 ems.
```

at the beginning of a new line, TeX will *ignore* the command `\hspace{3em}` and typeset the line flush with the left margin (assuming, of course, that no other protocols are in place). Using `\hspace*{3em}` instead of `\hspace{3em}` will produce the desired effect. In general then, I would stick with using `\hspace*` and `\vspace*` unless there is a very specific reason you would want to use the unstarred versions.

On the topic of spaces, there are also the commands the create a fixed amount of horizontal spacing upon their calling. These are given in Table 3.4.

These commands can be used in combination with each other to create small spaces of a variety of lengths. For instance, `\quad\!` will create a space of (positive) length 5/6 em. (Notice that the smallest positive length that you can create in this fashion is 1/18 em.) And, yes, there are times in which you may wish to seriously consider shrinking a certain space naturally provided by LaTeX. For instance, consider the output

$$a \equiv b \pmod{m}$$

provided by the code `a \equiv b \pmod{m}`. Here, LaTeX gives a horrendous amount of spacing between b and (mod m). To remedy this issues, we can make use of the negative spacing commands in Table 3.4. In particular, the code

```
a \equiv b \negthickspace\negthickspace\pmod{m}
```

[4] I would point out that in math mode, there exist units called "math units" (mu), where 18 mu = 1 em; you can access these units by using the analogous macro `\mspace{⟨length⟩}`.

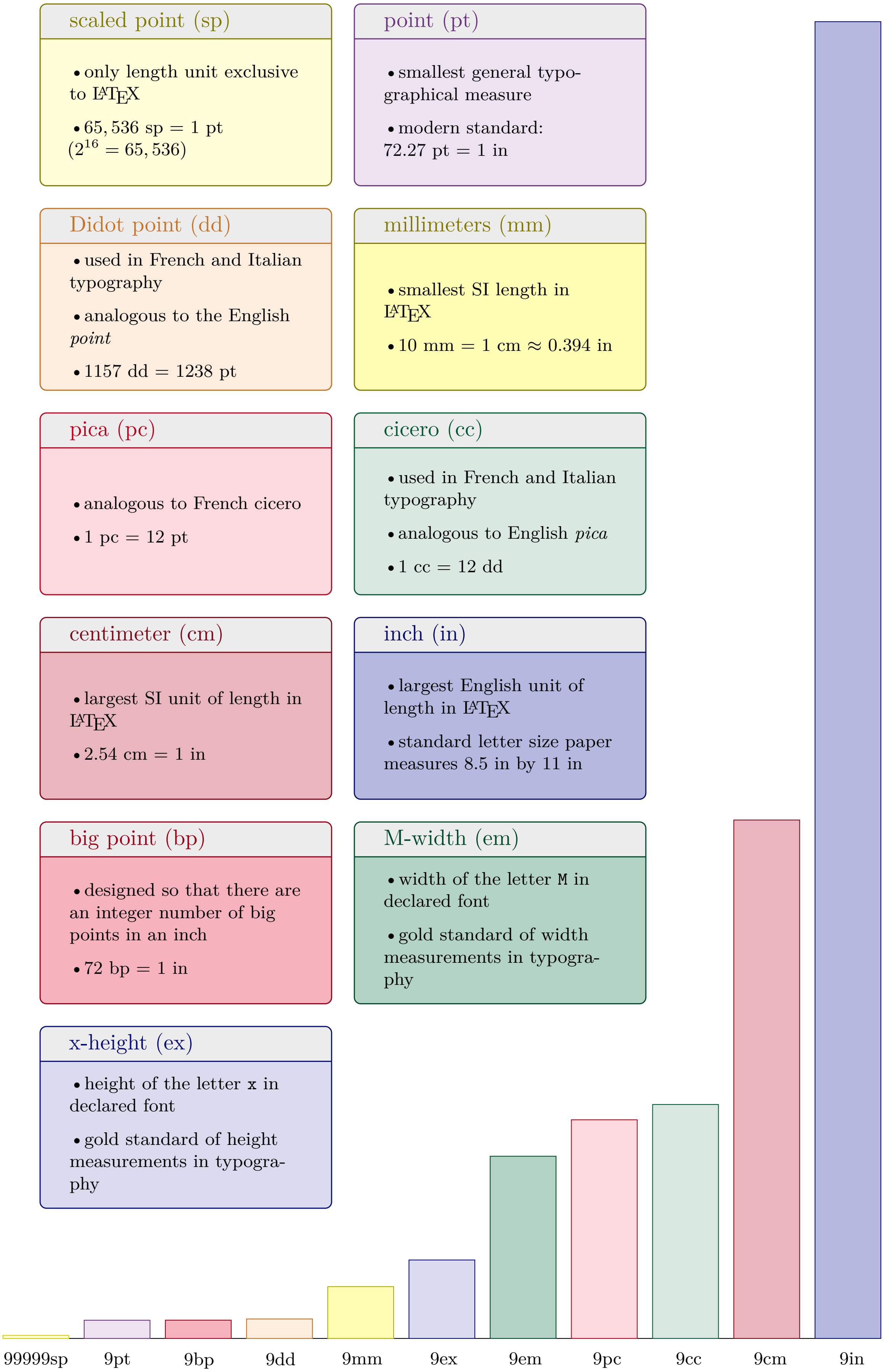


FIGURE 3.1: Length units in LaTeX

TABLE 3.4: Horizontal space commands in LaTeX

Command	Length	Visual
`\qquad`	2 em	→| |←
`\quad`	1 em	→| |←
`\enspace`	1/2 em	→| |←
`\ `	text space	→| |←
`\thickspace` or `\;`	5/18 em	→| |←
`\medspace` or `\:`	4/18 em	→| |←
`\thinspace` or `\,`	3/18 em	→| |←
`\negthinspace` or `\!`	-3/18 em	→||←
`\negmedspace`	-4/18 em	→| ←
`\negthickspace`	-5/18 em	→| ←

produces the (far more satisfying) output

$$a \equiv b \pmod{m}$$

Of course, some experimentation will probably be necessary to achieve the effect that you want.

And if you really want to push things...and I do mean *literally* push things...the macros `\hfill` and `\vfill` are designed to fill *all* of the available horizontal and vertical space, respectively, in a fixed space. For example, the code

```
Here is a \hfill large gap.
```

will produce the following result (without the frame):

Here is a large gap.

since the "fixed space" in this case is the single line that the sentence occupies (well, actually the invisible horizontal box that the line resides in, but that's a bit of a technical detail). Or say you want to spread the sentence evenly out across the entire line. Then use the code

```
Here\hfill is\hfill a\hfill large\hfill gap.
```

which produces (again without the frame)

Here is a large gap.

Not as though you are aiming to create sentences in your thesis or dissertation that look like either of the above examples, but `\hfill` and `\vfill` can be useful when it comes to the placement of other parts of your work, such as figures and tables.

3.3 Of Subscripts and Superscripts

Not a great deal of mathematics can be done without the ability to handle subscripts and superscripts. In LaTeX, if you wish to create a subscript, use the underscore key, as in `$x_i$` to produce x_i. On the other hand, if you wish to create a superscript, use the caret key, as in `$x^i$` to produce x^i. You can even do both at the same time; for example, x_i^j is created by `$x_i^j$`. Be careful however: if your subscript or superscript is more complicated than a single character (what pros call a "token"), you will need to encase the entire subscript or superscript in curly braces, as in `$x_{ij}^{k-m}$` to produce x_{ij}^{k-m}. This way LaTeX knows what your intentions are. More to the point, `$x_{i^j}$` will produce x_{i^j} and `$x^{i_j}$` will produce x^{i_j}. One of the ultimate examples of nested scripting is given by the cardinal number `$\aleph_{\aleph_{\aleph_{\ddots_{\aleph_0}}}}$` which produces

$$\aleph_{\aleph_{\aleph_{\ddots_{\aleph_0}}}}$$

Or perhaps you need to indicate that a certain operation is being conducted with regard to a range of values indexed by one or more variables (say, for example, a sum from $i = 1$ to $i = n$). Wonderfully, the LaTeX syntax is the same for handling this context as the previous one. See Figure 3.2, Figure 3.3, Figure 3.4, Figure 3.5, and Figure 3.6 for example.

By the way, do you see why the 13 needed to be surrounded by curly braces in Figure 3.4? That's right! because the number 13 consists of more than one character and you do not want LaTeX thinking that you want to multiply together fractions of the form $3m/(m + 1)$ from $m = 1$ to $m = 1$.

Sometimes it is relatively challenging to express certain operators with respect to a simple range of indexing values. For example, suppose you want to express a sum of all of the natural numbers less than or equal to 100 that are relatively prime to 100 (that is, have a greatest common divisor of 1 with 100). How would you accomplish this in LaTeX? The answer is by using the macro `\substack`. Specifically, if you wish to "stack" the mathematical expressions ⟨*expression 1*⟩, ⟨*expression 2*⟩, . . . , ⟨*expression n*⟩ in the subscript position, the syntax is

```
\substack{⟨expression1⟩\\⟨expression 2⟩\\...\\⟨expression n⟩}
```

So, with respect to our summing problem, we could write the expression

```
\[\sum_{\substack{n \leq 100 \\ \text{gcd}(n,100)=1}} n\]
```

which produces

$$\sum_{\substack{1\leq n\leq 100\\ \text{gcd}(n,100)=1}} n$$

The only rather unfortunate aspect of using `\substack`, as you may have noticed above, is that the operand is artificially shifted to (supposedly) accommodate the required width of the substacked expression. However, many times this shifting is really unnecessary. To alleviate the issue, use either the macro `\mathclap` or the macro `\smashoperator`. In this case, I would personally type

```
\[\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}\]
```

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

FIGURE 3.2: LaTeX code showing the use of _ and ^ on `\sum`

```
\[\sum_{n=1}^\infty \frac{1}{n} = \infty\]
```

$$\sum_{n=1}^\infty \frac{1}{n} = \infty$$

FIGURE 3.3: LaTeX code showing the use of _ and ^ on`\sum`

```
\[\prod_{m=1}^{13} \frac{m}{m+1} = \frac{1}{14}\]
```

$$\prod_{m=1}^{13} \frac{m}{m+1} = \frac{1}{14}$$

FIGURE 3.4: LaTeX code showing the use of _ and ^ on `\prod`

```
\[\bigcap_{k=1}^\infty [-1/k,1/k] = \{0\}\]
```

$$\bigcap_{k=1}^\infty [-1/k,1/k] = \{0\}$$

FIGURE 3.5: LaTeX code showing the use of _ and ^ on `\cap`

```
\[\lim_{n \to \infty} a_n = L\]
```

$$\lim_{n \to \infty} a_n = L$$

FIGURE 3.6: LaTeX code showing the use of _ on `\lim`

```
\[\smashoperator{\sum_{\substack{n \leq 100 \\ \text{gcd}(n,100)=1}}} n\]
```

which produces

$$\sum_{\substack{1 \leq n \leq 100 \\ \text{gcd}(n,100)=1}} n$$

(By the way, the value of this sum is 2000. Do you know why without actually adding up all the respective n's?)

3.4 Going Greek (and a Little Hebrew)

If the mathematics that you are dealing with in your thesis or dissertation is beyond basic algebra, I would be willing to bet that you are using some Greek letters. In fact, depending upon the area that you are studying, you may even need a little bit of Hebrew. So, this final section of Chapter 3 is dedicated to providing you with the corresponding commands through Table 3.5, Table 3.6, and Table 3.7. Be aware, though, that the commands in these tables must be used in math mode.

TABLE 3.5: Commands for lowercase Greek letters

Syntax	Output	Syntax	Output	Syntax	Output
`\alpha`	α	`\iota`	ι	`\sigma`	σ
`\beta`	β	`\kappa`	κ	`\tau`	τ
`\gamma`	γ	`\lambda`	λ	`\upsilon`	υ
`\delta`	δ	`\mu`	μ	`\phi`	ϕ
`\epsilon`	ϵ	`\nu`	ν	`\chi`	χ
`\zeta`	ζ	`\xi`	ξ	`\psi`	ψ
`\eta`	η	`\pi`	π	`\omega`	ω
`\theta`	θ	`\rho`	ρ		
`\varepsilon`	ε	`\varpi`	ϖ	`\varsigma`	ς
`\vartheta`	ϑ	`\varrho`	ϱ	`\varphi`	φ
`\digamma`	$\digamma$	`\varkappa`	$\varkappa$		

TABLE 3.6: Commands for some Hebrew letters

Syntax	Output	Syntax	Output
`\aleph`	$\aleph$	`\beth`	$\beth$
`\daleth`	$\daleth$	`\gimel`	$\gimel$

TABLE 3.7: Commands for uppercase Greek letters

Syntax	Output	Syntax	Output	Syntax	Output
`\Gamma`	Γ	`\Xi`	Ξ	`\Phi`	Φ
`\Delta`	Δ	`\Pi`	Π	`\Psi`	Ψ
`\Theta`	Θ	`\Sigma`	Σ	`\Omega`	Ω
`\Lambda`	Λ	`\Upsilon`	Υ		
`\varGamma`	$\varGamma$	`\varXi`	$\varXi$	`\varPhi`	$\varPhi$
`\varDelta`	$\varDelta$	`\varPi`	$\varPi$	`\varPsi`	$\varPsi$
`\varTheta`	$\varTheta$	`\varSigma`	$\varSigma$	`\varOmega`	$\varOmega$
`\varLambda`	$\varLambda$	`\varUpsilon`	$\varUpsilon$		

Chapter 4

On Display

It ain't what a man knows that gets him into trouble.
It's what he knows for sure that just ain't so.

Mark Twain

Now that you've got your feet wet with some of the basic tools of LaTeX, it's time to take you further into how to present certain expressions in LaTeX the way a professional mathematician would. In particular, I want to show you here how you can "display" material in LaTeX.

4.1 One Singular Expression

Often in mathematical writing, you want to separate out a special formula or statement from the rest of the surrounding text because either

(1) you wish to emphasize the importance of the formula or statement, or

(2) the mathematical expression is involved enough that leaving it inline with the rest of the surrounding text would make it very difficult for a reader to see the important details of the expression.

This separated expression is what we call a **displayed** formula or statement, and to create such a thing in LaTeX is a very straightforward matter.

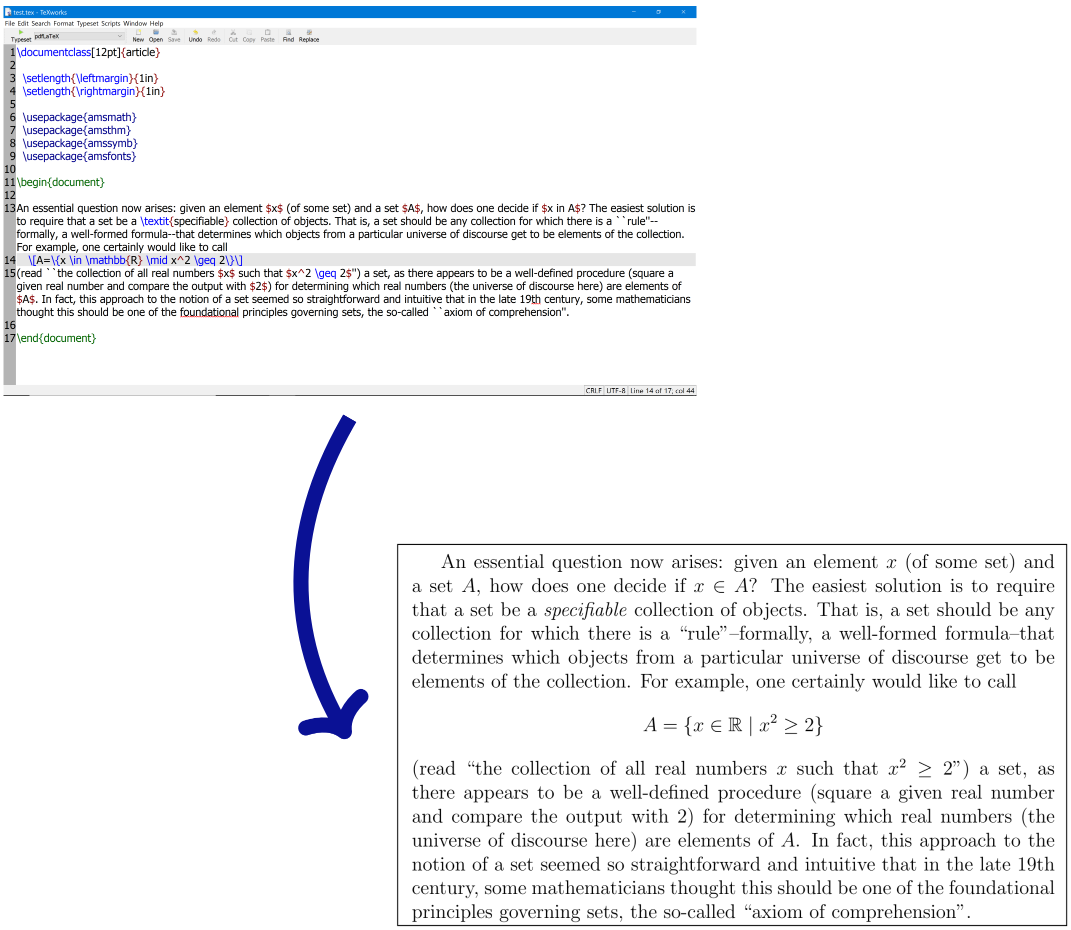
Let's return to our original sample text from Chapter 2. You know, the one that starts with "An essential question now arises". (Be sure your program still has all four packages `amsmath`, `amsthm`, `amssymb`, and `amsfonts` loaded.) Suppose you want to display the formula

```
A=\{x \in \mathbb{R} \mid x^2 \geq 2\}
```

(ironically the way we just did with the code). To do so, instead of single dollar signs surrounding the formula, encase the formula in the delimiters `\[` and `\]`, so that you now have

```
\[A=\{x \in \mathbb{R} \mid x^2 \geq 2\}\].
```

Compile and you should see the output given in Figure 4.1. Do you see now what we mean by a displayed formula? It's much easier to notice this formula with this type of presentation.




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

 \setlength{\leftmargin}{1in}
 \setlength{\rightmargin}{1in}

 \usepackage{amsmath}
 \usepackage{amsthm}
 \usepackage{amssymb}
 \usepackage{amsfonts}

\begin{document}

An essential question now arises: given an element $x$ (of some set) and a set $A$, how does one decide if $x in A$? The easiest solution is to require that a set be a \textit{specifiable} collection of objects. That is, a set should be any collection for which there is a ``rule''--formally, a well-formed formula--that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call
   \[A=\{x \in \mathbb{R} \mid x^2 \geq 2\}\]
(read ``the collection of all real numbers $x$ such that $x^2 \geq 2$'') a set, as there appears to be a well-defined procedure (square a given real number and compare the output with $2$) for determining which real numbers (the universe of discourse here) are elements of $A$. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called ``axiom of comprehension''.

\end{document}
```

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $x \in A$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call

$$A = \{x \in \mathbb{R} \mid x^2 \geq 2\}$$

(read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 4.1: Example of a displayed formula

You can also display important text, if you wish. For example, let's say that you want to display the statement

`given an element $x$ (of some set) and a set $A$, how does one decide if $x \in A$?`

in the first sentence. To do this, we need to exercise a little more care, as just offsetting the statement with `\[...\]` will compel LaTeX to believe that the entire thing should be in math mode and that certainly is not what we're going for here. We instead need to tell LaTeX what should be in math

Avoiding the almighty double dollar

If you are already quite familiar with LaTeX, you may be in the habit of displaying formulas by means of surrounding statements using double dollar signs $$...$$. While I personally have never had an issue with doing this, if LaTeX has a problem with what you are trying to display, you may get a cryptic error message that will have you pulling your hair out trying to determine what went wrong. It is considered better practice to use the delimiters `\[` and `\]` (or if you're a workaholic, the environment `\begin{displaymath}`...`\end{displaymath}`) instead of the double dollar signs.

An alternative to the dollar (sign)

Instead of using dollar signs for an inline math environment, you can instead use either the delimiters `\(` and `\)` or `\begin{math}`...`\end{math}` (analogizing the delimiters `\[...\]` and `\begin{displaymath}`...`\end{displaymath}`, respectively, for a displayed formula). In fact, an argument can be made for using `\(...\)` over `$...$`, as the former allows LaTeX to provide clearer error messages should something go wrong in an inline math environment.

font and what should be in normal font (sometimes called roman font). Since the entire statement will be offset using `\[...\]`, there will no need to offset the x, A, or $x \in A$ using dollar signs. So, what remains is to separate out the rest as normal text. LaTeX has the beautifully simple macro `\text` for doing just this:

```
\[\text{given an element }x\text{ (of some set) and a set}
A\text{, how does one decide if }x \in A\text{?}\]
```

Try it and you should get the output given in Figure 4.2.

An essential question now arises:

given an element x (of some set) and a set A, how does one decide if $x \in A$?

The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call $A = \{x \in \mathbb{R} \mid x^2 \geq 2\}$ (read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 4.2: Example of displayed text

4.2 The Equation Environment

Often when you display a formula, you need to make reference to it in the text somewhere. For thesis and dissertation writing, you are not permitted to simply say things like "the equation below shows that..." or "the above equation indicates..." as perhaps you would in article writing. It is considered to be just too informal. So, *if* you intend to refer to your displayed material in the text, you will need to also display a reference number to go with it. The standard numbering scheme along these lines is according to section, offset by parentheses, and right-justified on the same line as the displayed material (although all of these aspects can be changed if need be). How is this accomplished in LaTeX? Well, instead of the `\[...\]` to separate off a displayed formula, we use what is called an "equation environment" (which, incidentally, can also be used to number a displayed piece of text like the previous example). **Environments** are structures in LaTeX capable of handling multiple lines of code with the goal of creating a special section of mathematics and/or text in the work. Any environment in LaTeX has the following format:

```
\begin{⟨environment name⟩}
    ⟨text⟩
\label{⟨name⟩}
\end{⟨environment name⟩}
```

\label→p.43

This is the means we will use to provide a displayed reference number to our previous displayed formula. To that end, go back to the document that we've been working on and, instead of `\[...\]` around the formula, encase it using an environment that begins with `\begin{equation}` and ends with `\end{equation}`. Your code should then look like the following:

```
For example, one certainly would like to call
\begin{equation}
    A=\{x \in \mathbb{R} \mid x^2 \geq 2\}
\end{equation}
(read ``the collection of all real numbers $x$ such that $x^2 \geq 2$'')
a set, as there appears to be a well-defined procedure...
```

(note that you could have written

```
\begin{equation} A=\{x \in \mathbb{R} \mid x^2 \geq 2\}\end{equation}
```

but for the sake of readability, I prefer to place the beginning and ending commands for an environment on separate lines of code). Now compile. This time you should see a "(1)" to the far right of the displayed formula, as in Figure 4.3. So in the rest of your work, you can refer to the formula as "Equation (1)".

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $x \in A$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call

$$A = \{x \in \mathbb{R} \mid x^2 \geq 2\} \tag{1}$$

(read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 4.3: Example of some output of the `equation` environment

Whatever happens, don't blank out!

Make sure that you do not have any blank lines inside a displayed math or equation environment. Otherwise, LaTeX will throw up an error message. In addition, don't place a blank line between your text and the beginning of a displayed math or equation environment. Otherwise, you will end up with too much space above the environment when it is typeset.

4.3 References and Labels

Speaking of referring to your equation (or any other object in LaTeX for that matter), there is an easy way to do this that saves you from having to worry about mismatches if the numbers change (say after a late addition of another formula that you place before this one). We are going to give the equation a label and then make reference to the label in the discussion that follows. To provide the label, we use the macro

```
\label{⟨name⟩}
```

While we could give our equation any name we want (e.g., equation 1, the main equation, da bomb equation), it would be best to give it a name that is short and descriptive. Let's then call it “the A equation”. To make the associated label, we simply have to type `\label{the A equation}` somewhere inside the corresponding `equation` environment. Go ahead and do just that. (While LaTeX will recognize the label anywhere inside the `equation` environment, it may be best to get into the habit of placing the label *immediately before* the command `\end{equation}` as placement will make a difference for figures and tables; see page 113).

Now, just below the `equation` environment, we are going to refer to the equation in the parenthetical statement. This can be accomplished by the macro

```
\ref{⟨name⟩}
```

So, at the beginning of the parenthetical statement, write the following: `(\ref{the A equation} is read ``the collection of all real numbers...` and compile.

Twice as nice just before the end

! Many tasks in LaTeX, especially tasks as big as compiling a thesis or dissertation, require that you compile twice in order to get the correct effect. If you are not seeing what you want in your document after compiling—especially when it comes to the agreement of references and citations with their sources—try compiling a second time and see if that resolves the issue. If not, check to make sure that the corresponding `\label` commands appear just before the `\end{⟨environment name⟩}` commands in which they are contained.

What happened? Do you see two boldfaced question marks in place of the equation number? Does this mean we did something wrong in the coding? Have I been lying to you this entire time about the `equation` environment? Will you be able to trust anybody ever again? I'm sure your mind is whirling with questions and, while I can't speak to the last question, I can tell you that to get the correct connection between labels and references, LaTeX often requires a second compilation. So, go ahead and compile a second time. Now, there should be a 1 in place of the two question marks and the connection is now complete (see Figure 4.4). Feel better?

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $xinA$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call

$$A = \{x \in \mathbb{R} \mid x^2 \geq 2\} \tag{1}$$

(Equation **??** is read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $xinA$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call

$$A = \{x \in \mathbb{R} \mid x^2 \geq 2\} \tag{1}$$

(Equation 1 is read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

FIGURE 4.4: Agreement of reference and source number after second compilation

How to make your references clever

There are, in fact, even faster ways to create references that can automatically change if you decide to change the *name* of the object to which you are referring. This can be especially useful if you decide to, for example, turn a theorem into a proposition or turn a question into a conjecture. Such macros are called "clever references" and are called by using either `\Cref` or `\cref` (use the former anywhere you need to capitalize the name of the respective environment).

For example, instead of writing

```
Equation \ref{the A equation}
```

as discussed in Section 4.3, you could simply write

```
\Cref{the A equation}
```

(notice that you no longer need the name "Equation"—clever references will automatically provide the name). Then if you decide later to call it an example, `\Cref` will automatically change the reference name once you change the corresponding environment name from "equation" to "example".

`TNThesis` already has this feature built into it, but if you are going to try it outside the context of your thesis or dissertation, you will need to add `\usepackage{cleveref}` to your preamble. Also, don't forget that you will need to compile twice to see the overall result.

Dancing with the stars

! Almost any LaTeX environment (and many a macro) has an associated "starred" version. In particular, there is a LaTeX environment with the syntax

```
\begin{equation*}
    ⟨equation⟩
\label{⟨name⟩}
\end{equation*}
```

\label→p.43

This simply means that the associated type of environment there will be unnumbered. So, instead of our use of `\[...\]` above, we could have used the starred version of the `equation` environment (which, of course, would not be as efficient).

4.4 Size Matters

There are times when, depending upon what you're displaying, the delimiters (things like curly braces, square brackets, vertical bars, and parentheses) look rather puny next to the surrounding mathematical material. This can certainly happen when you have an indexed operator like $\sum$ or $\prod$ in your display, where LaTeX will naturally put the indexing limits above and below the operator. This causes the limits to poke out beyond the natural height of any delimiters in the vicinity. There are a couple of things that you can do to remedy this unsightly problem. In this section, we cover your options in this regard.

4.4.1 \left, \left, \left, \right, \left

The easiest thing to do is to replace any square brackets in your displayed formula with `\left[` and `\right]`, any parentheses with `\left(` and `\right)`, any vertical bars with `\left|` and `\right|`, and any *printed* curly braces with `\left\{` and `\right\}` (of course you do want to be careful not to replace any curly braces used to signify an argument to a macro with `\left\{` and `\right\}`). These modified macros will tell LaTeX to resize the delimiters to match the height of the tallest overall structure in the displayed formula.

For example, consider the following displayed equation:

$$\frac{\log x}{x}(\frac{1+x+x^2}{\log x + 1}).$$

While the parentheses are quite visible and communicate precisely what you want here, they are nevertheless completely overwhelmed by the rest of the mathematical structure. It would appear far better to make use of the `\left` and `\right` operators on these delimiters:

```
\[\frac{\log x}{x}\left(\frac{1+x+x^2}{\log x + 1}\right).\]
```

which produces

$$\frac{\log x}{x}\left(\frac{1+x+x^2}{\log x + 1}\right).$$

As another example, consider expressing a set of 2×2 (or larger) matrices using set notation:

$$A=\{\begin{bmatrix} a & b \\ c & d\end{bmatrix} : ad-bc=0\}.$$

My goodness that looks awful! The matrix is completely overwhelming the curly braces here. What is more desirable is to emphasize that A is a *set* of matrices rather than a set of *matrices*. To accomplish this, we simply use the following code:

```
\[A=\left\{\begin{bmatrix*}[r] a & b \\ c & d\end{bmatrix*} : ad-bc=0\right\}.\]
```

which produces the output

$$A=\left\{\begin{bmatrix} a & b \\ c & d\end{bmatrix} : ad-bc=0\right\}.$$

Take it to "de limit"

In addition to parentheses, brackets, braces, and vertical lines, LaTeX offers the following additional delimiters:

TABLE 4.1: Additional LaTeX delimiters

Macro	Output	Macro	Output
`\langle`	$\langle$	`\rangle`	$\rangle$
`\lfloor`	$\lfloor$	`\rfloor`	$\rfloor$
`\lceil`	$\lceil$	`\rceil`	$\rceil$
`\uparrow`	$\uparrow$	`\downarrow`	$\downarrow$
`\Uparrow`	$\Uparrow$	`\Downarrow`	$\Downarrow$
`\updownarrow`	$\updownarrow$	`\Updownarrow`	$\Updownarrow$
`\ulcorner`	$\ulcorner$	`\urcorner`	$\urcorner$
`\llcorner`	$\llcorner$	`\lrcorner`	$\lrcorner$

Of course, each of these macros may be prepended with either `\left` or `\right` (as appropriate) should the situation call for it.

All seems right with the world again, doesn't it?

I should point out that you really should not overuse `\left` and `\right`, but rather reserve them for times in which you know you need to increase the size of a delimiter (e.g., expressions involving fractions or matrices). This is because using `\left` and `\right` creates a certain amount of extra space around the corresponding delimiter (as LaTeX anticipates that you have a structure around the delimiter that warrants the increased space). So, if you overuse these macros, your expressions will definitely start to look like there's a little too much space separating them.

4.4.2 `\Bigg`*ie Size It*

There are times when the use of `\left` and `\right` simply will not suffice to achieve a certain desired look in your displayed formula. For example, say you want to communicate the fact that the absolute value of the absolute value of a real number x is the absolute value of x. If you type `[||x||=|x|\]`, you will get

$$||x|| = |x|$$

which, while correct, can potentially mislead a cursory reader to think you are talking about the *norm* of the *vector* x (and while, yes, the standard norm on the real numbers coincides with absolute value, that's not quite what you are trying to communicate here, now is it). Furthermore, this issue isn't fixed by using the above macros, for `\[\left|\left|x\right|\right| = \left|x\right|\]` yields the exact same thing!

In situations like this, you will need to tell LaTeX that you want a particular set of delimiters to be bigger than others for a given displayed equation. For the example above, the outer absolute value bars should be made to be slightly bigger than the inner absolute value bars. This can be effected by use of the macros `\big⟨token⟩`, `\Big⟨token⟩`, `\bigg⟨token⟩`, and `\Bigg⟨token⟩`, where ⟨*token*⟩ represents a delimiter such as `(`, `)`, `[`, `]`, `\{`, `\}`, and `|` (notice the lack of curly braces to signify input into these macros; the token you wish to enlarge is always placed immediately after the macro name here). So, to fix our problem regarding the absolute value of the absolute value of x, we could write `\[\big||x|\big| = |x|\]` which gives

$$\big||x|\big| = |x|$$

A side-by-side comparison of these particular macros is given in Table 4.2.

Table 4.2: Relative sizes of enlargement macros applied to the token |

`|`	`\big|`	`\Big|`	`\bigg|`	`\Bigg|`
$|$	$\big|$	$\Big|$	$\bigg|$	$\Bigg|$

Another use for these particular sizing macros is found by revisiting the previous problem of correctly sizing the curly braces for a set of 2×2 matrices. Suppose for this set you wish to use the "pipe" symbol | instead of the colon to express "such that". If you attempt to just slap the macro `\mid` (which in general is the appropriate macro to use for such circumstances) in place of the colon, this is what you will get:

$$A = \left\{\begin{bmatrix} a & b \\ c & d \end{bmatrix} \mid ad - bc = 0\right\}.$$

Yikes! To fix this issue, we need to forego the usual `\mid` macro and instead tap into the corresponding delimiter | along with a suitable sizing macro from Table 4.2. We certainly want something bigger than the plain | but not bigger than the surrounding curly braces for the set. After some trial and error, you will probably find that `\bigg` works best here (along with some extra space to the right of the pipe; see Table 3.4):

```
\[A=\left\{\begin{bmatrix*}[r] a & b \\ c & d\end{bmatrix*} \bigg|\,\,
ad-bc=0\right\}.\]
```

$$A = \left\{\begin{bmatrix} a & b \\ c & d \end{bmatrix} \bigg|\,\, ad - bc = 0\right\}.$$

If you prefer, you can use `\vert` in place of |; it will have exactly the same effect.

And just in case none of these particular enlargement macros is quite up to the task, I have created the following exclusively for `TNThesis`:

```
\SuperSize{⟨mag factor⟩}⟨delimiter⟩
```

where *⟨mag factor⟩* is the magnification factor beyond the size given by `\Bigg` (e.g., *⟨mag factor⟩* = 1.5 would correspond with a size 50% bigger than `\Bigg`) and *⟨delimiter⟩* is the delimiter that one wishes to resize. So, for example, `\SuperSize{1.2}\{` will create a left curly brace that is 20% larger than the one created by `\Bigg\{`.

Of course, size concerns are not merely limited to particular delimiters in a displayed formula. If you are using a formula in your work that frequently makes use of subscripts of subscripts or something on that level, it may be challenging for you reader to see these details even if you display the formula. In such a situation, you may wish to consider enlarging the *entire* formula. This can be accomplished through a set of sizing macros that are already part of the underlying `memoir` class to `TNThesis`. Recall that the default font size for `TNThesis` is 12 pt, which is then identified by the class through the macro `\normalsize`. Given this default, other commands and their corresponding font sizes are provided in Table 4.3.

TABLE 4.3: Sizing commands and their corresponding font sizes assuming a standard of 12 pt

Sizing command	Font size (pt)
`\miniscule`	7
`\tiny`	8
`\scriptsize`	9
`\footnotesize`	10
`\small`	11
`\normalsize`	12
`\large`	14
`\Large`	17
`\LARGE`	20
`\huge`	25
`\Huge`	30
`\HUGE`	36

Should you feel the need to implement a size change to a displayed formula, call the necessary command *just before* the displayed environment (i.e., just before `\[`, `\begin{equation}`, or any similar such environment beginning). Then call the command `\normalsize` *immediately after* the end of the environment (i.e., just after `\]`, `\end{equation}`, etc.)

Don't poke the bear!

! Technically, the Graduate School allows only 12 pt font to be used in a thesis or dissertation. However, a special case can certainly be made for mathematical formulae, as readability demands a bigger font size sometimes. But, the (real) point is not to get crazy with it. The Graduate School will allow you to increase the font only just to the point that your displayed equation can be comfortably read AND NOT BEYOND!

One is the loneliest delimiter

Usually delimiters must occur in corresponding pairs (i.e., every left bracket must have a corresponding right bracket), otherwise LaTeX pitches a fit. There is however a way to work around this if, say, you need a right bracket without the corresponding left bracket (for instance, as used in calculus to represent the difference of the evaluation of a function at the upper and lower limits). The trick is to fool LaTeX into thinking that the companion delimiter is really there when it's not. To accomplish this, use either the syntax `\left.` or the syntax `\right.` to hold the place of the missing delimiter. For example, the syntax `$\left. F(x) \right]_a^b$` will typeset as $\left. F(x) \right]_a^b$ and cause no errors in compiling.

Seeing the interval as both half-open and half-closed

Speaking of issues with typing one delimiter without its companion, you might wonder then how on earth one typesets a half-open, half-closed interval such as $[0, \infty)$. I'm glad you asked. `TNThesis` calls a special package `interval` that handles this situation nicely. In particular, this package provides the commands

```
\ointerval{⟨start⟩}{⟨end⟩}
\rinterval{⟨start⟩}{⟨end⟩}
\linterval{⟨start⟩}{⟨end⟩}
```

where `\ointerval`, `\rinterval`, and `\linterval` create an open interval, a right open-left closed interval, and a left open-right closed interval, respectively. So, the code for $[0, \infty)$ is given by `$\rinterval{0}{\infty}$`. (By the way, while you can use the standard set of parentheses `( , )` to create an open interval, using `\ointerval` provides better spacing around the starting and ending values.)

4.5 Alignment

The last issue concerning displayed material that I want to cover in this chapter concerns alignment of two or more formulas. As with a lot of things in LaTeX, there are oh so many options.

4.5.1 *The* `align` *Environment*

Say in our test document that we want to write

$$
\begin{aligned}
A &= \{x \in \mathbb{R} \mid x^2 \geq 2\} && (4.1)\\
&= \{x \in \mathbb{R} \mid x \geq \sqrt{2}\} \cup \{x \in \mathbb{R} \mid x \leq -\sqrt{2}\} && (4.2)
\end{aligned}
$$

so that the equals signs are lined up with each other. This can be achieved by means of, what else, but the "align" environment. Instead of using `\begin{equation}` and `\end{equation}`, replace these with `\begin{align}` and `\end{align}`, respectively. In addition, we need to tell LaTeX where to line up the formulas and where to transition from the first to the second line. For the former task, we use the ampersand symbol `&` and for the latter task, we use the basic line break command `\\`. The complete syntax for our example then looks like

```
\begin{align} \label{the A equation}
    A &=\{x \in \mathbb{R} \mid x^2 \geq 2\} \\
        &=\{x \in \mathbb{R} \mid x\geq\sqrt{2}\}
         \cup \{x \in \mathbb{R} \mid x\leq -\sqrt{2}\}
\end{align}
```

(By the way, since the ampersand symbol is reserved for this purpose, if you need to type the ampersand symbol in your text somewhere, use `\&`.)

What's that you say? you need to align several "if and only if" statements. No problem. The exact same structure can be used, just put the ampersand symbol `&` wherever you want the alignment to take place. So, if you need to align several "if and only if" statements at the double-headed arrow, it would look like the following example:

```
\begin{align} \label{set distributivity}
x \in A\cap(B\cupC) &\Leftrightarrow x\in A \text{ and } x\in B \cup C \\
        &\Leftrightarrow x\in A, \text{ and } x\in B \text{ or } x\in C \\
        &\Leftrightarrow x\in A \text{ and } x\in B, \text{ or } x\in A
                         \text{ and } x\in C  \\
        &\Leftrightarrow x \in (A\cap B)\cup(A\cap C)
\end{align}
```

And, if you do not want your equations to be numbered, use `\begin{align}*` and `\end{align}*` instead.

4.5.2 *The* `alignat` *Environment*

Let's say that you have several sets of equations that you want to display in multiple (invisible) columns. For example,

$$\begin{alignat*}{3}
&f(x) &&= x^2 \quad & h(x) &= x^4\\
&g(x) &&= x^3 \quad & k(x) &= x^5
\end{alignat*}$$

For this job, you need an adaptation of the traditional `align` environment called the `alignat` environment. The idea is basically the same as the `align` environment in that the ampersand symbol `&` will be used to determine where to line certain items up while the double backslash `\\` will be used to indicate where to move to the next line.

There is some additional care that has to be exercised in this environment: how do you tell LaTeX where to line up different *sets* of objects. The answer is found in the use of a mandatory argument that follows `\begin{alignat}`. In particular, by using the syntax `\begin{alignat}{n}`...`\end{alignat}`, you are telling the environment that you are creating n groupings, so that the environment will then line up the first ampersands `&` on each line with each other, the second ampersands `&` on each line with each other, etc. In particular, the above output was generated with the following code:

```
\begin{alignat*}{3}\label{power functions}
    &f(x) &=x^2 \quad & h(x)=x^4\\
    &g(x)&=x^3        \quad & k(x)=x^5
\end{alignat*}
```

`\quad`→p.31
The `\quad` is there simply to provide the desired spacing between the two groups of equations (and, in fact, we only needed one such `\quad` rather than two). If you wish even greater control on the spacing between your columns, use the `\hspace*` macro. By the way, the use of the asterisk after `alignat` is there to ensure our equations are unnumbered. If you prefer each row numbered, leave off the asterisk.
`\hspace`→p.31

4.5.3 *The* `aligned` *Environment*

One of the major downsides to the `align` environment is that it *cannot* be used in math mode—that is, you cannot put an `align` environment inside another displayed environment. This is precisely why we did *not* use the syntax `\[\begin{align}...\end{align}\]` in the "the A equation" above; the `align` environment in effect creates its own math mode.

However, there are times in which you might want something of a "mini" `align` environment inside a bigger one. Take, for instance, the issue of having to display a relatively lengthy formula as one in a series of aligned formulas (perhaps one of most troublesome general problems I've come across). One of the best solutions that I have found is to call into service another alignment-type environment that is designed to work in math mode, namely the `aligned` environment. This environment will let you set up a separate system of ampersand symbols `&` and basic line break commands `\\` for the overly long expression.

Here is an example of the `aligned` environment in action:

```
\begin{align*} \label{Example of Aligned}
\sum_{n \leq x} f(n) &=
    \begin{aligned}[t] f(x)\lfloor x \rfloor
        &+ \int_1^x (t- \lfloor t \rfloor)f^\prime(t)\,dt\\
        &- \left(xf(x) - f(1) - \int_1^x f(t)\,dt\right)\end{aligned}\\
    &= \int_1^x f(t)\,dt + \int_1^x (t - \lfloor t \rfloor)f^\prime(t)\,dt
        +f(x)(\lfloor x \rfloor - x) + f(1)
\end{align*}
```

And here is the corresponding printout:

$$\begin{aligned}\sum_{n \leq x} f(n) &= \begin{aligned}[t] f(x)\lfloor x \rfloor &+ \int_1^x (t- \lfloor t \rfloor)f^\prime(t)\,dt\\ &- \left(xf(x) - f(1) - \int_1^x f(t)\,dt\right)\end{aligned}\\ &= \int_1^x f(t)\,dt + \int_1^x (t - \lfloor t \rfloor)f^\prime(t)\,dt +f(x)(\lfloor x \rfloor - x) + f(1)\end{aligned}$$

If you're curious about the optional argument [**t**] immediately after `\begin{aligned}`, it is there to control vertical placement of the `aligned` system relatively to the surrounding display. Specifically, your options here are as follows:

t lines up the top of the `aligned` material with the surrounding material; most frequently used option for placing an `aligned` environment inside of an `align` environment

b lines up the bottom of the `aligned` material with the surrounding material

<no option given> a plain `\begin{aligned}`...`\end{aligned}` environment will line up the center of the `aligned` material with the surrounding material

4.5.4 *The* `gather` *Environment*

The `gather` environment is used when you want to list out a series of equations and simply want each one centered with respect to its geometric middle. For example, the code

```
\begin{gather*}\label{some functions}
    f(x)=2x^2 - 3x + 1 \\
    g(x)=x^3 \\
    h(x)=\sin(2x-1)
\end{gather*}
```

will produce the following output:

$$f(x) = 2x^2 - 3x + 1$$
$$g(x) = x^3$$
$$h(x) = \sin(2x - 1)$$

Once again, if you wish each equation to have a corresponding number listed right-justified on the page, simply remove the asterisk from `gather*`.

4.5.5 *The* `multline` *Environment*

The `multline` environment is specifically designed to handle lengthy expressions that would otherwise have gone into the margin. Here you use only the double backslash `\\` symbol to indicate where you want your expression broken. The protocol that `multline` follows is to typeset the initial part (i.e., before the first `\\`) flushleft in the displayed environment, the last part (i.e., after the last `\\` is typeset flushright in the displayed environment, and all middle parts are typeset halfway between the first and last parts.

For example, to typeset

$$\begin{multline*}
f(x) = 2020x^{2020} + 2019x^{2019} + 2018x^{2018} + 2017x^{2017} + 2016x^{2016} + 2015x^{2015} \\
+ 2014x^{2014} + 2013x^{2013} + 2012x^{2012} + 2011x^{2011} + 2010x^{2010} \\
+ \cdots + 2x^2 + x + 42
\end{multline*}$$

use the code

```
\begin{multline*}\label{a really long polynomial}
    f(x)=2020x^{2020}+2019x^{2019}+2018x^{2018}+2017x^{2017}+2016x^{2016}+
      2015x^{2015}\\+2014x^{2014}+2013x^{2013}+2012x^{2012}+2011x^{2011}+
      2010x^{2010}\\+\cdots+2x^2+x+42
\end{multline*}
```

As a cautionary point, `multline` cannot be used in math mode. In particular, you cannot use `multline` in an `align` environment to handle certain lines that are too long. For that you will need to use either the `aligned` or `split` environment.

4.5.6 *The* `split` *Environment*

The `split` environment works in a similar manner to the `align` environment, except

(1) it gives one number for the entire system of equations that you are displaying, and

(2) it *must* be used in math mode (which include other alignment-type environments)

For example, let's return to our original code of

```
\begin{align} \label{the A equation}
A&=\{x \in \mathbb{R} \mid x^2 \geq 2\} \\
  &=\{x \in \mathbb{R} \mid x\geq\sqrt{2}\}
    \cup \{x \in \mathbb{R} \mid x\leq -\sqrt{2}\}
\end{align}
```

If we instead replace the `align` environment with `split` and surround the entire structure with `\begin{equation}` and `\end{equation}`, we get the following code:

```
\begin{equation}
\begin{split} \label{the A equation}
A&=\{x \in \mathbb{R} \mid x^2 \geq 2\} \\
  &=\{x \in \mathbb{R} \mid x\geq\sqrt{2}\}
    \cup \{x \in \mathbb{R} \mid x\leq -\sqrt{2}\}
\end{split}
\end{equation}
```

This produces the following result:

$$\begin{split} A&=\{x \in \mathbb{R} \mid x^2 \geq 2\} \\ &=\{x \in \mathbb{R} \mid x\geq\sqrt{2}\} \cup \{x \in \mathbb{R} \mid x\leq -\sqrt{2}\} \end{split} \tag{4.3}$$

which may be more suitable for reference purposes in your work.

If number, reference you must

! Before you go hog-wild numbering every single displayed formula, bear in mind that proper mathematical writing requires that you number *only* those formulas and equations that are referenced in the actual text of your work. Otherwise, leave the formula unnumbered and use the display feature simply as a means to call greater attention to the particular expression.

Chapter 5

Diagrams

Always remember that you are absolutely unique.
Just like everyone else.

Margaret Mead

> In order to use the `tikz` package discussed in this chapter, you will need to specify it as an option in the `TNThesis` document class. In particular, you will need to use the syntax `\documentclass[tikz]{TNThesis}` at the beginning of `TNTmain`.

PERHAPS it's the algebraist in me, but one of the largest categories of figures that I think are perfectly suited for display in a mathematical work is the class of function diagrams. In particular, commutative diagrams such as

$$\begin{tikzcd} A \arrow[r, "\iota"] \arrow[dr, "\phi"] & F(A) \arrow[d, "\Phi"] \\ & G \end{tikzcd}$$

where A is a set, G is a group, $F(A)$ is the free group on A, $\phi\colon A \to G$ is a set map, $\iota\colon A \to F(A)$ is the canonical inclusion, and $\Phi\colon F(A) \to G$ is the unique group homomorphism such that $\phi = \Phi\iota$ leap to mind (as I'm sure they do for you, too). Diagrams like this are an important way for mathematicians to quickly and impactfully communicate certain relationships (in this case, between a free group and another group). The nature of a commutative diagram is that, for a fixed starting point and ending point, it does not matter over which "path" of functions an element travels, the functional result will be the same. In particular, the use of the "path" phrasing seems to indicate that a visual presentation is the best way to understand what is happening (especially for more complicated commutative diagrams). LaTeX can produce these diagrams through the `cd` library in the `tikz` package using many of the tools already presented in Chapter 4 *On Display*.

5.1 Simple Triangular Diagrams

All of the diagrams discussed in this chapter are created through a special environment:

```
\begin{equation}
\begin{tikzcd}
    ⟨code⟩
\end{tikzcd}
\end{equation}
```

equation→p.42

where ⟨*code*⟩ describes to the environment how to draw the particular diagram. It's important to point out that, for display purposes, the `tikzcd` environment should be encased in an environment such as `equation` or `\[...\]` (the latter if numbering of the diagram is not desired). I'm assuming, after all, that you're not trying to do something crazy like place a function diagram inline with your text.

Now the basic tools for constructing a function diagram such as the one above consist only of the ampersand symbol `&`, the double backslash `\\`, and the special macro `\arrow`. (Those of you who have read through the alignment-type environments from Section 4.5 will already be familiar with how the ampersand symbol and double backslash symbol will be used here.) The macro `\arrow` has the following syntax:

```
\arrow[⟨arrow options⟩]{⟨direction⟩}⟨label⟩
```

where `⟨arrow options⟩` is a comma-separated list of properties that the corresponding arrow should have (e.g., dashed, hookrightarrow) and ⟨*label*⟩, if you choose to use this option, takes the form

`{⟨label⟩}[⟨label options⟩]`⟨*other labels*⟩

`⟨direction⟩` is expressed by using a *combination* of the following letters:

r pointing right

l pointing left

d pointing down

u pointing up

For example, if you wish to have an arrow in your diagram that points up and to the right, you would type `\arrow{ur}` at the appropriate location.

Just as with the alignment-type environments of Section 4.5, the ampersand symbol `&` is used to indicate the vertical alignment of certain objects in the diagram while the double backslash `\\` is used to indicate a transition to the next line of the diagram.

Abbreviating too muc

! If you are already somewhat familiar with the `tikz` package, then you know that there are (very) shortened versions of the arrow macros. In particular, you can use `\ar` instead of `\arrow`, `\rar` for "right arrow", `\dlar` for "down left arrow", etc. However, I think it is possible to be overly brief with notation to the point where you can lose sight of what the notation represents. So, unless your thesis or dissertation is all about function diagrams, I would recommend sticking with the traditional `\arrow` macro.

Know your options first

It turns out that the `\arrow` macro is *very* picky when it comes to the ordering of arguments that follow the macro. Specifically, you must indicate ⟨*arrow options*⟩ *before* ⟨*direction*⟩, which itself must come *before* ⟨*label*⟩. Otherwise, LaTeX will throw up errors.

So, to draw the commutative diagram at the beginning of the chapter, use the following code:

```
\[
  \begin{tikzcd}
    A \arrow{rd}{\phi} \arrow{r}{\iota} & F(A) \arrow{d}{\Phi} \\
    & G
  \end{tikzcd}
\]
```

Of course, if you would rather have your diagram numbered, exchange the `\[...\]` for `\begin{equation}`...`\end{equation}`. Notice that the ampersand symbol `&` before $F(A)$ and G tells the environment to place $F(A)$ directly above G.

But what if you want to organize this diagram differently, say in the following way:

$$\begin{tikzcd}
 & F(A) \arrow[dd, "\Phi"] \\
A \arrow[ur, "\iota"] \arrow[dr, "\phi"'] & \\
 & G
\end{tikzcd}$$

This is an easy enough change to make. The ampersand symbol can be used multiple times to, in essence, shift objects horizontally to a different "column" of the diagram. For the above diagram, I would use the code

```
\[
    \begin{tikzcd}
        && F(A) \arrow{dd}{\Phi} \\
        A \arrow{rrd}{\phi} \arrow{rru}{\iota} \\
        && G
    \end{tikzcd}
\]
```

The double ampersand symbol in front of both $F(A)$ and G indicates to the environment that $F(A)$ and G are to be aligned vertically with each other, *but* occupy a space that is two columns to the right of the column occupied by A (since A has no ampersand symbol in front of it). Also notice that each of the arrow directions changed to account for $F(A)$ and G occupying a different column and different row than A. In particular, `\arrow{rrd}{\phi}` tells the environment that the arrow needs to be extended two columns to the right and one row down, which is what we need in order to get the arrow to "hit" G from A. Also, don't forget the need to add an extra double backslash `\\` to the system to indicate that A is now on a different row than $F(A)$.

5.2 Arrow Options

Upon a closer look at the diagram at the beginning of the chapter, you would probably agree with me that the function name ϕ would look better if it were on the *outside* of the diagram rather than the inside. This is an easy enough change to make by using one of the many optional arguments for the `\arrow` macro. In particular, simply replace `\arrow{rd}{\phi}` with `\arrow{rd}[swap]{\phi}` to produce

$$\begin{tikzcd} A \arrow[r, "\iota"] \arrow[dr, swap, "\phi"] & F(A) \arrow[d, "\Phi"] \\ & G \end{tikzcd}$$

Much nicer, right? But what if you want to change the style of the arrow, as well? I'm glad you asked! Table 5.1–Table 5.9 illustrate some of the different possibilities for *⟨options⟩* and the style of arrow that will result.

You can even *overlay* different styles on top of each other by using each table's "base" design (given by the red arrows in the tables). For example, `A \arrow[tail, two heads, dashed]{r} & B` produces

$$\begin{tikzcd} A \arrow[r, tail, two heads, dashed] & B \end{tikzcd}$$

So, using the convention that a hooked arrow represents an injection (i.e., a one-to-one function) and a two-headed arrow represents a surjection (i.e., an onto function), we can rewrite our commutative

TABLE 5.1: Standard left and right arrows in the **tikz** package

⟨*option*⟩	arrow
to head	$A \longrightarrow B$
rightarrow	$A \longrightarrow B$
leftarrow	$A \longleftarrow B$
leftrightarrow	$A \longleftrightarrow B$
Rightarrow	$A \Longrightarrow B$
Leftarrow	$A \Longleftarrow B$
Leftrightarrow	$A \Longleftrightarrow B$

TABLE 5.2: Mapping arrows in the **tikz** package

⟨*option*⟩	arrow
maps to	$A \longmapsto B$
mapsto	$A \longmapsto B$
mapsfrom	$A \longmapsfrom B$
Mapsto	$A \Longmapsto B$
Mapsfrom	$A \Longmapsfrom B$

TABLE 5.3: Hook arrows in the **tikz** package

⟨*option*⟩	arrow
hook	$A \hookrightarrow B$
hook'	$A \hookrightarrow B$
hookrightarrow	$A \hookrightarrow B$
hookleftarrow	$A \hookleftarrow B$

TABLE 5.4: Harpoon arrows in the **tikz** package

⟨*option*⟩	arrow
harpoon	$A \rightharpoonup B$
harpoon'	$A \rightharpoondown B$
rightharpoonup	$A \rightharpoonup B$
rightharpoondown	$A \rightharpoondown B$
leftharpoonup	$A \leftharpoonup B$
leftharpoondown	$A \leftharpoondown B$

TABLE 5.5: Squiggly arrows in the **tikz** package

⟨*option*⟩	arrow
squiggly	$A \rightsquigarrow B$
rightsquigarrow	$A \rightsquigarrow B$
leftsquigarrow	$A \leftsquigarrow B$
leftrightsquigarrow	$A \leftrightsquigarrow B$

TABLE 5.6: Headless/tailless arrows in the `tikz` package

⟨option⟩	arrow
no head	$A \;\text{———}\; B$
no tail	$A \longrightarrow B$
dash	$A \;\text{———}\; B$
equal	$A = B$

TABLE 5.7: Dashed arrows in the `tikz` package

⟨option⟩	arrow
dashed	$A \dashrightarrow B$
dashrightarrow	$A \dashrightarrow B$
dashleftarrow	$A \dashleftarrow B$

TABLE 5.8: Two-headed arrows in the `tikz` package

⟨option⟩	arrow
two heads	$A \twoheadrightarrow B$
twoheadrightarrow	$A \twoheadrightarrow B$
twoheadleftarrow	$A \twoheadleftarrow B$

TABLE 5.9: Tailed arrows in the `tikz` package

⟨option⟩	arrow
tail	$A \rightarrowtail B$
rightarrowtail	$A \rightarrowtail B$
leftarrowtail	$A \leftarrowtail B$

diagram:

$$\begin{tikzcd}
A \arrow[r, hook, "\iota"] \arrow[dr, two heads, "\phi"'] & F(A) \arrow[d, two heads, "\Phi"] \\
& G
\end{tikzcd}$$

But let's take this a step further, shall we. Technically, what I'm trying to express through this diagram is the *universal mapping property for free groups* which asserts that for any set map $\phi\colon A \to G$, there exists a *unique* map $\Phi\colon F(A) \to G$ that makes the diagram commute (i.e., $\phi = \Phi\iota$). Moreover, if ϕ is surjective, then so is Φ (as a consequence, any group is isomorphic to a factor group of a free group by taking $A = G$ and ϕ the identity map). Can I communicate all of this through a diagram? You bet! We just need to put two diagrams side-by-side through the following code:

```
\[
    \begin{tikzcd}
        A \arrow[swap, twoheadrightarrow]{rd}{\phi}
            \arrow[hookrightarrow]{r}{\iota} & F(A)\\
        & G
    \end{tikzcd}
    \qquad \implies \qquad
    \begin{tikzcd}
        A \arrow[swap, twoheadrightarrow]{rd}{\phi}
            \arrow[hookrightarrow]{r}{\iota} & F(A)
            \arrow[dashed, two heads]{d}{\exists!\,\Phi}\\
        & G
    \end{tikzcd}
\]
```

\qquad→p.31

and we get

$$\begin{tikzcd} A \arrow[swap, twoheadrightarrow]{rd}{\phi} \arrow[hookrightarrow]{r}{\iota} & F(A)\\ & G \end{tikzcd} \qquad \implies \qquad \begin{tikzcd} A \arrow[swap, twoheadrightarrow]{rd}{\phi} \arrow[hookrightarrow]{r}{\iota} & F(A) \arrow[dashed, two heads]{d}{\exists!\,\Phi}\\ & G \end{tikzcd}$$

In fact, if you really want to spice things up a bit, you can even add color to your arrows through *⟨arrow options⟩*. Simply list the color you want your arrow to be along with any of the other options. Some possibilities for color are given in Table 5.10. So, for example, if you want the arrow representing Φ to be red, then instead of `\arrow[dashed, two heads]`, write `\arrow[dashed, two heads, red]`. Additionally, let's replace `\exists!\,\Phi` with `\color{red}\exists!\,\Phi` since we will also want the label to be red, as well. Finally, let's space things out a little more by using the option `[column sep=large, row sep=large]` immediately after each `\begin{tikzcd}`.[1] All of these changes give

$$\begin{tikzcd}[column sep=large, row sep=large] A \arrow[swap, twoheadrightarrow]{rd}{\phi} \arrow[hookrightarrow]{r}{\iota} & F(A)\\ & G \end{tikzcd} \qquad \implies \qquad \begin{tikzcd}[column sep=large, row sep=large] A \arrow[swap, twoheadrightarrow]{rd}{\phi} \arrow[hookrightarrow]{r}{\iota} & F(A) \arrow[dashed, two heads, red]{d}{\color{red}\exists!\,\Phi}\\ & G \end{tikzcd}$$

A picture really is worth a thousand words, isn't it.

[1] Instead of `large`, you could use any sizing command from Table 4.3 without the preceding backslash.

TABLE 5.10: Standard colors in the `tikz` package

color name	appearance
red	
green	
blue	
cyan	
magenta	
yellow	
black	
gray	
darkgray	
lightgray	
brown	
lime	
olive	
orange	
pink	
purple	
teal	
violet	
white	

5.3 More Exotic Diagrams

As you can imagine, the tools we have thus far discussed can allow you to create a multitude of different diagrams. In this section, I want to take these ideas to the next level to show you what else is possible. I think a few examples, from the basic to complex, should suffice: see Figure 5.1, Figure 5.2, Figure 5.3, Figure 5.4, and Figure 5.5.

```
\[%
    \begin{tikzcd}
        A \arrow{r}{f}\arrow{d}{g} & B \arrow{d}{h}\\
        C \arrow{r}{k} & D
    \end{tikzcd}
\]
```

$$\begin{tikzcd} A \arrow{r}{f}\arrow{d}{g} & B \arrow{d}{h}\\ C \arrow{r}{k} & D \end{tikzcd}$$

FIGURE 5.1: Square diagram using the `tikzcd` package

```
\[%
    \begin{tikzcd}[column sep=large, row sep=large]
        A \arrow{r}{\beta} \arrow[swap]{d}{\delta}
            \arrow[crossing over]{rd}{\quad \epsilon}
        & B \arrow{d}{\alpha}
            \arrow[crossing over]{ld}{\quad \zeta}\\
        C \arrow[swap]{r}{\gamma}
        & D
    \end{tikzcd}
\]
```

$$\begin{tikzcd}[column sep=large, row sep=large]
A \arrow{r}{\beta} \arrow[swap]{d}{\delta} \arrow[crossing over]{rd}{\quad \epsilon} & B \arrow{d}{\alpha} \arrow[crossing over]{ld}{\quad \zeta}\\
C \arrow[swap]{r}{\gamma} & D
\end{tikzcd}$$

FIGURE 5.2: Diagram illustrating the use of the `swap` option for an arrow

```
\[%
    \begin{tikzcd}[column sep=large, row sep=large]
        & E & \\
        A \arrow{ur}{f} \arrow{r}{i_1} &
        C = A \amalg B
            \arrow[dashed]{u}{h} &
        B \arrow{l}{i_2}\arrow{ul}{g}
    \end{tikzcd}
\]
```

$$\begin{tikzcd}[column sep=large, row sep=large]
& E & \\
A \arrow{ur}{f} \arrow{r}{i_1} & C = A \amalg B \arrow[dashed]{u}{h} & B \arrow{l}{i_2}\arrow{ul}{g}
\end{tikzcd}$$

FIGURE 5.3: Diagram illustrating a dashed arrow

```
\[%
    \begin{tikzcd}
        \cdots \arrow{r} & F(i) \arrow{r}{F(g)} \arrow{dr}{\tau_i} &
            F(j) \arrow{r}{F(h)} \arrow{d}{\tau_j} & F(k) \arrow{r}
            \arrow{dl}{\tau_k} & \cdots \\
        && B
    \end{tikzcd}
\]
```

$$
\begin{tikzcd}
\cdots \arrow[r] & F(i) \arrow[r, "F(g)"] \arrow[dr, "\tau_i"] & F(j) \arrow[r, "F(h)"] \arrow[d, "\tau_j"] & F(k) \arrow[r] \arrow[dl, "\tau_k"] & \cdots \\
 & & B & &
\end{tikzcd}
$$

FIGURE 5.4: An example of an infinite diagram using the `tikzcd` package

```
\[%
    \begin{tikzcd}
        && A \\
        E \arrow[bend left]{urr}{f}
            \arrow[dashed]{r}{h}
            \arrow[bend right]{drr}{g} &
        D = A \,\Pi\, B \arrow{ur}{p_1}
            \arrow{dr}{p_2} \\
        && B
    \end{tikzcd}
\]
```

$$
\begin{tikzcd}
 & & A \\
E \arrow[urr, bend left, "f"] \arrow[r, dashed, "h"] \arrow[drr, bend right, "g"] & D = A \,\Pi\, B \arrow[ur, "p_1"] \arrow[dr, "p_2"] & \\
 & & B
\end{tikzcd}
$$

FIGURE 5.5: Diagram illustrating a bent arrow

5.4 Graphs

Along the lines of diagrams, there is perhaps one other area that I should address: how to create graphs (the vertex-and-edge kind, not the function kind) through `TNThesis`. These types of diagrams are not just critical for those doing work in networking or graph theory, but can be a valuable visual tool for those conducting research in the area of partially ordered sets. Nicely enough, the same base package, `tikz`, supplies an environment well-suited for this task.

The challenge here, though, is to determine not just the relative orientation of the vertices (or nodes) with respect to one another, but to determine what coordinates should be associated to each vertex to produce the desired appearance of the graph. (Although `tikz` does provide "help lines" to aid in this process, if you would like to have them.) The basic environment that we will use here is the `tikzpicture` environment:

equation→p.42

```
\begin{equation}
\begin{tikzpicture}[⟨options⟩]
    ⟨code⟩
\end{tikzpicture}
\end{equation}
```

where, just like with the `tikzcd` environment, ⟨*code*⟩ indicates how to draw the graph and `\begin{equation}`...`\end{equation}` is there simply to display the graph and provide an associated number to it. (Of course, if you would rather not have a number associated to your graph, you can either use the starred version of the `equation` environment or just use `\[...\]` instead of `\begin{equation}`...`\end{equation}`).

While `\arrow` was the essential macro for the `tikzcd` environment, `\draw` is the essential macro for the `tikzpicture` environment. The basic syntax for `\draw` as it pertains to a path is as follows:

```
\draw[⟨options⟩] ⟨point 1⟩ -- ⟨point 2⟩;
```

where ⟨*options*⟩ allows you to specify various characteristics of the line segment that will connect ⟨*point 1*⟩ with ⟨*point 2*⟩. Usually, points here are identified with ordered pairs of coordinates as if you would be specifying their locations on a Cartesian plane (although for the drawing of graphs, the coordinates are only meaningful in relation to other points). Be aware that `TNThesis` uses inches for its default measurement, and so that is what will be assumed for your coordinates if you do not explicitly identify some other measurement for them.

Let's do a very simple example: drawing a straight line path from a point A to a point B. To make the code a little more intuitive, we will use the command `\coordinate` to assign a label to a particular ordered pair, so that we can use the label instead of the ordered pair should we have need for the point again later. The basic syntax for `\coordinate` is

Drawing requires semicolons

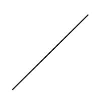
Don't miss the fact that a semicolon `;` is used at the end of every `\draw` command. This is an internal means by which the `tikzpicture` environment knows that you are really done with the present path. In fact, almost all commands in a `tikzpicture` environment require such semicolons.

`\coordinate[`⟨*options*⟩`]` ⟨*label*⟩ `at` ⟨*ordered pair*⟩;

So, our code would look like

```
\[
    \begin{tikzpicture}
        \coordinate (A) at (0,0);
        \coordinate (B) at (1,1);
        \draw (A) -- (B);
    \end{tikzpicture}
\]
```

which produces:



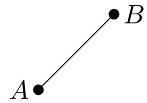
But this seems pretty mundane. What about actually showing the labels at these points along with some dots to represent the actual points. We can accomplish this by means of ⟨*options*⟩ in the `\coordinate` macro along with the `\fill` macro whose syntax is:

`\fill[`⟨*options*⟩`]`⟨*center*⟩ `circle` (⟨*radius*⟩);

Our code then becomes

```
\[
    \begin{tikzpicture}
        \coordinate [label=left:$A$] (A) at (0,0); \fill (A) circle (2pt);
        \coordinate [label=right:$B$] (B) at (1,1); \fill (B) circle (2pt);
        \draw (A) -- (B);
    \end{tikzpicture}
\]
```

which produces



With these simple macros, you really have all the tools you need to design any graph that you want. For instance, Figure 5.6 shows the code to create a rendering of K_5, the complete graph on five vertices:

```
\[%
    \begin{tikzpicture}
        \coordinate [label=left:$A$] (A) at (0,0); \fill (A) circle (2pt);
        \coordinate [label=left:$B$] (B) at (0,1); \fill (B) circle (2pt);
        \coordinate [label=above:$C$] (C) at (1,2); \fill (C) circle (2pt);
        \coordinate [label=right:$D$] (D) at (2,1); \fill (D) circle (2pt);
        \coordinate [label=right:$E$] (E) at (2,0); \fill (E) circle (2pt);
        \draw (A) -- (B); \draw (B) -- (C); \draw (C) -- (D); \draw (D) -- (E);
            \draw (E) -- (A);
        \draw (A) -- (C); \draw (A) -- (D); \draw (B) -- (D); \draw (B) -- (E);
            \draw (C) -- (E);
    \end{tikzpicture}
\]
```

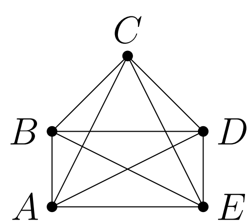


FIGURE 5.6: Code for drawing the complete graph on five vertices using the `tikz` package

However, say we want the above graph to be a *directed* graph–that is, we want each edge of the graph to exhibit a direction. This is usually accomplished by means of arrows instead of line segments for the edges. To replace each line segment with an arrow, we will make use of ⟨*options*⟩ for the `\draw` macro. Specifically, we will replace each `\draw` command with either `\draw [->]` or `\draw [<-]` depending upon which way we want the arrow to go (actually, you could also use `\draw [<->]` for a bidirectional edge). For example, if we want to indicate that the edge connecting points A and B goes *from* A *to* B, then we would use the command `\draw [->] (A) -- (B)` (actually, you may prefer to use `\draw [->] (0,0) -- (0,0.9)` or something similar since otherwise you will be unable to see the arrowheads).

Finally, to highlight certain features of your graph, you can add color, thickness, and a change in edge style. Returning to the original complete graph on five vertices, say you want to thicken the edges joining vertices B, D, and E and color them red while all other edges are dashed. The complete code with output is given in Figure 5.7. To achieve the first task, notice that

```
\[%
    \begin{tikzpicture}
        \coordinate [label=left:$A$] (A) at (0,0); \fill (A) circle (2pt);
        \coordinate [label=left:\textcolor{red}{$B$}] (B) at (0,1);
            \fill [red] (B) circle (2pt);
        \coordinate [label=above:$C$] (C) at (1,2); \fill (C) circle (2pt);
        \coordinate [label=right:\textcolor{red}{$D$}] (D) at (2,1);
            \fill [red] (D) circle (2pt);
        \coordinate [label=right:\textcolor{red}{$E$}] (E) at (2,0);
            \fill [red] (E) circle (2pt);
        \draw [dashed] (A) -- (B); \draw [dashed] (B) -- (C);
        \draw [dashed] (C) -- (D);
        \draw [thick, red] (D) -- (E);
        \draw [dashed] (E) -- (A); \draw [dashed] (A) -- (C);
        \draw [dashed] (A) -- (D);
        \draw [thick, red] (B) -- (D); \draw [thick, red] (B) -- (E);
        \draw [dashed] (C) -- (E);
    \end{tikzpicture}
\]
```

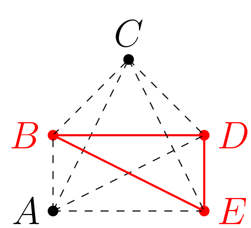


FIGURE 5.7: Code for drawing a red triangle in the complete graph on five vertices using the `tikz` package

each corresponding `\draw` command was replaced with `\draw [thick, red]` (we also colored the associated vertices and labels red, as well, which was done by using the option of `red` with `\fill` and using `\textcolor{red}{⟨label⟩}` in the optional argument to `\coordinate`). To achieve the second task, we replaced each corresponding draw command with `\draw [dashed]`.

Table 5.11 and Table 5.12 provide some other possibilities to consider for *⟨options⟩* with respect to the `\draw` macro.

TABLE 5.11: Line thickness options in the `tikz` package

⟨*option*⟩	example line
ultra thin	
very thin	
thin	
semithick	
thick	
very thick	
ultra thick	

TABLE 5.12: Line styles in the `tikz` package

⟨*option*⟩	example line
solid	
dotted	
densely dotted	
loosely dotted	
dashed	
densely dashed	
loosely dashed	
dash dot	
densely dash dot	
loosely dash dot	
dash dot dot	
densely dash dot dot	
loosely dash dot dot	

Chapter 6

Theorems and Friends

Mathematics may be defined as the subject in which we never know what we are talking about, nor whether what we are saying is true.

Bertrand Russell

6.1 The Theorem Environment

If you're writing a thesis or dissertation in mathematics, you will almost unquestioningly need to present some theorems, propositions, corollaries, etc. (and proofs hopefully). This is accomplished by use of the `theorem` environment (which, by the way, is also a generic term in LaTeX that can refer to proposition environments, corollary environments, etc.). Such an environment is created in a manner similar to that of an `equation` environment. In fact, for an actual theorem, the syntax is as follows:

\label→p.43

```
\begin{theorem}
    ⟨theorem text⟩
    \label{⟨name⟩}
\end{theorem}
```

Let's say we wanted to append the theorem that says the square root of 2 is irrational to the end of our test document from Chapter 2. We could do so by inserting the following code there:

```
\begin{theorem}
    $\sqrt{2}$ is irrational.
    \label{root2irrational}
\end{theorem}
```

(which of course should all come *before* the command `\end{document}`). Remember that you really want to get into the habit of providing a label to each of your environments so that you can refer to them later if need be. However, before you compile your document, you will need to tell LaTeX the style you want for your theorems. Go to your preamble and type in the following code:

```
\theoremstyle{plain}
    \newtheorem{theorem}{Theorem}
```

Compile your document and see what your theorem looks like.

But, of course, now you want to follow up your beautiful theorem with a succinct proof. So, after your `theorem` environment, create a `proof` environment by using `\begin{proof}` and `\end{proof}`. Here, just for the sake of seeing an example of the `proof` environment, type the following after your `theorem` environment:

```
\begin{proof}
    DUH
\end{proof}
```

and compile to get what is shown in Figure 6.1. Note that the classical end-of-proof box is right-aligned here on the last line of the proof.

An essential question now arises: given an element x (of some set) and a set A, how does one decide if $x \in A$? The easiest solution is to require that a set be a *specifiable* collection of objects. That is, a set should be any collection for which there is a "rule"–formally, a well-formed formula–that determines which objects from a particular universe of discourse get to be elements of the collection. For example, one certainly would like to call

$$A = \{x \in \mathbb{R} \mid x^2 \geq 2\} \tag{1}$$

(Equation 1 is read "the collection of all real numbers x such that $x^2 \geq 2$") a set, as there appears to be a well-defined procedure (square a given real number and compare the output with 2) for determining which real numbers (the universe of discourse here) are elements of A. In fact, this approach to the notion of a set seemed so straightforward and intuitive that in the late 19th century, some mathematicians thought this should be one of the foundational principles governing sets, the so-called "axiom of comprehension".

Theorem 1. *$\sqrt{2}$ is irrational.*

Proof. DUH $\square$

FIGURE 6.1: Example of `theorem` and `proof` environments in LaTeX output

In general, you can create propositions, corollaries, etc. by using the same format as the `theorem` environment above. Specifically,

I want my QED!

Sometimes a proof naturally ends with a displayed formula. However, by using only the command `\end{proof}` at the end of such a proof, the end-of-proof box will appear on the line *below* the displayed formula, which is rather unsightly. To remedy this problem, continue to use the standard `proof` environment, but also add the command `\QEDhere` immediately after the final expression in your displayed material. (This macro is an adaptation of the traditional `\qedhere` macro and is designed exclusively for `TNThesis`.) By doing this, LaTeX will move the end-of-proof box to a more appropriate position on the final line of displayed material. Be sure that you do *not* place the `\QEDhere` command in displayed material that is offset by double dollar signs `$$....$$` as this will cause spacing issues at the end of your proof (use instead the more appropriate delimiters `\[...\]` in this situation). Also, don't forget to still type the command `\end{proof}` at the conclusion of the proof; otherwise, LaTeX will take issue with a `\begin{proof}` that has no companion `\end{proof}`.

\label→p.43

```
\begin{⟨name of theorem type⟩}
    ⟨theorem text⟩
    \label{⟨name⟩}
\end{⟨name of theorem type⟩}
```

where ⟨*name of theorem type*⟩ can be any of the following:

> `conclusion, conjecture, corollary, definition, example,`
> `fact, hypothesis, lemma, methodology, problem, proposition,`
> `question, remark, theorem`

There are three standard arguments (i.e., style types) for `\theoremstyle`: `plain`, `definition`, and `remark`. To typeset a particular theorem-type environment in one of these styles, simply type

```
\theoremstyle{⟨style type⟩}
    \newtheorem{⟨name of theorem type⟩}{⟨printed version of name⟩}
```

in the preamble. For example, if you want the word "lemma" in all caps and the text of each lemma in the style of a definition, then you should write

```
\theoremstyle{definition}
    \newtheorem{lemma}{LEMMA}
```

in the preamble. By the way, LaTeX assumes that all of the `newtheorem`'s that you list immediately after `\theoremstyle{⟨style type⟩}` should be given this style until such time as you declare a different `theoremstyle`. (Keep in mind that the `TNThesis` class file will handle all of this when it comes time to type up your thesis or dissertation.)

6.2 Attribution of a Theorem

Sometimes you want to indicate in the theorem environment itself the popular name for a particular theorem (e.g., the Pythagorean theorem) or the original place (as a citation) where the reader can find the result (e.g., [5, Theorem 2.3]). Since `TNThesis` has the `thmtools` package called, each of these special insertions can be accomplished using the same optional argument to the `theorem` environment. That is, for `TNThesis`, every `theorem` environment really has the format

```
\begin{⟨name of theorem type⟩}[⟨attribution⟩]
    ⟨text⟩
    \label{⟨name⟩}
\end{⟨name of theorem type⟩}
```

\label→p.43

where ⟨*attribution*⟩ is either the popular (or, at least, more common) name for the result or a citation using the macro `\cite` (see *Citations* in Section 14.2 for more information about citations).

For example, to produce the printout

Theorem 6.2.1. (Euclid's lemma) *Let* $a, b, c \in \mathbb{Z}$ *such that* $a \,|\, bc$*. If* $(a, b) = 1$*, then* $a \,|\, c$*.*

use the following code:

```
\begin{theorem}[Euclid's lemma]
    Let $a,b,c \in \mathbb{Z}$ such that $a \,|\, bc$.
    If $(a,b)=1$, then $a \,|\, c$.
    \label{Euclid's Lemma}
\end{theorem}
```

If you would rather have a citation in place of the popular name, simply use the command `\cite[⟨text⟩]{⟨label⟩}` in place of the name *with the caveat* that the entire command should be encased in curly braces { }. So, for example, the code

\cite→p.188

```
\begin{theorem}[{\cite[Lemma 1]{BuchananHetzel}}]
    For all positive integers $k$ and $n$,
        \[\tau_k(n)\tau(n) = (1^{[k]}*k^\omega)(n).\]
    \label{Lemma1BH}
\end{theorem}
```

would produce the following output

Theorem 6.2.2. ([5, Lemma 1]) *For all positive integers* k *and* n*,*

$$\tau_k(n)\tau(n) = (1^{[k]} * k^{\omega})(n).$$

assuming that the label `BuchananHetzel` is connected to the fifth item in the bibliography.

6.3 Make Your Own Macros at Home

One of the best things you can do for yourself when creating a large work such as a thesis or dissertation is to create your own macros (you heard me right, chief) as shortcuts for more involved LaTeX commands. User-defined macros are placed in the preamble (that is, *before* the command `\begin{document}`), usually in their own little section for ease of finding them.

6.3.1 Operators

Let's say that you frequently use a particular operator in your work. There are a fair number of standard "built-in" macros representing operators provided by the package `amsmath` (which is automatically called by the package `mathtools` which is automatically called by `TNThesis`). Table 6.1 below gives a list of these "built-in" operators along with whether or not a given operator takes an optional subscript that is placed underneath the name in a displayed environment, such as `\lim_{n \to \infty}`, which produces $\lim\limits_{n\to\infty}$ in display mode.

Table 6.1: Standard operator macros provided by the `amsmath` package

Macro	Subscript?	Macro	Subscript?
`\arccos`	N	`\lim`	Y
`\arcsin`	N	`\liminf`	Y
`\arctan`	N	`\limsup`	Y
`\arg`	N	`\ln`	N
`\cos`	N	`\log`	N
`\cosh`	N	`\max`	Y
`\cot`	N	`\min`	Y
`\coth`	N	`\Pr`	Y
`\csc`	N	`\projlim`	Y
`\deg`	N	`\sec`	N
`\det`	Y	`\sin`	N
`\dim`	N	`\sinh`	N
`\exp`	N	`\sup`	Y
`\gcd`	Y	`\tan`	N
`\hom`	N	`\tanh`	N
`\inf`	Y	`\varinjlim`	Y
`\injlim`	Y	`\varliminf`	Y
`\ker`	N	`\varlimsup`	Y
`\lg`	N	`\varprojlim`	Y

(Of course, *any* of these operators can take a subscript in an absolute sense; for example, `\dim_k` produces $\dim_k$. The difference is whether or not the subscript is placed *underneath* the operator in display mode.)

However, let's say that an operator that you are using in your thesis or dissertation is not part of this list, such as `Tor` (representing the torsion submodule of a module) from homological algebra. You could use the code `$\text{Tor}\,M$` every time you want to typeset the operator acting on the module M. Unfortunately, if you use the operator a lot, it could be quite tedious to keep typing this code. Even worse, TEX isn't smart enough to know that the intention here is that the module M is associated with the `Tor` operator name. As such, if the expression $\operatorname{Tor} M$ is to be typeset at the end of a line, TEX may very well end up splitting the structure, so that Tor is on one line and M is on the next line. How unsightly and potentially confusing!

How can we remedy this issue, so that `Tor` is printed correctly (in roman text) with appropriate spacing between operator name and argument, and that TEX automatically groups the operator name and argument? The answer is found in a special set of commands also given by the `amsmath` package:

```
\DeclareMathOperator{⟨macro⟩}{⟨operator⟩}
\DeclareMathOperator*{⟨macro⟩}{⟨operator⟩}
```

where ⟨*macro*⟩ is the command name for your operator (preceded of course by a backslash) and ⟨*operator*⟩ is what will be typeset whenever you call the macro in your code. The starred version of the `\DeclareMathOperator` command is used when you want your operator to be able to take a subscript (that is, to be able to place a subscript *underneath* the operator name in display mode). So, for instance, by placing the syntax `\DeclareMathOperator{\Tor}{Tor}` in the preamble, the command `\Tor` will typeset the `Tor`-operator name with all of the desired characteristics.

6.3.2 General Macros

Much more generally, macros can be created by the command structure

```
\newcommand*{⟨name⟩}{⟨code⟩}
```

where ⟨*name*⟩ is the name that you want to give your macro (preceded of course by a backslash) and ⟨*code*⟩ is an existing sequence of LATEX commands that ⟨*name*⟩ will replace in your program. Say, for example, that you don't want to write out the phrase "The following are equivalent:" every time it appears in your program. You may choose instead to create a macro by placing into the preamble the following:

```
\newcommand*{\tfae}{The following are equivalent:}
```

Now, each time you need to type "The following are equivalent:" in your document, just type `\tfae` instead.

Be aware that you will need to post-type a backslash for your macro if your macro is not immediately followed by a limiting mark, such as a punctuation mark. Remember that LaTeX does not recognize small spaces. As such, if you define a new macro by

```
\newcommand*{\AJH}{Andrew J. Hetzel}
```

and type `\AJH is awesome!`, what you will get in your printout is

Andrew J. Hetzelis awesome!

To fix this, type instead `\AJH\ is awesome!` and LaTeX will know not to collapse the space between "Hetzel" and "is". If instead you want to use the macro `\AJH` in the sentence "Awesome is Andrew J. Hetzel!" no post-backslash should be used since the exclamation mark already stops any collapsing of space.

Along these lines, I have always found it extremely helpful to create my own macros that replace such commands as `\begin{theorem}` and `\end{theorem}` since they come up all the time in my mathematical writing. In particular, I use the macro `\bt` to stand in for `\begin{theorem}`, `\et` to stand in for `\end{theorem}`, and so on. These can all be created using the format of the above `\newcommand*` macro. For example,

```
\newcommand*{\bt}{\begin{theorem}}
```

Brigadier general with a `\newcommand`

There is, in fact, a standard `\newcommand` macro with the same basic functionality and syntax as `\newcommand*`. So why would I mercilessly have you tack on the additional asterisk when creating your own macros. The reason is that `\newcommand*` will **not** allow the TeX primitive `\par` in its ⟨*code*⟩, where `\par` is LaTeX's command for ending a paragraph. In fact, `\newcommand*` will not even allow a space to exist between lines of text. On the other hand, `\newcommand` will. So what?! you say. Well, the allowance of `\par` in the ⟨*code*⟩ can mask issues involving curly braces that could wreak havoc on your code down the road. Moreover, it may also be very challenging to track down precisely where the error happened in this situation. As such, it is generally advised to just use `\newcommand*` as a regular habit. For those of you who are already familiar with a bit of LaTeX, this is exactly why I would humbly ask that you *not* use the `\def` command despite being functionally equivalent.

Of course, `\newcommand*` can also be used to create macros that stand in for more complex mathematical expressions. For example, say you frequently make reference to the probability density function for a general normal distribution throughout your work. That is,

$$\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

whose syntax in LaTeX is given by

```
\frac{1}{\sigma\sqrt{2\pi}}\mathrm{e}^{-\frac{1}{2}
    \left(\frac{x-\mu}{\sigma}\right)^2}
```

Unless you have a peculiar masochistic streak, this is something for which you probably want to find a shortcut. Thankfully, you can type something like

```
\newcommand*{\normalpdf}{\frac{1}{\sigma\sqrt{2\pi}}
    \mathrm{e}^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}}
```

in the preamble and then anytime you need the expression, you can simply type `\normalpdf` in an appropriate math environment. Be aware though that if you place the macro outside a math environment, LaTeX will return an error as many of the component commands that comprise `\normalpdf` can be used only in math mode (e.g., between two dollar signs).

6.3.3 Macro Functions

Let me conclude this section with a small way that you can adapt `\newcommand*` to create actual macro *functions*, that is, macros that will take input(s) and return a certain expression based upon the input(s). The syntax for accomplishing this is given by

```
\newcommand*[⟨number⟩]{⟨macro function name⟩}{⟨code⟩}
```

Here ⟨*macro function name*⟩ is a macro that will take ⟨*number*⟩ inputs, each offset by its own set of curly braces { }and immediately following ⟨*macro function name*⟩. The ⟨*code*⟩ then involves ⟨*number*⟩ variables with the variables identified by #1, #2, #3, etc. corresponding with the order in which the inputs are given after ⟨*macro function name*⟩. (By the way, since the octothorp symbol # is reserved for this purpose, if you need the symbol in your text somewhere, use `\#`.)

For example, say you want to create a macro named `\LL` that takes two inputs and returns the expression given by the logarithm of the logarithm of the first input all raised to the second input power. In the preamble then, you would type the code

```
\newcommand*{\LL}[2]{(\log\log #1)^{#2}}
```

Once this is done, if you type, say, `\LL{\sin\theta}{4/3}` in your document, then the expression $(\log\log\sin\theta)^{4/3}$ will be typeset at the place where the macro is called in the code.

You can give your macro any name but...

When creating a name for your own macros, LaTeX limits you to the 52 alphabetic characters of the English language (26 lowercase and 26 uppercase). That is, you *cannot* use numbers anywhere inside your macro name. While there are ways around this for coders (using, for example, the command `\csname`), this really would become more complicated than I think you would care to have it for your thesis or dissertation.

Chapter 7

Making a List, Checking It Twice

> Seeing much, studying much, suffering much are the three pillars of learning.
>
> Benjamin Disraeli

The class file `TNThesis` automatically calls the package `enumitem`, which provides for tremendous control over the creation of lists. The three most commonly used list environments are `itemize`, `enumerate`, and `description`. They can be best illustrated in the following way:

Itemize

- Lions-yellow
- Tigers-orange
- Bears-brown

Enumerate

1. Lions-yellow
2. Tigers-orange
3. Bears-brown

Description

Lions yellow

Tigers orange

Bears brown

Essentially, the `itemize` environment prefaces each item with a labelling object (e.g., a bullet point), the `enumerate` environment prefaces each item with a counter, and the `description` environment prefaces each item with descriptive text. Each of these is an example of what is dubbed *seriation* in composition. Seriation is permissible under the APA 7 standard, assuming that proper grammar is used. In general, either each item is a separate sentence (with appropriate punctuation) or the items as a group are to be treated as part of the same sentence with appropriate use of commas, semicolons, and ending punctuation.

In LaTeX, a seriation is created by the following basic command structure:

```
\begin{⟨list name⟩}[⟨list options⟩]
    \item ⟨item 1⟩
    \item ⟨item 2⟩
    \item ⟨item 3⟩
        ⋮
    \item ⟨item n⟩
\end{⟨list name⟩}
```

where ⟨*list name*⟩ is one of the following: `itemize`, `enumerate`, or `description`. It should be noted that the `description` environment should use the modified command `\item[⟨label⟩] ⟨item⟩` for each item, where `⟨label⟩` here is the descriptive text associated to ⟨*item*⟩.

A very flexible item

Even through the `description` environment makes use of `[⟨label⟩]` with each item, technically `[⟨label⟩]` can be used with the `\item` macro in the `enumerate` and `itemize` environments, as well. So, for example, if you wanted to change the (default) labels 1., 2., and 3. in the `enumerate` environment to (a), (b), and (c), you could simply type `\item[(a)]`, `\item[(b)]`, and `\item[(c)]`, respectively, inside the `enumerate` environment. However, there are faster ways to accomplish this task, as you can see below.

Each of these environments can be modified extensively to suit whatever your wishes are for the corresponding list. As is typical for LaTeX environments, if you want something beyond the default options, these can be addressed at the beginning of the environment by changing `\begin{⟨name⟩}` to `\begin{⟨name⟩}[⟨list options⟩]`. Here `⟨list options⟩` takes the form

$$\text{key}_1 = \text{value}_1, \text{key}_2 = \text{value}_2, \ldots, \text{key}_n = \text{value}_n$$

where each key_i is a key from the previous list (e.g., **font**) and each corresponding value_i is an assigned value for the key for the given environment. This method of changing certain parameters in an instance of an environment is called the **key=value system.**[1] Bear in mind that

- values may be textual, numerical, dimensional, or a command depending upon the respective key,
- whatever values you set in this way are local, meaning they will not impact any other list that you create using these list environments, and
- if a particular key is not used in `⟨list options⟩`, it will have its default value.

[1] The “key=value” system was originally developed in the early 1990s by David Carlisle for use in the `graphics` package, but has now become the gold standard for specifying options in environments.

Some of the keys that can be used in ⟨*list options*⟩ are given in Table 7.1. Let's take a look at some examples of each type of list and how the lists can be modified.

Table 7.1: General keys for a list environment

left	the indentation of the label from the left margin (default is the normal paragraph indentation stored in the parameter `\parindent`)
label	the format for the respective default labels in each list (defaults for each type of list are given above in the "Lions, Tigers, and Bears" figure)
font	the font of the label (default is normal text font with the exception of the `description` environment which provides an additional boldfacing)
ref	the appearance of the item labels in a reference to the list (default is the appearance of the actual label)
start	indicates with which number an `enumerate` list should begin (default is 1)
style	sets certain options for the label in a `description` list

7.1 Itemize

In an `itemize` list, the choice really comes down what object you will choose for your label.

```
\begin{itemize}[
    label=\textbullet]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

- This is the first item.
- This is the second item.
- This is the third item.
- This is the fourth item.

Figure 7.1: Example of an `itemize` environment using •

```latex
\begin{itemize}[%
    label=\textasteriskcentered]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

- ∗ This is the first item.
- ∗ This is the second item.
- ∗ This is the third item.
- ∗ This is the fourth item.

FIGURE 7.2: Example of an `itemize` environment using ∗

```latex
\begin{itemize}[%
    label=\textgreater]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

- > This is the first item.
- > This is the second item.
- > This is the third item.
- > This is the fourth item.

FIGURE 7.3: Example of an `itemize` environment using >

```latex
\begin{itemize}[%
    label=\textdagger]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

- † This is the first item.
- † This is the second item.
- † This is the third item.
- † This is the fourth item.

FIGURE 7.4: Example of an `itemize` environment using †

```
\begin{itemize}[%
    label=\textdaggerdbl]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

‡ This is the first item.

‡ This is the second item.

‡ This is the third item.

‡ This is the fourth item.

FIGURE 7.5: Example of an `itemize` environment using ‡

```
\begin{itemize}[%
    label=\checkmark]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

✓ This is the first item.

✓ This is the second item.

✓ This is the third item.

✓ This is the fourth item.

FIGURE 7.6: Example of an `itemize` environment using ✓

```
\begin{itemize}[%
    label=\textopenbullet]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

◦ This is the first item.

◦ This is the second item.

◦ This is the third item.

◦ This is the fourth item.

FIGURE 7.7: Example of an `itemize` environment using ◦

```
\begin{itemize}[%
    label=\textasciitilde]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

~ This is the first item.

~ This is the second item.

~ This is the third item.

~ This is the fourth item.

FIGURE 7.8: Example of an `itemize` environment using ~

```
\begin{itemize}[%
    label=\textreferencemark]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

※ This is the first item.

※ This is the second item.

※ This is the third item.

※ This is the fourth item.

FIGURE 7.9: Example of an `itemize` environment using ※

```
\begin{itemize}[%
    label=\textlnot]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

¬ This is the first item.

¬ This is the second item.

¬ This is the third item.

¬ This is the fourth item.

FIGURE 7.10: Example of an `itemize` environment using ¬

```
\begin{itemize}[%
    label=\textrightarrow]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

→ This is the first item.

→ This is the second item.

→ This is the third item.

→ This is the fourth item.

FIGURE 7.11: Example of an `itemize` environment using →

```
\begin{itemize}[%
    label=\textsection]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

§ This is the first item.

§ This is the second item.

§ This is the third item.

§ This is the fourth item.

FIGURE 7.12: Example of an `itemize` environment using §

```
\begin{itemize}[%
    label=\textemdash]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

— This is the first item.

— This is the second item.

— This is the third item.

— This is the fourth item.

FIGURE 7.13: Example of an `itemize` environment using —

```latex
\begin{itemize}[%
    label=\textperiodcentered]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

· This is the first item.

· This is the second item.

· This is the third item.

· This is the fourth item.

Figure 7.14: Example of an `itemize` environment using ·

```latex
\begin{itemize}[%
    label=$\star$]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

$\star$ This is the first item.

$\star$ This is the second item.

$\star$ This is the third item.

$\star$ This is the fourth item.

Figure 7.15: Example of an `itemize` environment using $\star$

```latex
\begin{itemize}[%
    label=$\triangleright$]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

$\triangleright$ This is the first item.

$\triangleright$ This is the second item.

$\triangleright$ This is the third item.

$\triangleright$ This is the fourth item.

Figure 7.16: Example of an `itemize` environment using $\triangleright$

```
\begin{itemize}[%
    label=\maltese]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{itemize}
```

✠ This is the first item.

✠ This is the second item.

✠ This is the third item.

✠ This is the fourth item.

FIGURE 7.17: Example of an `itemize` environment using ✠

7.2 Enumerate

The basic styles for counting the items in an `enumerate` list are given in Table 7.2

TABLE 7.2: Commands for the **label** key in the `enumerate` environment

Command	Style	Examples
`\Alph*`	capital manuscript letters	A, B, C
`\alph*`	lowercase manuscript letters	a, b, c
`\Roman*`	uppercase Roman numerals	I, II, III, IV
`\roman*`	lowercase Roman numerals	i, ii, iii, iv
`\arabic*`	Arabic numerals	1, 2, 3, 4

Each of these commands can be used as part of the value of the **label** key in the `enumerate` environment. However, the number of ways to represent these styles is limited only by your imagination.

```
\begin{enumerate}[%
     label=(\alph*)]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

(a) This is the first item.

(b) This is the second item.

(c) This is the third item.

(d) This is the fourth item.

FIGURE 7.18: Example of an `enumerate` environment using lowercase letters encased in parentheses

```
\begin{enumerate}[%
    label=\Alph*., font=\bfseries]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

A. This is the first item.

B. This is the second item.

C. This is the third item.

D. This is the fourth item.

FIGURE 7.19: Example of an `enumerate` environment using boldfaced capital letters

```
\begin{enumerate}[%
    label=(\roman*)]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

(i) This is the first item.

(ii) This is the second item.

(iii) This is the third item.

(iv) This is the fourth item.

FIGURE 7.20: Example of an `enumerate` environment using lowercase Roman numerals

```
\begin{enumerate}[%
    label=-\arabic*-, start=13]
  \item This is the thirteenth item.
  \item This is the fourteenth item.
  \item This is the fifteenth item.
  \item This is the sixteenth item.
\end{enumerate}
```

-13- This is the thirteenth item.

-14- This is the fourteenth item.

-15- This is the fifteenth item.

-16- This is the sixteenth item.

FIGURE 7.21: Example of an `enumerate` environment using Arabic numerals starting at 13

```
\begin{enumerate}[label=\Roman*,
    font=\slshape]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

I This is the first item.

II This is the second item.

III This is the third item.

IV This is the fourth item.

FIGURE 7.22: Example of an `enumerate` environment using uppercase Roman numerals

```
\begin{enumerate}[%
    label=\#\arabic*]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

#1 This is the first item.

#2 This is the second item.

#3 This is the third item.

#4 This is the fourth item.

FIGURE 7.23: Example of an `enumerate` environment using Arabic numerals prefaced by the octothorp symbol #

```
\begin{enumerate}[%
    label=\textcircled{\arabic*}]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

① This is the first item.

② This is the second item.

③ This is the third item.

④ This is the fourth item.

FIGURE 7.24: Example of an `enumerate` environment using circled Arabic numerals

```
\begin{enumerate}[%
    label={[\arabic*]}]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

[1] This is the first item.

[2] This is the second item.

[3] This is the third item.

[4] This is the fourth item.

Figure 7.25: Example of an `enumerate` environment using Arabic numerals in square brackets

```
\begin{enumerate}[font=\bfseries,
    label=%
    \textlangle\Alph*\textrangle]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

⟨A⟩ This is the first item.

⟨B⟩ This is the second item.

⟨C⟩ This is the third item.

⟨D⟩ This is the fourth item.

Figure 7.26: Example of an `enumerate` environment using capital letters in angled brackets

```
\begin{enumerate}[%
    label=/\Roman*/]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

/I/ This is the first item.

/II/ This is the second item.

/III/ This is the third item.

/IV/ This is the fourth item.

Figure 7.27: Example of an `enumerate` environment using uppercase Roman numerals offset by forward slashes

```
\begin{enumerate}[%
    label=\arabic*:, font=\itshape]
  \item This is the first item.
  \item This is the second item.
  \item This is the third item.
  \item This is the fourth item.
\end{enumerate}
```

1: This is the first item.

2: This is the second item.

3: This is the third item.

4: This is the fourth item.

FIGURE 7.28: Example of an `enumerate` environment using italicized Arabic numerals

7.3 Description

In the `description` environment, the main choice you have is the presentation style of the label, which here is mainly expressed through the **font** and **style** keys. Here are some examples:

```
\begin{description}[%
    font=\normalfont]
  \item[First.] This is the first item.
  \item[Second.] This is the second item.
  \item[Third.] This is the third item.
  \item[Fourth.] This is the fourth item.
\end{description}
```

First. This is the first item.

Second. This is the second item.

Third. This is the third item.

Fourth. This is the fourth item.

FIGURE 7.29: Example of a `description` environment using normal font

```
\begin{description}[%
    style=nextline]
  \item[First] This is the first item.
  \item[Second] This is the second item.
  \item[Third] This is the third item.
  \item[Fourth] This is the fourth item.
\end{description}
```

First
This is the first item.

Second
This is the second item.

Third
This is the third item.

Fourth
This is the fourth item.

FIGURE 7.30: Example of a `description` environment using a next line format

```
\begin{description}[%
    style=standard]
  \item[First] This is the first item.
  \item[Second] This is the second item.
  \item[Third] This is the third item.
  \item[Fourth] This is the fourth item.
\end{description}
```

First This is the first item.

Second This is the second item.

Third This is the third item.

Fourth This is the fourth item.

FIGURE 7.31: Example of a `description` environment using standard style

```
\begin{description}[%
    font=\normalfont\slshape]
  \item[First] This is the first item.
  \item[Second] This is the second item.
  \item[Third] This is the third item.
  \item[Fourth] This is the fourth item.
\end{description}
```

First This is the first item.

Second This is the second item.

Third This is the third item.

Fourth This is the fourth item.

FIGURE 7.32: Example of a `description` environment using slanted font

When *not* to use a list environment

! In theses or dissertations in mathematics, you often present results with multiple displayed parts, as in the following:

> **Theorem.** *Let R be a commutative ring with nonzero unity. Then the following are equivalent:*
>
> *(1) R is a field;*
>
> *(2) 0 is a maximal ideal of R;*
>
> *(3) 0 is the only proper ideal of R.*

You may be tempted to use an environment like `enumerate` to list out the individual equivalent conditions (especially if you have a lot of conditions). *Resist this temptation.* If you use a list environment to accomplish this task, you will find the spacing to be far more than is reasonable for a set of conditions in a theorem environment. As such, here you should simply write out the labels (1), (2), and (3) and place the necessary blank line between the individual conditions in your code (for otherwise TeX will place each of these conditions on the same line). In particular, the code to create the above theorem looks like the following:

```
\begin{theorem}
    Let $R$ be a commutative ring with nonzero unity.
    Then the following are equivalent:

        (1) $R$ is a field;

        (2) $0$ is a maximal ideal of $R$;

        (3) $0$ is the only proper ideal of $R$.

\end{theorem}
```

```
\begin{description}[%
    font=\normalfont\scshape]
  \item[\fbox{First}] This is the first
  ↪  item.
  \item[\fbox{Second}] This is the second
  ↪  item.
  \item[\fbox{Third}] This is the third
  ↪  item.
  \item[\fbox{Fourth}] This is the fourth
  ↪  item.
\end{description}
```

FIRST This is the first item.

SECOND This is the second item.

THIRD This is the third item.

FOURTH This is the fourth item.

FIGURE 7.33: Example of a `description` environment using small caps type

```
\begin{description}[font=%
    \normalfont\slshape\bfseries,
    style=standard]
  \item[First] This is the first item.
  \item[Second] This is the second item.
  \item[Third] This is the third item.
  \item[Fourth] This is the fourth item.
\end{description}
```

First This is the first item.

Second This is the second item.

Third This is the third item.

Fourth This is the fourth item.

FIGURE 7.34: Example of a `description` environment using slanted boldfaced type

```
\begin{description}[font=%
    \normalfont\scshape\bfseries]
  \item[First] This is the first item.
\begin{enumerate}[%
    label=\arabic*)]
  \item First subitem
  \item Second subitem
\begin{itemize}
  \item First subsubitem
  \item Second subsubitem
\end{itemize}
\end{enumerate}
  \item[Second] This is the second item.
\end{description}
```

First This is the first item.

1) First subitem
2) Second subitem
    - First subsubitem
    - Second subsubitem

Second This is the second item.

FIGURE 7.35: Example of an environment using each of `itemize`, `enumerate`, and `description`

Chapter 8

It Figures!

> "Oh, figures!" answered Ned. "You can make figures do whatever you want."
>
> Jules Verne,
> *20,000 Leagues Under the Sea*

Figures are often a welcomed addition to any thesis or dissertation. They not only break up the monotony of a lot of verbiage, but provide a high-impact way of communicating complicated ideas. Especially in mathematics, one well-placed graph or histogram can immediately illuminate a concept's value. Nobody thought much of complex numbers until the very late 18th century when a means of graphing complex numbers on a plane was conceived (and no, not by Gauss...but don't get me started).

8.1 Captivating Captions

Before delving into the ways that you can insert a figure into your work, it seems prudent at the outset to discuss "captions", for any figure, table, or listing that you place into your work *must* have a caption associated with it. In composition, a **caption** for a special object like a figure consists of (1) a label (e.g., Figure) that identifies the type of object that it is, (2) a corresponding number based upon its position in the work relative to other similar objects, and (3) a short phrase (e.g., The iconic logo of Tennessee Tech University) referred to as the "caption title" that describes the content of the object .[1] The caption is also the means by which `TNThesis` will identify the page number associated to the figure, table, or listing for the sake of your requisite list of figures, list of tables, or list of listings, respectively.

Different standards (e.g., Chicago, MLA, APA) and disciplines each have their own style when it comes to the formatting of captions and corresponding lists of captions. The Graduate School at Tennessee Tech subscribes to the APA 7 (short for *American Psychological Association, version 7*)

[1] Often, people will use the term "caption" to refer only to this short descriptive phrase.

standard for theses and dissertations.[2] This means that captions for *any* type of special insertion are

- placed *above* the insertion,
- single-spaced, and
- aligned with the left margin (as is the corresponding insertion).

Furthermore,

- the label and number are boldfaced,
- the numbering is sequential according to the type of insertion,
- the caption title is italicized, and
- the caption title begins on the line just underneath the label.

Figure 8.1 showcases some of these features. In addition, let me point out that captions should not generally end with any concluding punctuation marks, as they are meant to be simple phrases and not full sentences.

Figure 1

Energy generation projections from various sources for the year 2050 in comparison with the year 2020

FIGURE 8.1: An example of a caption using APA 7 style

Before you get overwhelmed trying to figure out (pun intended) how you are going to create captions that meet the above requirements, don't fret! `TNThesis` has some built-in tools that will do all these things for you automatically. However, with great power comes great responsibility (to actually use these tools). In this chapter and the next two, I will describe these tools, along with more traditional approaches to inserting figures, tables, and listings, should the need arise.

8.2 The `\Graphic` Macro

In general, the insertion of a simple *image* in a LaTeX document is a pretty straightforward matter, requiring only the `\includegraphics` macro (see Section 8.3). However, a *figure* in a thesis

[2] If you want to know more about this standard, check out either tntech.edu or the book *Publication Manual of the American Psychological Association: The Official Guide to APA Style* (7th ed.) https://doi.org/10.1037/0000165-000.

or dissertation is a bit different. In fact, figures, tables, and listings belong to a class of structure called "floats". The moniker is due to the fact that TEX takes some liberty in moving them around, within certain limits, in order to accommodate other surrounding material. The overall aim that TEX has in this process is to create a visually cohesive document without any unsightly and confusing gaps in content.

This is not to say, of course, that TEX is always successful in this endeavor. Much depends on where you call the figure in your code and how big you want your figures to be (thankfully though, both of these parameters are in your control). That being said, there are some parameters that have been hard-coded into `TNThesis` regarding floats:

- A float may appear exactly at the place in the code where you call it, at the top of the next page, at the bottom of the current page, or on a separate page that consists only of floats (i.e., no text). This is designed to give TEX maximum flexibility in placing the float, which is as it should be! While you can circumvent this process (see the bang on page 114), you are admonished not to do so. I'm confident you will not like the results.

- Rules of English composition require that (1) a float must not appear before the first time it is referenced in the body of the work, and (2) a float must not appear in a subsequent division (e.g., chapter, section) of the work from where it is first referenced in the body of the work. `TNThesis` is coded so that, as long as you call the float in your code at a place shortly after the first reference to it, both conditions (1) and (2) will automatically be satisfied.[3]

- A certain amount of vertical shrink and stretch (formally called "glue" in TEX) has been built into floats in `TNThesis`, so it may be the case that surrounding text appears to be closer to one float than another. This is perfectly natural, and should not present as an issue with the Graduate School.

Now that we've got that out of the way, let's talk about how you actually call a figure in your work. If you've done any online research on this topic, I have no doubt that you've come across the standard figure environment `\begin{figure}`...`\end{figure}`. While this environment may be needed for relatively complicated figures, if you want to insert a simple figure, you need only use the following `TNThesis` macro:

```
\Graphic[⟨key=value options⟩]{⟨filename⟩}
```

where ⟨*filename*⟩ is the name of the file *including the extension* containing the image and ⟨*key=value options*⟩ allows you to specify a host of properties for your image using the standard key=value system of LATEX.

[3] This is partly the reason I rail against the use of subsubsections. Subsubsections tend to be very short on content in general practice, and so if you include a float inside a subsubsection, often the result is TEX creating a huge gap in content to accommodate the float.

The allowable keys for the `\Graphic` macro are provided below, with the first set (**caption**, **shortcaption**, **postcaption**, **note**, **callout**, and **label**) applicable to tables and listings, as well.

caption

Provides the *necessary* caption to the corresponding figure. The caption is automatically italicized and aligned with the left margin above the figure, per APA 7 requirements. The value given to the **caption** key should be encapsulated by curly braces `{...}` to avoid problems with LaTeXunderstanding what should and should not be part of the caption (particularly true if you have a comma in your caption). If no caption is given, an error will be produced.

> *Example.* `caption={The iconic Tennessee Tech logo, adopted in 2015}` will place the caption *The iconic Tennessee Tech logo, adopted in 2015* above the corresponding figure.

shortcaption

`TNThesis` will default to placing the caption for a figure in the list of figures (see Subsection 12.3.8) that appears in the front matter of your work. However, if you think that the caption for a particular figure is too long for the list of figures, you may choose to specify a shortened version of the caption that will appear in the list of figures by using the **shortcaption** key. Just as with the **caption** key, the **shortcaption** value should be encapsulated by curly braces `{...}`.

> *Example.* `shortcaption={The iconic Tennessee Tech logo}` will place "The iconic Tennessee Tech logo" in the list of figures instead of the actual caption.

postcaption This key will allow you to specify content that is to immediately follow the actual caption (note the deliberate space before the opening paren in the example below), but *not* appear in the list of figures. Such content sometimes consists of a parenthetical comment regarding the figure, although you ought to consider using the **note** key more often then the **postcaption** key to express this type of content. The value should be encapsulated by curly braces `{...}`.

> *Example.* `postcaption={ (taken using a high resolution camera)}` will place this parenthetical statement at the end of the actual caption, but not in the list of figures.

note Allows you to place a special comment underneath the figure. This may be a useful means of providing the reader with more information concerning the corresponding image. The value given to the **note** key should definitely be encapsulated by curly braces `{...}`, particularly if the note has a line break in it.

> *Example.* `note={ The absence of a plot corresponding to the year 2002 is due to a lack of data for that year.}` will place this statement beneath the corresponding figure immediately after the word *Note.* (note then the deliberate space before the word “The” in this example).

callout Allows you to provide a legend for certain aspects of the figure . This may be a useful means of providing references or statistical information with regard to particular parts of the figure. The value given to the **callout** key should definitely be encapsulated by curly braces `{...}`, particularly if you have more than one callout (and hence need a line break in the callout). Callouts will be placed underneath the figure. Moreover, if there is also a note to the figure provided by the **note** key, callouts will be placed underneath the note. It is important to understand that, if you have more than one callout, you should place all of them together in the same application of the **callout** key and separated by `\\`. See the example below.

> *Example.* `callout={ \textsuperscript{a}This data is found in \cite{DMZ}.\\ \textsuperscript{b}This data is found in \cite{OB1}.}` will place these references beneath the corresponding figure. You should then, of course, have the corresponding superscript marks in the figure.

label If you wish to refer to your figure elsewhere in your work, you can use this key to establish a label for your figure. The default is no label. See Section 4.3 for an explanation of how labels work in LaTeX.

> *Example.* `label=IconicTechLogo` will identify the corresponding figure with the name "IconicTechLogo" so that the command `\ref{IconicTechLogo}` in the text of your work will always point to that particular figure.

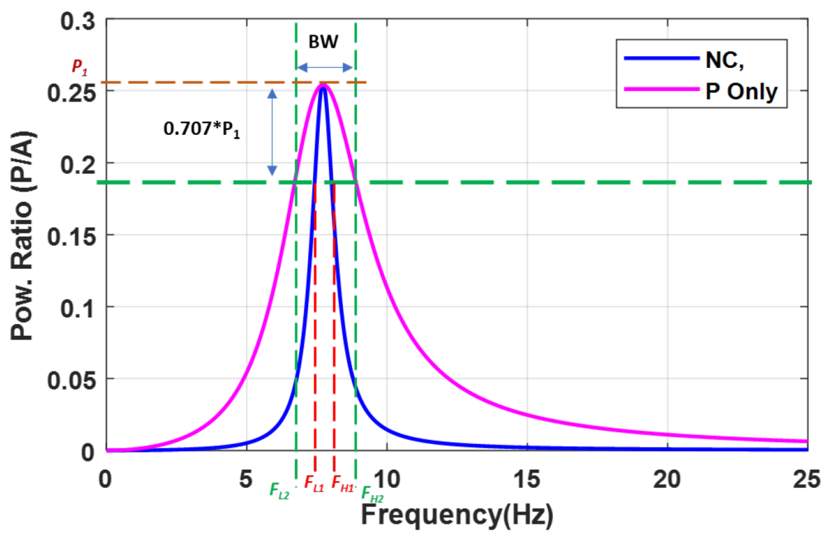
As an actual example of the use of the `note` key, consider Figure 8.2, whose code is:

```
\Graphic[caption={Diagram showing impact of the proportional
controller on the bandwidth of the modified energy harvester},
label={fig5.12}, note={\\$F_{L1}$ = Lower frequency corner of the
energy harvester system = 7.4 Hz\\$F_{H1}$ = Higher frequency corner
of the energy harvester system = 8.2 Hz\\$F_{L2}$ = Lower frequency
corner of the modified energy harvester system = 6.8 Hz\\$F_{H2}$
= Higher frequency corner of the modified energy harvester system =
8.7 Hz\\$P_1$ = 0.26 W}]{fig5.12.png}
```

The figure-specific keys in *⟨key=value options⟩* are given below:

Figure 59

Diagram showing impact of the proportional controller on the bandwidth of the modified energy harvester



Note.
F_{L1} = Lower frequency corner of the energy harvester system = 7.4 Hz
F_{H1} = Higher frequency corner of the energy harvester system = 8.2 Hz
F_{L2} = Lower frequency corner of the modified energy harvester system = 6.8 Hz
F_{H2} = Higher frequency corner of the modified energy harvester system = 8.7 Hz
P_1 = 0.26 W

FIGURE 8.2: An example of the `note` key for a figure

trim Specifies the amount to be removed (or added in the case of negative values) to each side of the image. The first value corresponds to the left side of the image, the second value corresponds to the bottom, the third value corresponds to the right, and the fourth value corresponds to the top. The default is no trim on any side.

> *Example.* `trim=1mm 2mm 3mm 4mm` will remove 1 mm from the left side, 2 mm from the bottom, 3 mm from the right, and 4 mm from the top of the figure.

clip Clips the graphic to the bounding box. Values here are `true` and `false` with the default being `true`.

> *Example.* `clip=false` will ensure that there is a space between the image and the surrounding (invisible) bounding box.

angle Rotates the image counterclockwise by the specified amount, which is assumed to be in degrees. If the value is negative, the image will be rotated clockwise. The default is no rotation.

> *Example.* `angle=90` will rotate the figure counterclockwise by 90° from the leftmost point on the baseline of the figure (which, be advised, is different than the bottom of the figure); see the key **origin** below for more options concerning the point of rotation.

origin Specifies the point about which the image is to be rotated. Values here involve strings containing one or two of the following characters, which indicate positions relative to the image: `l` for left, `r` for right, `b` for bottom, `c` for center, `t` for top, and `B` for baseline. The default is `lB`.

> *Example.* `origin=c` will rotate the image about the center point of the figure.

width Provides a desired width for the image. The default in `TNThesis` is the width of the textblock (i.e., 6 inches).

> *Example.* `width=50mm` will stretch or compress the natural width of the figure to a length of 50 mm.

height Provides a desired height for the image. (In LaTeX, the "height" of an object is defined to be the distance from the *baseline* to the top of the object. For instance, the height of the letter "g" is simply the vertical length of the upper circular portion of the letter. The "depth" is the vertical distance below the baseline subtended by an object. See the key **totalheight** below.)

> *Example.* `height=10cm` will stretch or compress the natural height of the figure to a length of 10 cm.

totalheight Provides the desired total height (i.e., height+depth) for the image. The default is the height of the textblock (i.e., 9 inches).

> *Example.* `totalheight=20pt` will stretch or compress the natural total height of the figure to a length of 20 pt.

keepaspectratio Used along with a specified **width** and either a specified **height** or **totalheight**, this key will have LaTeX make the graphic as large as possible *without distortion*, but not wider than the specified **width** or taller than the specified **height** (resp., **totalheight**). The possible values for this key are `true` and `false` with a default of `true`.

> *Example.* `keepaspectratio=false` will allow width and height to be specified independently of each other.

scale Provides the factor by which the graphic will be scaled. If the value is negative, the graphic will also be reflected. The default value for this key is 1.

> *Example.* `scale=1.5` enlarges the figure by 50%.

ext A way to specify the extension of the file containing the image so that is does not have to be specified in ⟨*filename*⟩. This should be used only in conjunction with the **type** key given below. The default is to have the extension specified in ⟨*filename*⟩.

> *Example.* `ext=.jpg` establishes an extension of `jpg` for the relevant filename.

type Specifies the graphic type.

> *Example.* `type=bitmap` establishes the relevant file to be of bitmap graphic type.

page Specifies which page out of a multi-page pdf file is to be displayed in the figure. The default is the first page.

> *Example.* `page=3` will show only the third page of the corresponding pdf figure.

You can use any comma-separated combination of these keys. However, since the keys are read in order from left to right, there are a few instances in which order matters. For instance,

```
\Graphic[width=2cm, angle=90]{techlogo.png}
```

will first stretch or compress the image contained in the file `techlogo.png` to a width of 2 cm and *then* rotate the image counterclockwise 90 degrees (about the point located at the leftmost place on the baseline), whereas

```
\Graphic[angle=90, width=2cm]{techlogo.png}
```

will first rotate the graphic 90 degrees counterclockwise, *then* stretch or compress the image to a width of 2 cm. You should also feel free to make use of the parameters `\textwidth` and `\textheight` as values for width and height, respectively. Personally, if I have a lot of images to insert relatively close to one another, I will try to control the height more than the width, and so I tend to frequently use

```
\Graphic[width=\textwidth, height=3in, keepaspectratio=true]{techlogo.png}
```

Otherwise you may find enormous graphics taking up half of a page and a whole lot of white space on the page just prior to the graphic.

By way of an actual example of the `\Graphic` macro, the code below will produce the output given in Figure 8.3.

```
This thesis on the evolution of the official seal of Tennessee
Tech University throughout its history actually begins with
the 2021 seal, as given in Figure \ref{TNTechSeal}.
\Graphic[caption={The 2021 official seal of Tennessee Tech
University},label=TNTechSeal, width=2.5in, height=2.5in]{TechSeal.jpg}
One notices a certain minimalism in this logo that is characteristic
of logos for the time period.
```

This thesis on the evolution of the official seal of Tennessee Tech University throughout its history actually begins with the 2021 seal, as given in Figure 1. One

Figure 1

The 2021 official seal of Tennessee Tech University

notices a certain minimalism in this logo that is characteristic of logos for the time period.

FIGURE 8.3: An example of output using the `\Graphic` command

What happens in an environment stays in an environment

The `\Graphic` macro is based upon an environment, and changes made inside an application of an environment in LaTeX are specific to that instance. As such, the value given to a key in one application of the `\Graphic` macro will *not* automatically be applied in a subsequent application of the `\Graphic` macro. If you would like a particular set of values to hold true for many figures in your work, you can choose to use the macro

```
\GlobalFloatParameters{⟨key=value options⟩}
```

where ⟨*key=value options*⟩ is a comma-separated list of key=value entries that you would like applied to each subsequent figure that is produced by the `\Graphic` macro. Simply place this macro somewhere before the first figure to which it should apply. If you do not want these options applied to *all* subsequent figures in your work, you will need to reassign the affected values using `\GlobalFloatParameters` again.
Also, note that any value assigned to a key using the `\GlobalFloatParameters` macro will *not* automatically apply to an explicit `\begin{figure}`...`\end{figure}` environment (see Section 8.3), should you decide to also use such an environment as well.

8.3 More Complicated Figures

If you intend to do something more exotic with some figures in your work (e.g., side-by-side figures), you will need to work under the hood a bit and use the actual **figure** environment itself. But first, I need to tell you about the general `\caption` command provided by the **memoir** class:

```
\caption[⟨short caption⟩]{⟨caption text⟩}
```

where ⟨*caption text*⟩ is a short description of the image, table, or listing that will appear with the float and ⟨*short caption*⟩ is an optional argument that allows you to specify a shortened version of your caption that will go in the appropriate float list in the front matter (see Subsection 12.3.8), if such a shortened version is desired there.

The `\caption` command is needed in the basic syntax for the **figure** environment when it is used to create figures for a thesis or dissertation. The syntax for this environment is given as follows:

\caption→p.112
\label→p.43

```
\begin{figure}[⟨placement⟩]
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \includegraphics[⟨key=value options⟩]{⟨filename⟩}
\end{figure}
```

where `⟨filename⟩` is the name of the file *including the extension* containing the image and `⟨key=value options⟩` is precisely as in the previous section, except that the **caption**, **shortcaption**, **postcaption**, **note**, **callout**, and **label** keys cannot be used here.

The caption is on top of the table. Figure that!

! For your thesis or dissertation, you will need to arrange for a caption to appear *above* the respective figure, table, or listing if you explicitly use the corresponding environments. Thankfully, the `\caption` command is sensitive to position. This means that to produce the appropriate placement of the caption for a figure, the `\caption` command should appear *before* the corresponding `\includegraphics` command but still inside the `figure` environment. For a table, the `\caption` command should be placed inside the `table` environment but *before* the corresponding `tabular` environment. The situation for a figure is demonstrated below with the "The official seal of Tennessee Tech University" and the situation for a table is illustrated in Chapter 9 with "Population of Cookeville, TN, 2016–2019".

I was referring to that table, not your figure

! Don't forget that if you intend to refer to your figure, table, or listing in the text of your work, you will need to use the command `\label{⟨name⟩}` inside the respective environment (but *not* inside the `tabular` environment for a table) and *after* the corresponding `\caption` command. By referencing the label, you will produce the appropriate identification of your figure, table, or listing in the text, as in "We see in Figure 3 that..." Phrases such as "We see in the following figure..." are to be avoided.

The optional argument `⟨placement⟩` immediately following `\begin{figure}` determines where TeX will try to place the figure. The options for `⟨placement⟩` are as follows:

t top of the page

b bottom of the page

h here on the page

p on its own page without any text on the page

In fact, you need (uh, should) not select just one of the above options. For instance, the default for `TNThesis` is ⟨*placement*⟩ = htbp (and I would admonish you to stick with that unless you have good reason to locally change it).[4] If you want to take it a step further, you can use an exclamation point, as in ⟨*placement*⟩ = !htbp, which tells TEX that you want it to make a special effort (i.e., release some of its restrictions) to try to place your figure at the "here" position.

What the H!

! One option for float placement that I did not mention previously is the `H` option, as in ⟨*placement*⟩ = `H`. The reason that I did mention it as an option amongst `h`, `t`, `b`, and `p` is because you should absolutely not use it. The `H` option is a way for you to demand that TEX place the figure *exactly* where you called it in your code, *and nowhere else*. Frankly, this is absurd, unnecessary, and really goes against the entire idea of a "float". Moreover, I am pretty sure you will not like the results if you attempt to use this option. If you are absolutely `H`-bent on placing an image at a very, very specific place in your work, then just use the `\includegraphics` command without encasing it in a `figure` environment.

8.3.1 Framed Figures

Say that you want your figure to be placed inside a frame. Then you can use the command structure

```
\begin{figure}[⟨placement⟩]
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \begin{framed}
        \includegraphics[⟨key=value options⟩]{⟨filename⟩}
    \end{framed}
\end{figure}
```

`\caption`→p.112
`\label`→p.43

8.3.2 Side-by-side Figures

Say you want some figures side-by-side in your work. Then you can use the command structure

[4] By the way, contrary to popular belief, the ordering of the options does not matter. The inclusion of the option in the list means that TEX will regard it as a possibility.

\caption→p.112
\label→p.43
\hspace→p.31
\hfill→p.33

```
\begin{figure}[⟨placement⟩]
    \caption{⟨text⟩}
    \label{⟨name⟩}
    \hspace*{fill}
    {\includegraphics[⟨options⟩]{⟨filename⟩}\hfill
        \includegraphics[⟨key=value options⟩]{⟨filename⟩}}
    \hspace*{fill}
\end{figure}
```

Chapter 9

Tabling Issues

It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.

Sir Arthur Conan Doyle, *Sherlock Holmes*

If your thesis or dissertation concerns data of any sort, you will probably want to showcase such data in your work using a table. Often such tables are relegated to appendices, but if the table is small enough and important to a discussion that you want to have in a chapter, it can certainly appear in the main matter. Unsurprisingly, LaTeX offers numerous ways to create a table for your data. It all depends on the look that you want and what data in your table you would like to highlight.

Let me say at the outset that much of what was said in the previous chapter concerning figures applies to tables as well. The main piece of information that I would reiterate here is that the captions for tables, like figures, must be placed *above* the tables. This is either accomplished by placing the `\caption` command *between* the `\begin{table}` and `\begin{tabular}` commands or automatically by using the `\Table` macro. The next two sections discuss these options, respectively.

9.1 The `Table` and `Tabular` Environments

The `table` and `tabular` environments are two closely connected environments in LaTeX that I would summarize this way:

(1) you cannot have a `table` environment without a `tabular` (or similar) environment, but

(2) you can have a `tabular` environment without a `table` environment, unless you want a label and a caption (and you most certainly do for a thesis or dissertation).

If this sounds confusing, hopefully the discussion presented here can clear things up a bit.

The basic command structure for a table in LaTeX is given by

```
\begin{table}[⟨placement⟩]
    \caption{⟨text⟩}
    \label{⟨name⟩}
\begin{tabular}{⟨column specs⟩}
    ⟨(1,1) entry⟩ & ⟨(1,2) entry⟩ & ··· & ⟨(1,n) entry⟩ \\
    ⟨(2,1) entry⟩ & ⟨(2,2) entry⟩ & ··· & ⟨(2,n) entry⟩ \\
        ⋮
    ⟨(m,1) entry⟩ & ⟨(m,2) entry⟩ & ··· & ⟨(m,n) entry⟩ \\
\end{tabular}
\end{table}
```

\caption→p.112
\label→p.43

Just as with the `figure` environment, ⟨*placement*⟩ is an optional argument by which LaTeX will try to place the table in certain positions; see page 113. In the absence of ⟨*placement*⟩, `TNThesis` will default to `htbp` to place a table.

The mandatory argument ⟨*column specs*⟩ will take some explanation. In LaTeX, to determine a table you need not pre-specify the number of rows, but you do need to pre-specify the number and type of columns. This is accomplished by means of ⟨*column specs*⟩ taking the basic form

$$x_1 \ \langle\textit{code 1}\rangle \ x_2 \ \langle\textit{code 2}\rangle \ x_3 \ \langle\textit{code 3}\rangle \ \cdots \ x_{n-1} \ \langle\textit{code n}\rangle \ x_n$$

where each x_i represents one of the letters `l` (left-aligned), `c` (centered), or `r` (right-aligned) indicating the type of data alignment for the respective column (the x_1 corresponds with the first column, the x_2 with the second columns, etc.) and the number of x_i's indicates the number of columns. Furthermore, what goes *between* these letters—the ⟨*code 1*⟩, ⟨*code 2*⟩, ...⟨*code n*⟩ in the above generic representation of ⟨*column specs*⟩—dictates other features of the columns. Such features include how far apart the columns are separated, the width of a particular column, and the font in a particular column. A few options for ⟨*code*⟩ are given in Table 9.1.

So, for instance, if you wish to have three columns where the first and third column data are centered while the second column data are left-aligned with no vertical separation lines between any columns, then the simplest thing to do is to set

⟨*column specs*⟩=`clc`

On the other hand, if you wish to have five columns all with their respective data centered with the data in the first column boldfaced and no extra spacing before the first column or after the last column, the simplest thing to do is to set

⟨*column specs*⟩=`@{}>{\bfseries}ccccc@{}`

As an actual example, the code below will produce the output given in Figure 9.1.

TABLE 9.1: Column parameters for a table

`m{⟨width⟩}`	defines a column of width ⟨*width*⟩ with each entry centered in proportion to the rest of the line
`p{⟨width⟩}`	defines a column of width ⟨*width*⟩ where each entry is top-aligned and can wrap to the next line
`b{⟨width⟩}`	defines a column of width ⟨*width*⟩ where each entry is bottom-aligned and can wrap to the next line
`>{⟨decl⟩}`	is used *before* an `l`, `c`, or `r` and inserts ⟨*decl*⟩ in front of the entries of the column
`<{⟨decl⟩}`	is used *after* an `l`, `c`, or `r` and inserts ⟨*decl*⟩ behind the entries of the column
`|`	is used to provide a vertical line to separate two columns
`!{⟨decl⟩}`	corresponds with the `|` option except that ⟨*decl*⟩ is inserted instead of a vertical line
`@{⟨decl⟩}`	suppresses inter-column space and inserts ⟨*decl*⟩ instead

```
This thesis on factors contributing to the growth of the population in
Cookeville, Tennessee begins with a look at the population there between
the years 2016 and 2019, as given in Table \ref{PopulationData}.
\begin{table}
   \caption{Population of Cookeville, TN, 2016--2019}
   \label{PopulationData}
   \begin{tabular}{cccc}
      \toprule
         2016 & 2017 & 2018 & 2019 \\
      \midrule
         26,723 & 27,104 & 29,001 & 32,754 \\
     \bottomrule
   \end{tabular}
\end{table}
As one can see from Table \ref{PopulationData}, the rate of growth in
Cookeville, Tennessee has been steadily increasing between 2016 and 2019.
```

Did you notice the lack of any vertical lines in the above table? Perhaps you thought I forgot about them or that a table of data must necessarily have “cells”, one for each piece of data. Well, stop it! Cells may be fine for Excel spreadsheets and the like, but your reader is not trying to add cell A31 to B2 in your work, now is she. Your thesis or dissertation should be about the *presentation* of the data and, frankly, many (including yours truly) feel that almost all data is best presented in a table with a complete absence of vertical divisions. In fact, in LaTeX circles, this is known as “Fear’s Rules”. I can tell that you still need some further convincing. All right then. The two tables

This thesis on factors contributing to the growth of the population in Cookeville, Tennessee begins with a look at the population there between the years 2016 and 2019, as given in Table 1. As one can see from Table 1, the rate of growth in Cookeville, Tennessee has been steadily increasing between 2016 and 2019.

Table 1

Population of Cookeville, TN, 2016–2019

2016	2017	2018	2019
26,723	27,104	29,001	32,754

FIGURE 9.1: An example of output using the `table` environment

below (which I obtained from the `memoir` class documentation) present the same data. Take a look at them and tell me which one you think presents the data the best.

gnats	gram	$13.65
	each	.01
gnu	stuffed	92.50
emu		33.33
armadillo	frozen	8.99

Item		
Animal	Description	Price ($)
Gnat	per gram	13.65
	each	0.01
Gnu	stuffed	92.50
Emu	stuffed	33.33
Armadillo	frozen	8.99

I'm glad we had this discussion.

9.2 The `\Table` Macro

Just as with figures, the `TNThesis` class file provides a means based upon the key=value system that allows for the quick insertion of simple tables into your work with automatic formatting that is consistent with the requisite APA 7 style for Tennessee Tech's theses and dissertations:

```
\Table[⟨key=value options⟩]{%
    ⟨(1,1) entry⟩ & ⟨(1,2) entry⟩ & ··· & ⟨(1,n) entry⟩ \\
    ⟨(2,1) entry⟩ & ⟨(2,2) entry⟩ & ··· & ⟨(2,n) entry⟩ \\
        ⋮
    ⟨(m,1) entry⟩ & ⟨(m,2) entry⟩ & ··· & ⟨(m,n) entry⟩ \\
}
```

where `⟨key=value options⟩` allows you to specify several essential aspects of your table using the key=value system. These include the keys **caption**, **shortcaption**, **postcaption**, **note**, **callout**, and **label**, as described on page 104. The table-specific options are as follows:

colspecs This key serves precisely the same role as `⟨column specs⟩` does in the `tabular` environment discussed in the previous section. In particular, it provides for the desired formatting of the columns of your table and moreover uses the same syntax as `⟨column specs⟩`. The only special requirement here is that the syntax be offset with curly braces `{...}`. There is no default for this key.

Example. `colspecs={crcl}` creates four columns, where the data in the first and third columns is centered, the data in the second column is aligned to the right edge of the column, and the data in the fourth column is aligned to the left edge of the column.

env This key allows you to specify an environment by which to create your table. The default for **env** is `tabular` (which was discussed in the previous section and is by far the most commonly used such environment). However, some other possibilities include `tabular*`, `tabularx`, and `tabulary`.[1]

tabwidth This key specifies the total desired width of the table and should be used only in conjunction with the `tabular*`, `tabularx`, or `tabulary` environments (see the **env** key above).

Example. `tabwidth=4in` creates a table whose total width is 4 inches.

[1] Each one of these environments represents an attempt to reformat the table based upon a desired total width. The environment `tabular*` does so by modifying the amount of intercolumn spacing. If you think you might be interested in one of the other two environments, you can find its documentation at ctan.org/pkg/tabularx or ctan.org/pkg/tabulary.

compress This is a Boolean key that will resize an entire table to have a width equal to the width of the textblock (i.e., 6 inches). The resizing is done proportionally across all columns of the table. The default value for **compress** is `false`. To activate it for a particular table, you may type either `compress=true` or just `compress`. Be aware that if you need to resize a table to fit on a page, then you will more than likely also need to restructure the contents of the table for the sake of readability.

> *Example.* `compress=true` or just `compress` resizes a table so that the total width is 6 inches.

Thus, to reproduce the table concerning the population of Cookeville, TN between 2016 and 2019 given in the previous section, you could do the following:

```
This thesis on factors contributing to the growth of the population in
Cookeville, Tennessee begins with a look at the population there between
the years 2016 and 2019, as given in Table \ref{PopulationData}.
\Table[caption={Population of Cookeville, TN, 2016--2019},
  label=PopulationData, colspecs={cccc}]{
      \toprule
         2016 & 2017 & 2018 & 2019 \\
      \midrule
         26,723 & 27,104 & 29,001 & 32,754 \\
     \bottomrule
}
As one can see from Table \ref{PopulationData}, the rate of growth in
Cookeville, Tennessee has been steadily increasing between 2016 and 2019.
```

A (rather involved) example of the **callout** key is given in Figure 9.2 based upon the following code snippet:

```
\Table[caption={Summary of energy harvesting sources, technologies, and
conversion efficiencies}, label={tab2.1}, compress, colspecs={@{}cclccc@{}},
callout={%
    \textsuperscript{a}\cite{zhou2014harvesting, matiko2013review,
        tan2010energy}\\
    \textsuperscript{b}\cite{zhou2014harvesting, moghe2009scoping,
        matiko2013review}\\
    \textsuperscript{c}\cite{moghe2009scoping, tan2010energy}\\
    \textsuperscript{d}\cite{zhou2014harvesting}\\
    \textsuperscript{e}\cite{akbari2014energy, zhou2014harvesting,
```

```
        matiko2013review}\\
    \textsuperscript{f}\cite{akbari2014energy, visconti2017overview}\\
    \textsuperscript{g}\cite{khan2013acoustic, visconti2017overview}\\
    \textsuperscript{h}\cite{matiko2013review}%
}]{%
    \toprule
        \headcell{Energy\\Source}²  & \multicolumn{2}{c}{\textbf{Operation}} &
        \headcell{Power\\Density} &
        \headcell{Conversion\\Efficiency} &
        \headcell{Energy\\Harvester} \bigstrut \\
    \midrule
        \multirow{2}{*}{\textbf{Light}} &
            \multicolumn{1}{r}{\multirow{2}{*}{photovoltaic
                effect}}\ldelim\{{2}{*} & indoor  & 10--100
                $\mu$W\cm \textsuperscript{3} &
                \multicolumn{1}{c}{\multirow{2}{*}{up to 40\%}} &
                \multicolumn{1}{c}{\multirow{2}{*}{photovoltaic
                \textsuperscript{a}}}\\
        & \multicolumn{1}{c}{} & outdoor &
            15 mW\cm \textsuperscript{2} & \multicolumn{1}{c}{} &
            \multicolumn{1}{c}{} \bigstrut \\
    \midrule
        ⋮
    \midrule
        \textbf{Soundwaves} & \multicolumn{2}{c}{Helmholtz effect}  &
            1.436 mW\cm \textsuperscript{2} @ 123 dB & 0.012\% &
            acoustic\textsuperscript{h} \bigstrut\\
    \bottomrule%
}
```

Just as with figures, you can use the macro `\GlobalFloatParameters` to set values that you want to apply to multiple tables created using the `\Table` macro; see the bolt on page 112 for more details. Also, note that any values assigned to keys using the `\GlobalFloatParameters` macro will *not* automatically apply to an explicit `\begin{table}`...`\end{table}` environment, should you decide to use such an environment as well.

[2] The `\headcell` macro is a user-created macro defined by

```
\newcommand{\headcell}[1]{%
    \begin{tabular}{@{}>{\bfseries}c@{}}#1\end{tabular}%
}
```

See Section 6.3 for more details on user-defined macros.

Table 1

Summary of energy harvesting sources, technologies, and conversion efficiencies

Energy Source	Operation		Power Density	Conversion Efficiency	Energy Harvester
Light	photovoltaic effect	indoor	10–100 μW/cm^3	up to 40%	photovoltaic[a]
		outdoor	15 mW/cm^2		
Vibration	piezoelectric effect		30 μW/cm^3	up to 30%	piezoelectric[b]
	Faraday's law	human	4 μW/cm^3 @ kHz	up to 67%	electromagnetic[c]
		machine	306 μW/cm^3 @ kHz		
	vibration dependent capacitors		50–100 μW/cm^3	9.5–23.6%	electrostatic[d]
Thermal	Seebeck effect	human	100 μW/cm^3	10–15%	thermoelectric[e]
		machine	100 mW/cm^3		
EM Waves	radio transmitters	GSM	0.1 μW/cm^2	50–70%	RF[f]
		wi-fi	0.01 μW/cm^2		
Crystals	Olsen cycle		3.5 μW/cm^3 @ 85 oC, 0.11 Hz	0.1325%	pyroelectric[g]
Soundwaves	Helmholtz effect		1.436 mW/cm^2 @ 123 dB	0.012%	acoustic[h]

[a] [39, 60, 72]
[b] [39, 41, 72]
[c] [41, 60]
[d] [72]
[e] [6, 39, 72]
[f] [6, 65]
[g] [29, 65]
[h] [39]

FIGURE 9.2: An example of the `callout` key in a table

9.3 A Few Basic Table Styles

In what follows, I give you a few more examples of tables, including some with vertical divisions (if you are absolutely determined to have them). Since I am not providing captions or labels here, I will just give you the corresponding `tabular` environments. However, if you intend to use one of the styles below in your thesis or dissertation, be certain to either encase the `tabular` environment in a `table` environment with the appropriate `\caption` and `\label` commands as shown in Section 9.1 or use the format of the `\Table` macro as shown in Section 9.2.

```
\begin{tabular}{%
>{\scshape}c >{\bfseries}l
    >{\itshape}c}
Apples & Bananas & Cantaloupe \\
100 & 10 & 1 \\
35 & 5 & 20
\end{tabular}
```

Apples	**Bananas**	*Cantaloupe*
100	**10**	*1*
35	**5**	*20*

Figure 9.3: Example of a simple table with different fonts in each column

```
\begin{tabular}{%
>{\scshape}c >{\bfseries}l
    >{\itshape}c}
\toprule
Apples & Bananas & Cantaloupe \\
\midrule
100 & 10 & 1 \\
35 & 5 & 20 \\
\bottomrule
\end{tabular}
```

Apples	**Bananas**	*Cantaloupe*
100	**10**	*1*
35	**5**	*20*

Figure 9.4: Example of a simple table using `\toprule`, `\midrule`, and `\bottomrule`

```
\begin{tabular}{%
  >{\scshape}c >{\bfseries}l
    >{\itshape}c}
  \toprule
  Apples & Bananas & Cantaloupe \\
  \midrule
  \midrule
  100 & 10 & 1 \\
  35 & 5 & 20 \\
  \bottomrule
\end{tabular}
```

Apples	**Bananas**	*Cantaloupe*
100	**10**	*1*
35	**5**	*20*

Figure 9.5: Example of a simple table using a double `\midrule`

```
\begin{tabular}{|c|l|c|}
Apples & Bananas & Cantaloupe \\
100 & 10 & 1 \\
35 & 5 & 20
\end{tabular}
```

Apples	Bananas	Cantaloupe
100	10	1
35	5	20

FIGURE 9.6: Example of a simple table using column dividers

```
\begin{tabular}{|c|l|c|}
\hline
Apples & Bananas & Cantaloupe \\
\hline
100 & 10 & 1 \\
35 & 5 & 20 \\
\hline
\end{tabular}
```

Apples	Bananas	Cantaloupe
100	10	1
35	5	20

FIGURE 9.7: Example of a simple table with column and row dividers

The case of the missing data

Sometimes you need to create a table where some of the data is nonexistent. In such a case, simply juxtapose the appropriate number of ampersand symbols (&) together in the affected row to indicate to LaTeX that you wish to skip over those particular cells. So, in the table above concerning apples, bananas, and cantaloupe, if you want to omit the number of apples on the second row, type `& 5 & 20 \\` for the second row. If you want to omit the number of apples and bananas on the second row, type `&& 20 \\` for the second row. Be sure, though, that every row has the same number of ampersand symbols on it; otherwise LaTeX will assume that you made a mistake and throw up an error.

9.4 Taking Your Table to the Next Row

If you are including a table in your work, hopefully the previous section provides you with a good start on how you might wish to present it. However, if you want to create a table that will make all of your friends jealous, then this section is for you.

Let's begin with some mild tweaking of the spacing in your table (after all, you have to sand your table before you stain your table, right?). Since page **??** provided some tools for controlling the intercolumn spacing, let's take a look at a few tools for controlling the spacing between rows. If you have mathematical notation that involves subscripts or superscripts in a table, you may find your rows looking a little cramped. To alleviate this issue, you can use

```
\renewcommand{\arraystretch}{⟨factor⟩}
```

where *⟨factor⟩* is the stretch factor for the rows of any table following the command. The default value of **\arraystretch** is 1.0. If you want enough space to comfortably accommodate subscripts or superscripts, try first setting *⟨factor⟩*=1.35.

Alternatively, you may wish to try

```
\setlength{\extrarowheight}{⟨length⟩}
```

where *⟨length⟩* is the fixed amount that the height of each row following this command will increase by. The default value of **\extrarowheight** is 0pt. If you want space to comfortably accommodate subscripts or superscripts, try first setting *⟨length⟩*=3pt.

With the macros `\arraystretch` and `\extrarowheight`, bear in mind that (1) since the approval page in the front matter is based upon a table, be sure to place these commands *after* the front matter (perhaps somewhere just before the first instance that you need the commands), and (2) if you do not want successive tables to be row-stretched in a similar fashion, you will need to set the commands back to their default values before the occurrence of the next table.

However, if you would prefer something that is more of a surgeon's scalpel, the `booktabs` package (loaded by the `memoir` class) provides such a macro. If you wish to insert a fixed amount of space between two rows of a particular table (or a fixed amount of space after the last row), use the following macro after the corresponding alignment marker `\\` in your table:

```
\addlinespace[⟨length⟩]
```

where *⟨length⟩* is the desired amount of space between the rows.

Now, let's say that you have a particular heading that applies to multiple columns. The `memoir` class provides the following macro for that purpose:

```
\multicolumn{⟨number⟩}{⟨alignment⟩}{⟨text⟩}
```

where ⟨*number*⟩ is the number of columns to which the heading applies, ⟨*alignment*⟩ is the alignment of the heading title (`c` for center, `l` for left, and `r` for right), and ⟨*text*⟩ is the heading title itself. Of course, you will want to place a nice sub-rule under such a heading title and over the columns to which the title applies (you wouldn't purchase a sports car without also springing for the leather interior, would you?). The following macro from the `booktabs` package will do just that:

```
\cmidrule[⟨length⟩](⟨trim⟩){⟨range⟩}
```

where ⟨*length*⟩ provides the length of the sub-rule, ⟨*trim*⟩ provides an optional side and amount by which the sub-rule is to be trimmed (e.g., `lr{0.75em}` provides a default left-side trim of 0.5 em and a right-side trim of 0.75 em), and ⟨*range*⟩ provides the range of column numbers over which the sub-rule applies (in the form `a-b`, where `a` is the number of the first applicable column and `b` is the number of the last). I think Figure 9.8 will illuminate things here.

```
\begin{tabular}{%
  @{} l c >{\bfseries}c c @{}}
  \toprule
  &\multicolumn{3}{c}{Fruit}\\
  \cmidrule(l){2-4}
  & Apples & Bananas
  & Cantaloupe \\
  \midrule
  Krogers & 100 & 10 & 1\\
  Sam's Club & 35 & 5 & 20\\
  \bottomrule
\end{tabular}
```

	Fruit		
	Apples	**Bananas**	Cantaloupe
Krogers	100	**10**	1
Sam's Club	35	**5**	20

FIGURE 9.8: Example of a table using `\multicolumn`

But what if you have a particular heading that applies to multiple *rows* of your table? For this task, we will make use of the following companion macro:

```
\multirow[⟨vposition⟩]{⟨numrows⟩}{⟨width⟩}[⟨vmove⟩]{⟨text⟩}
```

where the arguments are as follows:

⟨*vposition*⟩ defines the vertical position of ⟨*text*⟩ in the multirow block. The options for ⟨*vposition*⟩ are `c` for centered (the default), `t` for top-aligned, and `b` for bottom-aligned.

⟨*numrows*⟩ is the number of rows that the multirow block will span.

⟨*width*⟩ is the width to which the text is to be set. Other than indicating an actual width (e.g., ⟨*width*⟩=1in), you can set ⟨*width*⟩=$*$ to indicate that the text parameter's natural width is to be used.

⟨*vmove*⟩ is a length used for fine-tuning. In particular, the text will be raised or lowered (if ⟨*vmove*⟩ <0) by this length above or below, respectively, wherever it would have otherwise gone.

⟨*text*⟩ is the actual text that will be inserted into the `\multirow` cell.

Figure 9.9 and Figure 9.10 show `\multirow` in action. Man, we're cooking with charcoal now! I know you think that this can't possibly get any better, but just you wait.

Multi-multi

If you want to use both `\multirow` and `\multicolumn` on the same entry, you must put the `\multirow` command *inside* the `\multicolumn` command. For example,

```
\multicolumn{2}{c}{\multirow{3}{*}{Text}}
```

```
\begin{tabular}{@{} ccc @{}}
  \toprule
  Teacher & Class & Enrollment\\
 \midrule\midrule
  \multirow{3}{*}{Dr. Hetzel}
      & Calculus I & 23\\
  \cmidrule(l){2-3}
      & Algebra & 13\\
  \cmidrule(l){2-3}
      & Analysis & 7\\
  \midrule
  \multirow{2}{*}{Dr. Barker}
      & Statistics & 31\\
  \cmidrule(l){2-3}
      & Modeling & 22\\
  \bottomrule
\end{tabular}
```

Teacher	Class	Enrollment
	Calculus I	23
Dr. Hetzel	Algebra	13
	Analysis	7
Dr. Barker	Statistics	31
	Modeling	22

FIGURE 9.9: Example of a table using `\multirow`

```
\begin{tabular}{@{} ccc @{}}
  \toprule
  Teacher & Class & Enrollment\\
 \midrule\midrule
  \multirow{3}{*}{Dr. Hetzel}
      & Calculus I & 23\\
      & Algebra & 13\\
      & Analysis & 7\\
  \midrule
  \multirow{2}{*}{Dr. Barker}
      & Statistics & 31\\
      & Modeling & 22\\
  \bottomrule
\end{tabular}
```

Teacher	Class	Enrollment
Dr. Hetzel	Calculus I	23
	Algebra	13
	Analysis	7
Dr. Barker	Statistics	31
	Modeling	22

FIGURE 9.10: Example of a table using `\multirow` and without center midrule lines

Let's say that you wish to highlight various cells in your table. This can be accomplished by means of the following macros provided by the package `colortbl`:

```
\cellcolor{⟨color⟩}[⟨left overhang⟩][⟨right overhang⟩]
\columncolor{⟨color⟩}[⟨left overhang⟩][⟨right overhang⟩]
\rowcolor{⟨color⟩}[⟨left overhang⟩][⟨right overhang⟩]
```

where *⟨color⟩* is the name of the color you wish to place in the background of a cell, column, or row, respectively, while *⟨left overhang⟩* (resp, *⟨right overhang⟩*) is the amount that you want the color to extend past the respective cell(s) on the left (resp., right).

For instance, if you want to highlight the cells representing the smallest enrollments for each teacher, you could do something like Figure 9.11. Or if you wish to highlight the entire enrollment column, you could do something like Figure 9.12.

The application of `\rowcolor` is a bit trickier, as the command must be placed at the *beginning* of the appropriate row and, as such, is not generally well-suited for situations involving `\multirow` (you can however use `\cellcolor` as needed in this context). Figure 9.13 is an example illustrating the use of `\rowcolor`.

Rather than highlight a group of cells, you might want to simply indicate the group by some delimiter (such as a really large curly brace) possibly along with a comment concerning the group. For this task, we can use the following macros from the `bigdelim` package:

```
\begin{tabular}{@{} ccc @{}}
  \toprule
  Teacher & Class & Enrollment\\
 \midrule\midrule
  \multirow{3}{*}{Dr. Hetzel}
      & Calculus I & 23\\
      & Algebra & 13\\
      & Analysis &
      \cellcolor{green} 7\\
  \midrule
  \multirow{2}{*}{Dr. Barker}
      & Statistics & 31\\
      & Modeling &
      \cellcolor{green} 22\\
  \bottomrule
\end{tabular}
```

Teacher	Class	Enrollment
Dr. Hetzel	Calculus I	23
	Algebra	13
	Analysis	7
Dr. Barker	Statistics	31
	Modeling	22

FIGURE 9.11: Example of a table using `\cellcolor`

```
\begin{tabular}{%
  @{}cc>{\columncolor{yellow}}c@{}}
  \toprule
  Teacher & Class & Enrollment\\
 \midrule\midrule
  \multirow{3}{*}{Dr. Hetzel}
      & Calculus I & 23\\
      & Algebra & 13\\
      & Analysis & 7\\
  \midrule
  \multirow{2}{*}{Dr. Barker}
      & Statistics & 31\\
      & Modeling & 22\\
  \bottomrule
\end{tabular}
```

Teacher	Class	Enrollment
Dr. Hetzel	Calculus I	23
	Algebra	13
	Analysis	7
Dr. Barker	Statistics	31
	Modeling	22

FIGURE 9.12: Example of a table using `\columncolor`

```
\begin{tabular}{clc}
    \toprule
    Apples & Bananas & Cantaloupe\\
    \midrule
    100 & 10 & 1 \\
    \rowcolor{orange} 35 & 5 & 20\\
    \bottomrule
\end{tabular}
```

Apples	Bananas	Cantaloupe
100	10	1
35	5	20

FIGURE 9.13: Example of a table using `\rowcolor`

Make it a happy row

If you need to create some different hues or shades of certain colors or mixtures of two colors, you can do so by means of the following structure:

⟨*color1*⟩!⟨*percentage*⟩!⟨*color2*⟩

where ⟨*color1*⟩ is your base color and ⟨*percentage*⟩ is the amount of ⟨*color1*⟩ you wish to mix with ⟨*color2*⟩ (expressed as a number between 0 and 100). For example, if you want a lighter shade of orange for your row color, you can use the structure `orange!80!white` in place of `orange` in the command `\rowcolor`. For some of the options for ⟨*color1*⟩ and ⟨*color2*⟩, see Table 5.10.

```
\ldelim⟨delimiter⟩{⟨numrows⟩}{⟨width⟩}[⟨text⟩]
\rdelim⟨delimiter⟩{⟨numrows⟩}{⟨width⟩}[⟨text⟩]
```

where `⟨delimiter⟩` is the delimiter type (e.g., `⟨delimiter⟩=\{`) which will extend over the `⟨numrows⟩` rows starting at the one containing the command (corresponding cells in the subsequent rows must be given as empty cells). The parameter `⟨width⟩` is the space reserved for the delimiter itself and its text (if text is given). As with the `\multirow` macro, you can set `⟨width⟩=*` to indicate that the text parameter's natural width is to be used.

Once again, I believe that an example (Figure 9.14) will help you make sense of these macros. The real *pièce de résistance*, however, is in Figure 9.15.

Told you that you would make your friends jealous.

```
\begin{tabular}{@{}cccr@{}}
  \toprule
  Teacher&Class&Enrollment&\\
  \midrule\midrule
  \multirow{3}{*}{Dr. Hetzel}
      & Calculus I & 23 &
      \rdelim\}{3}{*}[full]\\
      & Algebra & 13 &\\
      & Analysis & 7 &\\
  \midrule
  \multirow{2}{*}{Dr. Barker}
      & Statistics & 31 &\\
      & Modeling & 22 &\\
  \bottomrule
\end{tabular}
```

Teacher	Class	Enrollment	
	Calculus I	23	
Dr. Hetzel	Algebra	13	full
	Analysis	7	
Dr. Barker	Statistics	31	
	Modeling	22	

FIGURE 9.14: Example of a table using a big delimiter

```
\hyperlabel{array}
\begin{equation*}
    \begin{array}{cc@{}c@{}c@{}cc}
    \ldelim[{3}{0em} & a & b & c
        & \rdelim]{3}{0em} &\\
    & e & \boldsymbol{f} & g & &
        \leftarrow \text{pivot row} \\
    & h & i & j & & \\
    & & \uparrow  & & &\\
    & & \text{pivot} & & &\\
    & & \text{column} & & &
    \end{array}
\end{equation*}
```

$$\begin{array}{cc} \begin{bmatrix} a & b & c \\ e & \boldsymbol{f} & g \\ h & i & j \end{bmatrix} & \leftarrow \text{pivot row} \\ \begin{array}{ccc} & \uparrow & \\ & \text{pivot} & \\ & \text{column} & \end{array} & \end{array}$$

FIGURE 9.15: Example of a matrix using big delimiters and the `array` environment

Chapter 10

Listing Your Listings

Programming is the art of telling another human being what one wants the computer to do.

Donald Knuth

As of version 5.0 of `TNThesis`, the `listings` package is not supported. Instead, students who wish to display code should plan on using the `minted` package. In order to use the `minted` package discussed in this chapter, you will need to specify it as an option in the `TNThesis` document class. In particular, you will need to use the syntax `\documentclass[minted]{TNThesis}` at the beginning of `TNTmain`. Documentation for the package beyond what is presented in this chapter can be found at ctan.org/pkg/minted.

THIS chapter is one of the newest in this manual. It emerged after a semester of guiding one of our statistics students through some programming issues he was encountering in LaTeX. Based upon my experience with him, I came to believe that a fair number of our graduate students at Tennessee Tech nowadays may be working on theses and dissertations where the subject matter itself concerns computer programming. For such students, I can envision that printed code snippets or segments—technically referred to as **listings** in programming circles—may populate their documents as much as figures and tables populate the works of students in other fields. However, even if a listing is relegated to an appendix, LaTeX has the means to generate some truly professional-looking code—code where you can emphasize keywords and thereby improve the reader's understanding of your work.

10.1 Putting Down Roots

Arguably, the main challenge in displaying computer code is to find a means to ensure that LaTeX does not inadvertently try to execute any part of the code. This requires the use of what is known as the `verbatim` environment, which is designed simply to output what is contained within its borders. Regrettably though, `verbatim` is a very sensitive kind of environment (the technical term is “fragile”), which means essentially that extreme care must be exercised when manipulating it. However, two packages, `listings` and `minted`, have found successful paths forward in this matter. After doing a thorough comparison of these two packages, I am now of the opinion that the `minted` package is the better one to use for my purposes, and so `TNThesis` now automatically loads the `minted` package. The flip side of the coin is that the `minted` package needs a little bit more of a running start before you can use it, particularly if you are compiling your work on your own computer.[1]

The idea behind the `minted` package is that the Python language is used to color (they use the term “highlight”) certain keywords in your displayed code. So, to make this work, you will need to do the following three things first:

Install the latest version of Python on your computer You can check which version of Python you have by opening a command prompt and typing `python -version`. Then compare your version to what is available at python.org/. If you need to install the latest version, be sure to check the appropriate box on the setup wizard to set the path to your installation of Python to be a value of the Path environment variable. If the latest version of Python is already on your system, you will need to manually arrange for this to be the case by going to your system’s control panel and editing your environment variables. For example, in my installation, I have

```
C:\Users\ahetz\AppData\Local\Programs\Python\Python311\Scripts
```

as the value to the Path variable. Be advised that your path may be slightly different.

Install the latest version of the Python module Pygments on your computer. You can check out the tutorial for doing this if you would like. However, most of the time, this part of the process involves nothing more than opening up a command prompt and typing `pip install Pygments`.

Allow your LaTeX editor to “talk” to Pygments This is accomplished by means of the `-shell-escape` switch. Most local installations of TeX default to a reserved version of `-shell-escape`, but you will need to arrange for the full version if you intend to use the `minted` package. The procedure depends a great deal on the LaTeX editor that you are using, but at least for TeXworks, start by following EDIT →PREFERENCES →TYPESETTING. Double click then on PDFLATEX to open up the TOOL CONFIGURATION menu. Then type `-shell-escape` just before `$fullname`, so that it will read

```
$synctexoption
-undump=pdflatex
--shell-escape
$fullname
```

[1] Those who are using Overleaf may skip this section, as Overleaf has everything needed to successfully run minted already set up on their system.

Apply these changes. Now, restart your computer and you should be good to go.

10.2 The Listing Environment

Assuming that you have everything set up correctly for you to use the `minted` package, you will unfortunately find that you still cannot use the package's native `listing` float environment for your work, as you cannot reformat the environment to produce displayed code in the required APA 7 style. No worries, though. `TNThesis` has you covered with its own listing float environment:

```
\begin{Listing}[⟨key=value options⟩]{⟨language⟩}
    ⟨first line of code⟩
    ⟨second line of code⟩
        ⋮
    ⟨last line of code⟩
\end{Listing}
```

where ⟨*key=value options*⟩ allows you to specify a myriad aspects of your code by employing the same key=value system used for both the `\Graphic` and `\Table` macros. In fact, the `Listing` environment shares the keys **caption**, **shortcaption**, **postcaption**, **note**, **callout**, and **label** with these two macros; see page 104 for more details. Your choices for the programming language ⟨*language*⟩ can be found at pygments.org/languages/.

Some of the basic listing-specific keys for ⟨*key=value options*⟩ are as follows:

autogobble This is a Boolean key whose default value is `false`. If set to `true`, all leading whitespace will be removed from each line of code.

breaklines This is a Boolean key whose default value is `true`, and so long lines of code will be broken at space characters. If a line is broken, the remainder will be indented by the value given in the key **breakindent** and a default carriage return arrow will indicate that the line has been broken.

bgcolor This key provides for a background color to the listing. The default is a white background (i.e., **bgcolor**=`white`). Other colors can be found in the table Table 5.10. It is strongly suggested that, if you would like a background other than white, choose a very light gray. In particular, I have created the color `codegray` in `TNThesis` for just this purpose.

escapeinside This key is more for the advanced user. It allows one to specify delimeters inside of which LaTeX code will become active again. For example, if one sets **escapeinside**=`||`, then anything set between two vertical bars in the listing will be interpreted as LaTeX code and typeset accordingly.

finalline The line number of the last line of output. The default value for this key is 9. Its only purpose is to allow LaTeX to adjust the horizontal spacing of the listing from the left margin if line numbers are displayed.

firstline The number of the first line to be output in the display. All lines before this one do not appear in the output. The default value for this key is 1.

firstnumber The line number of the first line. Options are `auto` (the default, which also means the first line number is always 1), `last` (if one wants the first line number to pick up where the last code of the same language left off), or an integer of one's choosing.

fontsize This allows you to adjust the size of the font in your displayed code. The default is **fontsize**=`\normalsize` (which corresponds with 10 pt font), but other acceptable options are **fontsize**=`\small` (for 11 pt font), or **fontsize**=`\footnotesize` (for 10 pt font).

frame Allows various types of frames to be put around the code. The default for `TNThesis` is `lines`, which provides a horizontal line above the code and a horizontal line below. Other options are `none`, `leftline`, `topline`, `bottomline`, and `single`.

framerule The width of the frame, if a frame is used. The default value is 0.4 pt.

framesep The distance between the frame, if one is used, and the code itself. The default value for `TNThesis` is 2 ex.

highlightcolor The color used for highlighting an entire line or set of lines of code. The default is `LightCyan`.

highlightlines This key specifies which lines of code are to be highlighted with the color given by the key **highlightcolor**. One can specify *either* a single line, as in **highlightlines**=`{3}`, *or* a range of lines, as in **highlightlines**=`{3-5}`. (Unfortunately, one cannot specify both types despite what the `minted` documentation says.)

lastline	The dual to the **firstline** key. This specifies the last line of code to be shown in the output.
numbers	This key determines if and where line numbers will be shown. The default is `left` but other options are `right`, `both`, and `none`.
numberblanklines	This is a Boolean key whose default value is `true`. It enables or disables the numbering of blank lines in the code.
numbersep	This is the gap between line numbers and the start of the lines of code. The default value for `TNThesis` is 1 em.
obeytabs	This is a Boolean key for treating tabs as tabs instead of converting them to actual spaces. The default value is `false`.
rulecolor	The color of the frame, if a frame is used. The default value is `black`.
showspaces	A Boolean key that enables visible spaces. It is `false` by default.
showtabs	A Boolean key that enables visible tabs. It is `false` by default.
spacecolor	This sets the color of visible spaces. By default, it takes the color of its surroundings. See the **showspaces** key.
style	This sets the style sheet to be used by Pygments in determining the coloring scheme for keywords. The default in `TNThesis` is `xcode`, but other (good) options include `default`, `vs`, `emacs`, `bw` (if you don't want any special coloring), `tango`, `perldoc`, `lovelace`, `rainbow_dash`, `abap`, `stata`, `gruvbox-light`, and `sas`. You are encouraged to experiment with these to find which one you like the best.
stepnumber	The interval at which line numbers appear. The default value is 1.
stripall	A Boolean key that determines whether or not all leading *and trailing* whitespace will be removed from the input (compare with the **autogobble** key). The default for this key is `false`.
tabcolor	The sets the color of visible tabs. By default it is set to `black`, but if it is set to `none`, visible tabs will simply take the color of their surroundings. See the **showtabs** key.
tabsize	The number of spaces to which a tab is equivalent. The default value for this key is 8. See also the key **obeytabs**.

Don't forget that the `Listing` environment is a float, just like the `figure` and `table` environments. Moreover, it has the same placement protocol as these other environments, namely htbp. This effectively means that the displayed code need not appear exactly where you place it in your LaTeX code relative to the surrounding text, but it should be close!

As with the `\Graphic` and `\Table` macros, use `\GlobalFloatParameters` if you wish to set up key-values that apply to many listings; see the bolt on page 112 for more information.

Here is an example of the `Listing` environment with the resultant output given in Figure 10.1:

```
\begin{Listing}[caption={An example of a listing}, highlightlines={3-4},
    finalline=29]{python}
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
\end{Listing}
```

Listing 1

An example of a listing

```python
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;

                if M is None:
                    M = np.copy(VT)
                else:
                    M = np.concatenate((M, VT), 1)

                VT = np.zeros((n*m,1), int)

    return M
```

FIGURE 10.1: An example of output using the `Listing` environment

10.3 Very Small and Very Big Listings

Admittedly, the `Listing` environment works well for what it was intended—displaying listings representing a few lines of code in a manner similar to that of figures and tables. However, what if you would rather highlight just a single line of code in text or, on the other extreme, showcase code that spans more than a page? Thankfully, the `minted` package already comes equipped with tools for dealing with these situations.

Let's say you want to single out a line of code for discussion in the text of your work. Moreover, the line of code should have an appearance that distinguishes it from the surrounding text (much the same way as in-text mathematical symbols use an italicized font that immediately distinguishes them from the surrounding text). To accomplish this, you can use the command structure

```
\mintinline[⟨options⟩]{⟨language⟩}⟨character⟩⟨code⟩⟨same character⟩
```

The reason that `\mintinline` has this atypical command structure is due to the fact that a curly brace may in fact be part of the line of code that you want to typeset. If this is the case, then the traditional use of curly braces `{...}` for mandatory arguments simply will not do, as LaTeX would be unable to distinguish a curly brace in the code from a curly brace intended as a delimiter. So, to get around this potential problem, you must choose here a character different from anything that appears in ⟨*code*⟩ to use as a delimiter. In fact, the delimiting character can be different for different instances of `\mintinline`.

Figure 10.2 shows the output of the following code:

```
The syntax \mintinline{R}!n <- readline(prompt="Enter an integer: ")! from the
R language is employed to solicit an integer value from the user and store
the resulting answer in a variable.
```

> The syntax `n <- readline(prompt="Enter an integer: ")` from the R language is employed to solicit an integer value from the user and store the resulting answer in a variable.

FIGURE 10.2: An example of output using the `\mintinline` command

Allow me to make a few points concerning the use of `\mintinline`:

- The only characters that *can never* be used as delimiting characters are alphanumeric characters and the symbols \, %, [, and # (since these symbols have special meaning in LaTeX).
- `\mintinline` uses its own set of defaults apart from the defaults for the `Listing` environment. To set the `\mintinline` defaults for your work, simply use the command

  `\setmintedinline{⟨key=value list⟩}`

  where ⟨*key=value list*⟩ is a comma-separated list of key=value options from the above list of keys.
- I would strongly advise you to set

  `\setmintedinline{breaklines=true}`

  at the beginning of your work. If you don't, you will probably find times in which your code runs off into the right margin.

On the other end of the spectrum, let's say you want to list a multi-page piece of code (which undoubtedly would go in an appendix in your thesis or dissertation). The way to include this lengthy code in your work is by means of the command

```
\inputminted[⟨options⟩]{⟨language⟩}{⟨filename⟩}
```

where ⟨*filename*⟩ is the filename *with the extension* of the code that you want inserted into your work. Moreover, the code will be listed at the precise spot where the `\inputminted` command is called.

Chapter 11

Do It With Style

Everywhere I go I'm asked if I think the university stifles writers. My opinion is that they don't stifle enough of them. There's many a best-seller that could have been prevented by a good teacher.

Flannery O'Connor

If you've been diligently working through this manual chapter by chapter, I imagine you are chomping at the bit to start typing up your thesis or dissertation. Patience young grasshopper. It may be helpful if you first appreciate some of the tenets of good composition. To that end, in the first section of this chapter I discuss how to address a few points of proper typographic style using LaTeX. In subsequent sections I focus specifically on mathematical writing, elaborating on certain standards and conventions therein.

11.1 Typographic Style in LaTeX

While LaTeX does a phenomenal job on its own typesetting your work, there are nevertheless a few stylistic rules to which you may need to pay attention. Much of what is given below is championed by famed typographer Robert Bringhurst and suggested in his book *The Elements of Typographic Style*.[1]

- Particularly when referencing the author(s) of a work, it is not uncommon to use both the first and middle initials followed by the last name(s). When typing such first and middle initials, you should either type them with no intermediate space (as in `H.L. Mencken`) or use a thin space `\,` between the initials (as in `H.\,L. Mencken`). Whichever you decide, be consistent throughout your work (including the bibliography).

[1] Bringhurst, R. (2012). *The Elements of Typographic Style*, fourth ed. Hartley & Marks. See, in particular, sections 2.1.4, 2.1.5, and 5.3.2. of his book.

- Speaking of capital letters, LaTeX handles the spacing after a piece of punctuation that itself immediately follows a capital letter differently than it would handle the corresponding situation for a lowercase letter (e.g., the spacing after "hill." is different than the spacing after "HTML."). This happens because LaTeX assumes that, for example, if you type a capital letter followed by a period, you intend for the capital letter to represent an initial, not the end of a sentence. So, if you intend such a period to truly represent the end of a sentence, you should indicate as much by using `\@.` instead of the usual period. For example, `The result follows from Condition A\@.` More generally, you should use the format `\@⟨punctuation mark⟩` when needed to essentially tell LaTeX to pretend that the letter in front of ⟨*punctuation mark*⟩ is a lowercase letter. For example, `My favorite programming languages are C\@, FORTRAN\@, and APL\@.`

  On the flip side of this coin, if you have a period that is not serving to end a sentence, you will need to append a backslash `\` to the period in order to produce the proper spacing. For example, `I am studying mathematics, science, art, etc.\ in school right now.`

- LaTeX automatically creates ligatures with the combinations "ff", "fi", "fl", "ffi" and "ffl", as in the word "difficult". While ligatures have been standard practice in typography for more than a century, if you don't want ligatures, there are two ways to handle this:

  1. In the preamble of `TNTmain`, place the following command (from the `microtype` package):

     ```
     \DisableLigatures[⟨characters⟩]{⟨encoding=T1⟩}
     ```

     where `⟨characters⟩` is an optional list of the first characters of the ligatures that you want removed (e.g., `⟨characters⟩`=f will remove all ligatures that start with an "f"). If `⟨characters⟩` is not specified, then all ligatures will be removed from the work. However, you should use this option *only* if you really know what you're doing. For instance, if you disable all ligatures, then typing something like `--` will in fact produce `-` instead of the traditional en-dash –.

  2. Insert a pair of curly braces `{}` (*without* any space between the braces) between the appropriate letters of any word that involves a ligature. For example, `dif{}f{}icult` produces difficult.

- If you've used the single/double quotation mark key on your keyboard to make quotation marks, you've probably noticed your quotation marks looking like "this" and 'this'. Pretty ugly, right? To remedy this problem, use the accent/tilde key (usually on the upper left side) on your keyboard to make the opening quotation mark (once for a single quotation mark and twice for a double quotation mark) and then use the standard single/double quotation mark key to make the closing quotation mark (once for a single quotation mark and twice for a double quotation mark). When done properly, your quotation marks should look like "this" and 'this'.

- In math mode, LaTeX automatically typesets all delimiters in upright font. However, outside of math mode, you need to continue to use upright (not *slanted*) parentheses, brackets, and braces, even if the context is italic. Thus, if for some reason you are typing $a(bc)$ outside of math mode, you will need to use the syntax `\textit{a}(\textit{bc})`.

- Depending upon the typesetting system, sometimes an italicized character immediately followed by an upright character has the effect of the italicized character "running into" the upright character, as in *f*H. To prevent this from happening, every character in a font system has an associated *italic correction*—that is, a corresponding amount of space that can be used in the aforementioned situation to prevent the italicized version of the character from making contact with a subsequent upright character. While LaTeX 2$_\varepsilon$ builds in an italic correction whenever `\textit` or `\itshape` is used, if you ever need to invoke the italic correction for a character, simply use the command `\/` between the italicized character and upright character, as in `{\it f}\/H`.[2]

11.2 Common Issues in Mathematical Writing

Even after 20 years of teaching classes in writing mathematics, it never ceases to amaze me how often I find students who think that mathematics is only about equation or symbol manipulation and has little, if anything, to do with explanatory statements (it's probably this misguided perspective that drove them into pursuing mathematics as a major in the first place). But this view is utterly without merit. I would wager to say that if a teacher walked into the classroom one day, said absolutely nothing, and just wrote symbols on the board, these same students would have no hesitation admitting that they would be absolutely lost. But why? Because *explanations matter*, whether verbal or written. In fact, one of the great advances in mathematical writing at the turn of the 20th century is that we do no computation before it's time—that is, we do no computation before justifying *why* we're doing it. And, yes, the "why" requires words.

I would go further to assert that the precision required of good writing in mathematics requires even more accuracy than that of normal English composition (although I anticipate getting a lot of hate mail over this statement from English professors everywhere). Every word you choose, every way you choose to phrase something, and every placement of a formula or theorem creates an impression in the reader's mind about what you are trying to communicate. And in mathematics, that impression is either spot on or it is dead wrong. Either what you say will help the reader's understanding or it will hurt the reader's understanding—there is nothing in between.

This is why good writing in mathematics is a skill that takes years to cultivate. However, I hope in this section to give you a little leg up on the process by sharing with you some common issues students have when they write their first serious work in mathematics (such as a thesis or dissertation).

Consider the following proof a student gave on the infinitude of the prime numbers:

---

[2] Please note that LaTeX 2.09 font selection commands such as `\it` for italics are deprecated in LaTeX 2$_\varepsilon$. Moreover, the `memoir` class (and thus the `TNThesis` class) does not support these commands.

> Assume there are finitely many primes. Then $p_1p_2\cdots p_n+1$ is divisible by one of the p's. But this means that p divides 1.

There are *at least* eight things wrong with this proof:

1. It's clear that the student is doing a proof by contradiction but has left it up to the reader to figure that out. It's possible that the student thought it was obvious that a proof by contradiction was what he or she was doing by the first sentence, but a lot goes wrong in mathematical writing whenever you believe that something is obvious. In fact, if a reader does not quickly catch onto the overarching proof technique that you are using, he or she really stands no chance at understanding anything else in your proof. I would advise an introductory sentence along the lines of *We proceed by contradiction.* or, if that's too wordy for you, do what the pros do and just write *Deny.*

2. The statement *assume there are finitely many primes* is patently **true**, and hence does not represent something to contradict against. For example, the set $\{2, 3, 5\}$ constitutes a finite set of primes. The proof writer needs to say *assume there are* <u>*only*</u> *finitely many primes* instead.

3. There needs to be some groundwork laid before diving into the second sentence. In particular, the proof writer needs to clarify what $p_1, p_2, \dots, p_n$ *are*. It would be better to simply append this information to the first sentence, as in *Assume there are only finitely many primes, say* $p_1, p_2, \dots, p_n$.

4. The second sentence should establish *why* a number like $p_1p_2\cdots p_n+1$ is divisible by some prime, since such an explanation requires some external information. So, the second sentence needs to say something like *Clearly* $p_1p_2\cdots p_n+1>1$*, and so the fundamental theorem of arithmetic guarantees that this number is divisible by a prime.* (By the way, it is not a misprint to use *fundamental theorem of arithmetic* rather than *Fundamental Theorem of Arithmetic*, but more on that later.)

5. The proof writer's second sentence needs a great deal more clarity to drive home the contradiction. In fact, the use of a transition word like *however* would be valuable to the reader at this point, as well as some specifics about who exactly this mysterious p really is. I would say something like *However, this prime must be one of the* p_i*'s.* (We could debate about whether an explanation is important at this point as to *why* the prime must be one of the p_i's, but since the reason has to do with something *internal* to the proof, specifically the fact that $p_1, p_2, \dots, p_n$ are assumed to be *all* the primes, I would feel comfortable leaving it up to the reader to fill in that particular gap.)

6. Now we most definitely need a transition to get to the point where we can clearly see why p_i divides 1. I would suggest *Since* p_i *also divides* $p_1p_2\cdots p_n$*, it follows that* p_i *must divide* 1. Of course, if you wish to provide more of the mechanics as to why it works this way, that would be fine, too. However, I would contend that any reader who is serious about understanding your proof would already have a basic knowledge of properties of "divides".

7. The proof writer stops well short of putting the nail in the coffin on his or her proof by contradiction. After all, it is not unreasonable for a reader to now ask, "Why is this a problem?!" I would suggest a statement here like *But* $p_i \geq 2$. (Notice that this simple statement simultaneously speaks to the nature of p_i as a prime and to the issue with p_i dividing 1.)

8. To appropriately finish the proof by contradiction, the proof writer should bring things full circle with a statement like *This contradiction shows that there must be infinitely many primes.*

So, the fully corrected proof would look like

> Deny. Assume that there are only finitely many primes, say $p_1, p_2, \ldots, p_n$. Clearly $p_1 p_2 \cdots p_n + 1 > 1$, and so the fundamental theorem of arithmetic guarantees that this number is divisible by a prime. However, this prime must be one of the p_i's. Since p_i also divides $p_1 p_2 \cdots p_n$, it follows that p_i must divide 1. But $p_i \geq 2$. This contradiction shows that there must be infinitely many primes.

Hopefully, this example gets your attention that good writing in mathematics isn't like falling off a log. It requires just the right explanation at just the right time worded in just the right way. I would also say it is a bit of balancing act to ensure you don't go overboard on your explanations. After all, I could have made the above proof more than a page long if I wanted to go into what "divides" means, what it means to be a "prime", etc. However, such a proof risks obscuring the beautiful simplicity in this proof by contradiction.

Based upon my experience over the years, here is a list of what I have found to be the most common issues (in no particular order) students have in general mathematical writing.

transition words When you use words like *so*, *thus*, and *then* at the beginning of a sentence (or anywhere for that matter), you are indicating that what you are about to say *logically follows* from what you just said at the end of the previous sentence. For example, if you write

Let n be an even integer. Then $n = 2k$ for some integer k.

you are indicating that the fact that $n = 2k$ for some integer k is a direct logical consequence of n being even (which it is, by definition). If, on the other hand, you wrote

Let n be an even integer. Next, $n = 2k$ for some integer k.

you are indicating that the fact that $n = 2k$ for some integer k is a separate piece of additional information from n being even (which it is not). In short then, use transition words like *so*, *thus*, *then*, *hence*, *therefore*, *it follows that*, and *as a result* **only** when a logical implication occurs. Otherwise, use transition words like *next*, *also*, and *in addition* to introduce extra information.

(By the way, in professional mathematical writing the powerful word therefore is usually reserved for the ultimate transition into the final sentence of a proof. It sounds anticlimactic to have what should be the crescendo of your proof introduced by such a pedestrian word as *then*.)

such that* versus *so that The problem involving logical consequence discussed above is not confined simply to the beginning of a sentence. Take, for example, the confusion over *such that* and *so that*. The phrase *such that* qualifies a particular property you have just mentioned in the initial clause of your sentence, while *so that* indicates that what follows is a logical consequence of what you have just stated in the prior clause. For instance, you could write *let n be an even integer, so that $n = 2k$ for some integer k* but you would *not* write *let n be an even integer such that $n = 2k$ for some integer k.* On the other hand, you could write *let n be an even integer such that n is divisible by* 4 since the set of all integers divisible by 4 is a subset of those that are even. The word *where* can also be used here instead of *such that* to qualify.

let* versus *put* versus *choose The word *let* is used to introduce a particular object to the reader along with some defining quality of that object, as in *let p be a prime number*. It is most often used when the *arbitrariness* of the new object needs to be highlighted (such as when showing that *all* objects in a particular set satisfy a certain property). On the other hand, *put* is used to create an assignment of value, as in *put* $n = k + 1$. It is most often used when the *specificity* of the new object needs to be highlighted. The word *choose* is used when the existence of a particular object is only an intermediate step towards a larger goal, as in *choose a natural number of the form p^2q^2, where p and q are distinct primes* when the ultimate goal of the proof is to prove that there exist natural numbers with exactly 9 positive divisors.

suppose* versus *assume I teach my own students that there really is no effective difference between introducing a statement with the word *suppose* and introducing a statement with the word *assume*. That being said, I tend to see the word *assume* used more often in statements that one plans to eventually contradict.

trivially* versus *clearly* versus *obviously As a general rule, you should use these adverbs sparingly in your work, particularly because these words by their very nature are there to introduce something that you do not intend to explain. (I always chuckle a little bit when I read a sentence like *Clearly,..., since....*) In fact, seasoned mathematicians know that these words are sometimes used to mask the fact that the writer just doesn't understand why a certain statement is true. More than once I have discovered a proof-ending error in a statement that began with the word *obviously*.

However, if you intend to use one of these words because, well, the thing you are about to say really is straightforward, be aware of the tone that your choice will project. The word *trivially* is generally used for statements that follow from the very definition or nature of the object being discussed. The word *obviously* is generally used for statement that follows very quickly from a well-known property of the object being discussed. The word *clearly* is generally used for a statement that follows by doing a very natural or intuitive process on the object being discussed. For example, *If p is a prime number, then p is trivially bigger than 1.*

If p is a prime number bigger than 2, then obviously $p = 4k + 1$ or $p = 4k + 3$ for some whole number k.

If p is a prime number bigger than 2 such that $p = a^2 + b^2$ for some integers a and b, then clearly $p = 4k + 1$ for some natural number k.

Of course, one man's "clearly" is another man's "what the heck?!" An understanding of your audience is paramount in these considerations.

notation and punctuation It is generally considered bad form to introduce a sentence with a mathematical symbol or use a mathematical symbol next to a piece of punctuation (like a comma) in the middle of a sentence. The main reason for this is because it can cause confusion over a proper understanding of the sentence, since a great deal of mathematical notation out there adapts punctuation symbols for ulterior purposes. For example, consider the sentences *How is it! n is not even an integer?* I think a publishing editor could be forgiven if, on a first pass, he or she thought that $!n$ might be some piece of mathematical notation and assume the writer really meant to write a single sentence *How is it $!n$ is not even an integer?*

misapplication of the comma One of the major roles of the comma in grammar is to offset a subordinate clause that is offering supplementary information in a sentence. That is, in this usage, whatever is offset by commas should be removable without affecting the essential understanding of the sentence or subsequent sentences. Unfortunately, all too often I come across students who want to place an important mathematical object between a set of commas in a sentence, as in *The prime number, p, is of the form $4k + 1$.* In such a situation, p makes clear *to which* prime number you are referring, and so should not be offset by commas. In fact, I have personally never come across a situation in which a piece of mathematics should be offset by commas.

11.3 Special Conventions in Mathematical Writing

In this section, I want to present a few things to keep in mind that are peculiar to mathematical writing. These conventions that mathematicians follow in their own writing are unfortunately a burr underneath a thesis reviewer's saddle since they differ so markedly from the normal rules of English composition. Along these lines then, if you find yourself at odds with the Graduate School over any of the following points, you should not fret over changing what you have written to suit the taste of the Graduate School. Any mathematician reading your work will still perfectly understand what you have written.

Latin abbreviations There was a time not too long ago when *all* works in mathematics and science were written purely in Latin. It was, after all, considered the language of the learned. Some could argue that the use of Latin abbreviations such as *e.g.* for "such as" are merely a throwback from that day in age and, as such, have no place in modern mathematical writing. Of course, if you carefully examine this user's manual, you will find that this is not a perspective to which I subscribe. Frankly, I think that Latin abbreviations are making a bit of a resurgence since the internet made looking up such abbreviations as easy as reaching for your smart phone. However, this is not a hill that I intend to die on (and neither should you). So, if the Graduate School insists that you make your *i.e.* "that is", I say graciously comply.

we* versus *he or she Perhaps the biggest bone that those outside of mathematics pick with us concerning our writing is our standard of first person plural. That is, we *never* use the pronoun *I* (even if we are the only author) and we certainly don't use *he*, *she*, or (my personal favorite) *the author*. Both mathematicians and grammarians agree that *I* has no place in formal writing, as it sounds too egotistical. However, for mathematicians, third person sounds either too distant or like someone else is writing the work. For us, the pronoun *we* communicates both a sense of community and intimacy at the same time. It's our way of expressing that, while I may be the one writing this paper, I did not get here alone.

lowercase for theorem names I mentioned earlier that in mathematics, we would write *fundamental theorem of arithmetic*, not *Fundamental Theorem of Arithmetic*. The reason for this is that *fundamental theorem of arithmetic* does not represent an "official" name for the particular result. Moreover, mathematical organizations are not in the business of creating such official names (I can hear the debate now over whether it should be called the *squeeze theorem* or the *sandwich lemma*). So, it does not rise to the level of a proper name that would require capitalization. That being said, you *must* capitalize proper name's that are part of the name of a result, such as *Euclid's theorem*.

emphasizing the defined term In an mathematics article, the author will typically wish to provide the definitions of a few terms used in the paper in the discussion part of the paper (that is, outside a formal definition environment). When this happens, the terms are italicized in order to make it easier for the reader to find later. For example, in the introduction to a paper in multiplicative ideal theory, you may find the sentence "A *prime ideal* P of a commutative ring R is an ideal of R for which whenever $a, b \in R$ with $ab \in P$, it must be the case that either $a \in P$ or $b \in P$".

quotes around concepts Technically, there is a difference between a mathematical object itself and the underlying concept that the object represents. As such, whenever you refer to the concept or definition of an object rather than the object itself, the corresponding term should be offset by quotation marks " ". For example, notice the difference in what is being expressed in the following two sentences:

> *Every prime ideal of a ring is an ideal of the ring.*
>
> *The notion of "prime ideal" of a ring is based upon the definition of an "ideal" of a ring.*

In the former, we are talking about the *object* of a prime ideal (notice that we can express the exact idea of this sentence if we rewrote it as *If P is a prime ideal of a ring R, then P is an ideal of R.*). However, in the latter, we are talking about the underlying *definition* of a "prime ideal".

This is precisely why the term *prime ideal* (and, for that matter, the word *ideal*) was offset by quotation marks.

italicizing subscripts and superscripts This is an issue that crops up more often in applied mathematical writing than in pure mathematics. Technically, if a subscript or superscript to a mathematical object refers to a variable, then the subscript or superscript should be italicized. Otherwise, it should be typeset in normal upright font. For example, say you wish to emphasize the fact that the constant in Hooke's law is a spring constant. Instead of typing `$k_{spring}$` which produces k_{spring}, you should instead type `$k_{\text{spring}}$` which produces k_{spring}. This principle also applies to say the "T" used to identify the transpose of a matrix or the "d" in a differential (such as in a derivative or integral), both of which should be typeset in normal upright font.

11.4 Considerations for Math Environments

One major issue facing writers of mathematics is how to appropriately integrate the mathematics environments themselves into the work so that (1) the usual rules of English composition are respected and (2) the reader has no trouble following the mathematics (an understanding of the mathematics, however, is an entirely different matter). It often requires a bit of work to make the transition between the discussion and the math environments seamless. In this section, I want to provide you with some tools to accomplish just that.

to display or not to display Sometimes it's a real judgement call on whether or not you want to display a piece of mathematics. Usually how important it is to the discussion and how much you want to emphasize it are considerations in this decision. However, there are times in which leaving the item in-text will make it challenging for the reader to easily assess it. You should definitely display your item if any of the following are true:

1. The item involves superscripts of superscripts or subscripts of subscripts.
2. The item involves fractions inside fractions or a relatively involved fraction.
3. The item involves certain pieces of notation such as l that could be confused for the number 1 if the item were placed in-text.
4. The item takes up more than half of a line if left in-text.

punctuation in displayed formulas If you are displaying a formula that happens to fall at the end of a sentence or, more generally, would have required punctuation had it occurred as an in-text expression, then you should include the corresponding punctuation with it even as a displayed formula. Some may argue that the virtue of being displayed inherently creates the requisite separation that a comma or period would have ordinarily provided. However, the convention in mathematics is that we continue to treat the displayed environment as if it were a normal part of the sentence in which it resides and that the displaying process is designed only for emphasis and clarity.

breaking a displayed formula In Section 4.5, various alignment-type environments were discussed, some of which deal specifically with the issue of breaking a long displayed formula. However, the conventions for breaking a such a formula are as follows:

1. Try to break the formula just *before* a binary relation or operation (like $+$ or $-$).
2. If you break a formula just before a $+$ or $-$, start the next line with `{}+` or `{}-`, respectively, instead of just $+$ or $-$.
3. If you break a formula within a bracket, indent the next line so that it begins to the *right* of the opening bracket.

definition environments When you use a formal definition environment it is helpful to a reader if you emphasize the term that you are defining. Common ways of doing this are by boldfacing the **term** or italicizing the *term* (assuming of course that your definition environment does not already consist of italicized text). For example,

> **Definition 11.4.1.** Let $p > 1$ be a natural number. We call p a **prime number** if the only positive divisors of p are 1 and p.

low dots versus center dots In mathematics, there are two general ways to express the continuation of a pattern, using an ellipses-like notation $\ldots$ given by the macro `\ldots` or using the raised version $\cdots$ given by the macro `\cdots`. The basic rule is that `\ldots` is used only for sequences, such as $a_1, a_2, a_3, \ldots, a_n$. Any other time, `\cdots` is applied. This is especially true if there is an operation involved, as in $a_1 + a_2 + a_3 + \cdots + a_n$.

Chapter 12

Front Matter Matters

Every year, many, many stupid people graduate from college. And if they can do it, so can you.

John Green

In this chapter, I let the rubber meet the road and discuss how to build the first part of your thesis or dissertation, what is often called the "front matter" material, using TNThesis. In the follow-up chapters to this one, I will show you how to do the same for your thesis or dissertation chapters and post-chapter material, i.e., the appendices, bibliography, and vita page.

12.1 Packages Loaded by TNThesis

TNThesis is built on top of the memoir class file and, as such, has access to all macros established by the memoir class.[1] In addition to the packages emulated by the memoir class, there are a fair number of others loaded directly by TNThesis. The complete list of all packages associated with TNThesis is given below, with those loaded directly by TNThesis boldfaced. Each member of the list (with the exception of the lmodern package which just specifies the font) is hyperlinked to its CTAN page, so that you can check out the package's documentation, if you would like. However, it is advisable that, for those packages that have been merely emulated by the memoir class (indicated by the lack of boldfacing), you instead check out the memoir class documentation.

abstract, **amsfonts**, **amssymb**, **amsthm**, appendix, array, **babel**[2] , **biblatex**, **bigdelim**, booktabs, ccaption, **centernot**, chngcntr, chngpage, **cleveref**, **colortbl**, **csquotes**[3] , dcolumn, delarray, enumerate, **enumitem**, epigraph, **fontenc**[4] , framed, **graphicx**, ifmtarg, ifpdf, index, **inputenc**[5] , **interval**, **latexsym**, **lmodern**, makeidx, **mathrsfs**,

[1] memoir is given the options 12pt, extrafontsizes, oldfontcommands, and twoside

[2] babel is loaded with the american option

[3] csquotes is loaded with the autostyle option

[4] fontenc is loaded with the OT1 option

[5] inputenc is loaded with the utf8 option

mathtools, **microtype**, **minted**[6] , moreverb, **multirow**, needspace, newfile, nextpage, parskip, patchcmd, **ragged2e**, setspace, shortvrb, showidx, tabularx, **thmtools**, **tikz**[7] , titleref, titling, tocbibind, tocloft, **totcount**, **url**, verbatim, verse, **xcolor**

The evils of `hyperref`

The `hyperref` package is one of the most popular packages to use in large works because it has the capability of creating pdf bookmarks and clickable hyperlinks from the table of contents, references, and citations to their corresponding chapters, results, and entries in the bibliography, respectively. However, `hyperref` can present some profound issues when used in combination with the `memoir` class, upon which `TNThesis` is based. Moreover, not only is the use of this package *unequivocally* discouraged here, but LaTeX simply will not compile `TNTmain` with this package loaded. In my experience, while there are enough internal workarounds that can be created to `TNThesis` to allow `TNTmain` to compile with `hyperref` loaded, (1) you probably will not like the way `hyperref` has to readjust the appearance of your work to suit its needs, and (2) when you are ready to submit your work to your committee or the Graduate School, you will need to remove `hyperref`, which will entail systematically and painstakingly undoing all of the changes it originally made. In my opinion, it's just not worth it.

12.2 Overview of `TNTmain`

All of the input that you will give to LaTeX will be done through the file `TNTmain`. In fact, all of the commands that you will issue to LaTeX in the subsections that follow will be placed in the preamble of `TNTmain` (that is, in the space *after* `\documentclass` and *before* `\ThesisContent` (which is a macro exclusive to `TNTmathTHSIS` that creates the necessary `\begin{document}` and `\end{document}` commands; see Section 2.2). Generally, the order in which you place the commands does not matter to LaTeX, although good programming practice is to preserve a certain sensible order to things so that you can quickly find particular commands later, if need be.

`TNTmain` begins with the command line

```
\documentclass{TNThesis}
```

and ends with the macro `\ThesisContent`, which is a macro representing the code

[6] **minted** is loaded only if the user specifies it as a documentclass option and is then loaded with the **cache** and **newfloat** options activated

[7] **tikz** is loaded only if the user specifies it as a documentclass option

```
\begin{document}
    \frontmattercontent
    \mainmatter
        \mainmattercontent
    \backmattercontent
\end{document}
```

Each of `\frontmattercontent`, `\mainmattercontent`, and `\backmattercontent` is itself a special macro exclusive to `TNThesis` that calls a certain sequence of parts of your thesis or dissertation. As such, you should *not* comment out (see page 24) the `\ThesisContent` macro *nor* add any macros of your own after this command (you may of course add macros before this command; see, in particular, *Make Your Own Macros at Home*, Section 6.3). If you would like to get an advanced look at `TNTmain`, see Appendix C.

In what follows, I describe some of the particulars of how you can create the front matter of your thesis or dissertation.

Doing the bare minimum

`TNThesis` is designed to allow you to compile your work even if you do not have all of the requisite information entered through `TNTmain`. However, if you do not have a particular item entered via the correct macro, you will find a warning message in the compilation log as well as a message in your output in the respective space of your work indicating that you are missing the item. For example, if you neglect to input your abstract file using the `\ThesisAbstract` command (described in **??**), then the message

$$\langle\langle\text{Abstract Not Provided}\rangle\rangle$$

will appear on the abstract page. You should ensure that no such messages are present before officially submitting your work to the Graduate School for approval.

12.3 Front Matter: The Preliminary Pages

The initial pages of your thesis or dissertation consist of the following items (in the order in which they must appear):

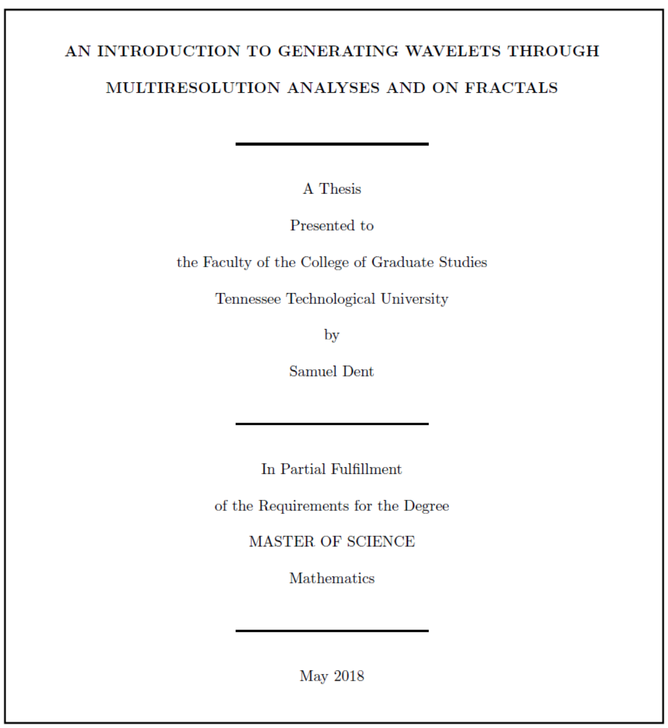
1. the title page
2. the abstract page
3. the copyright page
4. the table of contents
5. the approval page
6. the dedication page (optional)
7. the acknowledgments page (optional)

8. the list of figures (if needed)
9. the list of tables (if needed)
10. the list of listings (if needed)
11. any extra lists (optional)

Let's take a look at each of these in turn and how `TNThesis` creates them.

12.3.1 The Title Page

The title page of your work is the *first official* page of your thesis or dissertation, even though the page number does not appear on the page (technically though, it is still identified as page i by TeX). It is designed to clearly indicate what the title of the work is, who the author is, and for what purpose the work is done (basically, in partial fulfillment of the degree requirements of Tennessee Tech). The information is to be both horizontally and vertically centered and to spread vertically across the expanse of the page. An example of a title page is given in Figure 12.1.


AN INTRODUCTION TO GENERATING WAVELETS THROUGH
MULTIRESOLUTION ANALYSES AND ON FRACTALS

A Thesis
Presented to
the Faculty of the College of Graduate Studies
Tennessee Technological University
by
Samuel Dent

In Partial Fulfillment
of the Requirements for the Degree
MASTER OF SCIENCE
Mathematics

May 2018

FIGURE 12.1: An example of a title page

`TNThesis` is designed to allow you to input the relevant information through a series of commands and automatically place these items in the correct locations with the appropriate font size and indentation. After the `\documentclass` command, you can begin your program in `TNTmain` with the following macros:

```
\ThesisTitle{⟨title⟩}
\ThesisAuthor{⟨author⟩}
```

where ⟨*title*⟩ is the full title of your work and ⟨*author*⟩ is the name with which you enrolled at Tennessee Tech.

In addition to the above information, you will need to provide the document type, the degree type, your major, your graduation month, and your graduation year. Each of these pieces of information can be provided through the following macros:

```
\DocumentType{⟨type⟩}
\DegreeType{⟨degree⟩}
\Major{⟨major⟩}
\GraduationMonth{⟨month⟩}
\GraduationYear{⟨year⟩}
```

where ⟨*type*⟩ is either `Thesis` for a master's degree or `Dissertation` for a doctorate degree and ⟨*degree*⟩ is either `Master of Science`, `Master of Art`, or `Doctor of Philosophy`. ⟨*major*⟩ is the official name of your major (e.g., `Mathematics`). Also, ⟨*month*⟩ is your anticipated graduation month in standard form (e.g., `May`, `August`) and ⟨*year*⟩ is your anticipated graduation year in standard form (e.g., 2022). Each of these items (save of course the graduation year) should be capitalized. Once you have provided this information, `TNThesis` will be able to process the title page when you eventually compile your work.

The title of your work on the title page must be in all caps, boldfaced, and double-spaced in the form of (no joke) an inverted Mayan pyramid (see Figure 12.2). In addition, the title must appear the same on the abstract page and the approval page as it does on the title page, save single spacing instead of double spacing on the abstract page. Don't worry though, `TNThesis` will handle all of this special formatting for you!Moreover, `TNThesis` has an internal algorithm to find the optimally presented (relative to interword spacing) title *allowing*, to a certain extent, end-of-line hyphens.

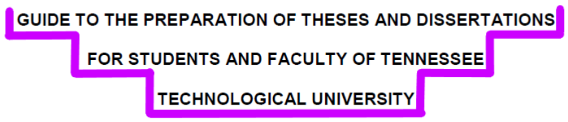


FIGURE 12.2: The inverted Mayan pyramid look of the title on the title page

It's okay to blank out

When TEX compiles your thesis or dissertation, you will find that there is a blank page between the title page and the abstract page. There is perfectly normal. In particular, the abstract page must begin on a recto (that is, an odd-numbered) page, and so `TNThesis` will insert a blank page to ensure that this happens.

12.3.2 The Abstract Page

The abstract page is a *single page* that introduces your entire work to the reader. Your abstract is supposed to briefly explain to a reader what problem or question you entertained in your thesis or dissertation and how you sought to answer the problem or question in your work. It should be clear and concise, providing just enough background information that a reader will at the very least understand the general context in which the problem exists. The abstract should then provide an outline of your methodology in solving this problem and any significant conclusions that resulted from your work.

The file is always in the last place you look

You will need to make certain that `TNTmain` can find all of the various parts of your thesis or dissertation. As such, it is a good idea to simply go ahead from the beginning and place the file containing your abstract (and all other parts of your work) in the same folder as `TNTmain` and `TNThesis`.

The abstract should *not* contain any citations, however it *may* contain references to numbered results in the work (e.g., Theorem 1.2.5) as long as such references are parenthetical in nature. However, it is generally considered to be better form to simply explain the result and forego any reference to its specific location in your work. Remember that an abstract should give a reader a broad picture of what it is that you have accomplished in your thesis or dissertation and thus should not be overly formal.

The abstract page itself consists of the title of your work in all caps, boldfaced, single-spaced in the form of an inverted Mayan pyramid at the top followed by your name, the degree sought, and then the abstract itself, also single-spaced. (The abstract page is the *only* page of the thesis or dissertation that is permitted to be single-spaced. The rest must be double-spaced. `TNThesis`

Before washing, separate your files

As with all of the essential parts of your thesis or dissertation, you will need to create separate `tex` files for each one. However, there is no need to worry about formatting, as `TNThesis` automatically handles this in light of Graduate School rules in these matters.

will handle the atypical spacing on the abstract page.) An example of an abstract page is given in Figure 12.3.

AN ABSTRACT OF A THESIS

RESIDUAL BASED FILTERING PROCEDURE FOR THE MOVING BLOCK BOOTSTRAP ON AR(1) DATA

Seth Enoch Brys

Master of Science in Mathematics

The moving block bootstrap is a resampling procedure for time-dependent data that uses randomly selected blocks of consecutive data points from the original time series to construct a bootstrap sample. Since the value of the current observation in a time series depends in some manner on the values at previous points in time, it is possible that randomly selecting blocks of data with the the moving block bootstrap could cause the dependency present in the original series to be lost in the resample. In this thesis, a procedure of block selection for the moving block bootstrap is proposed that aims to preserve the dependency of the original data by selecting blocks that will keep the dependency throughout the construction of the resample.

FIGURE 12.3: An example of an abstract page

Assuming that you already have given `TNTmain` the requisite information for the title page, all you need to do at this point is to input your abstract file by way of the macro

```
\ThesisAbstract{⟨filename⟩}
```

where *⟨filename⟩* is the name of the file containing your abstract, *without* the `tex` extension.

12.3.3 The Copyright Page

The copyright page immediately follows the abstract page of your work. In most cases, you as the thesis or dissertation writer reserve the rights to the new results espoused in your work (see Figure 12.4 for an example of what a copyright statement would look like in this case). However, this may not be the case if you have received funding for your research or if you made a special arrangement with another party for your research. If you have any question about who owns the copyright to your work, check with your major advisor before inputting the necessary information for this page. If you are the one who owns the copyright to your work and would like to formally register a copyright for it, visit **copyright.gov**.

Copyright © Andrew Depp McDole, 2021

FIGURE 12.4: An example of a copyright statement for an individual

It's mine and you can't have it

! Aside from some agreed-upon prior arrangement regarding your research, you own the copyright to your work *regardless* of whether or not you formally register for a copyright with the U.S. government. That being said, if you one day need to legally defend your copyright, you will first need to formally register your work by visiting **copyright.gov** and providing the appropriate information.

The information on the copyright page is both horizontally and vertically centered and declares who reserves the rights to the work contained therein. If you own the copyright to the work in your thesis or dissertation, there is nothing to do for the sake of creating this page, as `TNThesis` already has all it needs from the title page to create the copyright page. On the other hand, if someone else owns the copyright to your work, you will need to provide the corresponding information by using the macro

```
\CopyrightOwner{⟨name⟩}
```

where ⟨*name*⟩ is the name of the individual(s) or corporation that owns the rights to the results contained in your thesis or dissertation. As with all of the user-input macros, the `\CopyrightOwner` macro needs to be placed in the preamble of `TNTmain`.

12.3.4 The Table of Contents

Unlike the title page, the table of contents literally writes itself. It looks for commands throughout the work such as `\chapter`, `\section`, `\subsection`, etc. and identifies the pages on which those commands start creating the corresponding parts of your work.

\chapter→p.170

Some conventions about the table of contents that `TNThesis` creates bear mentioning:

- The table of contents starts listing elements of the work beginning *after* the table of contents. Thus, nothing from the prior front matter (i.e., the title page and copyright page) will show up in the table of contents.
- Sections are listed underneath their respective chapters, subsections are listed underneath their respective sections, and subsubsections are listed underneath their respective subsections. `TNThesis` limits the listing of elements of your work at the subsubsection level (that is, paragraphs and lower divisions are *not* listed in the table of contents).
- However you enter your chapter, section, etc. titles in the corresponding commands is how it will show up in the table of contents.
- Chapters, sections, subsections, and subsubsections are numbered in the table of contents. Appendices are lettered.
- The table of contents is double-spaced, save a lengthy title that occupies multiple lines, in which case the title is single-spaced.[8]
- The table of contents can take as many pages as it needs.

12.3.5 The Approval Page

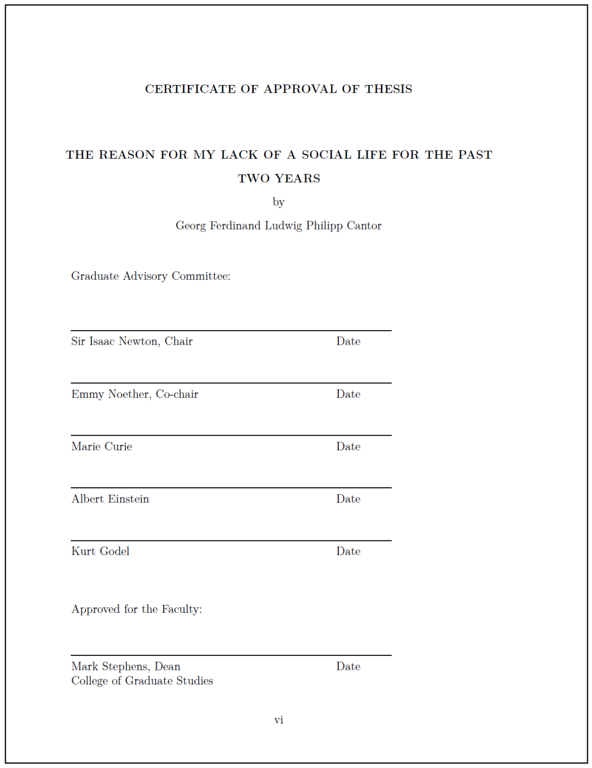
The approval page is where your thesis or dissertation committee members, as well as the Dean of the College of Graduate Studies (who is also the Senior Associate Vice President for Academic Affairs) at Tennessee Tech, sign off on your work, testifying to its completeness. An example of an approval page is given in Figure 12.5.

While you yourself will obviously need to obtain the signatures of your committee members on this page (usually after you have adequately addressed any concerns they have with your work), the Graduate School is responsible for obtaining the signature of the Dean of the College of Graduate Studies.

To have `TNThesis` create the approval page, you will need to supply the names of your committee members in `TNTmain`. The macro for this is

```
\CommitteeMember{⟨number⟩}{⟨name⟩}
```

[8] This feature can be changed if your editor at the Graduate School *really* insists that it be changed. However, I am of the opinion that your table of contents looks better the way `TNThesis` is originally programmed.



CERTIFICATE OF APPROVAL OF THESIS

THE REASON FOR MY LACK OF A SOCIAL LIFE FOR THE PAST
TWO YEARS

by

Georg Ferdinand Ludwig Philipp Cantor

Graduate Advisory Committee:

Sir Isaac Newton, Chair Date

Emmy Noether, Co-chair Date

Marie Curie Date

Albert Einstein Date

Kurt Godel Date

Approved for the Faculty:

Mark Stephens, Dean Date
College of Graduate Studies

vi

Figure 12.5: An example of an approval page

where ⟨*number*⟩ corresponds to the place in which the respective committee member's ⟨*name*⟩ will appear on the approval page. Of course, there are conventions for *how* the names must appear:

- ⟨*name*⟩ consists of the committee member's name in the format:

  first name ⟨*space*⟩ middle initial or middle name ⟨*space*⟩ last name

  It does *not* include any titles (e.g., Dr., Mr., Ms.).

- The first committee member's name is *always* the committee chairperson followed by a comma and the word "Chair" and the second committee member's name is *always* the committee co-chair (if one exists) followed by a comma and the word "Co-chair".

The seal of approval for your approval page

! *Before* obtaining any signatures, e-mail your approval page to Sharon Holderman at `sholderman@tntech.edu` to get approval on the formatting of the approval page. This will save you from potentially having to reobtain signatures if there is an issue with the formatting.

Since a chair and co-chairperson are assumed to have equal status on the committee, the determination of who is identified as chair and who is identified as co-chair is made simply by whose last name comes first alphabetically (and this person is then the chair). There is no such thing as co-co-chair or co-chairpeople.

- You should absolutely refrain from using the term “Chairman” or “Chairwoman”. “Chair” best reflects the egalitarian nature of the position.
- All other committee members are presented alphabetically by last name.

An example of the code in this matter is given as follows:

```
\CommitteeMember{1}{Albert Einstein, Chair}
\CommitteeMember{2}{Georg Cantor}
\CommittteeMember{3}{Emmy Noether}
\CommitteeMember{4}{Karl Weierstrass}
```

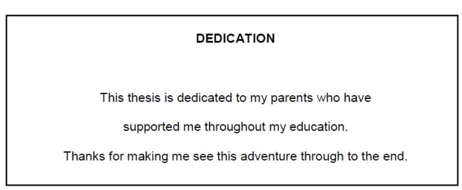
The rest of the information needed for the approval page has already been supplied on previous pages.

12.3.6 The Dedication Page

The dedication page is an *optional, single* page that is centered both vertically and horizontally, but is not fully justified in spite of being horizontally centered. Dedications are short one- to two-sentence statements that pay tribute to an individual or individuals for the work (see Figure 12.6 for an example of a dedication). Usually it is an individual close to you who provided some inspirational or emotional support along the way, but it does not absolutely have to be. Should you wish to include a dedication page, use the command

```
\Dedication{⟨text⟩}
```

in the preamble of `TNTmain`, where ⟨*text*⟩is the text of your dedication.


DEDICATION

This thesis is dedicated to my parents who have supported me throughout my education.

Thanks for making me see this adventure through to the end.

FIGURE 12.6: An example of a dedication

12.3.7 The Acknowledgments Page

The acknowledgments page is an *optional, single* page that provides an opportunity for you to thank those individuals who provided substantive support, usually academic or financial, for your work. Often this is a place where students thank their major advisors and the members of their committee. In addition, if the research leading to your thesis or dissertation was funded by an organization, the acknowledgments page would be the appropriate place to thank them. Acknowledgments usually are but a single paragraph in length and really never more than two (see Figure 12.7 for an example of an acknowledgment). If you wish to provide an acknowledgments page, use the command

```
\Acknowledgments{⟨filename⟩}
```

where ⟨*filename*⟩ is the filename (again, *without* the .tex extension) containing the text of your acknowledgments.

ACKNOWLEDGMENTS

Thank you to my adviser, Dr. Amy Chambers, for guiding me through the process of writing this thesis and for encouraging me to the end. Thank you to my committee for equipping me with the knowledge and skills to be able to take on this thesis topic, and for aiding me in editing this paper. I am grateful towards the Mathematics Department for providing me with educational opportunities and valuable life experiences over the past two years. I also extend my appreciation to my friends and family for encouraging me through this process. Thank you for your unending love and support.

FIGURE 12.7: An example of an acknowledgment

12.3.8 The List of Figures, List of Tables, and List of Listings

Much like the table of contents, the list of figures, list of tables, and list of listings writes itself. As a general rule of composition, if you have more than one figure (respectively, more than one table, more than one listing) in your work, you are required to put in a list of figures (respectively, a list of tables, a list of listings). However, `TNThesis` will automatically create the appropriate list if you have at least one item in your work of the corresponding type.

\caption→p.112

Each of these lists identifies the page number that the caption of a figure, table, or listing falls (a caption is required for any figure, table, or listing in your work) and supplies that page number in the list along with the corresponding caption (or short caption, if appropriate). Just as with the table of contents, the list of figures, list of tables, and list of listings can take as many pages as they need.

Who stole my list!

! Due to the internal means by which `TNThesis` automatically produces the list of figures, list of tables, and list of listings, it may be necessary to compile your work twice before you will see the appropriate list(s) appear in the front matter and table of contents of your work.

12.3.9 An Additional List (or Three)

To round off the front matter, `TNThesis` provides you with the means to produce any number of additional lists of your choosing, automatically formatted in the same basic style as the list of figures, the list of tables, and the list of listings. You may wish to include a list of symbols, a list of acronyms, a list of abbreviations, or any such items used in your work.

The macro in charge of producing everything that you will need for these additional lists is as follows:

```
\SpecialListItem{⟨macro name⟩}{⟨file extension⟩}{⟨type⟩}
```

where ⟨*macro name*⟩ is the name of the macro (*without* the backslash \) that you will use in your work at the places you wish to create a page reference in the respective list, ⟨*file extension*⟩ is the (usually) three letter extension where you want TeX to store the relevant information, and ⟨*type*⟩ is a name for the type of items being referenced (e.g., Symbol, Abbreviation, Acronym) *expressed as a singular noun*. Don't worry, `TNThesis` will pluralize it where appropriate. What you indicate for ⟨*type*⟩ will be used to provide the title of the additional list.

Be aware that you will need to use separate instances of the `\SpecialListItem` macro for each type of additional list that you wish to produce and that each one of these commands should be placed in `TNTmain` just before the `\ThesisContent` macro.

Now at the place in your work that you want a reference to the page number of a particular item to be placed in the appropriate list, use the command

```
\⟨name⟩[⟨description⟩]{⟨item⟩}
```

where \⟨*name*⟩ is the actual macro (automatically) created from the name that you chose when you set the corresponding `\SpecialListItem` macro in `TNTmain`, ⟨*item*⟩ is the actual symbol, acronym, etc. and ⟨*description*⟩ is the optional brief explanation that you want accompanying ⟨*item*⟩ in the special list.

For example, suppose that you are creating a special list of symbols used in your work where the corresponding page numbers represent the first occurrences of the symbols in the work. Say you want to include the "floor" function in this list. Type something like

```
\SpecialListItem{symb}{los}{Symbol}
```

somewhere just before the `\ThesisContent` macro in `TNTmain`. Then somewhere close to the first occurrence of the floor function in the body of your work, type

```
\symb[the floor of $x$]{$\lfloor x \rfloor$}
```

`TNThesis` will automatically flag the page on which the command falls and include this page number next to the listing of the floor function in the special list. Of course, if you wish to have the page reference be to another place in which the floor function appears (say to the actual definition of the floor function), simply move the `\symb` command to the desired instance of the floor function. Once you compile your work twice, you should see your special list appear just before the first chapter of your work.

To make sure everything looks as it should, keep the following in mind:

- Unlike indexing, the \⟨*name*⟩ command is designed to be used *only once* for a given item. So you will not be able to include all the occurrences of a particular item in your special list (unless of course you have only one occurrence of each such item in your work). So, be judicious about which page you would like to reference for each item.
- For the sake of the appearance of your special list, if you opt to provide a description for one item in your special list, you should provide a description for all the items in your special list. Otherwise, your special list will look as though you *forgot* to provide descriptions for certain items.
- If you find the items in your special list appearing cramped, you can always increase the width of the (invisible) box in which they are typeset by using the command

  ```
  \⟨name⟩width{⟨width⟩}
  ```

  where ⟨*name*⟩ is the same macro name created from the `\SpecialListItem` macro and ⟨*width*⟩ is the desired width of the box in which the items reside. The default is 7 em. If you use this change-in-width option, be sure to place the command in `TNTmain` before the `\ThesisContent` macro.

Chapter 13

We're Way Past Introductions

> Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time.
>
> Thomas Edison

Now we turn to the beating heart of your work, the research material that will (hopefully) fill most of the pages of your thesis or dissertation. In any type of written work, the major partitions of the work are given by the following hierarchy:

book
part
chapter
section
subsection
subsubsection
paragraph
subparagraph

although not all works have all of these divisions. In particular, more than likely, your thesis or dissertation will not have parts or subparagraphs (technically, the thesis or dissertation itself is considered at the level of book). In addition, by rule, a chapter may not have only one section, a section may not have only one subsection, and a subsection many not have only one subsubsection (otherwise, it should be left as a chapter without sections, a section without subsections, and a subsection without subsubsections, respectively.)

13.1 Chapters

You should type up each chapter in its own `tex` file with the first line of each chapter being the command

```
\chapter[⟨ToC title⟩]{⟨title⟩}
```

where *⟨title⟩* is the title that you want to appear on the chapter title page and *⟨ToC title⟩* is an optional shortened version of the title that will appear in the table of contents (useful if you feel that your chapter title is so long that it looks awkward in the table of contents). If *⟨ToC title⟩* is not used, then `TNThesis` will use the title given in *⟨title⟩* in the table of contents, as well. Of course, if you need to refer to the chapter in other places of your work, be sure to place the command `\label{⟨label name⟩}` right after the aforementioned `\chapter` command (the same principle applies to sections, subsections, subsubsections, figures, tables, etc.). \label→p.43

Follow the rules

! It is important that you use the macros `\chapter`, `\section`, `\subsection`, and `\subsubsection` when creating the various divisions of your thesis or dissertation. LaTeX specifically looks for these commands when creating the table of contents to your work.

An example of some code at the beginning of a chapter file is as follows:

```
\chapter{Ahmes Expansions of Rational Numbers of Length Two} \label{Ahmes}

In this chapter, we provide three algebraic characterizations of the rational
numbers $q$ for which there exist positive integers $m_1 < m_2$ such that $q =
1/m_1 + 1/m_2$. These characterizations lead to a determination of the minimum
``length'' $k$ for which a given rational number $q$ of the form $2/n$ or $3/n$,
with $n > 1$, can be expressed as $1/m_1 + \, \cdots \, + 1/m_k$, for positive
integers $m_1 < \, \ldots \, < m_k$.
```

The beginning of a chapter always occurs on an odd-numbered page (so `TNThesis` may insert a blank page between the end of one chapter and the beginning of the next to assure that this happens). The chapter label and title are both centered, boldfaced, and in all caps, as illustrated in Figure 13.1 (technically, APA 7 style does not require that the title be in all caps, but the Graduate School does).

The formatting of the chapter label and title are done automatically for you by `TNThesis`.

CHAPTER 1

INTRODUCTION

1.1 Background of Energy Harvesting From Vehicles

In recent years, transportation has been a leading contribution to the drastic climate change by way of increased greenhouse gas emissions. The impact of which has been felt globally through extreme hurricane seasons, rise in sea levels, dense

FIGURE 13.1: An example of the beginning of a chapter

13.2 Sections

Sometimes a chapter is relatively lengthy or deals with multiple major subtopics. In this case, it is appropriate to create sections within the chapter. To create a section within a chapter, simply place the command

```
\section[⟨ToC title⟩]{⟨title⟩}
```

at the location within the chapter where you want the section to begin. Just as with the `\chapter` command, *⟨title⟩* is the title that you want to appear in the text itself and *⟨ToC title⟩* is an optional shortened version of the title that will appear in the table of contents (useful if you feel that your section title is so long that it looks awkward in the table of contents). If *⟨ToC title⟩* is not used, then `TNThesis` will use the title given in *⟨title⟩* in the table of contents, as well. Of course, if you need to refer to the section in other places of your work, be sure to place the command \label→p.43 `\label{⟨label name⟩}` right after the aforementioned `\section` command.

An example of some code at the beginning of a section is as follows:

```
\section{Introduction} \label{Introduction}

The Egyptian, or Ahmes, algorithm  states that any positive rational number $q$
may be expressed as $q=1/m_1 + \, \cdots \, + 1/m_k$ for some positive integers
$m_1 < \, \ldots \, < m_k$. It is customary to refer to such a sum as an ``Ahmes
expansion (of $q$)'' and to the summands $1/m_i$ as ``unit'' fractions. The
``Ahmes'' terminology honors the appearance of a list of such sums in the Rhind
Papyrus. Several proofs of the Ahmes algorithm are accessible to students in a
course on elementary number theory.
```

The beginning of a section need not occur at any special place within a chapter, although it looks a little awkward if you do not have at least a sentence or two between the beginning of a chapter and the beginning of the first section in the chapter. The section label and title are centered, boldfaced, and in title case (but *not* in all caps to distinguish it from a chapter label and title), as illustrated in Figure 13.2.

> **1.2 Research Motivation**
>
> The resonant nature of vibration energy harvesters causes them to operate when the excitation frequency coincides with the harvester's resonant frequency. However, the energy harvested from a vehicle's suspension is non-resonant, and maximum power can be harvested from a vehicle's suspension system only when the excitation frequency matches the resonance frequency [28]. As a result, an energy harvester with only one fixed resonance frequency has a significantly lower efficiency because the energy harvester is not always in resonance with the excitation source. In moving

FIGURE 13.2: An example of the beginning of a section

Save the appropriate capitalization, the formatting of the section label and title are done automatically for you by `TNThesis`.

Get your title back with title case

! According to the APA 7 standard, section, subsection, and subsubsection titles need to be presented in "title case". By the APA 7 standard, this means that

- the first word is capitalized,
- every "major" word (i.e., noun, verb, adjective, adverb, and pronoun), including the second part of hyphenated major words, is capitalized, and
- every "minor" word (i.e., conjunction, article, preposition) consisting of four or more letters (e.g., "With", "From", "Between") is capitalized.

Moreover, every other word is not capitalized.

13.3 Subsections

Sometimes even the creation of sections within a chapter may not be enough to truly parse out the important points. You may need to create subsections within certain sections. To introduce a subsection, simply place the command

```
\subsection[⟨ToC title⟩]{⟨title⟩}
```

at the place in your code where you want the subsection to begin. Just as with both the `\chapter` and `\section` commands, ⟨*title*⟩ is the title that you want to appear in the text itself and ⟨*ToC title*⟩ is an optional shortened version of the title that will appear in the table of contents (useful if you feel that your subsection title is so long that it looks awkward in the table of contents). If ⟨*ToC title*⟩ is not used, then `TNThesis` will use the title given in ⟨*title*⟩ in the table of contents, as well. Of course, if you need to refer to the subsection in other places of your work, be sure to place the command `\label{⟨label name⟩}` right after the aforementioned `\subsection` command.

\label→p.43

An example of some code at the beginning of a subsection is as follows:

```
\subsection{Unit Fraction Expansions} \label{UnitFraction}

During the last half-century, several Ahmes-like expansions of rational numbers
have been studied in which $k$ is fixed but the unit fractions are not necessarily
distinct. In particular, Sierpi\'{n}ski  demonstrated that the set of all positive
rational numbers with an expansion involving at most $k$ unit fractions is
nowhere dense in the set of all positive rational numbers. More recently, Croot
\textit{et al.}  showed that for each $\varepsilon > 0$ and $k > 1$, the number of
positive rational numbers $a/n$ admitting an expansion involving precisely $k$
unit fractions is $\ll_{k,\varepsilon} n^{\alpha_k + \varepsilon}$ as a function of
$n$, where $\alpha_k$ is a rational number determined by a recurrence relation
and $0 \leq \alpha_k < 1$.
```

The beginning of a subsection need not occur at any special place within a section, although it looks a little awkward if you do not have at least a sentence or two between the beginning of a section and the beginning of the first subsection in the section. The subsection title is flushed left, boldfaced, and presented in title case (but *not* centered to distinguish it from a section label and title), as illustrated in Figure 13.3.

Save the appropriate capitalization, the formatting of the subsection title is done automatically for you by `TNThesis`.

2.3.2 Vibration Energy Transduction Mechanisms

Subjecting a device to vibration can result in linear movement through inertial mass. This movement can be converted to electrical energy using various transduction mechanisms. In various kinds of literature over the years, several transduction mechanisms have been proposed and developed for many applications. The major ones used are piezoelectric, electrostatic, and electromagnetic.

FIGURE 13.3: An example of the beginning of a subsection

13.4 Subsubsections (If You Must)

Avid readers of this manual know by this point my feelings concerning subsubsections. I understand that those in engineering and the sciences make frequent use of them, particularly for presenting the details of various experiments that were run or trials that were conducted. I get it. However, let me make the following points concerning subsubsections:

1. I would argue that for most readers, the very notion of a "subsubsection" is almost comical. Unfortunately, once your reader has reached this opinion, his or her overall impression of your work suffers.

2. As long as the details of the various trials, experiments, studies, etc. are not too lengthy, it would be more appropriate to simply present them using a list environment within the appropriate subsection; see Chapter 7.

3. If the details of the various trials, experiments, studies, etc. are lengthy, then it would seem that each deserves its own subsection.

4. Rules of English composition require that any items such as figures, tables, or listings cannot appear in a subsequent division of the work from where they are first mentioned, no matter how small that division that may be. So, if you place such an item in a (relatively short) subsubsection, you will force `TNThesis` to effectively "dump" it from storage before it begins the next division. Unfortunately, to accomplish this, `TNThesis` follows the standard LaTeX protocol of starting the next division on a new page. This often leads to unsightly blocks of whitespace right after the item is outputted.

However, if you (or rather your major advisor) are determined to have them, simply place the command

```
\subsubsection[⟨ToC title⟩]{⟨title⟩}
```

at the place in your code where you want the subsection to begin. Just as with both the `\chapter`, `\section`, and `\subsection` commands, *⟨title⟩* is the title that you want to appear in the text itself and *⟨ToC title⟩* is an optional shortened version of the title that will appear in the table of contents (useful if you feel that your subsubsection title is so long that it looks awkward in the table of contents). If *⟨ToC title⟩* is not used, then `TNThesis` will use the title given in *⟨title⟩* in the table of contents, as well. Of course, if you need to refer to the subsubsection in other places of your work, be sure to place the command `\label{⟨label name⟩}` right after the aforementioned `\subsubsection` command.

\label→p.43

The beginning of a subsubsection need not occur at any special place within a subsection, although it looks a little awkward if you do not have at least a sentence or two between the beginning of a subsection and the beginning of the first subsubsection in the subsection. The subsubsection title is flushed left, boldfaced, italicized (to distinguish it from a subsection label and title), and presented in title case, as illustrated in Figure 13.4.

> ***2.3.2.1 Piezoelectric Energy Generators***
>
> Piezoelectric materials can alter their electric polarization (charge) when strained. It also has a reverse piezoelectric effect by causing mechanical strain when an electric field is applied to the material, as seen in Figure 7. When pressure is applied, piezoelectric energy harvesters transform the electric charges created by piezo materials (PZT, PVDF, PbTiO3, ZnO) into energy. Numerous researchers have investigated the use of piezoelectric materials for energizing self-powered devices [39, 41, 72]. Due to its compact structure, this harvester is mostly used as an energy source for microelectromechanical systems (MEMS), wireless sensors, and smart structures [21].

FIGURE 13.4: An example of the beginning of a subsubsection

Save the appropriate capitalization, the formatting of the subsubsection title is done automatically for you by `TNThesis`.

13.5 Putting It All Together

Once you have your chapters typed up and ready to load into the preamble of `TNTmain`, use the macro

```
\MyChapter{⟨number⟩}{⟨filename⟩}
```

where ⟨*number*⟩ corresponds to the place the chapter will appear in your work and ⟨*filename*⟩ is the filename (say it with me, *without* the `tex` extension) of the file containing the corresponding chapter.

Get in line, chapter!

! The ordering of your chapters is dictated by ⟨*number*⟩ in the above macro, *not* the order in which you list the various `\MyChapter` commands in the preamble of `TNTmain`.

An example of such code would be

```
\MyChapter{1}{background}
\MyChapter{2}{history}
\MyChapter{3}{results}
\MyChapter{4}futurework}
```

where `background`, `history`, `results`, and `futurework` are the filenames for the `tex` files containing the first, second, third, and fourth chapters, respectively, of the work.

Stop trying to compile me!

To compile your chapters after you have loaded them into `TNTmain`, you must compile `TNTmain`, *not* the chapter files themselves. This holds true as well for your appendices (if you have any), your bibliography, and your vita page as well. Nothing that you are doing in the separate chapter, appendix, etc. files represents a compilable file by itself (notice that there is no `\documentclass` command or `\begin{document}`...`\end{document}` environment in any of the chapter files that you created). The reason you should place the major parts of your work in separate files is because it will allow for much quicker editing when the time comes.

To make the mistake-finding process easier on yourself, you should compile as you go with the development of your chapters. In particular, at various stages in the creation of your first chapter, go ahead and try to compile it to see if any errors pop up. If so, it should be a lot easier to pinpoint where the issue is than if you wait until you have *all* your chapters done and try to compile them all together. Once your first chapter compiles, comment out the line with the `\MyChapter{1}` command on it by putting the percentage symbol `%` in front of the command, so that LaTeX knows not to compile it in the next part of the process. Now, repeat this process for your second chapter, third chapter, and so on. Finally, once you know that all of the individual chapters compile, remove the percentage symbols in front of any of the `\MyChapter` commands and compile the work as a whole.

Chapter 14

Pulling Up the Rear

People love answers, but only as long as they are the ones who came up with them.

Criss Jami

THIS chapter is dedicated to showing you how to set up the all-important back matter of your work, that is, the appendices (if any), bibliography, and vita (uh, mini-biography of you). I chose to discuss this material with you in its own chapter since I field more questions concerning the back matter than anything else. Using these experiences, I have included information in each of the corresponding sections here that will hopefully allow you to head off any problems before they begin.

14.1 Appendices

An appendix to a work is a place following the regular chapters where technical information can be provided that would have otherwise interrupted the desired flow of a chapter. Such information can include large data sets, computer programs, and lengthy calculations or proofs. If you work has appendices, you should organize them much as you did your chapters, namely in separate files, one for each appendix. However, unlike your chapters, there is no special "appendix" command to place before the body of your appendix. Instead, the first line in each of your appendices should be

```
\chapter[⟨ToC title⟩]{⟨title⟩}
```

where ⟨*title*⟩ is the title of the corresponding appendix and ⟨*ToC title*⟩ is an optional shortened version of the title for the table of contents, if you feel it is necessary (don't worry, `TNThesis` has been designed so that LaTeX will understand that this is an appendix and not a regular chapter of your work). You may also choose to have sections and subsections in your appendices, as well as propositions, theorems, etc. Really anything that you would have put into a regular chapter of your work, you can put into an appendix. One of the distinguishing marks of an appendix is that it is "lettered" instead of numbered and, as such, all sections and results in an appendix are prefixed by the corresponding letter.

When it comes time to load your appendices into the preamble of `TNTmain`, the macro to load each of your appendices is given by

```
\MyAppendix{⟨number⟩}{⟨filename⟩}
```

where *⟨number⟩* is the appendix number corresponding to its place amongst all of your appendices (if you have just one appendix, as most humans do, then set *⟨number⟩* $= 1$) and *⟨filename⟩* is the filename (you guessed it, *without* the `tex` extension) containing the associated appendix. Just as with chapters, the ordering of your appendices is dictated by *⟨number⟩* in the above macro, *not* the order in which you list the various `\MyAppendix` commands in the preamble of `TNTmain`.

If you do not have an appendix, simply do not use the `\MyAppendix` command rather than set *⟨number⟩*$= 0$ (which will give LaTeX a fit).

An example of the code that you would use to load appendices into `TNTmain` would be

```
\MyAppendix{1}{popgrowth}
\MyAppendix{2}{empdata}
```

where the first appendix is contained in the file `popgrowth.tex` and the second appendix is contained in the file `empdata.tex`.

14.2 The Bibliography

The bibliography, or references section, of your work is the place where you identify the source of anything that you cited in the body of your thesis or dissertation. Typically, you want to develop an intuitive labeling scheme for your sources so that you can easily tell LaTeX how to connect an in-text citation with the appropriate source listed in your bibliography section.

There are several major ways, in general, by which you can create your bibliography, each with its own pros and cons. However, for your thesis or dissertation, `TNThesis` uses the package `biblatex` with a backend of the Biber software (recall the discussion from Chapter 2). This means that you will have to do some work up front to create what is called a `bib` file. But once this is done, `biblatex` (along with Biber) will order your sources for you and place them in the required APA 7 style (see Appendix B). Pretty sweet, huh?

14.2.1 Making a `bib`

Well, it's pretty sweet until you realize that implementing the procedure for actually outputting such a bibliography in your work requires its own learning curve. But don't worry, I'll guide you through this step by step.

1. Begin by opening a new document in your LaTeX editor. However, instead of typing any of the typical introductory commands that we would use if this were going to be a `tex` file, we are going to use this document to simply list bibliographic information related to your sources (in the appropriate format, of course). Let me dissect an example (provided in Figure 14.1) so that we can explore the anatomy of a typical entry in this document.

```
1  @book{2009solar,
2  address = {New Delhi},
3  author = {Solanki, Chetan Singh},
4  booktitle = {Solar Photovoltaics: Fundamentals, Technologies and Applications},
5  chapter = {12},
6  edition = {1},
7  isbn = {9788120337602},
8  pages = {478},
9  publisher = {Prentice-Hall of India},
10 title = {{Solar Photovoltaics: Fundamentals, Technologies and Applications}},
11 url = {http://books.google.ca/books?id=hdvYA9KsI2YC},
12 year = {2009}
13 }
```

FIGURE 14.1: An example of an entry in a `bib` file

- `@book` All entries in a `bib` file begin with the `@` symbol followed by the type of source (see the quasi-complete list in the subsection *Oh, the Possibilities!* below for allowable names of sources).
- `{...}` Even though `@book` does not look like a typical LaTeX command, here you will want to think of it as such. This means that following `@`⟨*source category*⟩, you will place an open curly brace {(see line 1 above), then a comma-separated list of "arguments", and you will finish the entry with a closed curly brace }(see line 13 above). You may then proceed to the next line to begin your next bibliographic entry.
- `2009solar` The very first "argument" needs to be the label that you will use to connect a citation with the particular source. For the above source, a citation in the body of your work could be created through using this label in commands such as `\cite{2009solar}`, `\citep{2009solar}`, or `\citet{2009solar}` depending upon your preference for the appearance of the citation (see the subsection *Citations* below for more information on these and related commands.)
- `author=` After the label, bibliographic categories are introduced with the format ⟨*field name*⟩=, followed by the corresponding information for the

particular source being supplied (see the quasi-complete list in the subsection *Oh, the Possibilities!* below for allowable names of fields).

- `,` As shown on line 5, don't forget that the various fields (and the label at the beginning) must all be separated by commas.
- `{...}` As shown on line 8, the actual information related to a given field must be offset with curly braces.

What year is it?

! In the above process, you are allowed to supply more information than is necessary for the creation of a bibliographic item in APA style. The `biblatex` package will automatically select only what is appropriate when your bibliography is printed. On the flip side of the coin, if a particular piece of information is unknown for a given source, you generally need not supply anything for the respective field (in fact, you should just omit the field entirely). For example, if no date is given for a source, `biblatex` will simply output `n.d.` for "no date" at the appropriate place in the entry.

2. Once you have finished inputting all of your entries, save the file by typing ⟨*filename*⟩`bib` in the Save field (yes, the `bib` extension with the filename is important here). However, don't attempt to compile your `bib` file! Also, be certain that the `bib` file is in the same folder as `TNTmain.tex`.

3. Now, to get `TNThesis` to recognize your `bib` file, you will need to use the following macro in the preamble of `TNTmain.tex`:

```
\MyBibReferences{⟨filename⟩}
```

where ⟨*filename*⟩ is the name of your `bib` file *without* the `bib` extension.

4. I know, I know, you're chomping at the bit to get this thing compiled. But, patience, grasshopper. Remember how I said that `TNThesis` uses the software Biber to get your bibliography organized. Well, now is the time that you're going to use it! Assuming that you are using TeXworks for your editor, first compile `TNTmain` as you normally would by selecting pdfLaTeX from the typesetting options and pressing the little green right-pointing arrow ▶. On the first compilation, you should *not* see your bibliography just yet. At this point, LaTeX is simply setting up the file from which it will read the bibliographic information. Now, select Biber from the typesetting options and typeset your document using this program. Here, Biber is writing the appropriate information to the aforementioned file. Finally, switch back to pdfLaTeX in the typesetting options and compile *twice* (hopefully

the graphic on the following page helps). You should see a beautiful-looking bibliography and all of your citations appropriately numbered in the text of your work. Congratulations! (By the way, the reason you need to compile twice at the end is not just so that LaTeX can read the file that contains all of the bibliographic information, but to ensure that all cross-referencing is correct in your work.) This process is showcased in Figure 14.2.

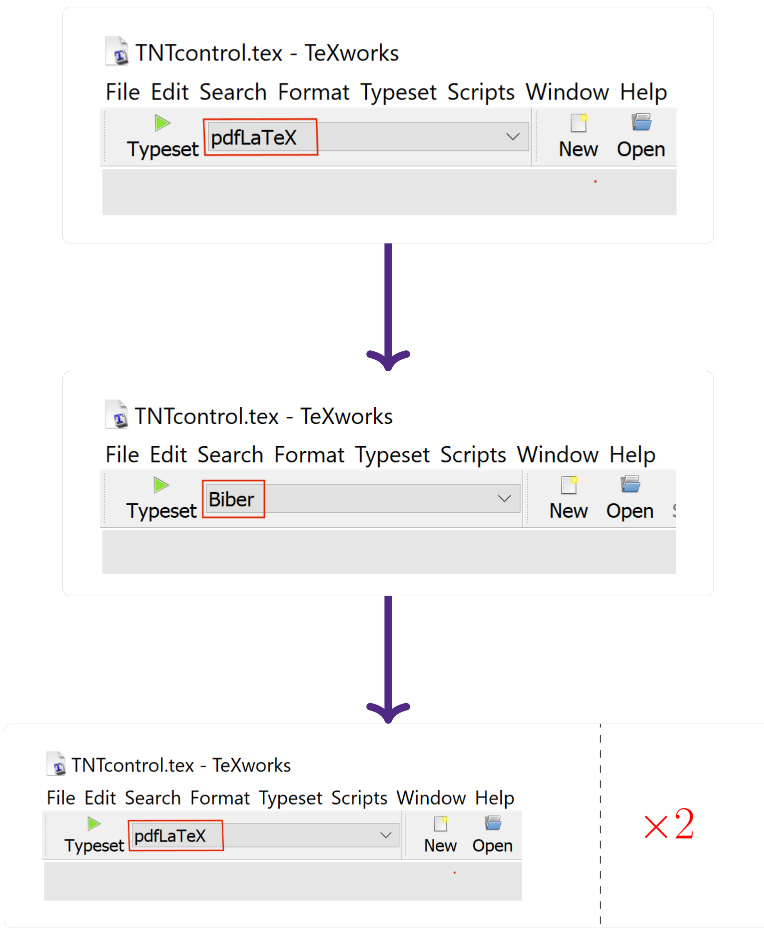


FIGURE 14.2: Graphic showing the procedure for compiling a `bib` file in TeXworks

To heck with BiBTEX!

If you are using an editor other than TEXworks, make sure that it is set up to process your bibliography using Biber, **not** BiBTEX. This may mean a trip into your editor's Preferences menu.

Let `biblatex` do its job!

! Particularly in engineering disciplines, I have found it common practice to copy and paste bibliographic information into one's `bib` file from an online database. However, you should *never* blindly do this! Far more often than not, such information is given in a manner inconsistent with the required APA style for Tennessee Tech theses and dissertations (e.g., journal titles come with certain words abbreviated). You should take care that you are providing as much information as possible for your `bib` file. This means no abbreviations whatsoever and a complete list of all authors related to a work. Trust me, `biblatex` will know how to properly handle the information that you give to it.

14.2.2 Oh, the Possibilities!

As promised, here then is a list of possibilities for sources in a `bib` file along with the associated required fields. Note that those "required" fields that are in parentheses are technically not required for proper compilation of the `bib` file, but *must* be provided, if available, by the APA 7 standard.

@article An article in a journal, magazine, newspaper, or other periodical which forms a self-contained unit with its own title. The title of the periodical is given in the `journaltitle` field. If the issue has its own title in addition to the main title of the periodical, it goes in the `issuetitle` field.
Required fields: `author, journaltitle, title, year/date`

@book A single-volume book with one or more authors share credit for the work as a whole.
Required fields: `author, title, year/date`

@mvbook A multi-volume book.
Required fields: `author, title, year/date`

@inbook A part of a book which forms a self-contained unit with its own title.
Required fields: `author, booktitle, title, year/date`

@suppbook Supplemental material in a book.
Required fields: `author, booktitle, title, year/date`

@booklet A book-like work without a formal publisher or sponsoring institution. Use the field `howpublished` to supply publishing information in free format, if applicable.
Required fields: `author/editor, title, year/date`

@collection A single-volume collection with multiple, self-contained contributions by distinct authors which have their own title.
Required fields: `editor, title, year/date`

@mvcollection A multi-volume collection.
Required fields: `editor, title, year/date`

@incollection A contribution to a collection which forms a self-contained unit with a distinct author and title. The `author` refers to the `title`, and the `editor` to the `booktitle`, i.e., the title of the collection.
Required fields: `author, booktitle, title, year/date`

@suppcollection Supplemental material in a collection.
Required fields: `author, booktitle, title, year/date`

@conference Equivalent to **@inproceedings**.

@dataset A data set or a similar collection of raw data.
Required fields: `author/editor, title, year/date`

@electronic Equivalent to **@online**.

@letter For personal correspondence such as e-mails, letters, memoranda, etc.

@manual Technical or other documentation, not necessarily in printed form.
Required fields: `(author/editor), title, year/date`

@mastersthesis Similar to **@thesis** except that the `type` field is optional and defaults to the term "Master's thesis".

@misc A fallback type for entries that do not fit into any other category. Use the field `howpublished` to supply publishing information in free format, if applicable. The field `type` may be helpful here, too.
Required fields: `(author/editor), title, (year)/date`

@online A purely online resource. This entry is intended for sources such as web sites which are intrinsically online resources.
Required fields: `(author/editor), date, doi/eprint/url, title, (year)`

@patent A patent or patent request. The number or record token is given in the `number` field. Use `type` to specify the type and `location` to indicate the scope of the patent, if different than the scope implied by `type`.
Required fields: `author, number, title, year/date`

@phdthesis Similar to **@thesis** except that the `type` field is optional and defaults to the term "PhD thesis".

@periodical A complete issue of a periodical, such as a special issue of a journal. The title of the periodical is given in the `title` field. If the issue has its own title in addition to the title of the periodical, it goes in the `issuetitle` field.
Required fields: `(editor), title, year/date`

@suppperiodical Supplemental material in a periodical.

@proceedings A single-volume conference proceedings.
Required fields: `(editor), title, year/date`

@mvproceedings A multi-volume conference proceedings.
Required fields: `title, year/date`

@inproceedings An article in a conference proceedings.
Required fields: `author, booktitle, title, year/date`

@reference A single-volume work of reference such as an encyclopedia or dictionary.
Required fields: `editor, title, year/date`

@mvreference A multi-volume work of reference such as an encyclopedia or dictionary.
Required fields: `editor, title, year/date`

@inreference An article in a work of reference.
Required fields: `author, booktitle, title, year/date`

@report A technical report, research report, or white paper published by a university or some other institution. Use `type` to specify the type of report. The sponsoring institution goes in the `institution` field.
Required fields: `author, institution, title, type, year/date`

@software Computer software.

@standard For national and international standards issued by a standards body such as the International Organization for Standardization.

@techreport Similar to **@report** except that the `type` field is optional and defaults to the term "technical report".

@thesis A thesis written for an educational institution to satisfy the requirements for a degree. Use `type` to specify the type of thesis.
Required fields: `author, institution, title, type, year/date`

@unpublished A work with an author and title that has not been formally published, such as a manuscript or the script of a talk. Use the fields `howpublished` and `note` to supply additional information in free format, if applicable.
Required fields: `author, title, year/date`

@www Equivalent to **@online**.

And here is a list of possible bibliographic fields:

`abstract`, `addendum`, `afterword`, `annotation`, `annotator`, `author`, `authortype`, `bookauthor`, `bookpagination`, `booksubtitle`, `booktitle`, `booktitleaddon`, `chapter`, `commentator`, `date`, `doi`, `edition`, `editor`, `editora`, `editorb`, `editorc`, `editortype`, `editoratype`, `editorbtype`, `editorctype`, `eid`, `entrysubtype`, `eprint`, `eprintclass`, `eprinttype`, `eventdate`, `eventtitle`, `eventtitleaddon`, `file`, `foreward`, `holder`, `howpublished`, `indextitle`, `institution`, `introduction`, `isan`, `isbn`, `ismn`, `isrn`, `issn`, `issue`, `issuesubtitle`, `issuetitle`, `iswc`, `journalsubtitle`, `journaltitle`, `label`, `language`, `library`, `location`, `mainsubtitle`, `maintitle`, `maintitleaddon`, `month`, `nameaddon`, `note`, `number`, `organization`, `origdate`, `origlanguage`, `origlocation`, `origpublisher`, `origtitle`, `pages`, `pagetotal`, `pagination`, `part`, `publisher`, `pubstate`, `reprinttitle`, `series`, `shortauthor`, `shorteditor`, `shorthand`, `shorthandintro`, `shortjournal`, `shortseries`, `shorttitle`, `subtitle`, `title`, `titleaddon`, `translator`, `type`, `url`, `urldate`, `venue`, `version`, `volume`, `volumes`, `year`

To make sure that `biblatex` does not misinterpret the data that you have given it:

- Anything that should be typeset exactly as it is inputted into the `bib` file should be surrounded by an extra set of curly braces. For example,

  ```
  author={{U.S. Department of Energy}}
  ```

- If a publisher has the word “and” literally in its name, wrap the word “and” in its own set of curly braces to make sure `biblatex` does not interpret this as multiple publishers. The same holds for organizations and institutions.

- Multiple authors must be separated by the word `and` with each author’s name given either in the form ⟨*first name*⟩ ⟨*last name*⟩ or ⟨*last name*⟩, ⟨*first name*⟩.

As a concluding remark for this section on making proper `bib` file entries, let me point out a particular work-around regarding the entry of source information for online resources that use the **@online** (or an equivalent) `bib` handle. APA 7 style mandates that any online source without an explicit author (e.g., an article from *Wikipedia*) use the format

⟨*Title*⟩. (⟨*date published*⟩). ⟨*Organization name*⟩. `Retrieved` ⟨*retrieval date*⟩ `from` ⟨*full url*⟩

To achieve this, you will need to use the fields `title`, `date`, `organization`, `urldate`, and `url` with the **@online** handle. Simply typing `author=`⟨*Organization name*⟩ will not work here.

Making a mad dash

In English composition, there are three types of dashes that get used: the hyphen (-), the en-dash (–), and the em-dash (—). Which you use depends on the situation. The hyphen is used for the purpose of creating a compound word from two separate words, such as "non-degenerate", and to separate the syllables of a word at the end of a line.. The hyphen can be typed in LaTeX by using a single instance of the hyphen key, as in `non-degenerate`. The en-dash is most often used for expressing a range of values, including page numbers. To typeset something like "pp. 123–141" in LaTeX, use a double instance of the hyphen key, as in `pp. 123--141`. The em-dash is most often used for separating off a part of a sentence designed to express a supplementary idea. For example, "My neighbor's dog—whose name I can't remember—just peed in my yard." This can be created in LaTeX by using (you guessed it) a triple instance of the hyphen key, as in `My neighbor's dog---whose name I can't remember---just peed in my yard.`

Talking with a foreign accent

Often you find references with author names or titles that involve special foreign language symbols. As always, LaTeX's got your back with regard to typesetting these symbols. Below is a table showcasing the breadth of what LaTeX is capable of along these lines. Bear in mind that just about any letter may be used as an argument for many of the macros in this table.

Table 14.2: LaTeX commands for foreign language symbols

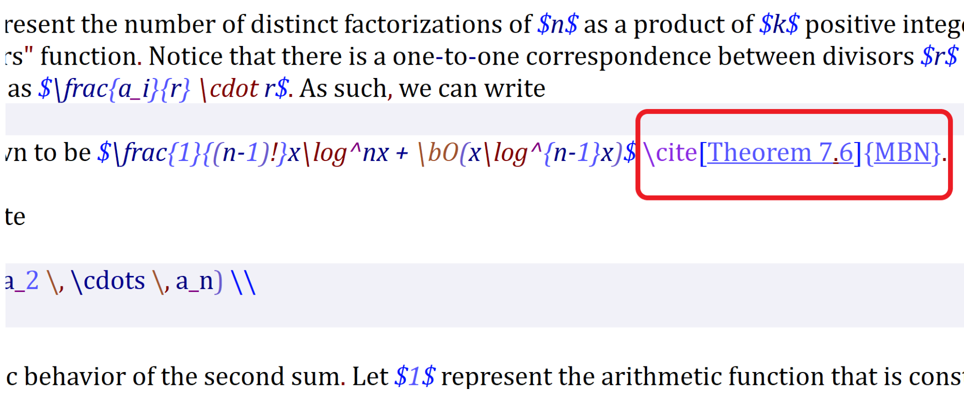
Syntax	Output	Syntax	Output	Syntax	Output
`\'{o}`	ó	`` \`{o} ``	ò	`\^{o}`	ô
`\~{o}`	õ	`\={o}`	ō	`\.{o}`	ȯ
`\"{o}`	ö	`\c{c}`	ç	`\u{o}`	ŏ
`\v{o}`	ǒ	`\H{o}`	ő	`\d{o}`	ọ
`\b{o}`	o̲	`\t{oo}`	o͡o	`\oe`	œ
`\OE`	Œ	`\ae`	æ	`\AE`	Æ
`\aa`	å	`\AA`	Å	`\o`	ø
`\O`	Ø	`\l`	ł	`\L`	Ł
`\i`	ı	`\j`	ȷ	`` !` ``	¡
`` ?` ``	¿	`\ss{}`	ß	`\k{e}`	ę
`\dh`	ð	`\DH`	Ð	`\dj`	đ
`\DJ`	Đ	`\ng`	ŋ	`\NG`	Ŋ
`\th`	þ	`\TH`	Þ		

14.2.3 Citations

`TNThesis` defaults to square brackets and numbers when creating citations, as in [2]. To create citations in the text, one need only use the command:

```
\cite[⟨text⟩]{⟨label⟩}
```

where *⟨label⟩* is the corresponding label to the appropriate entry in your `bib` file and *⟨text⟩* is optional text to be included with the particular citation (e.g., "Theorem 1.4" in the citation [2, Theorem 1.4]). Figure 14.3 shows what this looks like in your source code.


resent the number of distinct factorizations of n as a product of k positive intege
rs" function. Notice that there is a one-to-one correspondence between divisors r
as $\frac{a_i}{r} \cdot r$. As such, we can write

wn to be $\frac{1}{(n-1)!}x\log^nx + \bO(x\log^{n-1}x)$\cite[Theorem 7.6]{MBN}.

te

a_2 \, \cdots \, a_n) \\

c behavior of the second sum. Let 1 represent the arithmetic function that is const

FIGURE 14.3: An example of the `\cite` command in the source code

However, `biblatex` affords a number of additional options, depending upon the effect that you want to create. Here are a couple of variations on the `\cite` command idea:

```
\citep[⟨prefix⟩][⟨postfix⟩]{⟨label⟩}
\citet[⟨prefix⟩][⟨postfix⟩]{⟨label⟩}
```

where `\citep` institutes a *parenthetical* citation and `\citet` institutes a(n abbreviated) *textual* citation. The optional arguments *⟨prefix⟩* and *⟨postfix⟩* dictate what comes in front and behind, respectively, the number. However, if only one of these optional arguments is provided, it is assumed to be the *⟨postfix⟩*. As such, if you want a *⟨prefix⟩* without a *⟨postfix⟩*, you must use a set of square brackets [ ] with nothing between them as a place holder for the *⟨postfix⟩* (see the third example below). There is also a "starred" version of `\citet` that provides the complete list of authors associated to the citation.

Perhaps a few examples may help here:

TABLE 14.3: Examples of the use of `\citep` and `\citet`

`\citep{jon90}`	→	[3]
`\citep[chap.~ 2]{jon90}`	→	[3, chap. 2]
`\citep[see][]{jon90}`	→	[see 3]
`\citep[see][chap.~ 2]{jon90}`	→	[see 3, chap. 2]
`\citet{jon90}`	→	Jones et al. [3]
`\citet[chap.~ 2]{jon90}`	→	Jones et al. [3, chap. 2]
`\citet*{jon90}`	→	Jones, Baker, and Williams [3]

If you have a list of citations to provide at a particular place in your text, you can simply separate the various labels by commas, as in `\citep{jon90, jon92, hans98}`.

However, this really is the tip of the iceberg. Table 14.4 gives a basic summary of the different types of citation commands available to you, all of which follow the same basic command structure as `\citep` and `\citet` above.

TABLE 14.4: Variations on the `\cite` command

Command	Description
`\citeauthor`	author only (text citation)
`\citeyear`	year only (no parentheses)
`\citeyearpar`	year citation (with parentheses)
`\citealp`	parenthetical citation without the parentheses
`\citealt`	text citation without the parentheses
`\citenum`	number of the reference
`\nocite`	no citation, only reference list entry
`\Citep` or `\Citet`	Capitalized citation
`\citepalias`	parenthetical citation of alias
`\citetalias`	text citation of alias
`\citetext`	arbitrary text within citation parentheses

The value of issuing a citation

! In English composition, the only sources that may appear in the bibliography are sources that have been explicitly cited at least once in the body of the work and `TNThesis` is designed to respect this convention. As such, if you are not seeing a particular source appear in the references section, odds are pretty good that you have not yet cited the source.

14.3 The Vita Page

The very last page of your work is called the vita page and represents a *short, one-paragraph biography* of you. Strangely enough though, even though you will obviously be writing your own vita, by rule, the vita is written in third person. Often, people will put such facts in a vita as where they grew up, where they went to school, interesting details about their families, interesting hobbies, and what they plan to do career-wise after receiving their degrees. Write your vita in a separate file and use the command

```
\MyVita{⟨filename⟩}
```

where *⟨filename⟩* is the filename (shocker, *without* the `tex` extension) of the file containing your vita. An example of a vita is given in Figure 14.4.

VITA

Samuel Dent was born in New Orleans, Louisiana, on December 30, 1993. He attended Clinton Public Schools in Mississippi and graduated from Clinton High School as the valedictorian in 2012. Over the next four years, he attended the University of Southern Mississippi and in May of 2016, received a Bachelor of Science in Mathematics. He entered Tennessee Technological University in August of 2016 and received a Master of Science degree in Mathematics in May of 2018.

FIGURE 14.4: An example of a vita

At this point, all that's left is to compile `TNTmain`, cross your fingers, and hope everything compiles as it should. If not, see Chapter 15 on how to deal with certain errors in compiling that LaTeX gives you. Rest assured, it's bound to happen. God-speed!

Chapter 15

The Terror of Errors

> If you don't know, the thing to do is not to get scared, but to learn.
>
> Ayn Rand

OKAY, so I borrowed the title of this chapter from Peter Wilson, who wrote the manual to the `memoir` class file. But it really is apt. LaTeX can be downright unforgiving when it comes to what it expects from your code when you compile your work. Many different issues will force LaTeX to stop the compilation process in its tracks. And often times the reason LaTeX gives you for why there is an issue can be rather cryptic. (One time, LaTeX told me to figure out the problem myself—honestly!) So, based upon my experience, I decided to write this final chapter of the `TNThesis` manual describing some of the most common error descriptions that LaTeX gives and how to best deal with them.

15.1 Error Messages

LaTeX's error messages always begin with an exclamation mark followed by a message. The second line always starts with an `l` followed by a number to indicate the line upon which LaTeX first *realized* the error. Notice I didn't say the line where the error actually was. As I indicated in the first chapter of this manual, if there is an error in a lengthy set of nested commands, LaTeX will often indicate that the error is on line *xx*, where *xx* is the *last* line of the entire command structure. Then it's up to you to track down where in the command structure things went wrong. After the supposed line number where the mistake was made, LaTeX gives the text of the line itself up to the point where the error was detected followed by the rest of the erroneous line. The last line of the error message is always a question mark as LaTeX awaits a response from you on what you would like to do about the error.

I'll address this last bit at the end of the chapter, but for now let me go over some of the most common error messages that I have personally come across along with a little commentary regarding what each one means.

- `Extra }, or forgotten \endgroup, $, or \right`

  One of most infuriating error messages to get, particularly if you have a large nested command structure. Obviously, in general there should be a one-to-one correspondence between left curly braces and right curly braces. If not, you may get this error message. Unfortunately, the only real solution here is to start counting curly braces and see where you must have miscounted the first time. Hopefully, your LaTeX editor will help you out by not only highlighting the curly brace that your cursor is currently on, but the curly brace with which LaTeX thinks it corresponds.

- `File ended while scanning...`

  Sometimes when you forget a closing delimiter to a command, LaTeX will read well past where you intended it to stop (looking of course for a right curly brace or a right square bracket) and suddenly realize that the extra material is causing an issue with the definition of the command. Like the above error message, you will need to track down where you forgot an ending curly brace or square bracket.

- `Illegal unit of measure (pt inserted).`

  Usually this is just a simple mistake of misspelling a unit of length or forgetting to put the units with a particular length. Go back and make sure, for example, that you wrote "in" for inches.

- `Missing $ inserted.` or `Missing { inserted.` or `Missing } inserted.`

  Some macros can only be executed in math mode—that is, the command must be somewhere between a set of dollar signs `$...$` (or an equivalent) or between a set of display mode brackets `\[...\]` (or an equivalent). LaTeX is trying to remedy the problem by assuming that you wanted math mode where the macro is located. Most of the time, you don't. As such, it's best to go back and fix the problem by finding a different command to use in the problem spot.

- `Missing number, treated as zero.`

  This sometimes occurs when a left square bracket in the text was mistaken by LaTeX as the start of an optional argument for a particular command. For example, unbeknownst to even many die-hard TeXers, the new line command \\ can take an optional length argument (so the description of the macro really should be `\\[⟨length⟩]`). This means that if you wrote something like `\\ [Theorem 1.2]` intending for the expression [Theorem 1.2] to be on the next line of text, you will get this error message. To remedy the problem in this case, you will need to replace `\\` with `\newline`. Similar fixes hold for some other occasions in which this error message is produced.

  The other time that I've seen this error message pop up is when I've forgotten a mandatory argument involving a number or dimension to a particular macro. So, if you keep getting this error message, you may need to go back and check the documentation for the required arguments to the macros that you are using. Odds are pretty good that you've overlooked such an argument.

- `Overfull \hbox (... pt too wide).`

  This is simply a warning that LaTeX couldn't cram some piece of text into the allotted horizontal space (which TEX places in a thing called an hbox). It usually isn't a big concern and usually doesn't cause LATEX to stop compiling. Such instances are counted up and included in something called a "PDFTeXify Compilation Report" at the end of compiling your document.

- `Paragraph ended before...was complete.`

  Either a blank line or the macro `\par` (the LATEX command for "make a new paragraph") appeared in the argument to a macro that cannot handle the creation of a new paragraph, such as `\newcommand*`. You will need to then remove the offending blank line or `\par` command.

- `Runaway argument.`

  Just like it sounds. LATEX sees too much going into the argument of some command. Usually, this is caused by a forgotten right curly brace.

- `Too many }'s.`

  This is the same issue as the first error discussed above concerning an extra `}`.

- `Underfull \hbox (badness...).`

  This is LATEX's way of warning you that you might have some unwanted extra horizontal space, possibly caused by using two `\newline` or `\\` commands in succession with nothing between them. However, most of the time, this is in fact what you intended and it is not an error that causes LATEX to stop compiling.

- `\begin{...} ended by \end{...}.`

  The name of the environment associated with the `\begin` argument is not the same as that of the given `\end` argument. You may have either mistyped the name or forgot to even put an `\end` to your environment.

- `Command \... already defined or name \end... illegal.`

  A macro that you created already exists. Choose a different name for your macro.

- `File not found. Type X to quit or <RETURN> to proceed or enter new name (Default extension: ...).`

  This error is pretty self-explanatory. TEX can't find one of the files that it needs to call from `TNTmain` (such as a chapter or bibliography). Make sure that (1) your file is in the same folder as both `TNThesis` and `TNTmain`, and (2) you did *not* put any extension with any of your file names.

- `Missing \begin{document}.`

  Generally, this happens when there is something wrong with your preamble and TEX is trying to typeset something before it should. Check to see if you are missing the required backslash from the start of a command, misplaced braces around an argument, or something like this.

- `There's no line to end here.`

  This happens if you placed `\newline` or `\\` between paragraphs (that is, in "vertical mode"). Or you might have put `\\` immediately after the `\item` command in a list environment.

In the former case, if you need extra space, use the `\vspace*{⟨length⟩}` command. In the latter case, use `\item \mbox{} \\` instead if you want to drop a line immediately after the label corresponding with `\item`.

- `Citation...on page...undefined.`

  Self-explanatory. Make an entry in your `bib` file that corresponds with your `\cite` command.

- `Command...invalid in math mode.`

  You can't use certain commands in math mode. That is, you can't use certain commands in between two dollar signs `$...$` (or an equivalent) or in between display brackets `\[...\]` (or an equivalent). Use a corresponding command that is designed for math mode (there usually is one).

- `Reference...on page...undefined.`

  Self-explanatory. Make a corresponding result for your `\ref` command.

It's not your fault

Believe it or not, (on rare occasion) TeX can post an error that has absolutely *nothing* to do with a mistake you made in coding! One of the tell-tale signs of this type of error is when TeX throws up the message

```
File ended while scanning use of \@writefile. <inserted text> \par
```

You see, the compilation process involves TeX creating what is called an auxiliary file, which can be identified by the extension `aux` in the file that contains `TNTmain`. TeX then reads from this auxiliary file on a second compiling run. The process is partly designed to make compiling a bit faster, that is, it is designed so you are not compiling from scratch every time. However, the auxiliary file is also what can stop TeX in its tracks! Some code makes a semi-permanent mark in the auxiliary file which can create problems if certain changes are made between runs. The (only) solution then is to delete the corresponding auxiliary file and compile your document "fresh". Frankly, when I encounter this type of problem, I go ahead and delete everything except the original `tex` files and compile again.

15.2 Other Options for Dealing With Errors

When encountering an error, other than hitting , you have some other choices at the question mark. Here is a full list of what your options are at this point:

- *⟨Enter⟩*: LaTeX will try to work around the error and continue processing the document until it hits another error. (In my experience, though, all you will get by hitting ⟨*Enter*⟩ is another question mark, and then another, and another...)

- `H (help)`: The help message is given and then LaTeX waits for you to respond. In my experience, the help message is no more helpful than the original error message itself.
- `S (scroll)`: Continue processing, outputting any further error messages, but not stopping.
- `Q (quiet)`: Continue processing without stopping and with no further messages.
- `R (run)`: Like the Q(quiet) option above but not stopping even if your document requires some user input.
- `I (insert)`: This let's you insert some material for LaTeX to read but no changes are actually made to the source file.
- `E (edit)`: This should return you to your LaTeX editor so you can change the file.
- `X (exit)`: Stops this LaTeX run. No output file is generated.

Alternatively, before compiling, you could type one of the following commands into the preamble of your document:

- `\errorstopmode`: TeX will ask for user input on the occurrence of an error.
- `\scrollmode`: TeX fixes errors itself, but will ask the user for any missing files.
- `\nonstopmode`: TeX fixes errors itself and performs an emergency stop on serious errors such as missing input files.
- `\batchmode`: TeX fixes errors itself and performs an emergency stop on serious errors such as missing input files, but no terminal output is generated.

There are also a variety of plain TeX `\tracing`⟨*object*⟩ commands, where ⟨*object*⟩ is one of the following: **commands**, **macros**, **output**, **paragraphs**, **pages**, **lostchars**, **restores**, or **stats**. Placing one (or more) of these commands in your preamble, setting its value to be 1 (e.g., `\tracingcommands=1`), and then compiling your document will print *very* detailed information to the `log` file concerning the nitty-gritty details of how TeX is processing the corresponding object(s). Remember from the first chapter that the `log` file may be accessed simply by opening it from your LaTeX editor.

However, before you get so mad at compilation errors that you resort to one of the more heavy-handed options above, I would encourage you to just take a break and start again later. I understand how tempting it might be to hit S, Q, or R or use any of `\scrollmode`, `\nonstopmode`, or `\batchmode` in your preamble. And if you are utilizing these features to find a particular error, that's fine. The problem, especially with the latter three, is that eventually you very well may forget you put the command in your preamble. Moreover, some errors are not painfully obvious upon examining the corresponding `pdf` file. As such, you may be seduced into thinking everything in the output is fine when in fact it is not.

It may also be tempting to think that the various `\tracing` commands will clearly reveal the problem in the `log` file. However, you need to keep in mind that the `\tracing` commands will force potentially *millions* of lines of output to be produced in the `log` file, most of which is completely unintelligible except to experts in TeX.

Perhaps my best recommendation if you are struggling to find out why TeX keeps returning a particular error is simply to strategically comment out certain lines of code (using %; see page 24) around the problem area. Then recompile and see what happens. If you still find an error produced, comment out some other lines and recompile. Repeat this process until it (hopefully) becomes clear what the issue is. Of course, you can also do an internet search for the type of error you are encountering, but sometimes the answers you find do not directly address your situation.

Appendix A

Which Editor Is Right for Me?

> The highest form of ignorance is to reject something you know nothing about.
>
> — Wayne W. Dyer

You know, I get asked all the time, "Andrew, amongst the LaTeX editors out there, which one do you personally recommend?" (Actually, I have never been asked this question, but a guy can dream, right?) It really depends on which features you value the most. Certain editors are particularly well-suited for those just starting out with LaTeX, while others are better for those who are have a lot of experience with it. Some editors are great if you have multiple LaTeX documents going at once, while others are best at letting you find your mistakes. There is no one editor that has it all.

To help you out, I have put the five most popular editors to the test against 10 measures of an editor's worth. I then ranked them in each category as Great, Okay, or Poor. You can then assess what you find most important (which can change with increased experience) and decide which editor you want to use. All except WinEdt are free, and even WinEdt is just $20 for a student license.

	LyX	TeXnicCenter	TeXstudio	TeXworks	WinEdt
auto-suggestions	Okay	Great	Okay	Poor	Great
auxiliary file cleaning	Poor	Okay	Great	Poor	Great
customizability	Great	Poor	Great	Poor	Okay
grammar/spelling error detection	Okay	Poor	Okay	Poor	Great
help menu	Okay	Great	Poor	Poor	Okay
intuitiveness	Okay	Poor	Okay	Great	Okay
LaTeX error detection	Okay	Poor	Great	Okay	Okay
multi-program view	Poor	Great	Poor	Okay	Great
pdf document changes	Okay	Okay	Great	Great	Poor
tool options	Poor	Okay	Great	Okay	Great

Explanation of Categories

auto-suggestions Most editors will suggest certain completion options as you attempt to type out a LaTeX command, which can be quite helpful if you know how to start a particular command but are not quite sure of the exact spelling. After all, there are macros in packages out there whose syntax has a twist on the intuitive spelling of the command (purposefully designed to make sure that it will not conflict with other similar commands).

auxiliary file cleaning Not too infrequently, a compilation error occurs not because of any mistake that you made *per se*, but because LaTeX is trying to tap into the `aux` file from a previous compilation run and the `aux` file has information in it that is conflicting with edits you have since made to the source code. The only way to really deal with this problem is to delete the `aux` file and start with a fresh compilation. Some editors allow you to "clean" such files through a simple menu option while others force you to manually delete each one.

customizability Most editors allow you choose color schemes and features that you want to emphasize in your code, but some make this process easier than others.

grammar/spelling error detection Some editors have continuous alerts about grammar and spelling mistakes you have made, while others require you to do a separate check on these things. Even amongst those editors that do continuous alerts, some do a better job of warning you about potential mistakes than others.

help menu Every editor has its own quirks, and invariably you will need to reference the editor's HELP menu to find an answer to your question. But the real helpfulness of HELP menus varies wildly.

intuitiveness How easy is it for a beginner to the system to figure out what he or she needs to do in order to accomplish certain basic tasks?

LaTeX error detection Errors in LaTeX are sometimes like cockroaches, they keep coming back no matter what you seem to do. And often the error messages themselves are as cryptic as those of a first date. But some editors do a far better job than others in allowing you to efficiently track down and determine your mistakes.

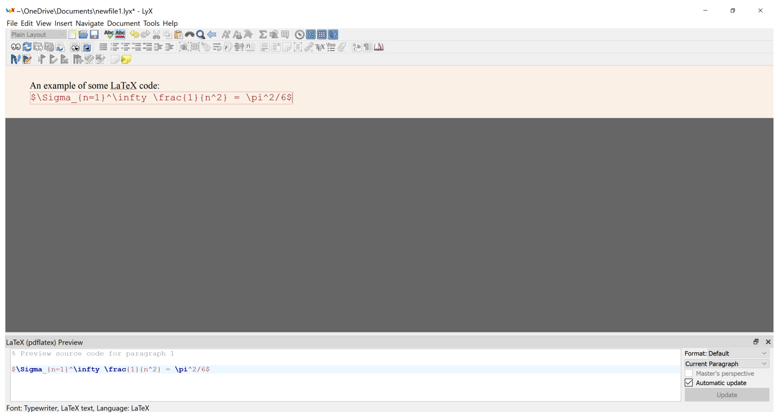
multi-file view If you are creating a work requiring multiple `tex` files (say a thesis or dissertation), you probably will want several of these files open at the same time for the sake of comparing what you wrote in one with what you wrote in the others. However, the various editors have different options as far as this goes.

pdf document changes When you edit a document, how easy is it to immediately see the changes you made in the corresponding `pdf` file? You may be surprised by how big the difference is in this area amongst the five editors.

tool options Some editors have a beautiful, icon-based approach to important tools that you need in your work, while others seem like their creators never really grew past MS-DOS. Some editors have a plethora of options in their TOOLS menu, while others do not.

A.1 LyX

Those familiar with LaTeX editors know that LyX is a bit of a different animal than the other four. In fact, reading through LyX's Introduction, I'm struck by how similar in tone it is to Overleaf in that it effectively bills itself as a LaTeX-based system designed for those who don't want to learn a lick of actual LaTeX. Think of it more as a word processing software with a vastly superior equation editor to Microsoft Word.



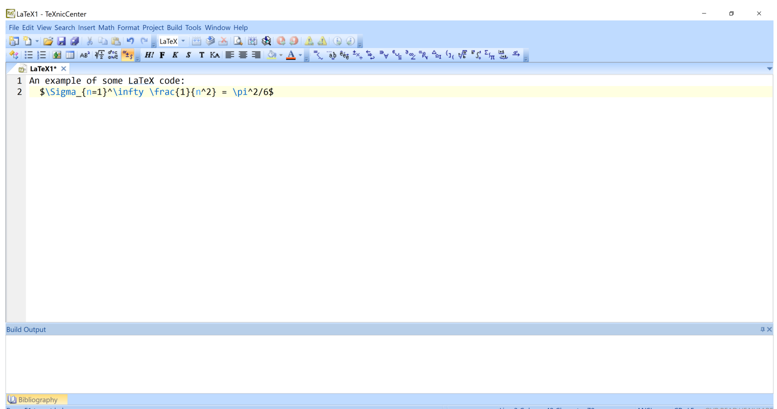
I was initially dumb-struck by LyX's graphical interface. Even with my laptop having one of the best screen resolution's on the market, I had to enlarge the icons to "giant-sized" to see on what I was clicking. Then there was the coding environment: one small beige strip at the top of the page juxtaposed by a massive amount of gray underneath. I quickly discovered that I could only type within the small beige strip with no way to move to another line unless I already had something on that line. Combined with the lack of an ability to tab over, it quickly became apparent to me that LyX was definitely not made for serious programmers. Moreover, it seems that the learning curve here is for those who are already comfortable with LaTeX! There is no typing of `\documentclass` at the beginning or any of the other standard protocols you follow with LaTeX code. If you want a certain documentclass, you choose from a drop-down menu under DOCUMENT. In fact, it appears as though LyX is designed *exclusively* for those who would rather click on something or select an option from a menu than type a command. Which is fine...as long as there is an option for what you want.

Perhaps the biggest disappointment came when I tried to typeset some actual LaTeX code. It turns out that it's not enough to offset a math environment with dollar signs and it's not enough to simply click on the TeX icon and type your code into the generated box, you need to do *both*. It's as if you need to tell the software what you intentions are and then reassert your intentions. I can easily see this becoming too much of a pain for anyone who has a lot of mathematics to typeset.

Summary: LyX is highly customizable and boasts tons of templates from which to choose. It may be suitable for someone who has just a handful of formulas in their document, and they want to render these formulas in a more professional way than what Microsoft Word can do. However, if this is not you, then you either should use Word or go with another editor.

A.2 TEXnicCenter

TEXnicCenter is one of those editors that has the potential to be truly great, if only there was a commitment on the part of the designers to modernize it! The initial wizard that popped up when I first opened TEXnicCenter claimed that if I am using MikTEX for my distribution (and I am), then most of what was needed by the wizard could be automatically obtained by the software...except the very first thing it needed (the file path for the MikTEX executables), it could not find. Funny, I never had an issue like this with any other editor. So, I extracted the file path from my MikTEX console and pasted it into the appropriate box in the wizard...only to find that it was the first in a long list of items the wizard could not find, thus relying on me to obtain the required information for it (doesn't this defeat the whole purpose behind a wizard). Which would not be that big of a deal for an individual well-versed with how LATEX works behind the scenes, but would be an absolute nightmare for a beginner.



Little did I realize that the issue with the so-called "wizard" was just the start of a host of technical issues that TEXnicCenter would have. The first several times I went into the Options menu to change a few things, the entire system crashed on me (hmm, maybe there's a message here). When I finally got past this point, an explanation for the crashing seemed to present itself when my eyes beheld icons that appeared to be reminiscent of Reagan-era Windows software. In fact, the most advanced version of Windows available under the Application Look menu was Windows 7 (how long has Windows 10 been out now?). All of this would not have bothered me so much, except for the fact that the core principle of LATEX is that aesthetics matter and the graphical interface for TEXnicCenter was screaming something quite different at me. Although, on a positive note, I would say that many of the options I had from the drop-down menus really made the coding environment look nice!

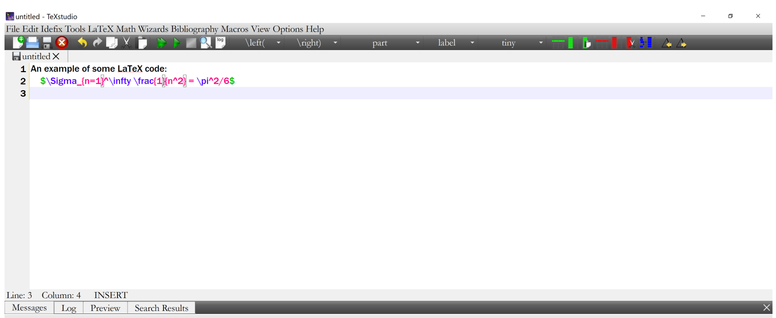
Perhaps the most odd (uh, concerning) thing I found about TEXnicCenter was the way it would automatically pass over minor (i.e., non-fatal) errors in the code in its zeal to produce some sort of output. Apparently, if it doesn't understand a command you typed (say you misspelled it), it will simply print nothing in its place. This can be an enormous problem if you have a pretty involved document for which your eye may not immediately catch the omission. Sure, the error is recorded in the `log` file and an indication that an error has occurred is in full view in the Build Output frame at the end of compilation, but I guess TEXnicCenter figures that if you really care about errors, you can track it down yourself.

Summary: If you can get it to work right out of the box, knock yourself out. But the pitiful LATEX error detection protocols are a deal-breaker for me. *Caveat emptor.*

A.3 TEXstudio

While certainly not perfect, TEXstudio boasts some impressive features that make it one of my absolute favorite editors to use. One of its greatest virtues is how customizable it is. The user can control practically every aspect of the coding experience, from the color of the commands to which packages it uses to make auto-suggestions. Furthermore, the customization dialog box is quite intuitive, so that even a novice user will enjoy selecting the features that he or she wants. This is most definitely an editor created by LATEX users for LATEX users.

However, my absolute favorite aspect of TEXstudio is the embedded pdf viewer that not allows the output to be in full view while you continue to edit your code, but upon recompiling, goes immediately to the first place in the output where a change was effected. And not only that, but the editor also allows you to highlight a small section of your code and compile just that portion on the spot! This is particularly helpful when you have a massive document that you don't want to fully compile just to see what a tiny part will look like. I know, I know, where do I sign, right?



But wait! there's more. TEXstudio has an auto-complete feature that automatically places a companion closing delimiter when you type an opening delimiter. So, for example, as soon as you type `{`, a companion `}` appears next to it. This tremendously reduces the chance of forgetting a closing delimiter and creating an error upon compiling. Speaking of errors, TEXstudio has perhaps the most comprehensive error detection scheme of any editor. TEXstudio color codes all of the errors (even the benign `underfull \hbox` and `overfull \hbox` errors) right in the code itself, making tracking down your mistakes as easy as it can possibly be.

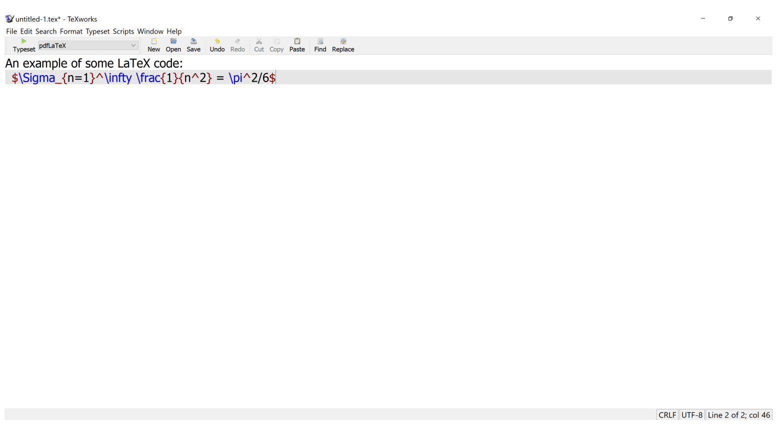
But for all of its positive qualities, TEXstudio has some significant shortcomings. Unlike WinEdt and TEXnicCenter, TEXstudio has no multi-file view, so if you have multiple files connected to your document, the best you can do here is to have tabs running at the top of your coding environment.

And perhaps the most annoying feature of all is the highly intrusive dialog boxes that provide in-depth definitions of certain LaTeX commands just by the user hovering the arrow over the command. I imagine that if I were new to LaTeX, this would be of some (marginal) benefit, but for those of us who are quite experienced, we don't need to be constantly reminded of what the command \\ represents. If the designers of TeXstudio would just allow me to turn this feature off, I would never use another editor again.

Summary: In spite of its noticeable faults, I am still a huge fan of TeXstudio. I imagine that once a user gets enough experience with LaTeX, he or she will find TeXstudio's features indispensable.

A.4 TeXworks

TeXworks is a marvelously simple LaTeX editor for those who are just starting out in LaTeX or who just don't want to fool around with a lot of bells and whistles. As such, if you have never really created a LaTeX document, TeXworks is the editor for you. There's really not a lot of extraneous options that you need to worry about here and, since it already comes with your MikTeX distribution, there's no pesky separate installation about which to worry: it is designed to work right out of the box, so-to-speak.



However, there is a price you pay for simplicity, I suppose. TeXworks has no spell checker or auto-suggestion features. Aside from changing font size and type, it is not customizable in the least bit. In fact, it doesn't even have a serious HELP menu (not that it really needs one). Essentially, almost everything about it screams "bare bones basic". That being said, I am grateful to see its designers recognize the value behind a multi-file view and an embedded pdf viewer that lets you keep the output document open while recompiling.

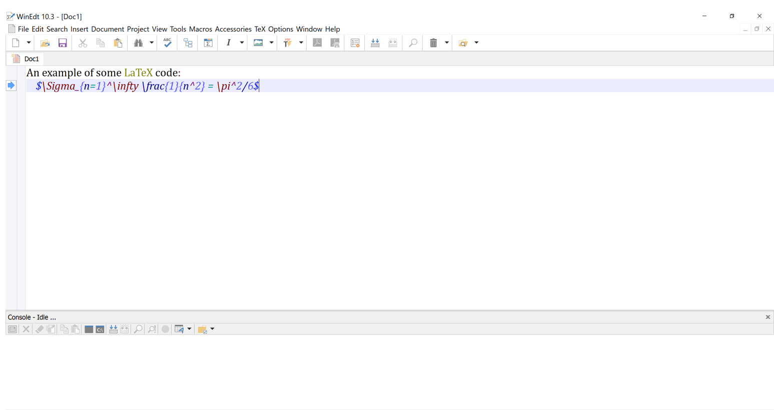
Summary: TeXworks is like the stripped down model of a car. It's fantastic for those just starting out and gives you the feel of successfully creating and compiling your own LaTeX documents. Of course, if you become a more serious LaTeX user, you will definitely want to consider an upgrade.

A.5 WinEdt

I would put WinEdt right up there with TEXstudio in terms of being an overall impressive LATEX editor that boasts quite a bit with which experienced LATEX users would salivate. Virtually every single menu category at the top is busting at the seams with options. It is also abundantly clear that a lot of attention was paid to the design and appearance of the editor. The graphics are simple and intuitive, while the coding environment is clean and bright. In addition, the unintrusive auto-suggestion feature is something over which I think all other editors should take a lesson.

However, I will freely admit that there was a time that I preferred TEXstudio to WinEdt, primarily due to WinEdt's lack of a dedicated pdf viewer (let along one as responsive as TEXstudio's). But version 11 of WinEdt now boasts such a pdf viewer. And, while it still needs a little work, it works well enough to woo me back.

Perhaps WinEdt's greatest shortcoming is that it is (still) relatively challenging to customize. Say you want to change the color of the LATEX macros from brown to red. To do so, you have venture into WinEdt's OPTIONS INTERFACE menu, select the submenu dealing with colors, at which point what you will see looks something like WinEdt's base code. As you start wading through the 1,381 lines of code here, you try to piece together to what each of the terms refers until you begin to wonder whether or not tinkering with anything in this code would even be worth it.



Summary: If you are an experienced LATEX user and like the default color scheme of WinEdt just fine, it is by all means worth it to get WinEdt.

Appendix B

APA 7 Style for Bibliographies

> It is nothing for one to know something unless another knows that you know it.
>
> Persian proverb

In this appendix, I briefly showcase APA 7 style for formatting bibliographies. This style is considered to be the gold standard of presenting bibliographic information by many professionals. The APA has a rigorous system for determining how such data is to be rendered in the bibliography, even when certain pieces of data are missing. Moreover, the defining characteristic of APA style is that bibliographic information is presented using the basic ordering *author(s)*, *year/date*, *title*, *source*, wherein if one of these pieces of information is unknown, the next takes its place. (If the date is not given for a source, check out the bang *What year is it?* on page 180.)

While `TNThesis` will automatically place the information from your `bib` file in the requisite APA 7 style and automatically order your entries according to the above rules, I want to illustrate what certain bibliographic items in your thesis or dissertation will look like using APA 7 style. *Bear in mind though*, unlike traditional APA style that uses no labels, each of your items will be labeled with a number, which will allow a reader to identify the corresponding source from the citation number in the text. Because of this use of labels, `TNThesis` will *not* indent lines after the first, as is standard practice for APA style, since the labels make a clear delineation of the various bibliographic items.

Journal Article In order of presentation, the components are as follows:

- The list of authors in the order in which they are presented in the article (alphabetical by last name) with the format of each author's name given by

  Last Name, ⟨*space*⟩ First Initial. ⟨*space*⟩ Middle Initial (if given).

  The list of authors is separated by commas with an ampersand & introducing the final author in the list.
- The year of publication in parentheses followed by a period.
- The title of the article in normal roman font with the convention that only the first letter of the title, the first letter of a subtitle, the first word after a colon or dash, and proper names in the title are capitalized. The title is then followed by a period.

- The full name of the journal in which the article is found followed by a comma. The font for the journal's name should be *slanted*.
- The volume number of the journal in which the article is found, given in *slanted* font. If the volumes of the journal are further parsed by issue, the issue number is represented immediately after the volume number in normal roman font offset by parentheses. Regardless, a comma follows this part.
- The inclusive page numbers of the article, separated by an en-dash (in LaTeX, an en-dash can be created by putting two hyphens next to each other, as in `--`). The page numbers are then followed by a period.
- If the article has a DOI (Digital Optical Identifier) code, the corresponding URL is placed at the very end without any punctuation following the URL. Such a URL will take the form

  https://doi.org/⟨*code 1*⟩/⟨*code 2*⟩

  Here ⟨*code 1*⟩ takes the form 10.*xxxx* or 10.*xxxx*.*xx*, where *xxxx* represents the registrant code and *xx* represents a subdivision, if one exists for the registrant code. The final piece of information, ⟨*code 2*⟩, is the DOI suffix.

Example.

[1] Lagarias, J. C., Montgomery, H. L., & Odlyzko, A. M. (1979). A bound for the least prime ideal in the Chebotarev density theorem. *Inventiones Mathematicae*, *54*(3), 271–296. https://doi.org/10.1007/BF01390234

Entire Book In order of presentation, the components are as follows:

- The author(s) in the order in which they are presented in the book with the format of each author's name given by

  Last Name, ⟨*space*⟩ First Initial. ⟨*space*⟩ Middle Initial (if given).

  The list of authors is separated by commas with an ampersand & introducing the final author.
- The year of publication in parentheses followed by a period.
- The title of the book in *slanted* font with the convention that only the first letter of the title, the first letter of a subtitle, the first word after a colon or dash, and proper names in the title are capitalized. If the book has multiple editions, a comma follows the title along with an indication of which edition was used as a source for the given work. Standard abbreviations are used to convey edition information, such as "ed." for "edition" and "rev." for "revised". A period always concludes this component of the bibliographic item.
- The name of the publisher followed by a period. Standard abbreviations are permitted here, as well. However, names of places (e.g., Cambridge) are never abbreviated.

- If the book has a DOI (Digital Optical Identifier) code, the corresponding URL is placed at the very end without any punctuation following the URL. See the information above on "journal article" for more explanation on DOI codes.

Example.

[1] Kaplansky, I. (1974). *Commutative rings*, rev. ed. Univ. Chicago Press.

Chapter in a Special Publication In order of presentation, the components are as follows:

- The author(s) in the order in which they are presented in the chapter with the format of each author's name given by

  Last Name, ⟨*space*⟩ First Initial. ⟨*space*⟩ Middle Initial (if given).

  The list of authors is to be separated by commas with an ampersand & introducing the final author.
- The year of publication in parentheses followed by a period.
- The title of the chapter in normal roman font with the convention that only the first letter of the title, the first letter of a subtitle, the first word after a colon or dash, and proper names in the title are capitalized. The title is then followed by a period.
- The word "In" followed by the editor's names in the format

  Last Name, ⟨*space*⟩ First Initial. ⟨*space*⟩ Middle Initial (if given).

  separated by commas with an ampersand & before the last editor. The list is followed by either (Ed.) or (Eds.), as appropriate, and a comma.
- The title of the special publication in *slanted* font.
- The inclusive page numbers of the chapter in normal roman font, offset by parentheses, and introduced by either "p." or "pp.", as appropriate. This part is followed by a period.
- The name of the publisher followed by a period. Standard abbreviations are permitted here. However, names of places (e.g., Cambridge) are never abbreviated.
- If the book has a DOI (Digital Optical Identifier) code, place the corresponding URL here without any punctuation following the URL. See the information above on "journal article" for more explanation on DOI codes.

Example.

[1] Dobbs, D. E. (1997). On Henselian pullbacks. In D. D. Anderson (Ed.), *Factorization in integral domains: Lecture notes in pure and applied mathematics* (pp. 317–326). Dekker.

Web page In order of presentation, the components are as follows:

- The author(s) in the order in which they are presented on the website with the format of each author's name given by

  Last Name, ⟨*space*⟩ First Initial. ⟨*space*⟩ Middle Initial (if given).

The list of authors is to be separated by commas with an ampersand & introducing the final author. If the website is created by an organization rather than an individual, then the name of the organization should be placed here instead. Regardless, this part ends with a period.

- The date of web-publication in parentheses followed by a period. Here the format of the date should be

  Year, ⟨*space*⟩ Month ⟨*space*⟩ Day

- The title of the web page in *slanted* font with the convention that only the first letter of the title, the first letter of a subtitle, the first word after a colon or dash, and proper names in the title are capitalized. The title is then followed by a period.
- The name of the website's sponsor in normal roman font followed by a period.
- The word "Retrieved" followed by the date the information from the website was downloaded. The date here should be in the format

  Month ⟨*space*⟩ Day, ⟨*space*⟩ Year

  followed by a comma and the word "from".
- The full URL of the web page with no concluding punctuation.

Example.

[1] Dineva, R. (2005, July 29). *The Euler totient, the Mobius and the divisor functions*. Mount Holyoke College. Retrieved February 3, 2020, from http://www.mtholyoke.edu/ robinson/reu/reu05/rdineva1.pdf

If you have a source type that is not addressed by one of the above categories, I would encourage you to check out the website owl.purdue.edu.

Appendix C

The TNTmain Program

> The eye sees only what the mind is prepared to comprehend.
>
> Robertson Davies

In Chapter 12, I gave you an overview of how you can use the **TNTmain** program to create the various parts of your thesis. Here, for your reference, I provide you with a listing of the initial code with which **TNTmain** comes equipped.

```
% This is the file `TNTmain.tex' version 5.1.0 designed
% to be the main file from which you will compile your
% entire thesis or dissertation.

% You should place all files relevant to your thesis or
% dissertation in the same folder as this tex file *and*
% the TNThesis class file so that this file can
% `recognize' all of the other files.

% To use a command below, simply remove the percentage
% symbol (%) from in front of it and edit appropriately

% Questions should be directed to Andrew J. Hetzel,
% ahetzel@tntech.edu

\documentclass{TNThesis}

%\ThesisTitle{THE REASON FOR MY LACK OF A SOCIAL LIFE FOR THE PAST TWO YEARS}

%\ThesisAuthor{Georg Ferdinand Ludwig Philipp Cantor}

```

```
%\DocumentType{Thesis}
%\DegreeType{Master of Science}
%\Major{Mathematics}
%\GraduationMonth{May}
%\GraduationYear{2023}

%\ThesisAbstract{abstract filename}

%\CommitteeMember{1}{Sir Isaac Newton, Chair}
%\CommitteeMember{2}{Emmy Noether, Co-chair}
%\CommitteeMember{3}{Marie Curie}
%\CommitteeMember{4}{Albert Einstein}
%\CommitteeMember{5}{Kurt Godel}

%\Dedication{For my parents, who loved me so much from the time that I was born.}
%\Acknowledgments{acknowledgments filename}

%\MyChapter{1}{chapter 1 filename}
%\MyChapter{2}{chapter 2 filename}
%\MyChapter{3}{chapter 3 filename}
%\MyChapter{4}{chapter 4 filename}
%\MyChapter{5}{chapter 5 filename}
%\MyChapter{6}{chapter 6 filename}
%\MyChapter{7}{chapter 7 filename}

%\MyAppendix{1}{appendix 1 filename}
%\MyAppendix{2}{appendix 2 filename}
%\MyAppendix{3}{appendix 3 filename}

%\MyBibReferences{bibliography filename}

%\MyVita{vita filename}

%       place any extra macros *before* the \ThesisContent macro below

\ThesisContent
```

Appendix D

The `TNThesis.cls` Code

Nothing in this world can take the place of persistence.

Calvin Coolidge

In this appendix, I'll show the morbidly curious out there (and you know who you are) what the underlying `TNThesis` code looks like in its entirety with some commentary along the way. This documented code was produced using the package `doc` included with the MikTEX distribution. While `TNThesis` does not come equipped with this package, if you have a program in LATEX related to your thesis or dissertation that you wish to present in this style, let me know and I will show you how it's accomplished.

Now turn the page...and prepare yourself.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesClass{TNThesis}[2023/03/03 LaTeX class file for %
TN Tech science theses and dissertations, v. 'November' 5.1.0]
```

establishes the TNThesis documentclass option “tikz” to load the `tikz` package for those who need it

```
    \newif\if@tnt@usetikz@
        \@tnt@usetikz@false
    \newif\if@tnt@useminted@
        \@tnt@useminted@false
    \DeclareOption{tikz}{\global\@tnt@usetikz@true}
    \DeclareOption{minted}{\global\@tnt@useminted@true}
```

`TNThesis` is built upon the `memoir` class (https://ctan.org/pkg/memoir). The “oldfontcommands” options allows for the use of deprecated commands such as `\bf`, `\sc`, etc.

```
\DeclareOption*{\PassOptionsToClass{\CurrentOption}{memoir}}
    \ProcessOptions\relax
\LoadClass[oldfontcommands, twoside, extrafontsizes, 12pt]{memoir}
```

This section loads all of the relevant packages. The packages `lmodern`, `microtype`, and `babel` should be loaded before `inputenc` and `fontenc`, textttminted should be loaded before `csquotes`, and `cleveref` should be loaded last. Also, the “newfloat” option for the `minted` package is important for creating a proper list of listings.

```
\RequirePackage{lmodern}
\RequirePackage{microtype}
\RequirePackage[american]{babel}
\RequirePackage[utf8]{inputenc}
\RequirePackage[T1]{fontenc}

\if@tnt@useminted@
    \RequirePackage[cache, newfloat]{minted}
\fi

\RequirePackage[autostyle=true]{csquotes}
\RequirePackage{ragged2e}

\RequirePackage{amsfonts, amssymb, amsthm, latexsym, mathtools}
    \mathtoolsset{mathic}
\RequirePackage{mathrsfs}

\RequirePackage{interval}
    \intervalconfig{soft open fences}
\RequirePackage{bigdelim, bigstrut, colortbl, multirow}
\RequirePackage{centernot}

\RequirePackage{thmtools}
\RequirePackage{enumitem}
    \setlist[description]{left=\parindent}
    \setlist[enumerate]{left=\parindent}
    \setlist[itemize]{left=\parindent}

\RequirePackage{xcolor, graphicx}

\if@tnt@usetikz@
```

```
    \RequirePackage{tikz}
        \usetikzlibrary{cd}
        \usetikzlibrary{decorations.pathmorphing}
        \usetikzlibrary{babel}
\fi

\RequirePackage{totcount}
\RequirePackage{url}
\RequirePackage{cleveref}
```

This is the newly established “command and control” section, which allows for the quick change of various parameters in the document.

```
%   general
\newdimen\tnt@parindent
    \tnt@parindent3em

%   title page
\newdimen\tnt@max@lineindent
    \tnt@max@lineindent2em
\newdimen\tnt@max@linestep
    \tnt@max@linestep3em
\newdimen\tnt@linestep@increment
    \tnt@linestep@increment1em
\newdimen\tnt@lineindent@increment
    \tnt@lineindent@increment1em
\newcount\tnt@title@hyphenpenalty
    \tnt@title@hyphenpenalty10

%   approval page
\newdimen\tnt@comember@liststd
    \tnt@comember@liststd1.9in
\newdimen\tnt@signature@linewd
    \tnt@signature@linewd4.5in

%   floats
\newskip\tnt@textfloatsep
    \tnt@textfloatsep1\onelineskip plus 0.5\onelineskip minus 0.5\onelineskip
\newskip\tnt@intextsep
    \tnt@textfloatsep1\onelineskip plus 0.5\onelineskip minus 0.5\onelineskip
\newskip\tnt@floatsep
    \tnt@floatsep1\onelineskip plus 0.5\onelineskip minus 0.5\onelineskip
\def\tnt@floatpagefrac{0.7}
\def\tnt@textfrac{0.2}
\def\tnt@topfrac{0.7}
\def\tnt@bottomfrac{0.3}
\def\tnt@minted@numberfont#1{%
    \sffamily%
    \footnotesize%
    \oldstylenums{%
            #1%
        }%
}
\newdimen\tnt@minted@numbersep
    \tnt@minted@numbersep1em
```

```

%   table of contents/list pages
\newdimen\tnt@std@pgboxwidth
    \tnt@std@pgboxwidth5em
\newdimen\tnt@spc@pgboxwidth
    \tnt@spc@pgboxwidth7em
\newif\if@tnt@single@spacing@
    \@tnt@single@spacing@true

%   chapter and friends
\newskip\tnt@chapskip
    \tnt@chapskip0ex plus 1ex minus 1ex
\newskip\tnt@secbeforeskip
    \tnt@secbeforeskip2ex plus 1ex minus .25ex
\newskip\tnt@secafterskip
    \tnt@secafterskip1ex plus .5ex

%   bibliography
\newdimen\tnt@labelwidth
    \tnt@labelwidth\tnt@parindent

%   theorem and friends
\newskip\tnt@displayaboveskip
    \tnt@displayaboveskip2ex plus 1ex minus 1ex
\newskip\tnt@displaybelowskip
    \tnt@displaybelowskip2ex plus 1ex minus 1ex
\newskip\tnt@displayshortaboveskip
    \tnt@displayshortaboveskip1.5ex plus 0.5ex minus 0.5ex
\newskip\tnt@displayshortbelowskip
    \tnt@displayshortbelowskip1.5ex plus 0.5ex minus 0.5ex
\newskip\tnt@abovethm
    \tnt@abovethm2ex plus 1ex minus 1ex
\newskip\tnt@belowthm
    \tnt@belowthm2ex minus 1ex
```

The following is a set of useful macros, particularly for the key=value systems used with the floats

```
\def\concatenate#1#2{%
    \expandafter\xdef\expandafter#1\expandafter{#1#2}
}

\def\xifsame#1#2#3#4{%
    \edef\@tempa{%
        \noexpand\detokenize{#1}
    }
    \edef\@tempb{%
        \noexpand\detokenize{#2}
    }
    \ifx\@tempa\@tempb%
        #3%
    \else%
        #4%
    \fi
}
```

```

\def\addtolist#1#2{%
    \xifsame{#1}{}{%
        \xdef#1{\zap@space#2 \@empty}%
    }{%
        \xdef#1{\zap@space#1,#2 \@empty}%
    }
}

\def\@firstofthree#1#2#3{#1}
\def\@secondofthree#1#2#3{#2}
```

The next section sets the general page properties for the document

```
%       global text spacing adjustments

\DoubleSpacing
\midsloppy
\hfuzz2.62pt
\vfuzz2.62pt


%       global demerit and penalty parameters

\doublehyphendemerits100000
\brokenpenalty10000
\widowpenalty10000
\clubpenalty10000


%       global page layout parameters

\paperwidth8.5in
\paperheight11in
\foremargin1.0in
\spinemargin1.5in
\textwidth6.0in

\newcommand*{\LineJump}[1]{%
    \vspace*{#1\baselineskip}
}
\headheight\baselineskip
\headsep0ex

\uppermargin1in
    \addtolength{\uppermargin}{-\topskip}
\lowermargin1in

\textheight\paperheight
    \addtolength{\textheight}{-\uppermargin}
    \addtolength{\textheight}{-\lowermargin}

\footskip2\onelineskip

\parindent\tnt@parindent

```

```
\checkandfixthelayout[nearest]

\makepagestyle{tech}
    \makeevenfoot{tech}{}{\thepage}{}
    \makeoddfoot{tech}{}{\thepage}{}
```

The following section uses code inspired by David Carlisle (https://tex.stackexchange.com/a/56853/82186) in order to create an optimal title appearance for the title page, abstract page, and approval page

```
\edef\newinnerdimen{\noexpand\newdimen}

\newdimen\tnt@fittedwidth
\newdimen\tnt@naturalwidth
\newdimen\tnt@deformation
\newdimen\tnt@prevdepth
\newdimen\tnt@lineindent
\newdimen\tnt@linestep

\newcount\tnt@penalty
\newcount\tnt@looseness
\newcount\tnt@linedemerit
\newcount\tnt@totalnumber@linedemerits
\newcount\tnt@savedpenalty
\newcount\tnt@linecount
\newcount\tnt@min@totalnumber@linedemerits
    \tnt@min@totalnumber@linedemerits100000000
\newcount\tnt@maxnumber@titlelines

\newskip\tnt@savedskip

\newbox\tnt@linebox
\newbox\tnt@parabox
\newbox\tnt@temp@sglspace@title
\newbox\tnt@temp@dblspace@title
\newbox\tnt@final@sglspace@title
\newbox\tnt@final@dblspace@title

\newif\if@tnt@incompatiblelines@
\newif\if@tnt@doublehyphens@

\def\tnt@LineDemeritAssessment#1#2{%
    \count@\numexpr\linepenalty+#1\relax
    \@tempcnta\numexpr\count@*\count@\relax
    \@tempcntb\numexpr#2*#2\relax
    \ifnum-1<#2
        \tnt@linedemerit\numexpr\@tempcnta+\@tempcntb\relax
    \else\ifnum-10000<#2
        \tnt@linedemerit\numexpr\@tempcnta-\@tempcntb\relax
        \else
            \tnt@linedemerit\@tempcnta
        \fi
    \fi
    \global\advance\tnt@totalnumber@linedemerits by \tnt@linedemerit
}
```

```

\def\tnt@AdjDemeritsAssessment#1#2{%
    \@tempcnta\tnt@looseness
    \ifnum#1<13
        \tnt@looseness\tw@
    \else\ifdim#2<0pt
        \tnt@looseness\thr@@
        \else\ifnum#1<100
            \tnt@looseness\@ne
            \else
                \tnt@looseness\z@
            \fi
        \fi
    \fi
    \ifnum\numexpr\@tempcnta-\tnt@looseness\relax>1
        \global\@tnt@incompatiblelines@true
    \fi
    \ifnum\numexpr\tnt@looseness-\@tempcnta\relax<-1
        \global\@tnt@incompatiblelines@true
    \fi
}

\def\tnt@HyphenPenaltyAssessment#1{% only works if \brokenpenalty is nonzero
    % and \hyphenpenalty=\exhyphenpenalty is nonzero
    \begingroup
        \count@\tnt@penalty
        \hyphenpenalty\tnt@title@hyphenpenalty
        \exhyphenpenalty\hyphenpenalty
        \brokenpenalty10000
        \ifnum#1=\brokenpenalty
            \global\tnt@penalty\hyphenpenalty
        \else
            \global\tnt@penalty\z@
        \fi
        \ifnum\count@=\tnt@penalty
            \ifnum\tnt@penalty=\hyphenpenalty
                \global\@tnt@doublehyphens@true
            \fi
        \fi
    \endgroup
}

\def\tnt@AssessLineBox#1{%
    \setbox0\hb@xt@\wd#1{\unhcopy#1}
    \global\tnt@fittedwidth\wd0
    \setbox0\hbox{\unhcopy#1}
    \tnt@naturalwidth\wd0
    \tnt@deformation\dimexpr\tnt@fittedwidth-\tnt@naturalwidth\relax
    \tnt@LineDemeritAssessment{\badness}{\tnt@penalty}
    \tnt@HyphenPenaltyAssessment{\tnt@savedpenalty}
    \tnt@AdjDemeritsAssessment{\badness}{\tnt@deformation}
}

\def\tnt@ExtraDemeritsAssessment{%
```

```
    \if@tnt@incompatiblelines@%
        \global\advance\tnt@totalnumber@linedemerits by \adjdemerits
    \fi
    \if@tnt@doublehyphens@%
        \global\advance\tnt@totalnumber@linedemerits by \doublehyphendemerits
    \fi
}

\def\tnt@DissectTitle#1\par{%
    \tnt@prevdepth\prevdepth%
    \global\setbox\tnt@temp@dblspace@title\vbox{%
        \setbox0\vbox{%
            \hbox{\vrule depth \tnt@prevdepth}%
            \linespread{2.0}\selectfont%
            #1\par\tnt@dblspace@assesstitle%
        }%
        \unvbox\tnt@parabox%
    }%
    \global\setbox\tnt@temp@sglspace@title\vbox{%
        \setbox0\vbox{%
            \hbox{\vrule depth \tnt@prevdepth}%
            \linespread{1.0}\selectfont%
            #1\par\tnt@sglspace@assesstitle%
        }%
        \unvbox\tnt@parabox%
    }%
}

\def\tnt@sglspace@assesstitle{%
    \loop
        \setbox\tnt@linebox\lastbox
        \tnt@savedskip\lastskip\unskip
        \tnt@savedpenalty\lastpenalty\unpenalty
        \ifvoid\tnt@linebox\else
            \tnt@AssessLineBox\tnt@linebox
            \setbox0\hb@xt@\textwidth{\hfill\hb@xt@\tnt@fittedwidth
                {\unhcopy\tnt@linebox}\hfill}
            \global\setbox\tnt@parabox\vbox{%
                \penalty\tnt@savedpenalty
                \vskip\tnt@savedskip
                \box0
                \unvbox\tnt@parabox
            }
    \repeat
    \tnt@ExtraDemeritsAssessment
}

\def\tnt@dblspace@assesstitle{%
    \loop
        \setbox\tnt@linebox\lastbox
        \tnt@savedskip\lastskip\unskip
        \tnt@savedpenalty\lastpenalty\unpenalty
        \ifvoid\tnt@linebox\else
            \tnt@fittedwidth\wd\tnt@linebox
```

```
            \setbox0\hb@xt@\textwidth{\hfill\hb@xt@\tnt@fittedwidth
                {\unhcopy\tnt@linebox}\hfill}
            \global\setbox\tnt@parabox\vbox{%
                \penalty\tnt@savedpenalty
                \vskip\tnt@savedskip
                \box0
                \unvbox\tnt@parabox
            }
    \repeat
}

\def\tnt@TitleCompare#1#2{%
    \ifnum#1<#2
        \global\tnt@min@totalnumber@linedemerits#1
        \global\setbox\tnt@final@sglspace@title\vbox{%
            \unvbox\tnt@temp@sglspace@title
        }
        \global\setbox\tnt@final@dblspace@title\vbox{%
            \unvbox\tnt@temp@dblspace@title
        }
    \else
        \@gobble\tnt@temp@sglspace@title
        \@gobble\tnt@temp@dblspace@title
    \fi
}

\begingroup
    \count@\z@
    \loop
        \ifdim\dimexpr\tnt@max@lineindent+\the\count@\tnt@max@linestep\relax>%
            \dimexpr\textwidth/2\relax\else
            \global\tnt@maxnumber@titlelines\count@
            \advance\count@\@ne
    \repeat
\endgroup

\def\tnt@parshape#1#2#3{%
    \@tempcnta\z@
    \def\tnt@parshapelist{}
    \@tempdima#2%
    \@tempdimb#3%
    \loop
        \ifnum\@tempcnta>#1\else
            \@tempcntb\numexpr2*\@tempcnta\relax%
            \concatenate\tnt@parshapelist{%
                \dimexpr\@tempdima+\the\@tempcnta\@tempdimb\relax %
                \dimexpr\textwidth-2\@tempdima-\the\@tempcntb\@tempdimb\relax
            }
            \advance\@tempcnta\@ne
    \repeat
    \parshape \the\@tempcnta\space \tnt@parshapelist
}

\def\tnt@InvPyrmdShape#1#2#3{%
```

```
        \parindent0em
        \leftskip0pt plus0.5fil
        \rightskip0pt plus-0.5fil
        \parfillskip0pt plus1fil
        \tnt@parshape{\tnt@maxnumber@titlelines}{#1}{#2}%
            \textbf{\boldmath\MakeTextUppercase{#3}}
}

\def\tnt@FindBestTitle#1{%
    \begingroup
        \overfullrule0em
        \hbadness1000000
        \hfuzz\maxdimen
        \clubpenalty0
        \widowpenalty0
        \tnt@lineindent0em
        \loop
            \ifdim\tnt@lineindent>\tnt@max@lineindent\else
                \tnt@linestep1em
                {%
                    \loop
                        \ifdim\tnt@linestep>\tnt@max@linestep\else
                            \global\tnt@penalty\z@
                            \global\tnt@looseness\tw@
                            \global\@tnt@incompatiblelines@false
                            \global\@tnt@doublehyphens@false
                            \global\tnt@totalnumber@linedemerits\z@
                            \tnt@DissectTitle%
                                \tnt@InvPyrmdShape{\tnt@lineindent}{\tnt@linestep}{#1}\par
                            \tnt@TitleCompare%
                                {\tnt@totalnumber@linedemerits}{\tnt@min@totalnumber@linedem
                            \advance\tnt@linestep by \tnt@linestep@increment\relax
                    \repeat
                }
                \advance\tnt@lineindent by \tnt@lineindent@increment\relax
        \repeat
    \endgroup
}
```

`\ThesisTitle` Special macros for the creating the front matter:

```
\newcommand*{\ThesisTitle}[1]{%
    \def\tnt@raw@title{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@raw@title}{%
        \ClassWarningNoLine{TNT}{%
            Title has not been provided through\MessageBreak
            the \protect\ThesisTitle\space macro%
        }
        \tnt@FindBestTitle{%
            \textlangle\textlangle Title Not Provided\textrangle\textrangle%
        }
    }{%
        \tnt@FindBestTitle{\tnt@raw@title}
    }
```

```
}
```

\ThesisAuthor

```
\newcommand*{\ThesisAuthor}[1]{%
    \def\tnt@author{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@author}{%
        \ClassWarningNoLine{TNT}{%
            Author has not been provided through\MessageBreak
            the \protect\ThesisAuthor\space macro%
        }
        \def\tnt@author{%
            \textlangle\textlangle Name Not Provided\textrangle\textrangle%
        }
    }{}
}
```

\DocumentType

```
\newcommand*{\DocumentType}[1]{%
    \def\tnt@doctype{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@doctype}{%
        \ClassWarningNoLine{TNT}{%
            Document type has not been provided through\MessageBreak
            the \protect\DocumentType\space macro%
        }
        \def\tnt@doctype{%
            \textlangle\textlangle Document Type Not Provided\textrangle\textrangle%
        }
    }{}
}
```

\DegreeType

```
\newcommand*{\DegreeType}[1]{%
    \def\tnt@degreetype{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@degreetype}{%
        \ClassWarningNoLine{TNT}{%
            Degree type has not been provided through\MessageBreak
            the \protect\DegreeType\space macro%
        }
        \def\tnt@degreetype{%
            \textlangle\textlangle Degree Type Not Provided\textrangle\textrangle%
        }
    }{}
}
```

\Major

```
\newcommand*{\Major}[1]{%
    \def\tnt@major{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@major}{%
```

```
        \ClassWarningNoLine{TNT}{%
            Major has not been provided though\MessageBreak
            the \protect\Major\space macro%
        }
        \def\tnt@major{%
            \textlangle\textlangle Major Not Provided\textrangle\textrangle%
        }
    }{}
}
```

\GraduationMonth

```
\newcommand*{\GraduationMonth}[1]{%
    \def\tnt@gradmonth{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@gradmonth}{%
        \ClassWarningNoLine{TNT}{%
            Graduation month has not been provided through\MessageBreak
            the \protect\GraduationMonth\space macro%
        }
        \def\tnt@gradmonth{May}
    }{}
}
```

\GraduationYear

```
\newcommand*{\GraduationYear}[1]{%
    \def\tnt@gradyear{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@gradyear}{%
        \ClassWarningNoLine{TNT}{%
            Graduation year has not been provided through\MessageBreak
            the \protect\GraduationYear\space macro%
        }
        \def\tnt@gradyear{\the\year}
    }{}
}
```

\ThesisAbstract

```
\newcommand*{\ThesisAbstract}[1]{%
    \long\def\tnt@abstract{%
        \input{#1}
    }
}
\AtBeginDocument{%
    \@ifundefined{tnt@abstract}{%
        \ClassWarningNoLine{TNT}{%
            Abstract has not been provided through\MessageBreak
            the \protect\ThesisAbstract\space macro%
        }
        \def\tnt@abstract{%
            \textlangle\textlangle Abstract Not Provided\textrangle\textrangle%
        }
    }{}
}
```

\CopyrightOwner

```
\newcommand*{\CopyrightOwner}[1]{%
    \def\tnt@copyowner{#1}
}
\AtBeginDocument{%
    \@ifundefined{tnt@copyowner}{%
        \def\tnt@copyowner{%
            \tnt@author%
        }
    }{}
}
```

\CommitteeMember

```
\newcount\tnt@totalnumber@members
    \tnt@totalnumber@members\z@
\begingroup
    \count@\@ne
    \loop
        \ifnum\count@>9\else
            \expandafter\def\csname
            tnt@committee@member\the\count@\endcsname{}
            \advance\count@\@ne
    \repeat
\endgroup
\newcommand*{\CommitteeMember}[2]{%
    \expandafter\def\csname tnt@committee@member#1\endcsname{%
        \parbox[t]{\tnt@signature@linewd}{%
            \rule{\tnt@signature@linewd}{0.5pt}\vskip0ex%
            #2\hfill Date\hspace*{0.25in}%
        }\\%
    }
    \global\advance\tnt@totalnumber@members\@ne
}

\AtBeginDocument{%
    \ifnum\tnt@totalnumber@members=\z@
        \ClassWarningNoLine{TNT}{%
            Committee members’ names have not been provided through\MessageBreak
            the \protect\CommitteeMember\space macro%
        }
        \expandafter\def\csname tnt@committee@member1\endcsname{%
            \textlangle\textlangle Committee Member Not
            Provided\textrangle\textrangle%
        }
        \global\advance\tnt@totalnumber@members\@ne
    \fi
}
```

\Dedication

```
\newcommand{\Dedication}[1]{%
    \long\def\tnt@dedication{#1}
}
```

\Acknowledgments

```
\newcommand{\Acknowledgments}[1]{%
```

```
    \long\def\tnt@acknowledgment{%
        \input{#1}
    }
}
```

\MyChapter Special macros for the creating the main matter:

```
\newcount\tnt@totalnumber@chapters
    \tnt@totalnumber@chapters\z@
\newcommand*{\MyChapter}[2]{%
    \expandafter\def\csname tnt@mychapter#1\endcsname{%
        \input{#2}
    }
    \global\advance\tnt@totalnumber@chapters\@ne
}

\AtBeginDocument{%
    \ifnum\tnt@totalnumber@chapters=\z@
        \ClassWarningNoLine{TNT}{%
            A chapter has not been provided through\MessageBreak
            the \protect\MyChapter\space macro%
        }
    \fi
}
```

\MyAppendix Special macros for the creating the back matter:

```
\newcount\tnt@totalnumber@appendices
    \tnt@totalnumber@appendices\z@
\newcommand*{\MyAppendix}[2]{%
    \expandafter\def\csname tnt@myappendix#1\endcsname{%
        \input{#2}
    }
    \global\advance\tnt@totalnumber@appendices\@ne
}
```

\MyBibReferences

```
\newcommand*{\MyBibReferences}[1]{%
    \def\tnt@bibreferences{#1}
    \addbibresource{#1.bib}
}
```

\MyVita

```
\newcommand*{\MyVita}[1]{%
    \long\def\tnt@vita{%
        \input{#1}
    }
}
\AtBeginDocument{%
    \@ifundefined{tnt@vita}{%
        \ClassWarningNoLine{TNT}{%
            Your vita has not been provided through\MessageBreak
            the \protect\MyVita\space macro%
        }
        \def\tnt@vita{%
            \textlangle\textlangle Vita Not Provided\textrangle\textrangle%
        }
    }{}
```

```
}

\strictpagecheck\checkoddpage
```

\tnt@titlepage

```
\long\def\tnt@titlepage{%
    \thispagestyle{empty}
    \centering
    \begin{minipage}[c][9in]{6in}
       \centering
        \unvcopy\tnt@final@dblspace@title
        \LineJump{1}
        \vfill
        \rule{2in}{0.5pt}
        \vfill
        A \tnt@doctype \\
        Presented to \\
        the Faculty of the College of Graduate Studies \\
        Tennessee Technological University \\
        by \\
        \tnt@author
        \vfill
        \rule{2in}{0.5pt}
        \vfill
        In Partial Fulfillment \\
        of the Requirements of the Degree \\
        \MakeTextUppercase{\tnt@degreetype} \\
        \tnt@major
        \vfill
        \rule{2in}{0.5pt}
        \vfill
        \tnt@gradmonth\ \tnt@gradyear
    \end{minipage}
    \clearpage
}
```

\tnt@abstractpage

```
\long\def\tnt@abstractpage{%
    \cleartorecto
    \centering
    \begin{SingleSpace}
        \textbf{AN ABSTRACT OF A \MakeTextUppercase{\tnt@doctype}} \\
         \LineJump{1}
        \unvcopy\tnt@final@sglspace@title
        \LineJump{1}
        \tnt@author \\
        \LineJump{1}
        \tnt@degreetype\ in \tnt@major \\
        \LineJump{2}
        \begin{minipage}{6in}
            \parindent\tnt@parindent
            \tnt@abstract
        \end{minipage}
    \end{SingleSpace}
    \clearpage
```

```
}
```

\tnt@copyrightpage

```
\long\def\tnt@copyrightpage{%
    \clearpage
    \centering
    \begin{minipage}[c][9in]{6in}
       \centering
        Copyright \copyright \thinspace \tnt@copyowner, \tnt@gradyear
    \end{minipage}
    \clearpage
}
```

\tnt@approvalpage

```
\newdimen\tnt@member@vspace
\def\tnt@process@membersep#1{%
    \global\tnt@member@vspace\dimexpr\tnt@comember@liststd/#1\relax
}
\long\def\tnt@approvalpage{%
    \clearpage
    \phantomsection
    \addcontentsline{toc}{chapter}{Certificate of Approval}
    \enlargethispage{\baselineskip}
    \centering
    \textbf{CERTIFICATE OF APPROVAL OF
        \MakeTextUppercase{\tnt@doctype}}\\
    \LineJump{0.5}
    \unvcopy\tnt@final@dblspace@title
    by\\
    \tnt@author\\
    \LineJump{0.5}
    \raggedright
    \begin{minipage}{\tnt@signature@linewd}
        \SingleSpacing
        \noindent Graduate Advisory Committee:\\
        \tnt@process@membersep\tnt@totalnumber@members
        \setlength{\extrarowheight}{\tnt@member@vspace}
        \begin{tabular}{@{} p{\tnt@signature@linewd} @{}}
            \csname tnt@committee@member1\endcsname
            \csname tnt@committee@member2\endcsname
            \csname tnt@committee@member3\endcsname
            \csname tnt@committee@member4\endcsname
            \csname tnt@committee@member5\endcsname
            \csname tnt@committee@member6\endcsname
            \csname tnt@committee@member7\endcsname
            \csname tnt@committee@member8\endcsname
            \csname tnt@committee@member9\endcsname
        \end{tabular}
        \LineJump{2}
        \parbox[t]{\tnt@signature@linewd}{%
            Approved for the Faculty:\\
            \LineJump{1}\\
            \rule{\tnt@signature@linewd}{0.5pt}\vskip0ex
            Mark Stephens, Dean\hfill Date\hspace*{0.25in}\\
            College of Graduate Studies%
```

```
        }
    \end{minipage}
    \clearpage
}
```

\tnt@dedicationpage

```
\long\def\tnt@dedicationpage{%
    \clearpage
    \phantomsection
    \addcontentsline{toc}{chapter}{Dedication}
    \centering
    \begin{minipage}[c][9in]{5in}
            \centering
            \textbf{DEDICATION} \\
            \LineJump{1}
            \tnt@dedication
    \end{minipage}
    \clearpage
}
```

\tnt@acknowledgmentspage

```
\long\def\tnt@acknowledgmentspage{%
    \clearpage
    \phantomsection
    \addcontentsline{toc}{chapter}{Acknowledgments}
    \begin{center}
        \textbf{ACKNOWLEDGMENTS}
    \end{center}
    \LineJump{1}
    \begin{minipage}{6in}
        \parindent\tnt@parindent
        \tnt@acknowledgment
    \end{minipage}
    \clearpage
}
```

\tableofcontents*

```
\settocdepth{subsubsection}
\setsecnumdepth{subsubsection}

\addtodef{\tableofcontents*}{%
    \clearpage
    \if@tnt@single@spacing@\SingleSpacing\fi
    }{
    \clearpage
    \DoubleSpacing
}
\addto{\captionsamerican}{%
    \renewcommand*{\contentsname}{Table of Contents}
}

\renewcommand{\printtoctitle}[1]{%
    \centering
    \textbf{\MakeTextUppercase{#1}}
}
```

```
\renewcommand*{\tocheadstart}{%
    \if@tnt@single@spacing@\LineJump{-0.5}\else\LineJump{-0.25}\fi
    \par
}
\renewcommand*{\aftertoctitle}{%
    \if@tnt@single@spacing@\LineJump{0.5}\else\LineJump{0.25}\fi
    \par
}
\renewcommand*{\cftchapterfont}{\normalfont}
\renewcommand*{\cftchaptername}{Chapter\space}
\renewcommand*{\cftappendixname}{Appendix\space}
\renewcommand*{\cftchapteraftersnum}{:}
\renewcommand*{\cftchapterpagefont}{\normalfont}
\renewcommand*{\cftchapterleader}{%
    \cftchapterfont
    \cftdotfill{\cftchapterdotsep}
}
\renewcommand*{\cftchapterdotsep}{\cftnodots}
\cftbeforechapterskip\f@baselineskip
\renewcommand*{\insertchapterspace}{}
\renewcommand{\@tocrmarg}{4em plus1fil}

\renewcommand*{\cftsectiondotsep}{\cftnodots}
\cftbeforesectionskip\f@baselineskip

\renewcommand*{\cftsubsectiondotsep}{\cftnodots}
\cftbeforesubsectionskip\f@baselineskip

\renewcommand*{\cftsubsubsectiondotsep}{\cftnodots}
\cftbeforesubsubsectionskip\f@baselineskip
```

This section sets parameters for floats. Also new float environments are established related to the `\Graphic` and `\Table` macros.

```
\newfloat{@Figure}{lof}{Figure}
\newfloat{@Table}{lot}{Table}
\newfloat{@Listing}{lol}{Listing}

\setFloatBlockFor{subsubsection}
\setFloatBlockFor{subsection}
\setFloatBlockFor{section}
\setFloatBlockFor{chapter}

\@fptop0ex

\textfloatsep\tnt@textfloatsep
\intextsep\tnt@textfloatsep
\floatsep\tnt@floatsep
\renewcommand{\bottomfraction}{\tnt@bottomfrac}
\renewcommand{\topfraction}{\tnt@topfrac}
\renewcommand{\textfraction}{\tnt@textfrac}
\renewcommand{\floatpagefraction}{\tnt@floatpagefrac}

\counterwithout{figure}{chapter}
\regtotcounter{figure}
```

```
\counterwithout{table}{chapter}
\regtotcounter{table}

\def\tnt@floats{%
    figure,@Figure,table,@Table%
}
\@for\type:=\tnt@floats\do{%
    \setfloatlocations{\type}{htbp}
    \setfloatadjustment{\type}{%
        \SingleSpacing
        \raggedright
    }
}

\if@tnt@useminted@
    \newfloat{@Listing}{lol}{Listing}

    \counterwithout{listing}{chapter}
    \regtotcounter{listing}

    \setfloatlocations{listing}{htbp}
    \setfloatadjustment{listing}{%
        \SingleSpacing
        \raggedright
    }
    \setfloatlocations{@Listing}{htbp}
    \setfloatadjustment{@Listing}{%
        \SingleSpacing
        \raggedright
    }
\fi

\renewcommand\@memfront@floats{}
\renewcommand\@memmain@floats{}
\renewcommand\@memback@floats{}
```

This section sets parameters for the captions.

```
\captionnamefont{%
    \bfseries
    \raggedright
}
\captiontitlefont{%
    \itshape
}
\captiondelim{\newline\newline}
\captionstyle{\raggedright}
\belowcaptionskip1.5\onelineskip plus 0.5\onelineskip minus 0.5\onelineskip
\abovecaptionskip0\onelineskip plus 0.5\onelineskip
```

This section sets parameters for the list of figures and list of tables (i.e., the standard lists).

```
\def\tnt@stdlist@ids{%
    {figure}{Figure}{lof},{table}{Table}{lot}%
}
\def\tnt@create@stdlists#1#2#3{%
```

```
    \expandafter\addtodef\csname listof#1s\endcsname{%
        \clearpage
        \if@tnt@single@spacing@\SingleSpacing\fi
    }{%
        \DoubleSpacing
        \clearpage
    }
    \expandafter\renewcommand\csname list#1name\endcsname{%
        List of #2s%
    }
    \expandafter\renewcommand\csname print#3title\endcsname[1]{%
        \centering
        \textbf{\MakeTextUppercase{##1}}
    }
    \expandafter\renewcommand\csname#3headstart\endcsname{%
        \if@tnt@single@spacing@\LineJump{-0.5}\else\LineJump{-0.25}\fi
        \par
    }
    \expandafter\renewcommand\csname after#3title\endcsname{%
        \if@tnt@single@spacing@\LineJump{2}\else\LineJump{1}\fi
        \par
    }
}
```

This section sets parameters for the list of listings if the “minted” option is used.

```
\if@tnt@useminted@
    \def\listlistingname{List of Listings}
    \newlistof{listoflistings}{lol}{\listlistingname}

    \def\tnt@create@listoflistings{%
        \addtodef\listoflistings{%
            \clearpage
            \if@tnt@single@spacing@\SingleSpacing\fi
        }{%
            \DoubleSpacing
            \clearpage
        }
        \renewcommand\printloltitle[1]{%
            \centering
            \textbf{\MakeTextUppercase{##1}}
        }
        \renewcommand\lolheadstart{%
            \if@tnt@single@spacing@\LineJump{-0.5}\else\LineJump{-0.25}\fi
            \par
        }
        \renewcommand\afterloltitle{%
            \if@tnt@single@spacing@\LineJump{2}\else\LineJump{1}\fi
            \par
        }
    }
\fi
```

Here the list specifications above are implemented and the entries in each list designed to suit APA 7 requirements.

```
\@for\type:=\tnt@stdlist@ids\do{%
```

```
    \edef\@typea{\expandafter\@firstofthree\type}%
    \edef\@typeb{\expandafter\@secondofthree\type}%
    \edef\@typec{\expandafter\@thirdofthree\type}%
    \tnt@create@stdlists\@typea\@typeb\@typec%
}
\if@tnt@useminted@
    \tnt@create@listoflistings
\fi

%   entry specifications in the list of figures
\def\tnt@@cftfigure\numberline#1#2#3{%
    \noindent%
    \makebox[\tnt@std@pgboxwidth][l]{%
        Page #3%
    }%
    \parbox[t]{\dimexpr\textwidth-\tnt@std@pgboxwidth\relax}{%
        Figure #1:\ #2%
    }%
    \vskip1\baselineskip%
}
\def\tnt@cftfigure#1#2{%
    \tnt@@cftfigure#1{#2}%
}
\let\c@figure\c@@Figure
\let\l@figure\tnt@cftfigure
\let\l@@Figure\tnt@cftfigure

%   entry specifications in the list of tables
\def\tnt@@cfttable\numberline#1#2#3{%
    \noindent%
    \makebox[\tnt@std@pgboxwidth][l]{%
        Page #3%
    }%
    \parbox[t]{\dimexpr\textwidth-\tnt@std@pgboxwidth\relax}{%
        Table #1:\ #2%
    }%
    \vskip1\baselineskip%
}
\def\tnt@cfttable#1#2{%
    \tnt@@cfttable#1{#2}%
}
\let\c@table\c@@Table
\let\l@table\tnt@cfttable
\let\l@@Table\tnt@cfttable

%   entry specifications in the list of listings if ``minted'' option is used
\if@tnt@useminted@
    \def\tnt@@cftlisting\numberline#1#2#3{%
        \noindent%
        \makebox[\tnt@std@pgboxwidth][l]{%
            Page #3%
        }%
        \parbox[t]{\dimexpr\textwidth-\tnt@std@pgboxwidth\relax}{%
            Listing #1:\ #2%
```

```
        }%
        \vskip1\baselineskip%
    }
    \def\tnt@cftlisting#1#2{%
        \tnt@@cftlisting#1{#2}%
    }

    \let\c@listing\c@@Listing
    \let\l@listing\tnt@cftlisting
    \let\l@@Listing\tnt@cftlisting
\fi
```

`\SpecialListItem` This section establishes user-defined special lists pages

```
\newcount\tnt@number@spc@lists
    \tnt@number@spc@lists\z@
\def\SpecialListItem#1#2#3{%
    \expandafter\def\csname list#1name\endcsname{List of #3}
    \newlistof{listof#1s}{#2}{\csname list#1name\endcsname}
    \newlistentry{#1}{#2}{0}
    \counterwithout{#1}{chapter}
    \regtotcounter{#1}
    \expandafter\addtodef\csname listof#1s\endcsname{%
        \clearpage
        \if@tnt@single@spacing@\SingleSpacing\fi
    }{%
        \DoubleSpacing
        \clearpage
    }
    \expandafter\renewcommand\csname print#2title\endcsname[1]{%
        \centering
        \textbf{\MakeTextUppercase{##1}S}
    }
    \expandafter\renewcommand\csname #2headstart\endcsname{%
        \if@tnt@single@spacing@\LineJump{-0.5}\else\LineJump{-0.25}\fi
        \par
    }
    \expandafter\renewcommand\csname after#2title\endcsname{%
        \if@tnt@single@spacing@\LineJump{1}\else\LineJump{0.5}\fi
        \par
    }
    \expandafter\newinnerdimen\csname tnt@#1width\endcsname
        \csname tnt@#1width\endcsname5em
    \expandafter\def\csname #1width\endcsname##1{%
        \csname tnt@#1width\endcsname##1
    }
    \expandafter\def\csname tnt@cft#1\endcsname##1##2{%
        \csname tnt@@cft#1\endcsname##1{##2}
    }
    \expandafter\def\csname tnt@@cft#1\endcsname\numberline##1##2##3{%
        \noindent\makebox[\tnt@spc@pgboxwidth][l]{%
            Page ##3%
        }%
        \makebox[\csname tnt@#1width\endcsname][l]{%
            \@firstoftwo##2%
        }%
```

```
        \parbox[t]{\dimexpr \textwidth-\tnt@spc@pgboxwidth%
            -\csname tnt@#1width\endcsname\relax}{%
            \phantom{M}%
            \@secondoftwo##2%
        }%
        \vskip1\baselineskip%
    }
    \expandafter\let\csname  l@#1\expandafter\endcsname%
        \csname tnt@cft#1\endcsname
    \expandafter\newcommand\csname#1\endcsname[2][]{%
       \refstepcounter{#1}
       \phantomsection
        \addcontentsline{#2}{#1}{%
            \protect\numberline{\value{#1}}{%
                {##2}{##1}
            }
        }
    }
    \advance\tnt@number@spc@lists\@ne
    \expandafter\def\csname
    tnt@speciallist@\the\tnt@number@spc@lists\endcsname{%
        \ifnum\totvalue{#1}>\z@%
            \phantomsection%
            \addcontentsline{toc}{chapter}{List of #3s}%
            \begingroup
                \renewcommand*{\addcontentsline}[3]{}%
                \csname listof#1s\endcsname
            \endgroup%
        \fi%
    }
}
```

This section creates universal keys for special float macros/environments.

```
\def\tnt@universalkeys{%
    caption,label,shortcaption,postcaption,note,callout%
}

\@for\key:=\tnt@universalkeys\do{%
    \begingroup
        \edef\ukey{%
            \endgroup%
            \noexpand\define@key{float}{\key}{%
                \noexpand\@namedef{tnt@\key}{##1}
            }
        }\ukey
}

\long\def\tnt@@note{%
    \@ifundefined{tnt@note}{}{%
            \vskip0.5\onelineskip%
            \parbox{\textwidth}{%
                \textit{Note}.\space%
                \tnt@note%
            }
    }
```

```
}

\long\def\tnt@@callout{%
    \@ifundefined{tnt@callout}{}{%
            \vskip0.5\onelineskip%
            \parbox{\textwidth}{%
                \tnt@callout%
            }
    }
}

\def\tnt@postfloat{%
    \tnt@@note%
    \tnt@@callout%
    \tnt@init@float@keyvalues%
}

\def\tnt@@postcaption{%
    \@ifundefined{tnt@postcaption}{}{%
        \tnt@postcaption%
    }
}
```

`\tnt@adjust@listingoptions` allows for the adjustment of the left-hand side of a listing float if line numbers are used. This is to ensure that the numbers do not spill into the left margin.

```
\if@tnt@useminted@
    \def\tnt@adjust@listingoptions{%
        \begingroup%
            \edef\nnin{%
                \endgroup%
                \noexpand\in@{numbers=none}{\tnt@listingoptions}%
            }\nnin%
        \ifin@%
            \addtolist{\tnt@listingoptions}{xleftmargin=\z@}%
        \else%
            \addtolist{\tnt@listingoptions}{xleftmargin=\tnt@minted@xleftmargin}%
        \fi%
    }
\fi

\def\tnt@prefloat#1{%
    \setkeys{float}{#1}%
    \if@tnt@useminted@\tnt@adjust@listingoptions\fi%
    \@ifundefined{tnt@caption}{%
        \ClassError{TNT}{%
            Every float requires a caption%
        }{}%
    }{%
        \@ifundefined{tnt@shortcaption}{%
            \def\tnt@shortcaption{\tnt@caption}
        }{}%
    }%
    \caption[\tnt@shortcaption]{\tnt@caption\tnt@@postcaption}%
```

```
    \@ifundefined{tnt@label}{}{%
        \label{\tnt@label}%
    }%
}
```

\Graphic The following section establishes the \Graphic macro.

```
\def\tnt@graphicsoptions{}

\def\tnt@figurekeys{%
    width,height,totalheight,keepaspectratio,origin,type,ext,page,clip,scale,angle%
}
\@for\key:=\tnt@figurekeys\do{%
    \begingroup
        \edef\fkey{%
            \endgroup%
            \noexpand\define@key{float}{\key}{%
                \noexpand\addtolist{\noexpand\tnt@graphicsoptions}{\key=##1}
            }
        }\fkey
}

\def\tnt@process@graphics#1{%
    \begingroup
        \edef\pg{%
            \endgroup%
            \noexpand\includegraphics\ifx\tnt@graphicsoptions\@empty\else%
                [\tnt@graphicsoptions]\fi{#1}%
        }\pg%
}

\newcommand{\Graphic}[2][]{%
    \@Figure%
        \tnt@prefloat{#1}%
        \tnt@process@graphics{#2}%
        \tnt@postfloat%
    \end@Figure%
}

\let\oldfigure\figure
\def\figure{%
    \oldfigure%
    \ClassWarning{TNT}{%
        It is strongly advised to use the \protect\Graphic\space macro\MessageBreak
        instead of the standard ``figure'' environment to get the\MessageBreak
        positioning of the figure and caption correct. See the\MessageBreak
        user's manual for details%
    }
}
```

\Table The following section establishes the \Table macro.

```
\define@key{float}{env}{%
    \def\tnt@env{#1}%
}

\define@key{float}{colspecs}{%
```

```
    \def\tnt@colspecs{#1}
}

\newif\if@tnt@compress@
    \@tnt@compress@false
\define@key{float}{compress}[true]{%
    \global\csname @tnt@compress@#1\endcsname
}

\define@key{float}{tabwidth}{%
    \def\tnt@tabwidth{#1}
}
\def\tnt@@tabwidth{%
    \@ifundefined{tnt@tabwidth}{}{%
        {\tnt@tabwidth}
    }
}

\def\tnt@table@compress#1#2#3{%
    \resizebox{\textwidth}{!}{%
        \begingroup%
            \edef\x{%
                \endgroup%
                \noexpand\begin{#1}{#2}%
            }\x%
                #3%
        \begingroup%
            \edef\y{%
                \endgroup%
                \noexpand\end{#1}%
            }\y%
    }%
}

\def\tnt@table@standard#1#2#3{%
    \setbox0\vbox{%
        \begingroup%
            \edef\s{%
                \endgroup%
                \noexpand\begin{#1}\tnt@@tabwidth{#2}%
            }\s%
            #3%
        \begingroup%
            \edef\w{%
                \endgroup%
                \noexpand\end{#1}%
            }\w%
    }\unvbox0%
}

\newcommand{\Table}[2][]{%
    \@Table%
        \tnt@prefloat{#1}%
        \if@tnt@compress@%
```

```
            \tnt@table@compress{\tnt@env}{\tnt@colspecs}{#2}%
        \else%
            \tnt@table@standard{\tnt@env}{\tnt@colspecs}{#2}%
        \fi%
        \tnt@postfloat%
    \end@Table%
}

\let\oldtable\table
\def\table{%
    \oldtable%
    \ClassWarning{TNT}{%
        It is strongly advised to use the ‘‘Table’’ macro\MessageBreak
        instead of the standard ‘‘table’’ environment to get the\MessageBreak
        positioning of the table and caption correct. See the\MessageBreak
        user’s manual for details%
    }
}
```

Listing The following section establishes the Listing environment

```
\if@tnt@useminted@
    \def\tnt@listingoptions{}
    \def\tnt@std@listingkeys{%
        baselinestretch,breakafter,breakaftersymbolpre,breakaftersymbolpost,%
        breakanywheresymbolpre,breakanywheresymbolpost,breakbefore,%
        breakbeforesymbolpre,breakbeforesymbolpost,breakindent,breakindentchars,%
        breaksymbolleft,breaksymbolright,breaksymbolsepleft,breaksymbolsepright,%
        breaksymbolindentleft,breaksymbolindentright,breaksymbol,breaksymbolsep,%
        breaksymbolindent,breaksymbolindentleftnchars,breaksymbolindentnchars,%
        breaksymbolindentrightnchars,breaksymbolsepnchars, breaksymbolsepleftnchars,%
        breaksymbolseprightnchars,bgcolor,codetagify,encoding,escapeinside,firstline,%
        firstnumber,fontfamily,fontseries,fontsize,fontshape,formatcom,frame,framerule,%
        framesep,gobble,highlightcolor,highlightlines,keywordcase,lastline,numbers,%
        numbersep,outencoding,rulecolor,space,spacecolor,style,stepnumber,tab,%
        tabcolor,tabsize,xleftmargin,xrightmargin%
    }
    \@for\key:=\tnt@std@listingkeys\do{%
        \begingroup
            \edef\ckey{%
                \endgroup%
                \noexpand\define@key{float}{\key}{%
                    \noexpand\addtolist{\noexpand\tnt@listingoptions}{\key=##1}
                }
            }\ckey
    }
    \def\tnt@bool@listingkeys{%
        autogobble,beameroverlays,breakaftergroup,%
        breakanywhere,breakautoindent,breakbeforegroup,breakbytoken,%
        breakbytokenanywhere,breaklines,curlyquotes,funcnamehighlighting,linenos,%
        numberfirstline,mathescape,numberblanklines,obeytabs,python3,%
        resetmargins,samepage,showspaces,showtabs,startinline,stepnumberfromfirst,%
        stepnumberoffsetvalues,stripall,stripnl,texcl,texcomments%
    }
    \@for\key:=\tnt@bool@listingkeys\do{%
        \begingroup
```

```
        \edef\bkey{%
            \endgroup%
            \noexpand\define@key{float}{\key}[true]{%
                \noexpand\addtolist{\noexpand\tnt@listingoptions}{\key=##1}
            }
        }\bkey
}

\def\tnt@prelisting#1#2{%
    \VerbatimEnvironment%
    \let\FVB@VerbatimOut\minted@FVB@VerbatimOut%
    \let\FVE@VerbatimOut\minted@FVE@VerbatimOut%
    \minted@configlang{#2}%
    \begingroup%
    \edef\skm{%
        \endgroup%
        \noexpand\setkeys{minted@opt@cmd}{#1}%
    }\skm%
    \minted@fvset%
    \begin{VerbatimOut}[codes={\catcode`\^^I=12},firstline,lastline]%
        {\minted@jobname.pyg}%
}
\def\tnt@postlisting{%
    \end{VerbatimOut}%
    \minted@langlinenoson%
    \minted@pygmentize{\minted@lang}%
    \minted@langlinenosoff%
}

\renewcommand{\theFancyVerbLine}{%
    \tnt@minted@numberfont{\arabic{FancyVerbLine}}%
}
\definecolor{codegray}{rgb}{0.97,0.97,0.97}

\newenvironment{Listing}[2][]
{%
    \@Listing%
    \tnt@prefloat{#1}%
    \tnt@prelisting{\tnt@listingoptions}{#2}%
}
{%
    \tnt@postlisting%
    \tnt@postfloat%
    \end@Listing%
}

\AtBeginDocument{%
    \let\oldlisting\listing
    \def\listing{%
        \oldlisting%
        \ClassWarning{TNT}{%
            It is strongly advised to use the ``Listing'' environment\MessageBreak
            instead of the ``listing'' environment to get the\MessageBreak
            positioning of the table and caption correct. See the\MessageBreak
```

```
                user’s manual for details%
            }
        }
        \def\lstlisting{%
            \ClassError{TNT}{%
                You *must* use the (provided) ‘‘minted’’ package and *not* the\MessageBreak
                ‘‘listings’’ package to display code. See the user’s manual for\MessageBreak
                more details%
            }{}
        }
    }
\fi
```

The following section sets default values for the special floats.

```
\if@tnt@useminted@%
    \newdimen\tnt@minted@xleftmargin

    \define@key{float}{finalline}{%
        \begingroup
            \setbox0\hbox{%
                \tnt@minted@numberfont{#1}%
            }
            \global\tnt@minted@xleftmargin%
                \dimexpr\tnt@minted@numbersep+\wd0\relax%
        \endgroup
    }
\fi

\def\tnt@default@float@keyvalues{%
    \if@tnt@useminted@%
        width=\textwidth, totalheight=\textheight, keepaspectratio=true,%
        env=tabular, compress=false,%
        breaklines=true, breakindent=3em, numbers=left, finalline=9,%
        numbersep=\tnt@minted@numbersep, xleftmargin=\tnt@minted@xleftmargin,%
        frame=lines, framesep=2ex, numberfirstline=true, style=xcode%
    \else%
        width=\textwidth, totalheight=\textheight, keepaspectratio=true,%
        env=tabular, compress=false%
    \fi%
}

\def\tnt@init@float@keyvalues{%
    \gdef\tnt@graphicsoptions{}%
    \gdef\tnt@listingoptions{}%
    \begingroup%
        \edef\skf{%
            \endgroup%
            \noexpand\setkeys{float}{\tnt@default@float@keyvalues}%
        }\skf%
}

\newcommand{\GlobalFloatParameters}[1]{%
    \gdef\tnt@default@float@keyvalues{#1}
}

```

```
\AtBeginDocument{%
    \tnt@init@float@keyvalues
}
```

The following section sets parameters for chapter, section, subsection, and subsubsection titles.

```
\makechapterstyle{tech}{%
    \beforechapskip\tnt@chapskip
    \midchapskip\tnt@chapskip
    \afterchapskip\baselineskip
    \renewcommand*{\printchaptername}{%
        \chapnamefont%
        \MakeTextUppercase{\chaptername}%
    }
    \renewcommand*{\chapnamefont}{%
        \centering%
        \bfseries%
        \boldmath%
    }
    \renewcommand*{\chapnumfont}{\chapnamefont}
    \renewcommand*{\printchapternum}{%
        \chapnumfont%
        \thechapter%
    }
    \renewcommand*{\printchaptertitle}[1]{%
        \chaptitlefont%
        \MakeTextUppercase{##1}%
    }
    \renewcommand*{\chaptitlefont}{%
        \centering%
        \bfseries%
        \boldmath%
    }
}

\setsecheadstyle{%
    \centering%
    \bfseries%
    \boldmath%
}
\setbeforesecskip{\tnt@secbeforeskip}
\setaftersecskip{\tnt@secafterskip}

\setsubsecheadstyle{%
    \raggedright%
    \bfseries%
    \boldmath%
}
\setbeforesubsecskip{\tnt@secbeforeskip}
\setaftersubsecskip{\tnt@secafterskip}

\setsubsubsecheadstyle{%
    \raggedright%
    \bfseries%
```

```
    \boldmath%
    \itshape%
}
\setbeforesubsubsecskip{\tnt@secbeforeskip}
\setaftersubsubsecskip{\tnt@secafterskip}
```

The following section calls the biblatex package and sets parameters for the bibliography.

```
\RequirePackage[style=apa, backend=biber, citestyle=numeric-comp,
language=american, defernumbers=true, natbib=true]{biblatex}

\DeclareLanguageMapping{american}{american-apa}
\DeclareBibliographyCategory{cited}
\def\ifcategoryempty#1{\ifcsvoid{blx@catg@#1}}

\AtBeginDocument{%
    \@ifundefined{tnt@bibreferences}{%
        \ClassError{TNT}{%
            Your bibliography file must be provided through\MessageBreak
            the \protect\MyBibReferences\space macro%
        }{}
    }{}
    \AtEveryCitekey{%
            \addtocategory{cited}{\thefield{entrykey}}%
    }%
    \DeclareFieldFormat{labelnumberwidth}{\mkbibbrackets{#1}}
    \defbibenvironment{bibliography}%
    {\list
        {%
            \printtext[labelnumberwidth]{%
                \printfield{labelprefix}%
                \printfield{labelnumber}%
            }
        }%
        {%
            \setlength{\labelwidth}{\tnt@labelwidth}
            \setlength{\leftmargin}{\labelwidth}%
            \setlength{\labelsep}{\biblabelsep}%
                \addtolength{\leftmargin}{\labelsep}%
            \setlength{\itemsep}{\bibitemsep}%
            \setlength{\parsep}{\bibparsep}%
        }%
        \renewcommand*{\makelabel}[1]{\hss##1}%
    }%
    {\endlist}%
    {%
        \item%
    }
}
\renewcommand{\bibname}{REFERENCES}
\defbibheading{bibliography}[\bibname]{%
    \chapter*{#1}%
}
```

\tnt@referencessection

```
\long\def\tnt@referencessection{%
    \cleardoublepage
    \phantomsection
    \addcontentsline{toc}{chapter}{References}
    \@ifundefined{tnt@bibreferences}{%
        \begin{center}
            \textbf{REFERENCES}
        \end{center}
        \LineJump{1}
        \textlangle\textlangle References Not Provided\textrangle\textrangle
    }{%
        \ifcategoryempty{cited}{%
            \begin{center}
                \textbf{REFERENCES}
            \end{center}
            \LineJump{1}
            \textlangle\textlangle References Not Cited\textrangle\textrangle
        }{%
            \begingroup
            \renewcommand*{\addcontentsline}[3]{}
            \bibitemsep\onelineskip
            \begin{SingleSpace}
                \printbibliography[%
                    category={cited},%
                    title={REFERENCES}
                ]
            \end{SingleSpace}
            \endgroup
        }
    }
    \clearpage
}
```

`\tnt@vitapage`

```
\long\def\tnt@vitapage{%
    \cleartorecto
    \phantomsection
    \addcontentsline{toc}{chapter}{Vita}
    \begin{center}
        \textbf{VITA}
    \end{center}
    \LineJump{1}
    \begin{minipage}{6in}
        \parindent\tnt@parindent
        \tnt@vita
    \end{minipage}
    \clearpage
}
```

The following sections call all requisite macros for the front matter, main matter, and back matter.

```
\long\def\frontmattercontent{%
    \pagenumbering{roman}
    \setcounter{page}{1}
    \tnt@titlepage
```

```
    \pagestyle{plain}
    \tnt@abstractpage
    \tnt@copyrightpage
    \tableofcontents*
    \tnt@approvalpage
    \@ifundefined{tnt@dedication}{}{\tnt@dedicationpage}
    \@ifundefined{tnt@acknowledgment}{}{\tnt@acknowledgmentspage}
    \ifnum\totvalue{figure}>\z@
        \phantomsection
        \addcontentsline{toc}{chapter}{List of Figures}
        \begingroup
            \renewcommand*{\addcontentsline}[3]{}
            \listoffigures
        \endgroup
    \fi
    \ifnum\totvalue{table}>\z@
        \phantomsection
        \addcontentsline{toc}{chapter}{List of Tables}
        \begingroup
            \renewcommand*{\addcontentsline}[3]{}
            \listoftables
        \endgroup
    \fi
    \if@tnt@useminted@
        \ifnum\totvalue{listing}>\z@
            \phantomsection
            \addcontentsline{toc}{chapter}{List of Listings}
            \begingroup
                \renewcommand*{\addcontentsline}[3]{}
                \listoflistings
            \endgroup
        \fi
    \fi
    \count@\@ne
    \loop
        \ifnum\count@>\tnt@number@spc@lists\else%
            \csname tnt@speciallist@\the\count@\endcsname%
            \advance\count@\@ne
    \repeat
}

\long\def\mainmattercontent{%
    \pagestyle{tech}
    \chapterstyle{tech}
    \parindent\tnt@parindent
    \justifying
    \indentafterchapter
    \ifnum\tnt@totalnumber@chapters=\z@
        \chapter{\textlangle\textlangle Chapter Not Provided\textrangle\textrangle}
    \else
        \begingroup
        \count@\@ne
        \loop
            \ifnum\count@>\tnt@totalnumber@chapters\else
```

```
                \csname tnt@mychapter\the\count@\endcsname
                \advance\count@\@ne
        \repeat
        \endgroup
    \fi
}

\long\def\backmattercontent{%
    \renewcommand*{\chaptername}{Appendix}
    \appendix
    \parindent\tnt@parindent
    \justifying
    \begingroup
    \@tempcntb\@ne
    \loop
        \ifnum\@tempcntb>\tnt@totalnumber@appendices\else
            \csname tnt@myappendix\the\@tempcntb\endcsname
            \advance\@tempcntb\@ne
    \repeat
    \endgroup
    \tnt@referencessection
    \tnt@vitapage
}
```

\ThesisContent

```
\newcommand{\ThesisContent}{%
    \begin{document}
        \frontmattercontent
        \mainmatter
        \mainmattercontent
        \backmattercontent
    \end{document}
}
```

This section sets parameters relevant for proclamation environments and displayed material.

```
\g@addto@macro\normalsize{%
    \abovedisplayskip\tnt@displayaboveskip%
    \belowdisplayskip\tnt@displaybelowskip%
    \abovedisplayshortskip\tnt@displayshortaboveskip%
    \belowdisplayshortskip\tnt@displayshortbelowskip%
}

\declaretheoremstyle[spaceabove=\tnt@abovethm,
    spacebelow=\tnt@belowthm,
    headfont=\normalfont\bfseries, headindent=0pt,
    headpunct={}, notefont=\mdseries,
    headformat={\NAME\ \NUMBER.\NOTE},
    notebraces={(}{)}, bodyfont=\normalfont\itshape,
    postheadspace=0.5em]{tntthmstyle}

\declaretheoremstyle[spaceabove=\tnt@abovethm,
    spacebelow=\tnt@belowthm,
    headfont=\normalfont\bfseries, headindent=0pt,
    headpunct={}, notefont=\mdseries,
```

```
    headformat={\NAME\ \NUMBER.\NOTE},
    notebraces={(}{)}, bodyfont=\normalfont,
    postheadspace=0.5em]{tntdefstyle}

\declaretheoremstyle[spaceabove=\tnt@abovethm,
    spacebelow=\tnt@belowthm,
    headfont=\normalfont\itshape, headindent=0pt,
    headpunct={}, notefont=\mdseries,
    headformat={\NAME\ \NUMBER.\NOTE},
    notebraces={(}{)}, bodyfont=\normalfont,
    postheadspace=0.5em]{tntremstyle}

\declaretheoremstyle[spaceabove=\tnt@abovethm,
    spacebelow=\tnt@belowthm,
    headfont=\normalfont\itshape, headindent=0pt,
    headpunct={}, notefont=\mdseries,
    headformat={\NAME. \NOTE},
    notebraces={(}{)}, bodyfont=\normalfont,
    postheadspace=0.5em]{tntconjstyle}

\declaretheorem[name=Theorem,
    refname={theorem, theorems},
    Refname={Theorem, Theorems},
    numberwithin=section,
    style=tntthmstyle]{theorem}

\declaretheorem[name=Lemma,
    refname={lemma, lemmas},
    Refname={Lemma, Lemmas},
    sibling=theorem,
    style=tntthmstyle]{lemma}

\declaretheorem[name=Corollary,
    refname={corollary, corollaries},
    Refname={Corollary, Corollaries},
    sibling=theorem,
    style=tntthmstyle]{corollary}

\declaretheorem[name=Proposition,
    refname={proposition, propositions},
    Refname={Proposition, Propositions},
    sibling=theorem,
    style=tntthmstyle]{proposition}

\declaretheorem[name=Definition,
    refname={definition, definitions},
    Refname={Definition, Definitions},
    sibling=theorem,
    style=tntdefstyle]{definition}

\declaretheorem[name=Example,
    refname={example, examples},
    Refname={Example, Examples},
    sibling=theorem,
```

```
    style=tntdefstyle]{example}

\declaretheorem[name=Fact,
    refname={fact, facts}, Refname={Fact, Facts},
    sibling=theorem, style=tntdefstyle]{fact}

\declaretheorem[name=Remark,
    refname={remark, remarks},
    Refname={Remark, Remarks},
    sibling=theorem,
    style=tntremstyle]{remark}

\declaretheorem[name=Conjecture,
    refname={conjecture, conjectures},
    Refname={Conjecture, Conjectures},
    numbered=no,
    style=tntconjstyle]{conjecture}

\declaretheorem[name=Question,
    refname={question, questions},
    Refname={Question, Questions},
    numbered=no,
    style=tntconjstyle]{question}

\declaretheorem[name=Problem,
    refname={problem, problems},
    Refname={Problem, Problems},
    numbered=no,
    style=tntconjstyle]{problem}

\declaretheorem[name=Hypothesis,
    refname={hypothesis, hypotheses},
    Refname={Hypothesis, Hypotheses},
    numbered=no,
    style=tntconjstyle]{hypothesis}

\declaretheorem[name=Conclusion,
    refname={conclusion, conclusions},
    Refname={Conclusion, Conclusions},
    numbered=no,
    style=tntconjstyle]{conclusion}

\declaretheorem[name=Methodology,
    refname={methodology, methodologies},
    Refname={Methodology, Methodologies},
    numbered=no,
    style=tntconjstyle]{methodology}
```

The following section sets the style of proof environments.

```
\newif\if@tnt@markproof
    \global\@tnt@markprooffalse
\newcommand*{\QEDhere}{%
    \hskip1em plus1em minus1em \hbox{\mathqed}

    \let\oldendalign\endalign
```

```
        \def\endalign{\oldendalign \vskip1ex plus 1ex}

    \let\oldendalignat\endalignat
        \def\endalignat{\oldendalignat \vskip1ex plus 1ex}

    \let\oldendaligned\endaligned
        \def\endaligned{\oldendaligned \vskip1ex plus 1ex}

    \let\oldendgather\endgather
        \def\endgather{\oldendgather \vskip1ex plus 1ex}

    \let\oldendmultline\endmultline
        \def\endmultline{\oldendmultline \vskip1ex plus 1ex}

    \let\oldendsplit\endsplit
        \def\endsplit{\oldendsplit \vskip1ex plus 1ex}

    \let\oldenddisplaymath\enddisplaymath
        \def\enddisplaymath{\oldenddisplaymath \vskip1ex plus 1ex}

    \let\oldendequation\endequation
        \def\endequation{\oldendequation \vskip1ex plus 1ex}

    \let\oldenddspmth\]
        \def\]{\oldenddspmth \vskip1ex plus 1ex}

    \global\@tnt@markprooftrue
}

\renewenvironment{proof}%
    {%
        \noindent\textbf{Proof.}\thickspace%
    }
    {%
        \if@tnt@markproof
            \global\@tnt@markprooffalse
            \let\endalign\oldendalign
            \let\endalignat\oldendalignat
            \let\endaligned\oldendaligned
            \let\endgather\oldendgather
            \let\endmultline\oldendmultline
            \let\endsplit\oldendsplit
            \let\enddisplaymath\oldenddisplaymath
            \let\endequation\oldendequation
            \let\]\oldenddspmth
        \else
            \hskip1em plus1em minus1em \hbox{\qedsymbol} \vskip1ex plus 1ex
        \fi
    }
```

`\SuperSize` This section provides a general sizing macro for delimeters.

```
\def\SuperSize#1#2{%
    \setbox0\hbox{\Bigg#2}
    \@tempdimb#1\ht0
    \@tempdimc#1\dp0
```

```
    \@tempdima\dimexpr\@tempdimb+\@tempdimc\relax
    \raisebox{-0.5\@tempdimc}{%
        \resizebox*{\wd0}{\@tempdima}{#2}
    }
}


\endinput
```

Appendix E

History

> Success is stumbling from failure to failure with no loss of enthusiasm.
>
> — Winston Churchill

03.06.2023 `TNThesis` version 5.1 'November' is launched

- Reintroduced the `biblatex` package option "natbib=true" so that commands such as `\citep` and `\citet` can be used
- The creation of `nullchapter.tex` has been eliminated due to consistent compilation errors; instead its contents is called directly in `TNThesis` upon the lack of any called chapter files
- The `minted` package is loaded by means of an eponymous class option rather than automatically
- The `minted` package option "cache=false" is changed to "cache" (which makes the option active) in order to prevent chronic `fvextra` package errors from appearing during compilation
- The `csquotes` package is called after the (possible) `minted` package to avoid a warning with the `lineno` package that `minted` calls

06.11.2022 `TNThesis` version 5.0 'Mike' is launched

- The name of the class file is changed to `TNThesis` and the name of the compilation file is `TNTmain`
- The problematic requirement of the `luximono` package is removed (Overleaf could not offer the package due to license requirements)
- The ability to create as many additional lists as one wishes is effected through the recreation of the maco `\SpecialListItem`
- The option to input a file to be used in the creation of an additional list is removed due to lack of interest and potential lack of consistency with the style of the standard lists

- The addition of the option defernumbers=true is effected for the biblatex package
- The macros \tnt@formattedtitleone and \tnt@formattedtitletwo are removed and replaced with the simple calls of \unvbox\tnt@finaltitleboxone and \unvcopy\tnt@finaltitleboxtwo, respectively
- Allowance is formally made for subsubsections with all the requisite changes in the table of contents
- Styling of section, subsection, and subsubsection headings is changed to be consistent with APA 7 style
- The bigstrut package is now automatically loaded
- The minted package is automatically loaded instead of the listings package
- The tikz package is loaded only by means of a documentclass option
- The inputenc package with the utf8 option is automatically loaded
- A command and control section is added to the beginning of the TNThesis class file to allow changes to be more easily effected, particularly by the user
- The macro \HyphenAllowance is eliminated due to the creation of the command and control section
- The code in TNThesis is streamlined by means of using the \define@key command of the keyval package
- The \Figure macro is relabelled \Graphic
- The Table environment is made into the macro \Table to facilitate compression, if needed

- TNThesis automatically loads the url package
- Glue is added above and beneath floats
- Warnings are issued if a student attempts to use the standard figure and table environments

08.19.2021 TNTmathTHESIS version 4.0 'Juliett' is launched

- The TNTmathTHESIS class file is now coded purely in Plain TEX and internal macros now feature the @ symbol, in keeping with good LATEX programming practices
- The package microtype is included to produce character protrusion (margin kerning) and font expansion
- The package fontenc is included with a T1 font encoding specification
- The package lmodern is included to make full use of the 8-bit T1 font encoding
- The package interval is included to facilitate the creating of half-open, half-closed intervals
- The package multirow and companion package bigdelim are included to expand the possibilities for table and array creation

- The package `centernot` is included to facilitate appropriate placement of the negation slash on certain operators
- The packages `csquotes` and `babel` (with the `american` package option) are included to facilitate the creation of more typographically accurate citations in the bibliography
- The bibliography no longer produces sources that are not cited in the work itself
- The packages `eso-pic`, `etoolbox`, `forloop`, `letltxmacro`, `xifthen`, `xparse`, `xstring`, `atbegshi`, `alphalph`, `adjmulticol`, `relsize`, and `datetime2` have been eliminated
- A dynamic title creater has been implimented that allows for `TNTmathTHESIS` to select the best-looking title from amongst several options
- The macro `\HyphenAllowance` is introduced to allow the user to decide how much he or she wants LaTeX to avoid producing end-of-line hyphens in his or her title
- The use of the command `\input` is relegated purely to `TNTmathTHESIS`
- The indefinite article "An" now introduces the header of the abstract page
- The rule of the section separation lines on the title page is reduced from 1 pt to 0.5 pt
- The macro `\Figure` and `Table` and `Listing` environments have been added to facilitate the insertion of the respective floats using the key=value system; the companion macros `\FigureSet`, `\TableSet`, and `\ListingSet` have also been added
- the macro `\Caption` has been eliminated
- The `figure`, `table`, and `listing` environments now produce output consistent with the APA 7 (est. Oct. 2019) standard
- The list of figures, list of tables, and list of listings is now automatically produced upon detection of the existence of at least one item of the respective type in the work; moreover these lists now abide by the APA 7 standard
- Float blocks have been implemented for chapters, sections, and subsections
- The table of contents no longer features the headers **Chapter** and **Appendix**; instead the chapter and appendix names are provided for each chapter and appendix
- The environments `\insfigleft`, `\insfigright`, `\instableft`, and `\instabright` have been eliminated, although the `wrapfig` package is still loaded by `TNTmathTHESIS`
- Glue has been added to the spacing above and below theorem environments
- The spacing above and below displayed material has been tightened up slightly and the values (including glue) for the macros `\abovedisplayskip`, `\abovedisplayshortskip`, `\belowdisplayskip`, and `\belowdisplayshort` have been changed to make them consistent with the spacing above and below theorem and proof environments
- The spacing after the issuing of the `\QEDhere` command has been tightened up and glue has been provided for this spacing
- The macro `\ThesisContent` has been created as a stand-in for the commands `\begin{document}`, `\frontmattercontent`, `\mainmatter`, `\mainmattercontent`, `\backmattercontent`, and `\end{document}`

- Dynamic spacing has been added between committee member names on the approval page
- Users may now compile their work without entering any information, although appropriate warnings will show up in the `log` file and in the output
- The user's manual is greatly enhanced:
  1. The introductory section *Basic Procedure* is revamped to address where to upload a thesis or dissertation once it's completed and to advise students to make a comment there concerning their work being typeset in LaTeX
  2. *Some Math Macros* is recast as *Examples of Math Into LaTeX* and expanded
  3. *Listing Your Listings* (Chapter 10) is added
  4. A section on Greek and Hebrew symbols is added to the end of Chapter 3
  5. The bomb *Seeing the interval as both half-open and half-closed* is included in Chapter 4
  6. The bolt *What happens in an environment stays in an environment* is added to Chapter 8
  7. The bolt *Setting a table* is added to Chapter 9
  8. The bang *The value of issuing a citation* is added to Chapter 14
  9. The bang *Who stole my list!* is added to Chapter 12
  10. Subsection 4.4.1 and Subsection 4.4.2 have been created in Section 4.4 with additional examples provided for the respective sizing macros; the sizing commands `\left.` and `\right.` for creating a delimiter without its companion are also discussed
  11. Section 6.3 has been partitioned into the Subsection 6.3.1, Subsection 6.3.2, and Subsection 6.3.3; Subsection 6.3.1 discusses the "built-in" operators of `amsmath`, as well as introducing the reader to the macro `\DeclareMathOperator`
  12. The bang *When not to use a list environment* in included at the end of Chapter 7
  13. The command from the `memoir` class is introduced to the user in the bang *Figures and tables float*
  14. Chapter 9 is revamped and now includes Section 9.4 *Taking Your Table to the Next Row*
  15. Chapter 11 is revamped and now includes Section 11.4 *Considerations for Math Environments* and Section 11.1 *Typographic Style in LaTeX*
  16. The bolt *Talking with a foreign accent* is added to Chapter 14
  17. Chapter 15 has been split into two sections: *Error Messages* and *Other Options for Dealing with Errors*; in the latter, the commands `\errorstopmode`, `\scrollmode`, `\nonstopmode`, and `\batchmode` are discussed, as well as several of the `\tracing` commands
  18. The glossary and index have been vastly improved for both readability, ease of access, and content

01.10.2021 `TNTmathTHESIS` version 3.3 'India' is launched

- The package `amsmath` is replaced with its extended counterpart `mathtools`, thereby correcting an oversight from previous versions of `TNTmathTHESIS`
- The macros `\mathclap`, `\smashoperator`, `\underbrace`, and `overbrace` are discussed in the manual

07.29.2020 `TNTmathTHESIS` version 3.2 'Hotel' is launched

- The package `atbegshi` is added to the preamble along with a line of code that ensures there is no page before the title page
- The `problem` environment is added to the list of theorem environments
- The theorem style `techconjstyle` is introduced specifically for the environments `conjecture, question, problem, hypothesis, conclusion, methodology` since none of these environments are numbered. Moreover, the introduction of this theorem style ensures that the period following the name in the respective label is placed immediately after the name without a space

06.12.2020 `TNTmathTHESIS` version 3.1 'Golf' is launched

- The abstract page is moved to just after the title page, per the Graduate School's request
- The macro `\QEDhere` is introduced to improve the placement of the end-of-proof box when a proof ends with a displayed environment
- The proof environment is retooled in `TNTmathTHESIS`
- For figures and tables, the recommendation for placing the label immediately after `\begin{figure}` or `\begin{table}` command is changed to after the `\caption` command to make sure that reference numbers are correct in the body of the text

04.15.2020 `TNTmathTHESIS` version 3.0 'Foxtrot' is launched

- The chart *Basic Procedure* is added to the front matter of the manual
- The page *Manual Legend* is added to the front matter of the manual
- *Preface to version 1.0* and *Preface to version 2.3* are removed and replaced with *Preface to version 3.0*
- The marginal indicators for those using `bib` files has been removed from the user's manual
- The chapter *In the Beginning…* has been added
- The chapter *Getting Started* has been renamed *…There Was LaTeX*
- The appendix *Which Editor Is Right for Me?* has been added to help students in the selection of an appropriate LaTeX editor
- The chapter *We're Way Past Introductions* has been split to allow for the creation of the chapter *Pulling Up the Rear* dedicated specifically to the back matter
- The appendix *Shaking the Packages* has been removed

- Foxtrot now makes standard the use of `biblatex` with Biber as the backend to create *all* bibliographies
- Foxtrot makes the table of contents single-spaced to allow for titles that proceed beyond a single line
- Foxtrot removes excess spacing after the titles *Dedication* and *Acknowledgments* on their respective pages
- Foxtrot numbers subsections

03.20.2020 `TNTmathTHESIS` version 2.3 ‘Echo’ is launched

- Echo corrects a bug in the `natbib` package concerning the `\setcitestyle` macro, which apparently cannot take a comma-separated list
- Echo adds `\usetikzlibrary{cd}` and `\usetikzlibrary{decorations.pathmorphing}` to the call of the `tikz` package, in order to allow for the flexibility of creating commutative diagrams
- The manual is updated with a chapter on diagrams and a chapter on mathematical writing
- Echo makes the necessary last minute changes to the approval form created by the Office of Graduate Studies
- Echo changes the macro `\ParentheticalCite` to `\CitationStyle`, wherein one can provide an argument of **round** or **authoryear** if one wants parentheses or author-year formatting, respectively, for the citations

03.09.2020 `TNTmathTHESIS` version 2.2 ‘Delta’ is launched

- Delta corrects a bug concerning a double period placed at the end of theorem names
- Delta corrects a bug concerning incorrect section numbering in appendices
- Delta introduces the environments `insfig` and `instable` for giving using more control over wrapping text around figures and tables, respectively
- The macro `\ParentheticalCite` is created for those using a .bib file who want parenthetical or textual citations; otherwise, `TNTmathTHESIS` defaults to square brackets and numbers
- The chapter *Getting Started* in the manual is broken into two chapters with the second chapter entitled *The Basic Tools*
- The section *Alignment* in the chapter *On Display* is greatly expanded to include all of the standard types of alignment from the package `amsmath`
- In light of the bug correction concerning section numbering in the appendices, the table of contents for theses and dissertations is redesigned to accentuate exactly where chapters begin and where appendices begin
- In light of the bug correction concerning section numbering in the appendices, the macro `\MyAppendices` will no longer take an argument for the table of contents title

02.28.2020 `TNTmathTHESIS` version 2.1 'Charlie' is launched

- Charlie allows for the possibility of creating a bibliography through a `bib` database file by means of the new macro `\MyBibReferences`; `TNTcontrol` is adjusted accordingly
- The approval page specifications are changed slightly for consistency with other discipline's approval pages
- The package `apacite` is added if a student chooses `\MyBibReferences` in order to automatically create APA style from the `bib` file

02.14.2020 `TNTmathTHESIS` (formerly `TECHthesis`) version 2.0 'Bravo' is launched

- The name of the class file has been changed from `TECHthesis` to `TNTmathTHESIS` to emphasize that the class file is intended primarily for the mathematics graduate students at Tennessee Tech University
- The name of the compilation file has been changed from `TECHcontrol` to `TNTcontrol` to reflect the change in the name of the corresponding class file.
- Specific error messages are introduced to `TNTmathTHESIS` to make sure students know what minimum information they need to provide for compilation
- The packages `listings`, `relsize`, and `mathrsfs` are incorporated into `TNTmathTHESIS`
- The underlying code for `TNTmathTHESIS` is supplied in the manual as an appendix (see Appendix D)
- The `TNTcontrol` program is supplied in the manual as an appendix (see Appendix C)
- The section "Front Matter: The Preliminary Pages" (Section 12.3) is changed to reflect a new ordering of items in the front matter.
- The section "Back Matter: Appendices, Bibliography, and Vita" (Chapter 14) is changed to reflect a fixed ordering of appendices before the bibliography.
- The subsection *The Table of Contents* in Section 12.3 is revised to reflect changes in the page identification of both the bibliography and appendices. In particular, neither item will have a separate title page any longer
- A *History* appendix is added to the manual
- The package `mathtools` replaces the package `amsmath`
- The sizing macros `\giant`, `\Giant`, `\goliath`, and `\Goliath` have been created for the class file
- The *Quick Reference* pages are added before the Preface in the manual (see page 2)
- The syntax `$$...$$` is replaced with `\[...\]` as the recommendation for creating displayed equations
- An index is supplied with the manual

- A glossary is supplied with the manual
- An appendix detailing how to write the individual items of a bibliography is included (see Appendix B)
- The macros `\CountCitations` and `\CountReferences` have been added to `TNTmathTHESIS`
- The macros `\InsertSpecialList` and companion `\SpecialListItem` have been added and their documentation included in Chapter 12
- The macro `\Caption` has been added for controlling the size of a lengthy caption and its documentation included in Chapter 8
- The *Beyond the Basics* chapter has been split apart into five chapters: *On Display*, *Theorems and Friends*, *Making a List*, *It Figures!*, and *Tabling Issues*
- The *Creating the Parts of Your Thesis* has been split into two chapters: *Front Matter Matters* and *We're Way Past Introductions*
- The gold-bordered display boxes have been retooled for better readibility
- The purple-bordered display boxes (i.e., bangs, bolts, and bombs) have titles attached to them for ease of locating

01.02.2020 `TECHthesis` version 1.0 'Alfa' is launched and there was much rejoicing in the land

Reference

Glossary

A B C D E F G H I J K L M N O P R S T U V W X

A

abstract

Definition A summary of the thesis or dissertation that provides the reader with an overview of what question(s) the author entertains in the thesis or dissertation and what the significant results are

See also front matter

Manual p. 160

Notes
- By rule, the abstract must not be longer than a page in length

`\addcontentsline`

Definition This macro allows the user to manually place an additional item in list-type files such as the one that generates the table of contents (extension `.toc`).

See also `\addtocontents`, `\phantomsection`

Syntax `\addcontentsline{⟨ext⟩}{⟨level⟩}{⟨content⟩}`
- `⟨ext⟩` is the extension of the file to which you wish to add `⟨content⟩`
- `⟨level⟩` determines the placement of `⟨content⟩` in the file

Example `\addcontentsline{toc}{chapter}{Dedication}` will place the title "Dedication" at the level of a chapter in the table of contents

Notes
- The macro should be called at a place in the code where the effect will be created at the proper place in the external file
- If a page number is required (e.g., in the table of contents), the macro should be prefaced with `\phantomsection` so the page number can be tagged

`\addtocontents`

Definition This macro allows the user to manually place code in certain files that are created during compilation.

See also `\addcontentsline`, fragile, `\protect`

Syntax `\addtocontents{⟨ext⟩}{⟨code⟩}`

- ⟨*ext*⟩ is the extension of the file into which ⟨*code*⟩ is inserted

Example `\addtocontents{toc}{\protect\mbox{}\protect\hrulefill\par}` places a horizontal line the length of the textblock in your table of contents

Notes
- The macro should be called at a place in the code where the effect will be created at the proper place in the external file
- Any fragile commands in ⟨*code*⟩ must be prefaced by `\protect`

`\addtodef`

Definition This command allows the user to insert code both at the very beginning and at the very end of macros that call environments.

See also `\addtocontents`, environment

Syntax `\addtodef{⟨macro⟩}{⟨beginning code⟩}{⟨end code⟩}`

- ⟨*macro*⟩ is the name of the macro into which the code will be inserted
- ⟨*beginning code*⟩ is the code that will be inserted at the beginning of the environment called by ⟨*macro*⟩
- ⟨*end code*⟩ is the code that will be inserted at the end of the environment called by ⟨*macro*⟩

Example `\addtodef{\tableofcontents}{\bfseries}{\normalfont}` will insert the command `\bfseries` at the beginning of the table of contents and the command `\normalfont` at the end of the table of contents. Effectively, this will boldface everything (including the title) in the table of contents and then return to the regular document font at the end of the table of contents.

Notes
- Since all arguments of `\addtodef` are mandatory, if you wish to, say, place code at the beginning of a macro definition, but not at the end, then you must set {⟨*end code*⟩}={}

`\addtolength`

Definition A macro that adds a specific length to an established length parameter macro.

See also `\setlength`

A B C D E F G H I J K L M N O P R S T U V W X

Syntax `\addtolength{⟨length macro⟩}{⟨length⟩}`

- ∘ *⟨length⟩* is the measurement to be added to *⟨length macro⟩*

Example `\addtolength{\textwidth}{0.5in}` will add 0.5 inches to the text width at the place where the command is called

Notes
- Useful for local expansions or contractions with certain lengths

APA style

Definition APA (American Psychological Association) style represents a particular style for composing works such as essays, reports, and articles. It's popularity stems from the APA's exacting standards for referencing sources.

See also bibliography, dissertation, thesis

Manual p. ??

Notes
- Since the Graduate School insists on this style for theses and dissertations, `TNThesis` automatically uses APA style for float and bibliographic information
- As of October 2019, APA style is in its 7th edition

appendix

Definition An appendix in a thesis or dissertation is an optional part appearing as the first item in the back matter. It is designed to be a place for information related to the work but of a more technical and ancillary nature. For example, certain large tables of data or computer codes are often placed in an appendix.

See also back matter, chapter

Manual p. 177

Notes
- Unlike chapters, appendices are lettered in sequential order rather than numbered
- Other than labelling, an appendix is formatted in exactly the same fashion as a chapter and may contain sections and even subsections
- Appendices are not a required part of a thesis or dissertation

Arabic numerals

Definition Arabic numerals are precisely the whole numbers 0,1,2,3, etc. Such numerals are used for the pagination of the main matter and back matter pages in a work such as a thesis or dissertation.

See also back matter, folio, main matter, pagination, Roman numerals

Notes
- In contrast, lowercase Roman numerals such as i, ii, iii, iv, etc. are used for front matter pages

A B C D E F G H I J K L M N O P R S T U V W X

array

Definition An `array` is a LaTeX environment that differs from a `tabular` environment in that the data in an `array` is automatically placed into math mode. The environment uses the same column specifications that the `tabular` environment uses and the data is entered in the same way, as well (using ampersand symbols (%) to separate data horizontally and line breaks (\\) to separate data vertically).

See also environment

Manual p. 133

Notes
- The `array` environment is particularly well-suited for creating matrices, although the `pmatrix`, `bmatrix`, and `vmatrix` environments from the `mathtools` package are quicker and cleaner ways to create matrices

asterisk (*)

Definition The presence of an asterisk at the end of a command name indicates that it is a variation on the "unstarred" version of the command.

See also `\newcommand`

Manual p. 46

Example The macro `\newcommand*` does the same thing as `\newcommand` with the additional feature that if the respective code contains the TeX primitive `\par` anywhere (even if automatically placed into the code by virtue of a line break), an error message will be returned (as opposed to the original `\newcommand` which allows the primitive `\par` in its argument).

Notes
- Even the line break macro `\\` has a starred version that disallows page breaks following the line break
- Another way to create a variant of a particular macro is to capitalize some of the letters in the name of the macro

B

back matter

Definition The concluding material of your thesis or dissertation. It typically includes (in order of presentation) the bibliography (i.e., the references section), appendices (if any), and the vita page.

See also appendix, bibliography, front matter, main matter, vita

Manual p. 177

Notes
- Pagination in the back matter is in Arabic numerals continuing in sequential order from where the main matter left off

A B C D E F G H I J K L M N O P R S T U V W X

badness

Definition TeX has a pretty sophisticated algorithm for determining the amount of spacing between letters and words on a given line in order to create an aesthetically pleasing appearance. In fact, each line in the printout is placed into an invisible horizontal box (called an “hbox”) for the purposes of such measurements. The “badness” of a horizontal box is an integer from 0 to 10000 that measures the quality of the spacing in the box.

See also demerit, penalty, `\pretolerance`,`\tolerance`

Manual p. 14

Example A badness of 10000 on a line is the worst case scenario in which TeX could not find a way to appropriately move the letters around to suit its spacing requirements, and so either part of the line had to spill outside of its allotted area (usually into the margin) or there simply was way too much white space on the line for the given content.

Notes
- TeX will report a badness of 10000 on a line in its log after compilation; however, a badness of 10000 is not considered a critical error that will keep TeX from compiling your document
- The package `microtype` lets you make certain fine adjustments to the interword and interletter spacing of your document

bang !

Definition A special alert in this manual that a particular fact needs to be taken into account when typing your work in LaTeX, otherwise your printout will not be as desired.

See also bolt, bomb

Manual p. 8

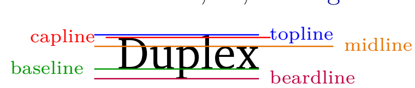
Notes
- Look at the list immediately after the table of contents (page xi) in this manual for all of the bangs, bolts, and bombs covered in the manual

baseline

Definition In a given line of text, it is the invisible line upon which the individual letters rest.

See also `\baselineskip`, `\baselinestretch`, ex, leading

Example



A B C D E F G H I J K L M N O P R S T U V W X

Notes
- In typography, the baseline is distinguished from the capline (which marks the tops of uppercase letters), the topline (which marks the tops of ascending lowercase letters like h), the midline (which marks the tops of lowercase letters like x), and the beardline (which marks the bottoms of lowercase letters like p)
- In Computer Modern font (the default for LaTeX), the capline and topline coincide

\baselineskip

Definition This TeX primitive stores the minimum vertical distance between the baselines of two adjacent lines of text.

See also baseline, `\baselinestretch`, leading, `\setlength`

Example `\setlength{\baselineskip}{15pt}` or `\baselineskip=15pt` will set the difference between adjacent baselines to be 15 pt

Notes
- `\baselineskip` may be changed by using the `\setlength` command or simply using the plain TeX "equals" formulation (as in the above example); however, this is usually ill-advised as LaTeX may automatically reset the value of `\baselineskip` at certain places in the text (e.g., at font changes)
- It is deemed preferable to change the value of `\baselinestretch` just prior to a section of your document if the leading there needs to be changed, and then restore the default value of `\baselinestretch` of 1.0 immediately afterwards

\baselinestretch

Definition A macro that scales that value of `\baselineskip`.

See also `\baselineskip`

Example `\renewcommand{\baselinestretch}{2.0}` doubles the value of `\baselineskip` where it is called and thus creates "double spacing" in the document after this point

Notes
- The default value of `\baselinestretch` is 1.0, but may be changed by using `\renewcommand`

\begingroup...\endgroup

Definition A primitive TeX environment designed so that any changes to commands made within this environment apply *only* within the environment. These commands then will have their "normal" (i.e., unchanged) values outside of the environment.

See also `\renewcommand`

Example

```
\begingroup\renewcommand{\alpha}{\beta} $\alpha$\endgroup$\alpha$
```

$\beta\alpha$

A B C D E F G H I J K L M N O P R S T U V W X

`\bfseries`

Definition A macro that allows an entire piece of your work to be typeset boldfaced.

See also `\boldmath`, `\boldsymbol`

Manual p. 28

Example

```
{\bfseries This part of the sentence} will be typeset boldfaced.
```

This part of the sentence will be typeset boldfaced.

```
\bfseries This part\normalfont\ will be typeset boldfaced.
```

This part will be typeset boldfaced.

Notes

- To ensure that other parts of your work remain unaffected by `\bfseries`, you should either set this command and the intended scope inside its own set of curly braces or use the command `\normalfont` at the place where you no longer wish `\bfseries` to apply

bibliography

Definition Also known as the references section, the bibliography is the place in the back matter where all sources cited in the body of the work are listed.

See also APA style, back matter, citation

Manual p. 178

Notes

- While publisher conventions differ considerably on requirements for bibliographies, the Graduate School stipulates that for theses and dissertations (1) APA (American Psychological Association) style must be used, (2) the sources must be listed in alphabetical order according to the first letter of the first piece of information that appears in a source's listing (usually the first letter of the first-named author's surname), and (3) only sources that are cited in the work may appear in the bibliography

`\boldmath`

Definition A command used to ensure that any mathematical notation placed within its scope (usually offset by curly braces) is typeset boldfaced.

See also `\bfseries`, `\boldsymbol`

Example

```
\boldmath$a^2 + b^2 = c^2$
```

$\boldsymbol{a^2 + b^2 = c^2}$

Notes

- Be sure to declare `\boldmath` *outside* of math mode and just *before* the mathematics to which it will apply

\boldsymbol

Definition A macro used to boldface any math type within its argument.

See also `\bfseries`, `\boldmath`

Manual p. 29

Syntax `\boldsymbol{⟨symbols⟩}`

- ⟨*symbols*⟩ consists of mathematical notation that is intended to be boldfaced

Example

```
$\boldsymbol{\sqrt{\pi}}$
```

$\boldsymbol{\sqrt{\pi}}$

Notes

- As opposed to `\boldmath`, this macro is used exclusively *inside* math mode

bolt

Definition A special piece of information in this manual that will allow you to work more efficiently in LaTeX.

See also bang, bomb

Manual p. 8

Notes

- Look at the list immediately after the table of contents (page xi) in this manual for all of the bangs, bolts, and bombs covered in the manual

bomb

Definition A special alert in this manual that a particular action will result in LaTeX failing to compile your document.

See also bang, bolt

Manual p. 8

Notes

- Look at the list immediately after the table of contents (page xi) in this manual for all of the bangs, bolts, and bombs covered in the manual

\brokenpenalty

Definition A macro that stores an integer-valued parameter that indicates to TeX how bad it should interpret having a page end with a hyphen.

See also `\clubpenalty`, `\displaywidowpenalty`, `\interlinepenalty`, `\outputpenalty`, penalty, `\widowpenalty`

Example `\brokenpenalty=10000` makes TeX extremely resistant to ending a page with a hyphen.

A B C D E F G H I J K L M N O P R S T U V W X

Notes • `\brokenpenalty` has a default value of 100

A B C D E F G H I J K L M N O P R S T U V W X

C

caption

Definition Generally speaking, a caption is a short descriptive phrase beneath a figure or above a table that describes the figure or table. Formally, a caption consists of a name (e.g., FIGURE), a number based upon the section in which the figure or table is found, a delimiter (e.g., a colon :), and a title in which the descriptive phrase is expressed.

See also float

Manual p. 112

chapter

Definition A major division in a work at a level just underneath that of a *part*.

See also section, subsection, subsubsection

Manual p. 170

Notes
- A chapter may consist of multiple sections, but there is no requirement that a chapter must have sections
- Typically, in a thesis or dissertation, *chapter* is the largest division in the work itself

citation

Definition An indication in the body of a work regarding which source listed in the bibliography is associated to a particular statement.

See also bibliography

Manual p. 188

Syntax `\cite[⟨result⟩]{⟨label⟩}`

- ⟨*result*⟩ is an optional declaration of the specific theorem, lemma, proposition, page, etc. that is being cited
- ⟨*label*⟩ is the internal identification used to allow LaTeX to connect the citation with the corresponding source listed in the bibliography

Notes
- The ⟨*result*⟩ will be printed out in the document, along with the external identification in the bibliography with which ⟨*label*⟩ is associated (typically the corresponding number in the bibliography or letters of the authors' surnames)

\cleardoublepage

Definition This LaTeX command will end the current page and cause all floats in LaTeX's buffer to be printed. In two-sided printing, it also makes the next page a recto (that is, odd-numbered) page, producing a blank page if necessary.

See also `\clearpage`, float, `\newpage`, recto page

\clearpage

Definition This LaTeX command ends the current page where it is called with the caveat that floats like figures and tables still in LaTeX's buffer will be immediately printed.

See also `\cleardoublepage`, float, `\newpage`

Notes
- `\clearpage` contrasts with the command `\newpage` in that `\newpage` will not immediately print floats in the buffer at the point it is called

\cline

Definition In LaTeX's `tabular` environment, the `\cline` command will issue a thin horizontal line that spans only a select number of columns.

See also `\hline`, `\vline`

Syntax `\cline{⟨a-b⟩}`
- ⟨a-b⟩ indicates that the horizontal line will begin with column #a and end with column #b

\clubpenalty

Definition A "club" is Donald Knuth's term for an orphan. The `\clubpenalty` macro is an integer-valued parameter between 0 and 10000 that indicates to TeX how bad it should interpret having an orphan created at the bottom of a page.

See also orphan

Example `\clubpenalty=10000` will set `\clubpenalty` to its maximum value

Notes
- The default value of `\clubpenalty` is 150
- If an orphan is still produced despite `\clubpenalty` at its maximum value, the Graduate School will require you to edit the affected paragraph(s) to ensure that this ultimately does not happen

\colorbox

Definition A box designed to color the background of the box with a specific color

See also `\fbox`, `\fcolorbox`, `\hbox`, `\mbox`, `\parbox`, `\vbox`

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
R
S
T
U
V
W
X

Manual See p. 64 for available standard color options in `TNThesis` and p. 132 for how to create different color mixtures

Syntax `\colorbox{⟨color⟩}{⟨text⟩}`

- ⟨*color*⟩ is a defined color from a loaded color package (e.g., `xcolor`) or a user-defined color using the macro `\definecolor`
- ⟨*text*⟩ is the text that will be contained inside the colored box

Example

```
\colorbox{yellow}{highlight this text}
```

highlight this text

compile

Definition The process by which TEX processes the information that you have given it and produces a printout of the work, usually as a pdf file.

Manual p. 11

Computer Modern

Definition The name of LATEX's default font. Latin Modern is an update to Computer Modern and features the option of a T1 font encoding scheme.

Notes

- A work may be typeset in other fonts, usually by loading the appropriate package in the preamble of the document.

control sequence

Definition A generic term used by Knuth himself to denote any group of characters that is executable in TEX.

See also cs, token

Example `\hspace{0.5in}` is a control sequence since it is executable; in particular, it will move what immediately follows the command half of an inch to the right

Notes

- Control sequences include both primitives and macros and are almost always prefaced by a backslash

copyright

Definition The term copyright refers to ownership of (some) ideas contained within a written piece of work, usually by virtue of being the originator of the ideas. It is a means to guard against others duplicating your ideas and claiming credit for them. By copyright law, if another person intends to use an idea of yours in a work of his or her own, he or she must give due credit to you. Furthermore, if the individual intends to use a significant portion of your work verbatim, then the person must ask for your permission first.

A B C D E F G H I J K L M N O P R S T U V W X

Manual p. ??

Notes

- Copyright law protects your work as soon as you put your ideas on paper
- If you intend to challenge another legally for violation of copyright, you will need evidence that you are the originator of the ideas and will need to have a formal copyright issued to you from the United States government

corollary

Definition In mathematics, a corollary is a relatively immediate consequence of a preceding theorem or proposition.

Manual p. 74

Notes

- Traditionally, lemmas do not have corollaries

cs

Definition An acronym for control sequence. It is often used in commands that are designed to create certain types of control sequences, as in `\csgdef` (short for "control sequence global definition") and `\csname` (short for "control sequence name").

See also control sequence, `\csgdef`, `\csuse`

`\csgdef`

Definition A primitive which defines a particular control sequence globally.

See also `\gdef`

Notes

- This macro is particularly useful when defining several control sequences at once whose names are indexed by some counter

`\csuse`

Definition A primitive which allows access to the value of a control sequence defined previously by the `\csgdef` primitive.

See also `\csgdef`, `\expandafter`

Notes

- The primitive `\expandafter` may need to be used in conjunction with `\csuse` if a counter value needs to be expanded first

(curly) braces

Definition In LaTeX commands that *require* user input will usually have such input entered inside a set of curly braces `{ }` immediately following the command name. Curly braces may also be used to limit the scope in which a particular macro applies.

A B C D E F G H I J K L M N O P R S T U V W X

See also mandatory argument, optional argument

Example [1] The macro `\CommitteeMember` in `TNThesis` takes two required arguments, the list position of the committee member and the committee member's name; so,

```
\CommitteeMember{2}{Johann Bernoulli}
```

will put Johann Bernoulli as the second committee member

[2] The syntax

```
{\bfseries This part of the sentence} is boldfaced.
```

limits the scope of the macro `\bfseries` to "This part of the sentence".

Notes
- By contrast, optional arguments are offset by square brackets `[ ]`

D

`\def`

Definition Similar in effect to LaTeX's `\newcommand`* macro, `\def` is a TeX primitive command that creates a user-defined macro.

See also `\let`, `\newcommand`

Manual p. 79

Syntax `\def`*⟨command⟩*`{`*⟨code⟩*`}`
- *⟨command⟩* is the name of the new macro (preceded by a backslash and **not** offset by curly braces)
- *⟨code⟩* is the code that will be implemented when *⟨command⟩* is called

Example

```
\def\TTU{Tennessee Tech University} \TTU
```

Tennessee Tech University

Notes
- In parallel with the unstarred version of `\newcommand`, if the primitive `\par` is used in *⟨code⟩* (even implicitly), then `\def` should be preceded by the command `\long`

delimiter

Definition A delimiter is a general term for a character that defines how far a particular operation applies. In LaTeX, it is often used to define the extent to which a given macro operates.

See also curly braces

Manual p. 47

A B C D E F G H I J K L M N O P R S T U V W X

Example In the command `\Verb+\macro{argument}+`, the plus symbol and the curly braces are both used as delimiters, as the plus symbol signifies how far the command `\Verb` applies and the curly braces signify how far the command `\macro` applies

Notes
- Curly braces represent the most used pair of delimiters in LaTeX, as they define the extent of a mandatory argument for a macro

demerit

Definition A demerit is a parameter calculated using the sum of the squares of the penalties associated with the line breaks in a paragraph and the badness of each line. It is the main value that TeX tries to minimize when determining line breaks.

See also badness, `\doublehyphendemerits`, penalty

Notes
- Demerits are assessed for each line in a paragraph and for the paragraph as a whole
- The three types of demerits for paragraphs as a whole are `\adjdemerits` for any two adjacent lines that are not "visually pleasing", `\doublehyphendemerits` for any two consecutive lines that end in a hyphen, and `\finalhyphendemerits` if the penultimate line of a paragraph ends with a hyphen

deprecated

Definition A piece of code (e.g., a package) is called deprecated if it is considered obsolete. Usually, this is either because nobody is maintaining the code or because the code was replaced by something that absorbed its functions. Commands may also become deprecated.

Example The two-letter commands `\bf`, `\em`, `\sl`, `\sf`, and `\it` from LaTeX2.09 for changing textual font are considered deprecated. In fact, `\bf` is no longer recognized in the current version of LaTeX

display

Definition The process by which a piece of mathematics or text is separated from the surrounding text and placed on its own line. Displayed material is automatically centered on its line and a vertical spacing roughly equal to a line space is placed between it and the surrounding text. It is designed for emphasis and readability of the material.

Manual p. 39

Notes
- In LaTeX any content placed in between `$$...$$` or `\[...\]` is displayed

A B C D E F G H I J K L M N O P R S T U V W X

\displaystyle

Definition A TEX primitive used to format mathematical notation within its scope (here, usually offset by either `$...$` or `\(...\)`) to a version equivalent to that which would appear in a displayed environment.

See also display

Example

```
$\displaystyle\sum_{i=1}^n a_i$ vs. $\sum_{i=1}^n a_i$
```

$\displaystyle\sum_{i=1}^n a_i$ vs. $\sum_{i=1}^n a_i$

Notes
- The previous and successive lines of text will be adjusted to accommodate the change that `\displaystyle` produces
- The use of this macro should be exercised in a thesis or dissertation only if absolutely necessary, as it can make line spacing look awkward where it is invoked

\displaywidowpenalty

Definition The additional penalty for breaking a page before the last line above a display formula.

See also `\brokenpenalty`, `\clubpenalty`, display, `\interlinepenalty`, `\outputpenalty`, penalty, `\widowpenalty`

Notes
- The default value of `\displaywidowpenalty` is 50

dissertation

Definition A dissertation is a term used for a major piece of work used to demonstrate both depth and originality of knowledge for someone seeking a doctorate degree. A dissertation generally must represent novel work on the part of the candidate.

See also thesis

Notes
- A thesis is the term used for a major piece of work by one seeking a master's degree and need not present new information

distribution

Definition A distribution is a group of ready-to-load packages along with an editor for the user interface with LATEX.

See also editor, package

Notes
- Some of the most popular distributions currently are *MikTeX*, *TEXLive*, and *ProTeXt*
- Packages not included with a particular distribution can still be downloaded to your computer, but the installation is generally a little bit more complicated

A B C D E F G H I J K L M N O P R S T U V W X

document class

Definition A declaration of the type of document being created, e.g., an article, a poster, a report. The document class is given on the first line of code in any compilable LaTeX program by using the command `\documentclass` followed by the class name offset by curly braces.

See also preamble

Manual p. 17

Example `\documentclass{article}` will implement a set of predetermined commands and parameters related to the creation of a typical article in mathematics

Notes
- Many default values of a document class can be changed by using commands such as `\setlength`, `\renewcommand`, and `\renewenvironment`

`\doublehyphendemerits`

Definition One of three special demerit parameters, it is added to a paragraph when a pair of consecutive lines of the paragraph ends in a hyphen.

See also demerit

Notes
- The default value of `\doublehyphendemerits` is 10000, but can be set as high as 100000

E

`\edef`

Definition A primitive that acts just as `\def` to define a macro, except the tokens involved are expanded at the moment the definition is *made* rather than at the moment the macro is *used*. Said another way, `\edef` allows the definition of a macro to be “permanent”, in the sense that any future changes to the constituent macros will not affect the value of the defined macro.

See also `\def`, `\let`, `\newcommand`, token, `\xdef`

Example

```
\def\State{Tennessee}%
\def\City{Cookeville,\ }%

\edef\Home{\City\State}%

\def\City{Nashville}%

\Home
```

Cookeville, Tennessee

editor

Definition In the context of LaTeX, an editor represents the software that allows you to interface with the LaTeX system. Editors typically feature an environment for typing up your code, as well as icons to facilitate important actions such as compiling your code, saving your program, etc.

Manual p. 16

Notes
- Some of the most popular editors currently are *TeXstudio*, *TeXnicCenter*, *TeXworks*, and *WinEdt*

em

Definition A unit of measurement in typography equal to the width of the uppercase letter "M" in the given font. It is a valid unit of measurement in LaTeX.

See also Computer Modern, ex, sp

Manual p. 32

Notes
- Since the value of an em depends on the font, it is the standard way of measuring width in typography
- In 12 pt normal Computer Modern font, 1 em is approximately 11.74983 pt; in 10 pt normal Computer Modern font, 1 em is exactly 10 pt

em-dash

Definition A dash that is one em wide (that is, a dash that is the width of a capital "M" in the selected font). It is typically used in a sentence when a pause longer than that afforded by a comma is desired and the material offset by the em-dash is supplementary in nature. It is created in LaTeX by juxtaposing three hyphens.

See also en-dash, hyphen

Manual p. 187

Example "My hair—at least, what little of it is left—is dark brown."

`\emergencystretch`

Definition If TeX cannot set a paragraph below the `\tolerance` level of badness, rather than make overfull boxes, it tries an extra pass "pretending" that every line has an additional `\emergencystretch` of stretchable glue. This allows the overall badness to be kept below 1000 and stops TeX from "giving up" and putting all the stretch into one line.

See also badness, glue, `\tolerance`

A B C D E F G H I J K L M N O P R S T U V W X

A B C D E F G H I J K L M N O P R S T U V W X

emphasis

Definition The emphasis macro changes text to italics if the macro is called in a normal (i.e., upright) font environment but will change text to normal font if the macro is called in an italicized font environment.

See also `\itshape`, `\normalfont`, `\slshape`, upright

Syntax `\emph{⟨text⟩}`

- `⟨text⟩` is the corresponding text to be emphasized

Example

```
\itshape This \emph{word} will typeset in normal font.
```

This word *will typeset in normal font.*

```
This \emph{word} will typeset in italicized font.
```

This *word* will typeset in italicized font.

encoding (font)

Definition Encoding is the process by which the characters of a given font are assigned numbers, so that computer programs can identify these characters.

Notes
- The three main encoding schemes in typography are T1, OpenType (OT), and TrueType (TT)
- The default encoding in TEX is referred to as OT1; it is 7-bit and contains 128 separate characters per font layout (e.g., italic font is a font layout)
- A T1-encoding (which `TNThesis` features) is 8-bit and features 256 separate characters per font layout
- Unicode, by contrast, is a scheme with the aim of creating a standardized coding of all the characters in all the world's various scripts

en-dash

Definition A dash that is half the width of an em-dash (that is, a dash that is 0.5 em wide). It is typically used to specify a range of values. It is created in LATEX by juxtaposing two hyphens.

See also em-dash, hyphen

Manual p. 187

Example "The answer is found on pp. 12–13 in the textbook."

environment

Definition A series of commands that operate together to create a type of special feature usually within a small section of the printout.

Manual p. 42

Syntax `\begin{environment name}`

⟨*code*⟩

`\end{environment name}`

- ⟨*environment name*⟩ is the name of the environment being created
- ⟨*code*⟩ is the code executed when the environment is called

Notes
- Environments with particular applicability to the creation of theses and dissertations include `align`, `align*`, `thebibliography`, `center`, `corollary`, `document`, `equation`, `equation*`, `figure`, `flushleft`, `flushright`, `justify`, `lemma`, `math`, `proposition`, `quotation`, `quote`, `proof`, `table`, `tabular`, `theorem`, `vplace`

---

ex

Definition A unit of measurement in typography equal to the height of the lowercase letter “x” in the given font. It is a valid unit of measurement in LaTeX.

See also Computer Modern, em, sp

Manual p. 32

Notes
- Since the value of an ex depends on the font, it is the standard way of measuring height in typography
- In 12 pt normal Computer Modern font, 1 ex is approximately 5.16667 pt; in 10 pt normal Computer Modern font, 1 ex is exactly 4.3055 pt

---

`\exhyphenpenalty`

Definition A primitive storing an integer parameter that indicates to TeX how bad it should interpret having a line break at an explicitly inserted hyphen.

Example `\exhyphenpenalty=10000` will make TeX highly resistant to breaking a line at the hyphen in a hyphenated word

Notes
- The default value of `\exhyphenpenalty` is 50

`\expandafter`

Definition A TeX primitive that is often used when you have a command as the argument of another command and you need to tell TeX to evaluate (or “expand”) the inner command before the outer command.

Example

```
\def\test{I want all caps!}%
\uppercase{\test}\par%
\uppercase\expandafter{\test}
```

I want all caps!
I WANT ALL CAPS!

A B C D E F G H I J K L M N O P R S T U V W X

Notes
- The command `\expandnexttwo` is a variation on `\expandafter` from the `etextools` package
- In more complicated command expressions, it may be the case that more than one `\expandafter` is necessary and possibly some `\noexpand` macros in the proper order

A B C D E F G H I J K L M N O P R S T U V W X

expansion (font)

Definition Font expansion is the process of using a wider or narrower variant of a font to make interword spacing more even in a document.

F

`\fbox`

Definition A macro with that creates a framed horizontal box whose length is given by the natural length of the argument. It is a robust alternative to `\framebox`, although `\framebox` allows optional arguments.

See also `\colorbox`, `\fcolorbox`, `\hbox`, `\mbox`, `\parbox`, `\vbox`

Syntax `\fbox{⟨text⟩}`

Syntax `\framebox[⟨width⟩][⟨position⟩]{⟨text⟩}`
- ⟨*width*⟩ is the desired width of the box
- ⟨*position*⟩ gives the position of text within the box: **c** for centered text in the box (the default), **l** for flushed left text, **r** for flushed right text, and **s** for text that is spread out in the box

Notes
- If a frame around multiple lines of text is desired, then the text needs to be first placed in the argument of a `\parbox` before then being placed in the argument of `\fbox`

`\fcolorbox`

Definition A macro with that creates a framed horizontal box where both the frame and the background of the box are colored. The length of the box is given by the natural length of the argument.

See also `\colorbox`, `\hbox`, `\mbox`, `\parbox`, `\vbox`

Manual See p. 64 for available standard color options in `TNThesis` and p. 132 for how to create different color mixtures

Syntax `\fcolorbox{⟨frame color⟩}{⟨bkgrnd color⟩}{⟨text⟩}`
- ⟨*frame color*⟩ is the color of the border of the box
- ⟨*bkgrnd color*⟩ is the color of the background of the box
- ⟨*text*⟩ is the text around which a frame will be created

Example

```
\fcolorbox{violet}{yellow}{Golden Eagles!}
```

Golden Eagles!

Notes

- If a colored frame box around multiple lines of text is desired, then the text needs to be first placed in the argument of a `\parbox` before then being placed in the argument of `\fcolorbox`
- The thickness of the border and the distance between the box content and the border can be modified beforehand by applying the `\setlength` command to the parameter macros `\fboxrule` and `\fboxsep`, respectively

fill

Definition As it relates to the ⟨*stretch*⟩ component of glue, TEX has three ways that allow for the filling of *all* available space in a box: `fil`, `fill`, and `filll`.

See also glue, skip

Manual p. 33

Example

```
This will \hskip0pt plus 1fil push these words.
```

This will push these words.

Notes

- If you have multiple glues working on the same box, then
  1. A fill of any sort always completely overrides anything without a fill
  2. Any fill with more l's in its name always completely overrides a fill with fewer l's in its name
  3. Two fills with same number of l's distribute the affected contents in proportion to their multipliers
- You can even change the multiplier in front on each type of fill in a box so as to create the desired proportion of space
- `\hfil` is equivalent to `\hskip0pt plus 1fil` and `\hfill` is equivalent to `\hskip0pt plus 1fill`

float

Definition A "float" refers to an object that LATEX will move around the surrounding text—within certain limits—in order to create a more aesthetically pleasing appearance and in order to accommodate other similar objects in the vicinity. Figures and tables fall under the heading of floats.

See also `\floatingpenalty`

Manual p. 103

Notes
- Since a figure or table is a float, it may not appear exactly where you placed it in your code, but should be relatively close
- LaTeX will ensure that a float does not appear before the place where it is referenced in the text, per norms of English composition
- If need be, the `memoir` class allows you to issue the command `\FloatBlock` at a place in your code where all floats in LaTeX's buffer should be printed before this point

`\floatingpenalty`

Definition This is a penalty that is added when a float must be split between pages.

See also float, penalty

Notes
- The total of all float penalties is given by `\insertpenalties`.

folio

Definition The term *folio* refers to the page number.

See also pagination

Notes
- A page with a "blind folio" means the page number does not actually appear on the page; the title page of your thesis or dissertation is such a page

footer

Definition The footer is a space situated underneath the textblock and just above the bottom margin of the page. It usually contains the current page number in its center, but can be repurposed for a variety of tasks.

See also header, pagination, textblock

`\footnotesize`

Definition A macro that `TNThesis` uses to set the font size of anything within its scope to be 10 point. This is the default font size for footnotes and captions in theses and dissertations.

See also `\normalsize`, `\small`

foremargin

Definition In typesetting, the foremargin is the "outside" margin. That is, when a book is opened, it is the margin on the outside of the textblock, so that it is the right margin on a recto page and the left margin on a verso page.

See also recto page, spine margin, textblock, verso page

A B C D E F G H I J K L M N O P R S T U V W X

fragile

Definition In LaTeX, commands are either fragile or robust. Fragile commands are ones that will not be properly executed when used inside the argument of a "moving" macro, that is, a macro that sends its argument to a different location in the printout than where it was originally placed in the code (`\caption` is a moving macro since it sends its argument not just to a location immediately above or underneath a figure or table but also to the list of figures or the list of tables).

See also environment, optional argument, `\protect`

Example Fragile arguments include environments, any macro with an optional argument, and line breaks `\\`

Notes
- Fragile arguments need to be prefaced with `\protect` in order to ensure correct execution

`\frenchspacing`

Definition This command forces TeX to treat spaces between words and spaces between sentences equally with regard to stretching and shrinking rules. TeX's default is to give greater priority to spacing rules between words than spacing rules between sentences.

Notes
- If `\frenchspacing` is invoked, it applies until a `\nofrenchspacing` command is issued

front matter

Definition The initial parts of a thesis or dissertation. For Tennessee Tech theses and dissertations, it consists of the following items (in order): title page, abstract page, copyright page, table of contents, , approval page, dedication page, acknowledgments page, list of figures (if appropriate), and list of tables (if appropriate).

See also abstract, back matter, copyright, main matter

Manual p. 157

Notes
- To distinguish the front matter from the rest of the work, the pagination in the front matter is often lowercase Roman numerals as opposed to Arabic numerals for the main matter and back matter

`\fussy`

Definition A LaTeX macro that sets parameters governing how much spacing between letters and words on each line of text will be allowed. In particular, `\fussy` sets `\tolerance` to be 200 with an `\emergencystretch` of 0 em.

See also `\emergencystretch`, `\midsloppy`, `\sloppy`, `\tolerance`

Notes
- The `\fussy` parameters are, in fact, the default parameters for typesetting in LaTeX

A B C D E F G H I J K L M N O P R S T U V W X

A B C D E F G H I J K L M N O P R S T U V W X

fuzz

Definition A parameter that represents the amount of overrun allowed before overfull boxes are reported.

See also `\hbox`, `\vbox`

G

`\gdef`

Definition A TEX primitive equivalent to `\global\def`. This allows definitions made outside certain environments to be valid inside those environments, as well.

See also `\def`, `\global`

`\global`

Definition This TEX primitive can preface other commands that involve setting the values of parameter macros or defining macros; for example, `\def` itself (although the abbreviated command `\gdef` is more commonly used in this case), `\setlength`, `\setcounter`, `\deflength` and `\defcounter` (from the `etoolbox` package), and `\booltrue` and `\boolfalse` (also from the `etoolbox` package). It is designed to ensure that values set by the command that immediately follows `\global` can be accessed outside the group in which the command resides.

See also `\def`, `\gdef`

glue

Definition Glue is a means for TEX to stretch or shrink the blank space in an underfull or overfull box up to a specified amount in order to prevent errors from occurring.

See also fill

Syntax ⟨*fixed part*⟩ `plus` ⟨*stretch*⟩ `minus` ⟨*shrink*⟩

- ⟨*stretch*⟩ is the maximum length beyond ⟨*fixed part*⟩ that the space may be extended
- ⟨*shrink*⟩ is the maximum length less than ⟨*fixed part*⟩ that the space may be shortened

Notes
- Any of ⟨*fixed part*⟩, ⟨*stretch*⟩, and ⟨*shrink*⟩ may be 0
- If a box is underfull, TEX expands all glue in the box which has a nonzero stretchable part, up to the amount specified in that glue; similarly, if a box is overfull, TEX shrinks all glue in the box which has a nonzero shrinkable part, up to the amount specified in that glue.

A B C D E F G H I J K L M N O P R S T U V W X

glyph

Definition A glyph is a version (incarnation) of a particular character in a given font and size.

See also encoding (font)

Example An accented letter such as "ö" is considered a glyph of the character "o".

H

`\hbox`

Definition A TEX primitive that sets up a horizontal box with width given by the natural width of its contents.

See also `\colorbox`, `\fbox`, `\fcolorbox`, `\parbox`, `\mbox`, minipage, `\vbox`

Manual p. 14

Syntax `\hbox⟨to width⟩{⟨text⟩}`

- `⟨text⟩` is the content to be placed inside the box
- `⟨to width⟩` is an optional specification of the desired width of the box

Example

```
\hbox{This text} is %
technically in an hbox.
```

This text
is technically in an hbox.

```
\hbox to 1in{This text}%
is also in an hbox.
```

This text
is also in an hbox.

header

Definition The header is a space situated above the textblock and just below the top margin of the page. It usually contains the current page number, if one chooses to place the page numbers in the upper corners of the pages. However, it can be repurposed for a variety of tasks.

See also footer, textblock

`\hline`

Definition In LATEX's `tabular` environment, the `\hline` command will issue a thin horizontal line that spans the width of the table at the place where it is called.

See also `\cline`, `\vline`

Manual p. 126

Notes
- `\hline` contrasts with `\cline` in that `\cline` is designed to span only a select number of columns

---

`\hsize`

Definition A primitive used to determine the width of an `\hbox` that holds a line of text from a document.

See also hbox

`\hss`

Definition A shorthand TEX primitive equivalent to `\hskip0pt plus 1fil minus 1fil`.

See also fill, glue, `\hbox`, skip, `\vss`

Notes
- in effect, `\hss` can be used to create a horizontal box with “infinite shrinkage” and “infinite stretch”

---

hyphen

Definition A dash used to join words to make a single, compound word, as in “egg-beater”, or to separate the syllables of a word at the end of a line.

See also em-dash, en-dash

Manual p. 187

`\hyphenpenalty`

Definition A primitive that stores an integer-valued parameter indicating to TEX how bad it should interpret having a line break at an automatically inserted hyphen.

See also `\exhyphenpenalty`, hyphen, penalty

Example `\hyphenpenalty= 10000` will make TEX highly resistant to hyphenating a word at the end of a line, which is desirable for titles

Notes
- The default value of `\hyphenpenalty` is 50

I

`\if...\else...\fi`

Definition A TEX primitive conditional environment that executes a particular code if a certain Boolean condition is met and executes another code if the Boolean condition is not met.

A B C D E F G H I J K L M N O P R S T U V W X

See also \ifthenelse

Syntax \if⟨*Boolean condition*⟩⟨*code if true*⟩\else⟨*code if false*⟩\fi

Notes
- A much more versatile and intuitive means of producing the same conditional is afforded by the **xifthen** package's macro \ifthenelse

---

\ifthenelse

Definition A macro from the **xifthen** package that executes a particular code if a certain Boolean condition is met and executes another code if the Boolean condition is not met.

See also \if. . . \else. . . \fi, \isundefined

Syntax \ifthenelse{⟨*Boolean condition*⟩}{⟨*code if true*⟩}{⟨*code if false*⟩}

Notes
- ⟨*Boolean condition*⟩ can be specified in a variety of ways using commands from the **xifthen** package, such as \isundefined, \isempty, and \equal

---

\indent

Definition Places a horizontal distance equal to that of the current value of \parindent in the printout wherever it is called in the code.

See also \noindent, \parindent, \setlength

Example Writing \setlength{\parindent}{3em} before invoking \indent will produce a distance equal to that of 3 ems wherever \indent is called in the code

---

\input

Definition A TEX primitive that inserts the contents of a file at the spot in the code where the primitive is called

Syntax \input{⟨*filename*⟩}
- ⟨*filename*⟩ is the name of the file *without* the extension

Notes
- The primitive \input differs from other commands with similar purpose in that \input has the same effect as if you would have written all the code in ⟨*filename*⟩ at the spot where \input is called; by contrast, the macro \include executes a \clearpage before and after the input of ⟨*filename*⟩

\interlinepenalty

Definition The penalty for breaking a page between the lines of a paragraph.

See also \brokenpenalty, \clubpenalty, \displaywidowpenalty, \outputpenalty, penalty, \widowpenalty

Notes
- The default value of \interlinepenalty is 0

`\isundefined`

Definition A macro from the `etoolbox` package that evaluates whether or not a given macro has been defined.

See also `\ifthenelse`

Syntax `\isundefined{⟨macro⟩}`

- `\isundefined` returns a Boolean value of *true* if `⟨macro⟩` has not be defined by the point `\isundefined` is called
- `\isundefined` returns a Boolean value of *false* if `⟨macro⟩` has been defined by that point

Example

```
\ifthenelse{\isundefined{\mymacro}}{%
My macro is not defined}{%
My macro is indeed defined}%

\def\mymacro{This is my macro!}%

\ifthenelse{\isundefined{\mymacro}}{%
My macro is not defined}{%
My macro is indeed defined}
```

My macro is not defined

My macro is indeed defined

`\itshape`

Definition A macro that allows an entire piece of your work to be typeset in italics.

See also emphasis, `\slshape`

Manual p. 28

Example

```
{\itshape This part of the sentence} will be typeset in italics.
```

This part of the sentence will be typeset in italics.

```
\itshape This part\normalfont\ will be typeset in italics.
```

This part will be typeset in italics.

Notes

- In order to ensure that other parts of your work remain unaffected by this macro, you should either offset this command and its affected text by curly braces or use the command `\normalfont` at the place you no longer wish italicized text
- Italicized font is the standard type of font for theorem environments in mathematics
- Slanted font (given by either `\textsl` or `\slshape`) is typically not as slanted as italicized font

A B C D E F G H I J K L M N O P R S T U V W X

\jobname

Definition A TeX primitive that holds the underlying file name for the current document *without* the extension.

Example If the name of your file is `AndrewsThesis.tex`, then `\jobname` will expand to `AndrewsThesis` any place that it is called

Notes
- Having such a command can be particularly useful when you need to call a related file from your `.tex` file; for example, if you also called your `.bib` file `AndrewsThesis.bib`, then you can call this file from your `.tex` file simply by typing `\jobname.bib`

\justifying

Definition A macro found in the `ragged2e` package that is designed to create (fully) justified text in the area to which it is applied; usually this area is offset by curly braces.

See also `\raggedleft`, `\raggedright`

Example

```
{\justifying This is a dummy%
sentence merely meant to%
showcase various types of%
text alignment.}
```

This is a dummy sentence merely meant to showcase various types of text alignment.

```
{\raggedright This is a dummy%
sentence merely meant to%
showcase various types of%
text alignment.}
```

This is a dummy sentence merely meant to showcase various types of text alignment.

```
{\raggedleft This is a dummy%
sentence merely meant to%
showcase various types of%
text alignment.}
```

This is a dummy sentence merely meant to showcase various types of text alignment.

Notes
- Alternatively, one can use the environment `\begin{justify}`...`\end{justify}` also found in the `ragged2e` package

K

kern

Definition In typography, kerning is the process of adjusting the space between certain pairs of letters in order to create a more visually pleasing, balanced appearance to the white space between the letters of a word.

See also ligature

Syntax `\kern`*⟨distance⟩*

- *⟨distance⟩* is the desired distance between the pair of letter blocks that `\kern` has been placed (note that if *⟨distance⟩* is negative, then the letter blocks—but perhaps not the letters themselves—will overlap)

Example Consider the word “WAR” below.

WAR

WAR

The first presentation of the word involves no kerning between the letters “W” and “A” (notice that the red line highlights the default arrangement between these letters, in that the “A” is not allowed to protrude into the box created by the “W”). However, the second presentation of the word involves kerning between the “W” and the “A”. Notice how much more balanced the intermediate white space is amongst the letters of the word.

Notes

- Usually, kerning involves shrinking the standard space between two letters

---

key

Definition In LaTeX, a key is a parameter in some optional arguments that can take on a certain set of values. The “key system” is generally expressed as ⟨*key*⟩ = ⟨*value*⟩.

See also optional argument

Example In the optional arguments for a list environment, `left = 3em` showcases the key `left` along with an assigned value (3 em) to the key; here the effect is to indent the list by 3 em from the left margin

L

`\large`

Definition A macro that creates a font size of 14 point within its scope.

See also `\Large`, `\LARGE`,`\normalsize`

Notes

- Usually the scope of this macro is offset by curly braces
- In general, the actual font size that this macro creates depends upon the document class and the initial options that are set for that class
- For theses and dissertations, the Graduate School will allow you to use this macro only in very limited places and only if absolutely necessary for readability

A B C D E F G H I J K L M N O P R S T U V W X

`\Large`

Definition A macro that creates a font size of 17 point within its scope.

See also `\large`, `\LARGE`,`\normalsize`

Notes
- Usually the scope of this macro is offset by curly braces
- In general, the actual font size that this macro creates depends upon the document class and the initial options that are set for that class
- For theses and dissertations, the Graduate School will allow you to use this macro only in very limited places and only if absolutely necessary for readability

`\LARGE`

Definition A macro that creates a font size of 20 point within its scope.

See also `\large`, `\Large`,`\normalsize`

Notes
- Usually the scope of this macro is offset by curly braces
- In general, the actual font size that this macro creates depends upon the document class and the initial options that are set for that class
- For theses and dissertations, the Graduate School will allow you to use this macro only in very limited places and only if absolutely necessary for readability

LaTeX

Definition A system of commands introduced in 1984 by Leslie Lamport as an overlay to the existing TeX program. LaTeX allows the user to create the desired content, which TeX then places in the appropriate places in the printout.

See also TeX

Manual p. 11

Notes
- The name LaTeX comes from prepending the last two letters of Leslie Lamport's last name to the word TeX

leading

Definition The vertical distance between two adjacent baselines.

See also baseline

`\leavevmode`

Definition A command that exits the code from vertical mode at the place of its calling.

Notes
- This primitive is useful for when subsequent code requires horizontal mode

A B C D E F G H I J K L M N O P R S T U V W X

lemma

Definition In mathematics, a lemma is a technical result that serves to streamline the proof of a subsequent proposition or theorem.

See also corollary, proposition, theorem

Manual p. 74

`\let`

Definition A TEX primitive that acts similarly to `\def` except its assignment takes place at the moment the ascription is *made*, whereas `\def`'s assignment takes place at the moment the new macro is *invoked*.

See also `\def`

Example

```
\def\bar{hello}%
\def\fooone{\bar}%
\let\footwo\bar%
\fooone, \footwo%

\def\bar{goodbye}%
\fooone, \footwo
```

hello, hello

goodbye, hello

Notes
- Unlike `\def`, you should not offset the macro argument of `\let` with curly braces
- The `\GlobalLetLtxMacro` is a variation on the `\let` macro from the `letltxmacro` package; it is particularly useful when you are dealing with commands that have optional arguments

ligature

Definition In typography, a ligature is a joining of a pair of letters, such as æ. LATEX automatically creates ligatures with certain pairs of letters, such as “ff” in words like “iff”.

See also kern

Manual p. 146

`\linebreak`

Definition An alternative to `\newline` for breaking a line at the point in which the macro is invoked. Unlike `\newline` however, `\linebreak` will space out the words in the part of the line preceding the command so as to fill up the entire line.

See also `\newline`

Example

```
This example shows what happens when\linebreak you use this macro.
```

This example shows what happens when
you use this macro.

`\linepenalty`

Definition A penalty value associated with each line break. It is one of the values that TEX uses in assessing the demerits of accorded to each line.

See also badness, demerit, penalty

Notes
- The default value of `\linepenalty` is 10

`\list...\endlist`

Definition A LATEX environment that creates a basic list structure where each item (called by the command `\item`) is introduced by a new paragraph.

`\long`

Definition A TEX primitive that is often used in conjunction with the command `\def` in order to create a definition that will allow blank lines in its corresponding code, e.g., blank lines as generated (implicitly or explicitly) by the command `\par`.

See also `\def`, `\newcommand`, `\par`

Notes
- Compare with the LATEX macro `\newcommand`

`\MakeTextLowercase`

Definition A macro that will reexpress the text in its argument in lowercase but leave any mathematical notation untouched.

See also `\MakeTextUppercase` indexed under “uppercase”

Syntax `\MakeTextLowercase{⟨text⟩}`

Example

```
\MakeTextLowercase{THIS, BUT NOT $A$} will be made lowercase.
```

this, but not A will be made lowercase.

Notes
- This macro from the `textcase` package is similar in effect to the TEX primitive `\lowercase` and the macro `\MakeLowercase` except it will not change any mathematical notation in its argument *unless* it is offset by curly braces

macro

Definition [1] Informally, another name for a command in LaTeX. Macros in LaTeX are always introduced by a backslash \ followed by the name of the command. Such names are case-sensitive, so that, for example, `\omega` is a different macro than `\Omega`.

[2] Formally, the term "macro" is used for commands that are built from those commands that TeX understands, referred to as primitives (e.g., `\par`). As such, any command in LaTeX is a macro.

See also control sequence, primitive

Manual p. 11

main matter

Definition The essential body of your thesis or dissertation. This is where the actual chapters of the work reside.

See also back matter, chapter, front matter

Manual p. 169

Notes
- Pagination in the main matter is in Arabic numerals beginning with "1" on the first page of the first chapter
- Each chapter in the main matter must begin on a recto (odd-numbered) page, meaning that `TNThesis` will add a blank page between chapters, if necessary

`\makeatletter...\makeatother`

Definition Some commands in LaTeX are designed to "internally" manipulate certain macros, that is, they are designed to process certain macros in an underlying style or class file. Each such internal command involves the `@` symbol somewhere in its name. Moreover, the `@` symbol is given a different category code in a style or class file than in a document (i.e., `.tex`) file. This design is not meant to simply indicate which commands are internal but also to ensure that a user does not accidentally alter an internal command somewhere in his or her code. However, this means that in order to change or use such internal commands, you must preface your corresponding arguments with the command `\makeatletter` and conclude them with `\makeatother`. This effectively lets LaTeX know that you are deliberately manipulating an internal command.

Example

```
\newcount\nothing %

\makeatletter %
\nothing=\z@ %
\makeatother %

\number\nothing
```

0

A B C D E F G H I J K L M N O P R S T U V W X

mandatory argument

Definition Certain macros in LaTeX require input from the user in order to execute their functions. Generally, one will enter such arguments immediately after the command name, each argument offset by its own set of curly braces `{ }`.

See also optional argument, token

Example The macro `\frac` needs two arguments to typeset the appropriate fraction, the first argument represents the numerator and the second argument represents the denominator. So

```
$\frac{a+b}{2}$
```

$\frac{a+b}{2}$

Notes
- Mandatory arguments stand in contrast to optional arguments that are offset by square brackets `[ ]`
- It should be noted that if mandatory arguments are *not* offset by curly braces, the command will assume that whatever (non-space) token(s) immediately follow the command are to be used for the required input; as a case in point, `$\frac ab$` will still produce $\frac{a}{b}$

`\mathclap`

Definition A macro in the `mathtools` package designed to eliminate excessive white space between an operator (usually with a subscript and no superscript or a superscript and no subscript) and adjacent mathematical notation. It places its argument in a zero-width box and centers it.

See also `\smashoperator`

Manual p. 34

Syntax `\mathclap{⟨argument⟩}`
- ⟨*argument*⟩ is the relevant subscript or superscript on the operator.

`\mbox`

Definition A macro that creates an (invisible, unbreakable) horizontal box whose length is given by the natural length of the textual argument. It is a robust alternative to `\makebox`, although `\makebox` allows optional arguments.

See also `\colorbox`, `\fbox`, `\fcolorbox`, `\hbox`, `\parbox`, `\phantom`, `\vbox`

Syntax `\mbox{⟨text⟩}`

Syntax `\makebox[⟨width⟩][⟨position⟩]{⟨text⟩}`
- ⟨*width*⟩ is the desired width of the box
- ⟨*position*⟩ gives the position of text within the box: **c** for centered text in the box (the default), **l** for flushed left text, **r** for flushed right text, and **s** for text that is spread out in the box

A B C D E F G H I J K L M N O P R S T U V W X

Notes

- Often times the command `\mbox{}` is used in places where you do not wish to typeset anything, but you need a placeholder so that LaTeX can order things properly
- You can place a piece of text in an mbox that you do not want to break across two lines

---

`\midsloppy`

Definition A special macro of the `memoir` class that sets certain parameters concerning `\tolerance` and `\emergencystretch`. It compromises between the extremes of `\sloppy` and `\fussy`.

See also `\emergencystretch`, `\fussy`, `\sloppy`, `\tolerance`

---

minipage

Definition An environment that creates a vertical box very similar to `\parbox` but with more versatility. Just as with `\parbox`, the intention of a minipage is to allow you to create a miniature document (with different parameters) within your larger work.

See also `\parbox`

Syntax `\begin{minipage}[⟨outer align⟩][⟨height⟩][⟨inner align⟩]{⟨width⟩}`

⟨*text*⟩

`\end{minipage}`

- ⟨*outer align*⟩ is one of **c** (for center, which is the default), **t** (for top), or **b** (for bottom) and represents the alignment of the environment relative to the page
- ⟨*height*⟩ is the height of the minipage
- ⟨*inner align*⟩ is one of **c** (the default), **t**, **b**, or **s** (for vertical spread)
- ⟨*width*⟩ is the width of the minipage

Notes

- Whatever changes are made inside a minipage remain confined to the minipage environment
- One big advantage that a minipage has to `\parbox` is that lists are allowed in a minipage

N

`\newcommand`

Definition A LaTeX macro similar to the TeX primitive command `\def` by which one can define new macros for use in a document.

A B C D E F G H I J K L M N O P R S T U V W X

See also `\def`

Manual p. 78

Syntax **`\newcommand`**`{⟨command⟩}{⟨code⟩}`

- `⟨command⟩` is the new macro being created (preceded by a backslash)
- `⟨code⟩` is the code that `⟨command⟩` is designed to replace

Example

```
\newcommand{\TTU}{\textbf{Tennessee Tech University}} \TTU
```

Tennessee Tech University

Notes

- The associated starred version `\newcommand*` has the same essential functionality as `\newcommand` except it does not allow the (implicit or explicit) use of the primitive `\par` in the `⟨code⟩`

`\newcounter`

Definition A macro that establishes a particular control sequence as representing a counter.

See also `\setcounter`, `\stepcounter`

Syntax **`\newcounter`**`{⟨name⟩}[⟨supercounter name⟩]`

- `⟨name⟩` is the name you wish to give to the new counter (*without* the traditional backslash in front of the name)
- `⟨supercounter name⟩` is the name of a counter that forces `⟨name⟩` to reset whenever `⟨supercounter name⟩` increments

Example

```
\newcounter{MyCounter}%
\setcounter{MyCounter}{2}%
\stepcounter{MyCounter}%

\theMyCounter
```

3

Notes

- Once the counter is established, its typical default initial value is 1, although this can vary depending upon the document class

---

`\NewDocumentCommand`

Definition A fantastic improvement upon the standard `\newcommand` macro provided by the `xparse` package.

See also `\newcommand`

Syntax **`\NewDocumentCommand`**\⟨*new macro name*⟩`{⟨input list⟩}{⟨code⟩}`

- \⟨*new macro name*⟩ is the command that you are defining
- `⟨input list⟩` is a series of **m**'s (for mandatory argument) and **O**{⟨*default*⟩}(for optional argument with ⟨*default*⟩ the corresponding value if the optional argument is not given)
- `⟨code⟩` is the code that is executed when ⟨*new macro name*⟩ is called (the order in which the **m**'s and **O**'s are listed in `⟨input list⟩` determines which variable number such an input will correspond to in the `⟨code⟩`)

A B C D E F G H I J K L M N O P R S T U V W X

Example

```
\NewDocumentCommand \mymacro %
    {m O{\bfseries} m} {%
        This is my #1 macro.%

        {#2 Are you #3?}}%

\mymacro{awesome}[\itshape]{amazed}%

\mymacro{cool new}{apoplectic}
```

This is my awesome macro. *Are you amazed?*

This is my cool new macro. **Are you apoplectic?**

Notes

- Actually, there are quite a few more options than just **m**'s and **O**'s—see pp. 2–3 of the documentation of `xparse` for more possibilities

---

`\newenvironment`

Definition A macro that allows the user to create new environments for his or her code, just as `\newcommand` allows the user to create new commands.

See also environment, newcommand

Syntax `\newenvironment{⟨name⟩}[⟨number⟩]{⟨start code⟩}{⟨end code⟩}`

- `⟨name⟩` is the name of the new environment
- `⟨number⟩` is the number of user-inputted parameters
- `⟨start code⟩` represents the code that is called when the user types `\begin{<name>}`
- `⟨end code⟩` represents the code that is called when the user types `\end{<name>}`

Notes

- There is also a companion `\renewenvironment` command that allows for the redefining of an existing environment
- Technically, `⟨start code⟩` creates the macro \⟨*name*⟩ with the corresponding code, while `⟨end code⟩` creates the macro `\end`⟨*name*⟩ with the corresponding code

---

`\newlength`

Definition A macro that establishes that a particular command will take on only length values.

See also `\setlength`

Syntax `\newlength{⟨command name⟩}`

- `⟨command name⟩` is the name of the macro (*with* the backslash) that will assume only length values

Example

```
\newlength\mylength%
\setlength{\mylength}{1.5cm}%
\the\mylength
```

42.67912pt

A B C D E F G H I J K L M N O P R S T U V W X

A B C D E F G H I J K L M N O P R S T U V W X

`\newline`

Definition A LaTeX macro used to create a line break where it is called. It inserts some glue and penalties at that point into the horizontal material, so that when the paragraph does end, a line break will occur at that point, with the short line padded with white space.

See also glue, `\linebreak`, `\par`, penalty

Manual p. 31

Notes
- Unlike `\par`, `\newline` does not end horizontal mode or end a paragraph and unlike `\linebreak`, it does not space out the words preceding it to fill out the broken line
- Often the use of the simpler macro `\\` has the same effect, although there are places where one can be used and the other cannot
- If you want to prohibit a page break right after the line break, you need to use the variant `\\*`)

`\newpage`

Definition This LaTeX command ends the current page where it is called with the caveat that floats like figures and tables are *not* cleared from LaTeX's buffer. This feature contrasts with the command `\clearpage`.

See also `\cleardoublepage`, `\clearpage`, float

`\newtheorem`

Definition This is a LaTeX macro that is used to create the heading (name and number) for proclamation statements such as theorems, propositions, and lemmas.

See also lemma, proposition, theorem, `\theoremstyle`

Manual p. 75

Syntax `\newtheorem{⟨type⟩}{⟨name⟩}[⟨numbering⟩]`

- `⟨type⟩` is the type of result (e.g., `theorem`, `lemma`, `proposition`)
- `⟨name⟩` is the corresponding appearance of the heading name in the printout
- `⟨numbering⟩` is according to what you want LaTeX to number the particular type of result (e.g., section, theorem).

Example If you want your theorem names in all caps (and boldfaced) and numbered by section, you would use the command

```
\newtheorem{theorem}{THEOREM}[section].
```

If you want your lemmas numbered in sequence with the theorems, you would write

```
\newcommand{lemma}{Lemma}[theorem].
```

Notes
- More complicated adjustments to the heading, body, and numbering of your results requires a different procedure altogether and is best done with a package like `thmtools`

`\nobreak`

Definition A Plain TEX command that is defined as `\penalty10000`, which effectively means that the place where is called is an infinitely bad place to break a line (if in horizontal mode) or break a page (if in vertical mode).

See also penalty, tie

`\noindent`

Definition Withholds the indentation of something that would have normally been indented

See also `\indent`

Notes
- Most often used at the beginning of paragraphs that you do not want to undergo any indentation

`\normalfont`

Definition This macro is used in circumstances where one wishes to revert back to the original (text) font with which the given document class came equipped. Usually, the normal font is an upright, non-boldfaced, non-italicized font.

See also `\normalsize`

Example

```
\itshape This sentence \normalfont ends with normal font.
```

This sentence ends with normal font.

`\normalsize`

Definition A macro that sets the font size to be the same as the font size originally declared in the options to the given document class, if such an option was exercised, or the default font size for the document class if no such option was exercised.

See also `\footnotesize`, `\large`, `\Large`, `\LARGE`, `\small`

`\notag`

Definition A command which is often issued on a particular line in an equation environment (involving multiple equations) to suppress the numbering of that line in the output.

See also `\tag`

A B C D E F G H I J K L M N O P R S T U V W X

Example	`\begin{equation}%`
	`A = \{n \in \mathbb{N} %`
	`    \mid n>1\}%`
	`\end{equation}`
		$A = \{n \in \mathbb{N} \mid n > 1\} \qquad \text{(E.1)}$

	`\begin{equation}%`
	`A = \{n \in \mathbb{N} %`
	`    \mid n>1\}\notag %`
	`\end{equation}`
		$A = \{n \in \mathbb{N} \mid n > 1\}$

Notes
- If no tag is desired on any line of output, you can use the starred version of the particular environment, e.g., `align*`

O

octothorp(#)

Definition In TEX the octothorp symbol is used to indicate variables (i.e., parameter tokens) within the scope of a `\def` or `\newcommand` command, amongst others.

See also `\def`, `\newcommand`, `\NewDocumentCommand`, `\newenvironment`, token

Manual p. 80

Example

```
\newcommand{\Ali}[2]{%
    Float like a #1, sting like a #2}%

\Ali{butterfly}{bee}
```

Float like a butterfly, sting like a bee

Here the `[2]` immediately following the macro name `\Ali` indicates that `\Ali` will take 2 mandatory arguments (each offset by its own set of curly braces). The use of `##1` in the code indicates the place(s) where the first mandatory argument will go and the use of `##2` in the code indicates the place(s) where the second mandatory argument will go.

Notes
- The use of a double octothorp (e.g., `##1`) represents a variable within the scope of a `\newenvironment` or within the scope of a `\newcommand` inside another `\newcommand`

optional argument

Definition In LATEX some commands come equipped with the ability to input additional information as part of the command, if one would wish. Such optional arguments in LATEX are always given inside square brackets `[ ]` immediately following the command name and most often preceding the input of any mandatory arguments, which are always given inside curly braces { }.

See also mandatory argument

Example The `\cite` command comes equipped with the option to indicate the designation of the specific result being cited. For example, `\cite[Theorem 1.2]{RJH}` will print [⟨*number*⟩, Theorem 1.2] at the place where the command is called. Here ⟨*number*⟩ is the associated number to the source in the bibliograpy that was given the label RJH (see page **??** for more information on how labels factor into the creation of a citation). However, given that the information concerning the specific result being cited was optional, for the `\cite` command it turns out that `\cite{RJH}` will simply print [⟨*number*⟩].

orphan

Definition In composition, an orphan occurs when the first line of a paragraph occurs on the last line of a page. This is considered so aesthetically displeasing that a rewrite is considered mandatory to keep this from happening.

See also `\clubpenalty`, widow

Notes
- TeX has a system in place through `\clubpenalty` to make the compilation "allergic" to orphans with varying levels of aversion. However, there is no way to guarantee that TeX will never create an orphan, so be on the lookout.

orthogonal

Definition Two commands in LaTeX are considered to be orthogonal if the result that the two together produce is independent of the order in which the commands are called.

Example `\large` and `\bfseries` are orthogonal since `\bfseries\large`⟨*text*⟩ will produce the same result as `\large\bfseries`⟨*text*⟩:

```
compare {\bfseries\large this} and {\large\bfseries this.}
```

compare **this** and **this.**

`\outputpenalty`

Definition The value of the penalty at the current page break, or 10000 if the break was not at a penalty.

See also `\brokenpenalty`, `\clubpenalty`, `\displaywidowpenalty`, `\interlinepenalty`, penalty, `\widowpenalty`

`\overbrace`

Definition A macro designed to put a curly brace above several pieces of mathematical notation along with a possible descriptor of the pieces.

See also `\underbrace`

A B C D E F G H I J K L M N O P R S T U V W X

Syntax `\overbrace{⟨collection⟩}[⟨^{descriptor}⟩]`

- ⟨*collection*⟩ is the relevant set of notation
- ⟨*^{descriptor}*⟩ is the optional descriptor of the set

Example

```
$\overbrace{f \circ f \circ\cdots\circ f}^{n\ \text{functions}}$
```

$$\overbrace{f \circ f \circ\cdots\circ f}^{n\ \text{functions}}$$

`\overfull`

Definition An hbox is considered to be overfull if TEX's spacing requirements could not be satisfied with the given content inside the hbox. As such, the content spills out of its own hbox.

See also badness, `\hbox`, underfull

Manual p. 14

Example

```
\fbox{\hbox to 1cm{This hbox is overfull, as you can see.}}
```

This hbox is overfull, as you can see.

Notes

- An overfull hbox on a line usually indicates that content has spilled into the margin; for theses and dissertations, this problem must then be rectified
- Overfull hboxes are reported in TEX's log, but it is not considered to be a critical error that will prevent TEX from compiling

P

package

Definition A package in LATEX is an add-on program that allows for the use of additional macros in your program. Packages are always loaded in the preamble.

Manual p. 11

Syntax `\usepackage{⟨name⟩}`

- ⟨*name*⟩ is the name of the respective package

Notes

- Class and style files use the synonymous `\RequirePackage{⟨name⟩}` to call packages
- Most packages can be found at ctan.org, although it is likely that the distribution being used (*MikTeX*, for example) already has the needed package ready to be loaded

A B C D E F G H I J K L M N O P R S T U V W X

`\pagestyle`

Definition A macro from the `memoir` class that can be used anywhere in a document to set a certain style for all subsequent pages after the command is issued.

See also footer, header

Syntax `\pagestyle{⟨style name⟩}`

- ◦ *⟨style name⟩* is the name of a preset or user-created style for pages, e.g., **empty** (which will make the headers and footers empty) and **plain** (which will make the header empty and the page number will be centered in the footer)

Notes
- A custom page style may be created using the command `\makepagestyle`
- If a style change only to a particular page is desired, the companion macro `\thispagestyle` can be used instead at the place in the code corresponding to the respective page

pagination

Definition The style of numbering the pages in a work.

See also Arabic numerals, folio, Roman numerals

Notes
- In a thesis or dissertation, front matter pages are typically numbered in lowercase Roman numerals, while main matter and back matter pages are numbered sequentially in Arabic numerals
- The title page, while technically page number i, does not have its page number displayed

`\par`

Definition A TeX primitive command typically used to indicate that the end of a paragraph has been reached and a new paragraph needs to begin. It ends horizontal mode precisely where it is called. TeX automatically inserts this command at the end of a line of textual input if there is more than a one-line space between this line of input and the next line of textual input.

See also `\linebreak`, `\newline`

Notes
- The significance of `\par` is found in the fact that TeX's typesetting is done paragraph by paragraph, and so the use of `\par` tells TeX to execute its algorithms (e.g., to determine letter spacing, hyphenation, etc.) to the previous paragraph

`\parbox`

Definition A LaTeX command that creates an invisible vertical box containing multiple lines of text for the purpose of applying a set of macros that affect only the text within the box.

See also \colorbox,\fbox, \fcolorbox, \hbox, \mbox, minipage, \vbox

Syntax **\parbox**[⟨*outer align*⟩][⟨*height*⟩][⟨*inner align*⟩]{⟨*width*⟩}{⟨*text*⟩}

- ⟨*outer align*⟩ is one of **c** (for center, which is the default), **t** (for top), or **b** (for bottom) and represents the alignment of the box relative to the page
- ⟨*height*⟩ is the height of the box
- ⟨*inner align*⟩ is one of **c**, **t**, **b**, or **s** (for vertical spread)
- ⟨*width*⟩ is the width of the box
- ⟨*text*⟩ is the text that the box will contain

Notes
- `\parbox` and the `minipage` environment have precisely the same list of mandatory and optional arguments
- Just like `minipage`, text if free to wrap around just as it would in the document at large (this contrasts with the situation for `\hbox`)
- Unfortunately, there are a number of limitations to using a `\parbox` (e.g., the text in a `\parbox` cannot contain lists) and, as such, for general use in a LaTeX document, the `minipage` environment is encouraged

\parfillskip

Definition This TeX primitive controls how much horizontal space will remain between the end of a paragraph and the right margin.

See also fill, \parskip, skip

Notes
- The default value of `\parfillskip` is 0 pt `plus 1fill`, but it can be changed through the use of the `\setlength` macro

\parindent

Definition This TeX primitive stores the amount of paragraph indentation for a section of a document.

See also \indent, \noindent

Notes
- `\parindent` can be changed by using the command `\setlength`
- In English composition, paragraphs are typically indented by 3 ems, which is roughly 0.5 inches
- Bear in mind that `\parindent` only applies locally, so if you are inputting separate files into a master file, you will need to set the value of `\parindent` in each file

\parshape

Definition A TeX primitive designed to handle unusual left and right indentations of paragraphs, e.g., to give a paragraph a certain geometrical shape.

See also \indent, \vbox

A B C D E F G H I J K L M N O P R S T U V W X

Syntax **\parshape**⟨*number*⟩
⟨*left indent 1*⟩ ⟨*line width 1*⟩
⟨*left indent 2*⟩ ⟨*line width 2*⟩
⋮
⟨*left indent n*⟩ ⟨*line width n*⟩
⟨*text*⟩

- ⟨*number*⟩ is the number of lines in the desired paragraph
- ⟨*left indent i*⟩ and ⟨*line width i*⟩ are the desired left indentation and line width, respectively, on line *i*, where *i*=1, 2, 3, . . . , ⟨*number*⟩
- ⟨*text*⟩ is the actual text for the paragraph

Notes
- It is often recommended to encase the entire system (including ⟨*text*⟩) in a `\vbox` so that subsequent lines are not affected
- The use of `\parshape` is precisely how the "inverse Mayan pyramid" effect is achieved with respect to the title of your thesis or dissertation

\parskip

Definition This TEX primitive controls how much vertical space *beyond the normal line drop* will exist between paragraphs in your work.

See also leading, skip

Notes
- The default value of `\parskip` is 0 pt, but it can be changed through the use of the `\setlength` macro

penalty

Definition [1] Penalties are one of the types of values that TEX tries to minimize when line- or page-breaking. Some penalties are built into TEX and inserted automatically.

[2] The command `\penalty` can be assigned any integer value (although usually between -10000 and 10000) and placed anywhere in the text by the user. This value is then the "aesthetic cost" of breaking the line at that place.

See also badness, demerit

Syntax **\penalty**⟨*integer*⟩

- ⟨*integer*⟩ is the desired value of the penalty; it is typically an integer between -10000 and 10000

Example

```
Notice that the line %
\penalty-10000 is broken %
precisely where I indicated %
a penalty value of -10000.
```

Notice that the line
is broken precisely where I indicated a penalty value of -10000.

A B C D E F G H I J K L M N O P R S T U V W X

Notes
- The default value of `\linepenalty` is 10, meaning that 10 is added to the badness of each line within a paragraph
- Increasing the value of `\linepenalty` will make TeX try to have fewer lines in each paragraph
- A `\penalty` value of 10000 inhibits a break at the point the command is called while a `\penalty` value of -10000 forces a break at the point the command is called

---

`\phantom`

Definition A macro that creates horizontal and vertical space equal to that of its argument without actually printing the argument.

See also `\mbox`

Syntax `\phantom{⟨text⟩}`

Example

```
$\sqrt{\phantom{abcde}}$
```

$\sqrt{}$

`\phantomsection`

Definition This command is the way `TNThesis` can identify certain parts of your work (e.g., the approval page) to place into the table of contents that would not have naturally been placed there due to the lack of an introducing command such as `\chapter` or `\section`.

See also `\addcontentsline`

Notes
- The `\phantomsection` command should generally be placed in the code as soon as the corresponding part is introduced

---

preamble

Definition The code of a LaTeX document written between the initial `\documentclass` command and the subsequent `\begin{document}` command.

Manual p. 19

Notes
- Commands in the preamble are intended to apply to the entire document and are initialized before TeX begins the typesetting process at the `\begin{document}` command.

A B C D E F G H I J K L M N O P R S T U V W X

`\pretolerance`

Definition An integer-valued parameter that can range from -1 to 10000 and is used as part of TEX's line-breaking algorithm. If `\pretolerance` is greater than or equal to 0, TEX first attempts to break up a paragraph without hyphenation by minimizing the badness of each line. If TEX can break up a paragraph so that none of the lines have badness greater than `\pretolerance`, then TEX accepts it. If this is not possible, TEX will attempt to break up the paragraph with the allowance of hyphenated words. In this second attempt, `\tolerance` is used instead of `\pretolerance`. If TEX is still unsuccessful, an error is reported in the log file. If `\pretolerance` is -1, TEX bypasses the first attempt at breaking a paragraph without hyphenation.

See also badness, `\tolerance`

primitive

Definition A primitive is one of the about 300 commands that TEX itself can recognize. All other commands are called macros and are built from primitives. Both Plain TEX and LATEX introduce macros on top of the TEX primitives in order to greatly expand the capabilities of the entire system.

See also macro

Example `\relax` is a primitive while `\colorbox` is a macro

proclamation

Definition A generic term that refers to any "theorem-like" environment such as a lemma, a corollary, a proposition, and of course a theorem. Proclamations are introduced by `\begin {⟨name⟩}` and concluded by `\end{⟨name⟩}`, where ⟨*name*⟩ is typically one of the following: **lemma**, **corollary**, **proposition**, or **theorem**.

See also corollary, lemma, proposition, theorem

proposition

Definition In mathematics, a proposition represents an independent result of lesser status than a theorem. Usually only a handful of results in a piece of mathematical work are designated theorems while the remaining independent results are designated propositions.

See also corollary, lemma, theorem

Manual p. 74

\protect

Definition A command used to ensure that fragile macros that are "moved" (e.g., passed to an external file like the table of contents file) can be correctly executed when the entire document is compiled. The command is designed to ensure that LaTeX avoids expansion without execution. In effect, `\protect` tells LaTeX not to expand the macros that follow it so that LaTeX can execute a proper expansion with execution at the destination.

See also `\expandafter`, fragile, robust

Notes
- `\protect` only applies to the command immediately following it, so multiple `\protect`'s may be needed in the same argument.

protrusion

Definition In typography, a (character) protrusion occurs when part or all of a character extends into a place that it is not supposed to be. Usually, this happens when part of a character spills into a margin.

See also kern

Notes
- The `microtype` package is designed to allow some very subtle character protrusion in order to (ironically) create sharper-looking edges between the textblock and the margins; by doing this, lines look fuller and tighter, as well

\providecommand

Definition A LaTeX macro similar to the macro `\newcommmand` except it will create the corresponding macro only if it is not already defined by the given document class or a loaded package. Otherwise, LaTeX will simply ignore the (re)definition.

See also `\newcommand`, `\par`

Syntax `\providecommand{⟨command⟩}{⟨code⟩}`
- ⟨*command*⟩ is the new macro being created (preceded by a backslash)
- ⟨*code*⟩ is the code that ⟨*command*⟩ is designed to replace

Notes
- The associated starred version `\providecommand*` is similar to `\newcommand*` in that it does not allow (implicitly or explicitly) the use of the primitive `\par` in the ⟨*code*⟩

R

\raggedleft

Definition A macro that *right* aligns text within its scope.

See also `\justifying`, `\raggedright`

Example

```
\raggedleft This sentence will be right aligned.
```

This sentence will be right aligned.

Notes

- The companion environment is given by `\begin{flushright}`...`\end{flushright}`
- The affected text is usually offset by curly braces to ensure other parts of the work remain unaffected.

\raggedright

Definition A macro that *left* aligns text within its scope.

See also `\justifying`, `\raggedleft`

Example

```
\raggedright This sentence will be left aligned.
```

This sentence will be left aligned.

Notes

- The companion environment is given by `\begin{flushleft}`...`\end{flushleft}`
- The affected text is usually offset by curly braces to ensure other parts of the work remain unaffected.

recto page

Definition In typesetting, a recto page is on the right side when the book is laid open facing the reader. Recto pages are always odd-numbered.

See also verso page

Notes

- Each page in a work is either a recto page or a verso page

reference

Definition A reference is an appeal to a part of a work elsewhere in the work.

See also citation

Manual p. 43

Syntax `\ref{⟨label name⟩}`

- `⟨label name⟩` occurs in the corresponding macro `\label{⟨label name⟩}` that is placed in the code at the location to which the reader will be directed

Example The statement “As in Theorem 1.2.3, we use the quadratic formula to show that...” makes a reference to Theorem 1.2.3, presumably established elsewhere in the work itself

Notes

- A reference may also be to a chapter, a section, etc.
- By contrast, a citation is an appeal to material *outside* of a work

A B C D E F G H I J K L M N O P R S T U V W X

\refstepcounter

Definition A macro that increments a given counter by 1 whenever it is called.

See also `\newcounter`, `\ref`, `\stepcounter`

Syntax `\refstepcounter{⟨counter name⟩}`

- ◦ *⟨counter name⟩* is the name of the counter *without* any backslash in front of the name

Notes
- Like`\stepcounter`, `\refstepcounter` will increment *⟨counter name⟩* by 1 and resets the value of any counter numbered *within* *⟨counter name⟩* (see the optional argument of `\newcounter`) whenever it is called
- Unlike `\stepcounter`, `\refstepcounter` defines the current value of `\ref` to the value of *⟨counter name⟩*; this is particularly useful when the numbering of tables and figures needs to reset after a chapter or section change

\relax

Definition A command whose only function is to indicate where the expansion of some previously called macro should stop.

See also `\expandafter`, glue

Notes
- `\relax` is often used in conjunction with the command `\expandafter`, certain length and numerical calculations (particularly those using the primitives `\dimexpr` and `\numexpr`), and certain glue expressions whenever there is a danger that one of these commands may assume that a subsequent command is part of the argument.

\renewcommand

Definition A LaTeX macro designed to replace an existing command with a new meaning.

See also `\newcommand`, `\par`

Syntax `\renewcommand{⟨command⟩}{⟨code⟩}`

- ◦ *⟨command⟩* is the existing macro
- ◦ *⟨code⟩* is the new code that *⟨command⟩* will stand for when *⟨command⟩* is called after the point of its redefintion

Example

```
$\sigma$ \renewcommand{\sigma}{\alpha} $\sigma$
```

σ α

Notes
- Typically `\renewcommand` is implemented in the preamble
- The starred version `\renewcommand*`has the same functionality as `\renewcommand` except it does not allow (implicitly or explicitly) the primitive `\par` in the *⟨code⟩*
- Unless you know what you're doing, this is a macro that is best left untouched, as unintended consequences may ensue

A B C D E F G H I J K L M N O P R S T U V W X

`\resizebox`

Definition A standard LaTeX macro that enables the resizing of the natural box that a particular piece of text is placed in.

Syntax `\resizebox{⟨width⟩}{⟨height⟩}{⟨text⟩}`

- ⟨*width*⟩ is the desired width of the new box
- ⟨*height*⟩ is the desired height of the new box
- ⟨*text*⟩ is the text that will fill the new box

Example

```
\fbox{T} \fbox{\resizebox{1cm}{1cm}{T}}
```

T T

Notes

- It is incredibly useful for typesetting expressions that need to be much smaller or larger than would normally be allowed by the standard font sizes of a given class
- Instead of ⟨*height*⟩, you may use `!` if you wish to keep the aspect ratio of ⟨*text*⟩

`\rmfamily`

Definition A macro that allows an entire piece of your work to be typeset in roman font (usually, the normal, or default, font of the class).

See also `\normalfont`, upright font

Example

```
\rmfamily This sentence will be typeset in roman font.
```

This sentence will be typeset in roman font.

Notes

- To ensure that other parts of your work remain unaffected by this, you should either offset this command and the affected text by curly braces or call a different font macro at the point you wish to end roman font
- More often than not, you would use this command to restore the rest of your work to the normal font after using a font macro such as `\bfseries` or `\itshape`

robust

Definition In LaTeX, commands are either fragile or robust. A robust command is one that will be properly executed even when used inside "moving" macros (e.g., `\caption` or `\footnote`).

See also fragile, `\protect`

Example Commands such as `\textit` that change the size and style of font are robust. However, commands such as `\begin` and `\end` that open and close environments are not robust.

Notes

- Robust arguments do not need to be protected

A B C D E F G H I J K L M N O P R S T U V W X

Roman numerals

Definition A Roman numeral is a number expressed according to the system developed by the ancient Romans. Such numbers have both a lowercase and uppercase version. They are based upon the notion of using certain “anchor” values as I (1), V (5), X (10), L (50), C (100), etc. A number is then given by prepending or appending these anchor values to the number’s closest anchor value so the difference (if prepending) or sum (if appending) of these anchor values is the value of the number. For example, the Roman numeral XIV is the number 14 since X (10) is the closest anchor value to 14 and the appending of IV indicates that 4 is to be added to 10 (notice that IV is the number 4 since the anchor value of 4 is V and I is prepended to V indicating that 1 is to be subtracted from 5).

See also Arabic numerals

Notes
- Lowercase Roman numerals such as i, ii, iii, iv, v, etc. are used to paginate the front matter of a thesis or dissertation

`\rule`

Definition This macro creates a horizontal or vertical line in the printout at the location its called in the code.

Syntax `\rule[⟨raise⟩]{⟨length⟩}{⟨width⟩}`
- ◦ `⟨raise⟩` specifies how far up (if positive) or down (if negative) from the baseline you want the line to be

S

sans serif font

Definition In typography, a serif is a short appendage off of one of the main strokes of a letter. As such, a letter which is “sans serif” is one without any such appendages.

See also `\sffamily`

Manual p. 28

Example


Notes
- All major font families have a sans serif variant
- LaTeX also features the command `\mathsf` for typesetting mathematical notation in sans serif font

A B C D E F G H I J K L M N O P R S T U V W X

scaled point (sp)

Definition A unit of measurement exclusive to LaTeX. There are $2^{16} = 65536$ sp in 1 pt.

See also em, ex

Manual p. 32

Notes
- The scaled point is by far the smallest unit of measurement featured in TeX
- TeX converts all measurements into integer multiples of scaled points
- 1 sp is approximately 0.0000002111359 inches or about 5 nanometers, which represents the accuracy of measurements in LaTeX
- The largest measurement that can be created in TeX is $2^{30} - 1$ scaled points, or about 5.7583174 meters

`\scshape`

Definition A macro that allows an entire piece of your work to be typeset in small caps.

Manual p. 28

Example

```
{\scshape This part of the sentence} will be typeset in small caps.
```

THIS PART OF THE SENTENCE will be typeset in small caps.

```
\scshape This part\normalfont\ will be typeset in small caps.
```

THIS PART will be typeset in small caps.

Notes
- In order to ensure that other parts of your work remain unaffected by this command, you should either offset this command and the affected text inside curly braces or use the `\normalfont` command at the place where you no longer wish small caps

section

Definition [1] A major partition of a chapter at the level just underneath that of *chapter* itself. If a chapter has sections, the sections together comprise the chapter, save some possible minor introductory remarks at the beginning of the chapter.

[2] A generic term in LaTeX that refers to any literary division of a work. In particular, the term "section" in this context may refer to any of the following (in order of highest to lowest level): *book, part, chapter, section, subsection, subsubsection, paragraph, subparagraph.*

See also chapter, subsection, subsubsection

Manual p. 171

Notes
- By rules of English composition, a chapter cannot have only one section

A B C D E F G H I J K L M N O P R S T U V W X

A B C D E F G H I J K L M N O P R S T U V W X

`\setcounter`

Definition A macro that establishes a certain starting value for a counter previously established with `\newcounter`.

See also `\newcounter`

Syntax `\setcounter{⟨counter name⟩}{⟨value⟩}`

- ◦ *⟨value⟩* is the new starting value that you wish to give to *⟨counter name⟩*

Notes
- In most document classes, the default starting value for a counter is 1
- Be aware that, in some cases, TEX may increment the value of the counter before it is used

`\setlength`

Definition A macro that allows you to change the length of a measurement stored in a parameter macro.

See also `\addtolength`, `\newlength`

Syntax `\setlength{⟨command⟩}{⟨length⟩}`

- ◦ *⟨command⟩* is the LATEX command that stores the length you wish to change
- ◦ *⟨length⟩* is the new length that you wish to assign to *⟨command⟩*

Example `\setlength{\leftmargin}{2in}` will change the left-hand margin of each page to 2 inches when placed in the preamble

`\settowidth`

Definition A macro that sets the value of a parameter macro (previously established by `\newlength`) to the width of the argument.

See also `\newlength`, `\setlength`

Syntax `\settowidth{⟨parameter macro⟩}{⟨text⟩}`

Example

```
\newlength\MyWidth%

\settowidth{\MyWidth}{%
    How long is this text?}
\the\MyWidth
```

96.22227pt

Notes
- Actually, *⟨text⟩* may be a box, in which case `\settowidth` gives the width of the box
- There are also companion macros `\settoheight` and `\settodepth` with similar syntax

`\sffamily`

Definition A macro that allows an entire piece of your work to be typeset in sans serif font.

See also sans serif

Manual p. 28

Example

```
{\sffamily This part of the sentence} will be typeset sans serif.
```

This part of the sentence will be typeset sans serif.

```
\sffamily This part\normalfont\ will be typeset sans serif.
```

This part will be typeset sans serif.

Notes

- In order to ensure that other parts of your work remain unaffected by this macro, you usually should set either offset this command and its affected text by curly braces or use the `\normalfont` command at the place you no longer wish sans serif font

`\shipout`

Definition A TEX primitive command that sends the contents of a box to the `dvi` file for printing.

skip

Definition Refers to an immediate injection of a fixed amount of space where the corresponding macro is called. LATEX features a fair number of commands that can fall in this category.

See also `\baselineskip`, `\parskip`

Example

```
The command \vskip 0.25in %
creates a downward vertical %
jump of 0.25 inches at the %
point where it is called in %
the code.
```

The command

creates a downward vertical jump of 0.25 inches at the point where it is called in the code.

Notes

- The length measurement associated with a skip may be negative if one wishes to jump in the opposite direction

`\sloppy`

Definition A LATEX macro that sets parameters governing how much spacing between letters and words on each line of text will be allowed. In particular, `\sloppy` sets `\tolerance` to be 9999 with an `\emergencystretch` of 3 em.

See also `\emergencystretch`, `\fussy`, `\midsloppy`, `\tolerance`

A B C D E F G H I J K L M N O P R S T U V W X

`\slshape`

Definition A macro that allows an entire piece of your work to be typeset slanted.

See also `\emphasis`, `\itshape`

Manual p. 28

Example

```
{\slshape This part of the sentence} will be typeset slanted.
```

This part of the sentence will be typeset slanted.

```
\slshape This part\normalfont\ will be typeset slanted.
```

This part will be typeset slanted.

Notes
- In order to ensure that other parts of your work remain unaffected by this macro, you should either offset this command and its affected text by curly braces or use the command `\normalfont` at the place you no longer wish slanted text

`\small`

Definition A macro that creates a font size of 11 point within its scope.

See also `\footnotesize`, `\normalsize`

Notes
- Usually the scope of this macro is offset by curly braces
- In general, the actual font size that this macro creates depends upon the document class and the initial options that are set for that class

`\smashoperator`

Definition A macro in the **`mathtools`** package designed to eliminate excessive white space between an operator and adjacent mathematical notation.

See also `\mathclap`

Manual p. 34

Syntax **`\smashoperator`**`[⟨position⟩]{⟨operator⟩}`

- ◦ *⟨operator⟩* is the relevant mathematical operator *with limits*
- ◦ *⟨position⟩* specifies from which side of the operator excess white space is to be eliminated (that is, smashed); *⟨position⟩* values are as follows: **l** for left, **r** for right, **lr** (default) for left and right

Notes
- `\smashoperator` is especially useful when ignorance of the width of *both* the subscript and superscript of the operator is desired

spine margin

Definition In typesetting, the spine margin is the "inside" margin. That is, when a book is opened, it is the margin on the inside of the textblock, so that it is the left margin on a recto page and the right margin on a verso page.

See also recto page, textblock, verso page

spurious

Definition In LaTeX the term "spurious" is used to identify characters that inadvertently mess with a command. It is most often used to refer to blank spaces that are accidentally put inside a mandatory argument and ruin the intended function of the corresponding macro.

See also mandatory argument

Notes
- In order to avoid problems created by spurious spaces, you should use the "comment out" symbol % at the end of a line of input if the next line is part of the same argument.

\stepcounter

Definition A command that increments the value of a given counter by 1.

See also \newcounter, \refstepcounter

Syntax `\stepcounter{⟨counter name⟩}`
- ⟨*counter name*⟩ is the name of the counter (*without* any backslash in front of the name)

\strut

Definition This is a rule with no width and a total height given by the value of `\baselineskip`. It is used to guarantee that an element on a page has a certain minimum total height. In LaTeX, a strut is defined to be `\rule[-0.3\baselineskip]{0pt}{\baselineskip}`.

See also \baselineskip, \rule

subsection

Definition A partition of a section at the level underneath that of *section* but above that of a *subsubsection*.

See also chapter, section, subsubsection

Manual p. 173

Notes
- By rule, a section cannot have just one subsection
- `TNThesis` limits parts of the work identified in the table of contents to the subsubsection-level or higher

A B C D E F G H I J K L M N O P R S T U V W X

subsubsection

Definition A partition of a subsection at the level just underneath that of *subsection* but above that of a *paragraph*.

See also chapter, section, subsection

Manual p. 174

Notes
- Generally, the level of subsubsection is the lowest titled division in a thesis or dissertation

T

`\tag`

Definition A macro which, when placed on a particular line of an equation environment, will replace the corresponding number of the line in the output with a given label.

See also `\notag`

Syntax `\tag{⟨new label⟩}`

Example

`\begin{equation}% A = \{n \in \mathbb{N} % \mid n>1\}% \end{equation}`	$A = \{n \in \mathbb{N} \mid n > 1\}$ (E.2)
`\begin{equation}% A = \{n \in \mathbb{N} % \mid n>1\}\tag{Eq1} \end{equation}`	$A = \{n \in \mathbb{N} \mid n > 1\}$ (Eq1)

TeX

Definition A program designed by Donald Knuth in 1978 for the purpose of typesetting complex mathematical formulae in an aesthetically pleasing fashion. The name TeX comes from the Greek word τεχνη, meaning “art” ,“craft”, or “skill”. TeX itself recognizes about 300 commands (called primitives).

See also LaTeX, primitive

Manual p. 10

Notes
- Plain TEX is an overlay system on top of TEX with an additional 600 commands

`\text`

Definition A LaTeX macro that is typically used inside math environments to typeset part of the corresponding expression in standard roman font.

See also `\normalfont`, `\rmfamily`

Manual p. 41

Syntax `\text{⟨text⟩}`
- ⟨*text*⟩ is the expression that will be typeset in standard roman font

Notes
- The macro `\textrm` has the same overall effect as `\text` except that `\text` respects small spaces in its argument, whereas `\textrm` does not

textblock

Definition In typesetting, the textblock refers to the actual space on a page where the text is printed.

Notes
- For your thesis or dissertation, the textblock measures 6 inches wide and 9 inches high

`\the`

Definition A command that when prefacing the name of a counter or parameter control sequence (e.g., a macro that stores a length) will output the current value of the counter or control sequence.

See also `\newcounter`, `\setcounter`

Example

```
The page \# is \thepage\ and the text width is \the\textwidth
```

The page # is 316 and the text width is 433.0pt

Notes
- Typically, if you need to use the current value of a counter in some computation or condition, you should use the command `\value` instead, as in `\value{page}`
- One can also use the command `\arabic` to output the value of a counter, as in `\arabic{page}`

A B C D E F G H I J K L M N O P R S T U V W X

theorem

Definition [1] In the context of LaTeX syntax, `theorem` represents a particular environment for rendering the actual statement of a theorem. The default for this environment is to typeset the name and number boldfaced with the numbering according to the section in which the theorem is placed, along with italicized font for the body of the theorem statement.

[2] In mathematics, a theorem represents a significant result. In a piece of mathematical work, usually only a handful of results are designated theorems.

See also corollary, lemma, proclamation, proposition

Manual p. 73

`\theoremstyle`

Definition A LaTeX command that prefaces a `\newtheorem` command and specifies a certain style for the proclamations given in the subsequent `\newtheorem` commands.

See also `\newtheorem`

Manual p. 75

Syntax `\theoremstyle{⟨style⟩}`

- ⟨*style*⟩ is one of the following: **plain**, **definition**, or **remark**; the **plain** option is traditionally used for lemmas, propositions, corollaries, and theorems

Example `\theoremstyle{plain}\newtheorem{lemma}{Lemma}` will set the lemma statements in italics

thesis

Definition A thesis is a term used for a major piece of work used to demonstrate depth of knowledge for someone seeking a master's degree. A thesis generally does not have to represent original work on the part of the candidate, but should illustrate a firm command of graduate-level information. A dissertation, on the other hand, is the term used for a major piece of work by one seeking a doctorate degree.

See also dissertation

tie (~)

Definition When the tilde symbol ~is placed in normal text, it indicates to TeX not to break the line at that particular point.

See also `\nobreak`

Example `H.~L. Menken` will typeset as H. L. Menken and also ensure that TeX does not break the line immediately after typing the “H.” (which, if it did, would look unsightly)

Notes
- If an actual tilde is desired in text, then the macro `\textasciitilde` should be used

token

Definition Roughly, a “token” represents the most basic object that TeX processes to create output. More technically, a token is created when a character code (from a given encoding) is assigned to a category code so that TeX knows exactly what the function of the object should be.

See also encoding (font)

Example The letter t_{11} and the command $\texttt{\textbackslash par}_{16}$ are tokens, where the subscripts indicate the category code to which the objects belong.

Notes
- There are three classes of token: character tokens, parameter tokens, and control sequence tokens
- If need be, the category code of a particular object can be changed by means of the command `\catcode‘`

`\tolerance`

Definition An integer-valued parameter that can range from 0 to 10000 and indicates to TeX how much badness is allowable without error.

See also badness, `\pretolerance`

`\ttfamily`

Definition A macro that allows an entire piece of your work to be typeset in teletype font. Teletype font is a monospaced font that makes text appear as it would on a typewriter.

See also `\normalfont`, `\rmfamily`

Manual p. 28

Example

```
{\ttfamily This part of the sentence} will be typeset teletype.
```

`This part of the sentence` will be typeset teletype.

Notes
- In order to ensure that other parts of your work remain unaffected by this macro, you usually should set this command and its affected text inside its own set of curly braces

`\underbrace`

Definition A macro designed to put a curly brace below several pieces of mathematical notation along with a possible descriptor of the pieces.

See also `\overbrace`

Syntax `\underbrace{⟨collection⟩}[⟨_{descriptor}⟩]`

- ⟨*collection*⟩ is the relevant set of notation
- ⟨*_{descriptor}*⟩ is the optional descriptor of the set

Example

```
$\underbrace{f \circ f \circ\cdots\circ f}_{n\ \text{functions}}$
```

$\underbrace{f \circ f \circ\cdots\circ f}_{n\ \text{functions}}$

`\underfull`

Definition An hbox is considered to be underfull if there is too little content for the size of the box, resulting in too much white space to satisfy TEX's requirements.

See also badness, `\hbox`, overfull

Manual p. 14

Example

```
\fbox{\hbox to 4in{This hbox is underfull.}}
```

This hbox is underfull.

Notes

- An underfull hbox on a line usually indicates that an unnecessary line break command (such as `\\`, `\linebreak`, or `\newline`) was issued on the previous line, and will result in extra spacing at the affected place in the output
- Underfull hboxes are reported in TEX's log, but it is not considered to be a critical error that will prevent TEX from compiling

`\MakeTextUppercase`

Definition A macro which will reexpress the text in its argument in all caps but leave any mathematical notation untouched.

See also `\MakeTextLowercase` indexed under "lowercase"

Syntax `\MakeTextUppercase{⟨text⟩}`

Example

```
\MakeTextUppercase{this, but not $a$} will be capitalized
```

THIS, BUT NOT a will be capitalized

upright font

Definition A generic term for font that is non-slanted and non-italicized.

See also emphasis, `\itshape`, `\normalfont`, `\rmfamily`, `\slshape`

Manual p. 28

Notes
- In Computer Modern font, the upright font style is the same as the standard roman font style; as such, `\textup` has precisely the same effect as `\textrm`

V

`\vbox`

Definition A TEX primitive that sets a vertical box the natural height of its contents.

See also `\colorbox`, `\fbox`, `\fcolorbox`, `\hbox`, `\mbox`, `\minipage`, `\parbox`

Syntax `\vbox`⟨*to height*⟩{⟨*text*⟩}
- ⟨*text*⟩ is the content to be placed inside the box
- ⟨*to height*⟩ is an optional specification of the desired height of the box

Example

```
%
\vbox{This text} is in a vbox.%
%
```

This text
is in a vbox.

```
%
%
\vbox to 0.5in{This text} %
is also in a vbox. %
%
%
```

This text

is also in a vbox.

Notes
- If you also want to control how wide a vbox is, you can simply add the command `\hsize=`⟨`width`⟩inside the vbox
- In general, the width of a vbox is determined by the width of the widest hbox inside of it

verso page

Definition In typesetting, a verso page is on the left side when the book is laid open facing the reader. Verso pages are always even-numbered. Each page in a work is either a recto page or a verso page.

See also recto page

vita

Definition A paragraph that appears at the very end of a thesis or dissertation that provides a short biography of the work's author.

A B C D E F G H I J K L M N O P R S T U V W X

A B C D E F G H I J K L M N O P R S T U V W X

Manual p. ??

Notes
- A vita is always written in third person

\vline

Definition In LaTeX's `tabular` environment, the `\vline` command will issue a thin vertical line that spans the length of the table.

See also `\cline`, `\hline`

Notes
- `\vline` can be used only in the column specification part of the environment and is sometimes substituted for | there

vplace

Definition A `memoir` class environment name designed to control the vertical placement of material within the scope of the environment.

Syntax `\begin{vplace}[⟨number⟩]`
⟨*material*⟩
`\end{vplace}`
  - ⟨*number*⟩ is the ratio of (vertical) space on the page above the material to space below the material (default value is 1 which vertically centers the relevant material on the page)

\vss

Definition A shorthand TeX primitive equivalent to `\vskip0pt plus 1fil minus 1fil`

See also fill, glue, `\hbox`, `\hss`, skip, `\vbox`

Notes
- In effect, it can be used to create a vertical box with "infinite shrinkage" and "infinite stretch"

W

widow

Definition In composition, a widow occurs when the last line of a paragraph occurs on the first line of a page. This is considered so aesthetically displeasing that a rewrite is considered mandatory to keep this from happening.

See also orphan, `\widowpenalty`

Notes

- TeX has a system in place through `\widowpenalty` to make the compilation "allergic" to widows with varying levels of aversion. However, there is no way to guarantee that TeX will never create a widow, so be on the lookout.

`\widowpenalty`

Definition A macro that stores an integer-valued parameter between 0 and 10000 that indicates to TeX how bad it should interpret having a widow created at the top of a page.

See also orphan, widow

Example `\widowpenalty=10000` sets `\widowpenalty` to its maximum value

Notes

- The default value of `\widowpenalty` is 150
- Bear in mind that even with a `\widowpenalty` at 10000, TeX's internal typesetting algorithms may still force a widow to be produced depending upon the values of other parameters; in such a case, you will need to edit the affected paragraph to ensure that this ultimately does not happen

X

`\xdef`

Definition A primitive that is equivalent to `\global\edef`. That is, not only are the constituent macros of the defined macro expanded at the time of definition, but this value for the defined macro holds in any environment. As such, `\xdef` is the most "permanent" of all ways to define a macro.

See also `\def`, `\edef`, `\global`

`\xleftarrow`

Definition An extension of the traditional `\leftarrow` command that allows for the placement of material both above and below the arrow

See also `\xrightarrow`

Syntax `\xleftarrow[⟨below⟩]{⟨above⟩}`

- ⟨*below*⟩ is the material to be placed beneath the arrow
- ⟨*above*⟩ is the material to be placed above the arrow

Notes

- True to conventions for optional arguments and mandatory arguments, if [⟨*below*⟩] is not given, nothing is placed beneath the arrow and if {⟨*above*⟩}= {}, nothing is placed above the arrow

A B C D E F G H I J K L M N O P R S T U V W X

\xrightarrow

Definition An extension of the traditional **\rightarrow** command that allows for the placement of material both above and below the arrow

See also \xleftarrow

Manual p. 3

Syntax **\xrightarrow**[⟨*below*⟩]{⟨*above*⟩}

- ⟨*below*⟩ is the material to be placed beneath the arrow
- ⟨*above*⟩ is the material to be placed above the arrow

Notes
- True to conventions for optional arguments and mandatory arguments, if [⟨*below*⟩] is not given, nothing is placed beneath the arrow and if {⟨*above*⟩}= {}, nothing is placed above the arrow

Analphabetic Index

\| 4
\; 33
\\ 3–5, **31**, 52–55, 58–60, 62–66, 106, 117, 118, 120, 125, 126, 128–133, 192–194
& 3–5, **52**, 52–54, 58–60, 62–66, 117, 118, 120, 125, 126, 128–133, 205–208
\left(\right) 2, **47**
\left\{ \right\} 5, **47**
⚡ 8, 264
! 8, 261
💣 8, 264
\␣ **11**, 21, 33, 79
$ **11**, 41, 192
! 22, **191**
? 22, **191**, 194
% **24**, 176
\: 33
\! 33
\{ \} 6, 35
\[\] **39**, 42, 46, 59, 255
$$ **41**, 255
?? 44
\left[\right] 47
\left| \right| 47, 49
{ } 104–106, 121, 142, 146, 186, 192, 269
\, 2, **33**, 145
@ 179
\' (ó) 187
\` (ò) 187
\^ (ô) 187
\~ (õ) 187
\= (ō) 187
\. (ȯ) 187
\" (ö) 187
!` (¡) 187
?` (¿) 187
\(\) 41
\left. **51**, 252
\$ 11
\# 80
\% 24
\& 52
\@ 146

Alphabetic Index

A

\AA (Å) 187
\aa (å) 187
abstract 257
abstract page **160**
  title 159, 161
acknowledgment 166
\Acknowledgments 166
acknowledgments page 166
\addcontentsline 257
\addlinespace 127

\addtocontents 258
\addtodef 258
\addtolength 258
\AE (Æ) 187
\ae (æ) 187
Albee, Edward 15
\aleph (ℵ) 34, **36**
\alpha (α) 3, **36**
\amalg (⨿) 65
ampersand symbol (&) **52**, 52–54, 58–60, 62–66, 117, 118, 120, 125, 126, 128–133
  bibliography 205–208
  double (&&) 59, **60**, 66
angle key 108
APA 7 style 120, 137, 170, 172, 178, 180, 182, 251, 255
  bibliography 12, 205
  captions 102, 104
  seriation 83
  titles 250
APA style 259
appendix **177**, 259
  loading 178
  none 178
  ordering 178
  title 178
approval page
  approval of 164
  signatures 163
  title 159
\ar 59
Arabic numerals 259
\arccos 77
\arcsin 77
\arctan 77
\arg 77
\arraystretch 127
\arrow **58**, 59, 64, 66
  colors 63, 64
  crossing over 65
  direction letters 58, 60
  styles
    bend left 66
    bend right 66
    dash 62
    dashed 62, 65, 66
    dashleftarrow 62
    dashrightarrow 62
    equal 62
    harpoon 61
    hook 61
    hookleftarrow 61
    hookrightarrow 61–63
    Leftarrow 61
    leftarrow 61
    leftarrowtail 62
    leftharpoondown 61
    leftharpoonup 61
    Leftrightarrow 61
    leftrightarrow 61
    leftrightsquigarrow 61
    leftsquigarrow 61
    Mapsfrom 61
    mapsfrom 61
    Mapsto 61
    mapsto 61
    no head 62
    no tail 62
    Rightarrow 61
    rightarrow 61
    rightarrowtail 62
    rightharpoondown 61
    rightharpoonup 61
    rightsquigarrow 61
    squiggly 61
    tail 60, 62
    to head 61
    two heads 60, 62
    twoheadleftarrow 62
    twoheadrightarrow 62, 63
  swap **60**, 62, 63, 65
article in a bib file 182
asterisk (∗) 260

@ symbol 179
@article 182
@book 182
@booklet 183
@collection 183
@conference 183
@dataset 183
@electronic 183
@inbook 182
@incollection 183
@inproceedings 184
@inreference 185
@letter 183
@manual 183
@mastersthesis 183
@misc 184
@mvbook 182
@mvcollection 183
@mvproceedings 184
@mvreference 185
@online 184
@patent 184
@periodical 184
@phdthesis 184
@proceedings 184
@reference 185
@report 185
@software 185
@standard 185
@suppbook 183
@suppcollection 183
@suppperiodical 184
@techreport 185
@thesis 185
@unpublished 185
@www 185
autogobble key 137
aux file 12, 194

B

\b (o̱) 187
back matter 177, 260
\backmattercontent 157
backslash 193
  defining a macro **11**, 21, 22, 78
  double 31
  following a macro 79
\backslash (\\) 5
badness 261
bang 261
\bar ($\bar{z}$) 4
baseline 261
\baselineskip 262
\baselinestretch 262
\batchmode 195
bbl file 12
bcf file 12
\begin{document} **18**, 156, 176, 193
\beta (β) 36
\beth ($\beth$) 36
\bfseries 29, 263
bgcolor key 137
bib file 12, 255
  article 182
  book 182
  booklet 183
  collection 183
  conference 183
  creating 178
  dataset 183
  electronic 183
  fields 186
  inbook 182
  incollection 183
  inproceedings 184
  inreference 185
  letter 183
  manual 183
  mastersthesis 183
  misc 184
  mvbook 182
  mvcollection 183
  mvproceedings 184

`mvreference` 185
`online` 184
`patent` 184
`periodical` 184
`phdthesis` 184
`proceedings` 184
`reference` 185
`report` 185
`software` 185
source types 182
`standard` 185
`suppbook` 183
`suppcollection` 183
`suppperiodical` 184
`techreport` 185
`thesis` 185
`unpublished` 185
`www` 185
Biber **17**, 178, 181, 182, 254
bibliographic databases 182
bibliographic fields 186
bibliography 17, 176, **178**, 205, 263
book 206
chapter in a book 207
journal article 205
webpage 207
BiBTEX 17, 182
`\Big` **49**
`\big` **49**
big point 32
`\bigcap` ($\bigcap$) 5, **35**
`\bigcup` ($\bigcup$) 5
`\Bigg` **49**
`\bigg` 2, **49**
`\bigoplus` ($\bigoplus$) 5
`\bigotimes` ($\bigotimes$) 5
`\binom` 5
blank page 160, 170
`blg` file 12
`bmatrix*` 4
`\boldmath` 263
`\boldsymbol` 133, 264
bolt 8, 264
bomb 8, 264
book in a `bib` file 182
booklet in a `bib` file 183
books
Math Into LATEX 25
More Math Into LATEX 25
Practical LATEX 25
wikibooks 25
`\bottomrule` **125**, 128–133
bp 32
braces *see* curly braces
breaklines key 137
`\brokenpenalty` 264
`bst` file 12
`bsx` file 12

C

`\c` (ç) 187
calculus commands 2
callout key 106, 122
`\cap` ($\cap$) 52
`\Caption` 256
caption **101**, 167, 265
`\caption` **112**, 113, 117, 118
caption key 104
cases 3
cc 32
`\cdot` ($\cdot$) 2
`\cdots` ($\cdots$) 4, 55, 154
`\cellcolor` **130**, 131
`\centernot` 5
centimeter 32
chapter **170**, 176, 265
loading 175
ordering 176
title 170
`\chapter` **170**, 177
`\checkmark` (✓) 87
`\chi` (χ) 36
Churchill, Winston 249

cicero 32
\circ ($\circ$) 319
\circ ($\circ$) 299
citation 44, 178, **188**, 194, 265
  multiple 189
  parenthetical 188
  postfix 188
  prefix 188
  textual 188
\cite 76, 188, 194, 265
\citealp 189
\citeauthor 189
\citenum 189
\Citep 189
\citep 188
\citepalias 189
\Citet 189
\citet 188
\citetalias 189
\citetext 189
\citeyear 189
\citeyearpar 189
\cleardoublepage 266
\clearpage 266
\cline 266
clip key 108
cls file 13
\clubpenalty 266
cm 32
\cmidrule 128, 129
collection in a bib file 183
\colon 3
\colorbox 266
colspecs key 121
\columncolor **130**, 131
command and control 250
commenting 24
committee members 163–165
  chair 165
  co-chair 165
  name format 164, 165
\CommitteeMember 163, 165
commutative diagram 57, 59, 62
  side-by-side 62
compiling **11**, 16, 19, 24, 44, 46, 74, 176, 181, 190, 193
  error **22**, 24, 44, 176, 191
    line number 24, 191
Comprehensive TeX Archive Network 25, 155
compress key 122
Computer Modern 267
computer programs, displaying 135
conclusion 74
conference in a bib file 183
congruent ($\equiv$) 5
conjecture 74
control sequence (cs) 267
Coolidge, Calvin 211
\coordinate **67**, 68–70
\coprod ($\coprod$) 5
copyright 267
copyright page 162
\CopyrightOwner 162
corollary 74, 268
\cos 77
\cosh 77
\cot 77
\coth 77
\CountCitations 256
\CountReferences 256
\Cref 46
\cref 46
cs (control sequence) 268
\csc 77
\csgdef 268
\csuse 268
csx file 13
CTAN 25
\cup ($\cup$) 52
curly braces 268

D

\d (ọ) 187

\daleth (ℸ) 36
dataset in a bib file 183
Davies, Robertson 209
dd 32
\ddot ($\ddot{y}$) 3
\ddots ($\ddots$) 4, 34
Dean, College of Graduate Studies 163
\DeclareMathOperator 78
\Dedication 165
dedication page 165
\def 79, 269
definition 74
\deg 77
\DegreeType 159
delimiter 47–51, 130, 138, 142, 147, 269
\Delta (Δ) 37
\delta (δ) 36
demerit 270
deprecated 270
derivative 3
\det 5, 77
determinant 5
\DH (Ð) 187
\dh (ð) 187
diagonal dots 34
Didot point 32
\digamma ($\digamma$) 36
\dim 77
\DisableLigatures 146
display 270
displayed formula **39**, 40, 42, 153, 270
  aligning equations 52
  breaking inside 154
  delimiters 47, 49
  double dollar signs 41
  numbering 42
  punctuation 153
  subscript position 77, 78
displaying code 135
\displaystyle 271
\displaywidowpenalty 271
Disraeli, Benjamin 83
dissertation 271
distribution 11, 15, 271
\DJ (Đ) 187
\dj (đ) 187
\dlar 59
document
  compiling 267
document class 17, 272
  tikz option 57
\documentclass **17**, 19, 156, 159, 176
\DocumentType 159
DOI 206
dollar sign 192
  double **41**, 255
  math environment **11**, 19, 39, 41
  text 11
\doublehyphendemerits 272
\Downarrow ($\Downarrow$) 48
\downarrow ($\downarrow$) 48
Doyle, Sir Arthur Conan 117
\draw **67**, 69, 70
  dash dot 71
  dash dot dot 71
  dashed 71
  densely dash dot dot 71
  densely dash dot 71
  densely dashed 71
  densely dotted 71
  dotted 71
  loosely dash dot 71
  loosely dash dot dot 71
  loosely dashed 71
  loosely dotted 71
  semithick 71
  solid 71
  thick 71
  thin 71
  ultra thick 71
  ultra thin 71
  very thick 71
  very thin 71

dtx file 13
Dyer, Wayne W. 197

E

E (edit) 195
\edef 272
Edison, Thomas 169
editor 273
electronic in a bib file 183
\ell (ℓ) 5
em 32, 273
em-dash (—) 187, 273
\emergencystretch 273
\emph 28, 274
emphasis 274
en-dash (–) 187, 206, 274
encoding (font) 274
end-of-proof box 74
\end{document} **18**, 74, 156, 176
\endgroup 192
\enspace 33
Enter 194
env key 121
environment 18, 31, **42**, 46, 112, 193, 274
  align 52, 54
  alignat 53
  aligned 53, 54
  array 133, 260
  \begingroup...\endgroup 262
  conclusion 74
  conjecture 74
  corollary 74
  definition 74, 154
  description 83, 84, 95
  displaymath 41
  document **19**, 176
  enumerate 83, 91
  equation **42**, 44, 46, 58, 59, 67, 133
  example 74
  fact 74
  figure **112**, 113, 114, 250
    caption 101, 112, 113
    framed 114
  gather 54
  hypothesis 74
  itemize 83, 85
  lemma 74
  \list...\endlist 289
  Listing 137
  listing 113
    caption 101, 112, 113
  math 22, 24, 41, 192, 194
  methodology 74
  minipage 292
  multline 55
  problem 74, 253
  proof 74
  proposition 74
  question 74
  remark 74
  split 55
  starred **46**
  table 113, **117**, 118, 125, 126, 250
    caption 101, 112, 113
    column options 118
    highlight cells 130–132
    horizontal lines 118, 120, 125, 128–133
    missing data 126
  tabular **117**, 118, 125, 126, 128–133
  theorem **73**, 74, 79
  tikzcd **58**, 59, 60, 62–64, 66
    column sep, row sep 63, 65
  tikzpicture **67**, 68–70
  verbatim 136
  vplace 321
\epsilon (ϵ) 36
\equiv ($\equiv$) 5
\errorstopmode 195
escapeinside key 138
\eta (η) 36
etex package error 16
ex 32, 275

example 74
exclamation mark
  bang 8
  error 22, **191**
\exhyphenpenalty 275
\exp 77
\expandafter 275
expansion (font) 276
ext key 110
\extrarowheight 127

F

fact 74
\fbox 276
\fcolorbox 276
fd file 13
Fear's Rules 120
\Figure 251
figures
  caption 112
  framed 114
  general keys 104
  label 113
  placement options 113
  requirements 103
  side by side 114
fill 277
\fill **68**, 70, 114
finalline key 138
firstline key 138
firstnumber key 138
float 277
\floatingpenalty 278
floats **103**
  caption 112
  general keys 104
  label 113
  placement options 113
  requirements 103
folio 278
font
  changing 50
  math 41
    blackboardbold 5, 19, **30**
    boldface 2, 3, 29, 30, 263, 264
    caligraphic 30
    fraktur 30
    italics 30
    roman 30
    sans serif 30
    script 2, 30
    teletype 30
  size 159
  size change 17
  text
    boldface 28, 29, 85, 125, 128, 154, 263
    emphasis 28, 274
    italics 19, 28, 29, 125, 147, 152, 154, 284
    middle weight 28, 29
    normal 28, 29
    roman 28, 29, 41, 78
    sans serif 28, 29, 309, 312
    slanted 28, 29, 313
    small caps 28, 29, 125, 310
    typewriter 28, 29, 318
    underline 28
    upright 28, 29, 147, 319
fontsize key 138
footer 278
\footnotesize 50, 278
foreign language symbols 187
foremargin 278
formatting 161
\frac 291
\frac 2, 3, 5, **35**
fraction 35
fragile 279
frame key 138
\framebox 276
framerule key 138
framesep key 138
\frenchspacing 279

front matter 157, 279
\frontmattercontent 157
function 3
\fussy 279
fuzz 280

G

\Gamma (Γ) 37
\gamma (γ) 36
\gcd 77
\gdef 280
\geq ($\geq$) 6
\gimel ($\gimel$) 36
glo file 13
\global 280
\GlobalFloatParameters **112**, 123, 140
glue 280
glyph 281
Grätzer, George 25
gradient 3
\GraduationMonth 159
\GraduationYear 159
\Graphic **103**, 111, 250
    figure-specific keys 106
graphs 67, 70
Greek letters
    lowercase 36
    uppercase 37
Green, John 155

H

\H (ő) 187
H (help) 195
\hbox 193, 281
header 281
Hebrew letters 36
height key 109
\hfill **33**, 114
highlightcolor key 138
highlightlines key 138
\hline 126, 281
Holderman, Sharon 164
\hom 77
horizontal mode 14
\hsize 282
\hspace **31**, 53, 114
\hss 282
\HUGE 50
\Huge 50
\huge 50
hyphen (-) 187, 282
\HyphenAllowance 250, 251
\hyphenpenalty 282
hypothesis 74

I

I (insert) 195
\i (ı) 187
idx file 13
\if...\else...\fi 282
if and only if ($\iff$) 5
\iff ($\iff$) 5
\ifthenelse 283
\iiint ($\iiint$) 2
\iint ($\iint$) 2
ilg file 13
\implies ($\implies$) 5, 62
\in ($\in$) 5, 21, **52**
in (inch) 32
inbook in a bib file 182
\includegraphics 103, **112**, 114
incollection in a bib file 183
ind file 13
\indent 283
indentation 159
indexing values 34, 47
\inf 77
\infty (∞) 2, 5, **35**
\injlim 77
inproceedings in a bib file 184
\input 283
\inputminted 142

inreference in a bib file 185
ins file 13
\InsertSpecialList 256
\int ($\int$) 2
integral
  double ($\iint$) 2
  line ($\oint$) 2
  triple ($\iiint$) 2
\interlinepenalty 283
intersection ($\cap$) 35, **52**
interval notation 6, 51
\iota (ι) 36
ist file 13
\isundefined 284
italic correction 147
\item 83, 84
\itshape 29, 284

J

\j (ȷ) 187
Jami, Criss 177
\jobname 285
\justifying 285

K

\k (ę) 187
\kappa (κ) 36
keepaspectratio key 109
\ker 77
kern 285
key 286
key=value system **84**, 112, 120, 121, 137, 251
Knuth, Donald 6, 10, 135

L

\L (Ł) 187
\l (ł) 187
l. 24
label
  citations 178
  divisions 170, 171, 173, 175
  environments 74, 106, 113
  list items 84, 85
\label **43**
label key 106
\Lambda (Λ) 37
\lambda (λ) 36
Lamport, Leslie 10
\langle ($\langle$) 3, 48
\LARGE 50, 287
\Large 50, 287
\large 50, 286
lastline key 139
LaTeX 10, 287
  editor 1, 14, **16**, 26, 192, 195
  symbols 25
LaTeX 2_ε 11
LaTeX3 Project 11
Latin Modern *see* Computer Modern
\lceil ($\lceil$) 48
\ldelim **130**, 133
\ldots ($\ldots$) 3, 154
leading 287
\leavevmode 287
\leftarrow 133
\Leftrightarrow ($\Leftrightarrow$) 52
lemma 74, 288
length, units 31, 192
\leq ($\leq$) 54
\let 288
letter in a bib file 183
\lfloor ($\lfloor$) 48
\lg 77
ligature 146, 288
lightning bolt 8
\lim 2, 77
\liminf 77
limit notation 2
\limsup 77
line break 31
\linebreak 288
\linepenalty 289

list
  additional 167
  figures 101
  listings 101
  tables 101
list of figures 166
list of listings 166
list of tables 166
Listing 251
listings 135
  caption 112
  general keys 104
  label 113
  placement options 113
  requirements 103
\llcorner (⌞) 48
\ln 77
lof file 13
lol file 13
\log 54, **77**
log file 13, 195, 252
\long 289
lot file 13
\lrcorner (⌟) 48
LyX 16, 199

M

macro **11**, 18, 25, 79, 290
  execution 12
  expansion 12
  optional parameter 17
  user-created 77
MacTEX 11
main matter 169, 290
\mainmatter 156
\mainmattercontent 157
\Major 159
\makeatletter...\makeatother 290
\makebox 291
MakeIndex 17
\MakeTextLowercase 289
\MakeTextUppercase 319
\maltese (✠) 91
mandatory argument 291
manual in a bib file 183
mastersthesis in a bib file 183
math mode 36, 194
math unit 31
\mathbb 5, 19, **30**
\mathbf 2, 3, 29, **30**, 263, 264
\mathcal 30
\mathclap **34**, 253, 291
\mathfrak 30
\mathit 30
\mathrm 30
\mathscr 2, **30**
\mathsf 30
\mathtt 30
matrix 4
\max 77
Mayan pyramid, inverted **159**, 161
\mbox 194, 291
\mdseries 29
Mead, Margaret 57
\medspace 33
memoir class **155**, 156, 191
methodology 74
\mid (|) 6
\midrule **125**, 128–133
\midslopy 292
MikTEX 1, **11**, 15, 16, 25, 211
  MikTEXConsole 1
  MikTEX Console 16
  TEXworks 16, 24
Miller, Dennis 27
millimeter 32
\min 77
\miniscule 50
\mintinline 141
  \setmintedinline 142
misc in a bib file 184
modulo 5
\mp (∓) 5

\mspace 31
mu 31
\mu (μ) 36
\multicolumn **127**, 128
\multirow **128**, 129–131, 133
mvbook in a bib file 182
mvcollection in a bib file 183
mvproceedings in a bib file 184
mvreference in a bib file 185
\MyAppendix 178
\MyBibReferences **180**, 255
\MyChapter **175**, 176
\MyVita 190

N

\nabla (∇) 3
\⟨*name*⟩ 167
natbib 249
n.d. 180
\negmedspace 33
\negthickspace 5, **33**
\negthinspace 33
\newcommand **78**, 79, 80, 292
\newcounter 293
\NewDocumentCommand 293
\newenvironment 294
\newlength 294
\newline **31**, 192–194, 295
\newpage 295
\newtheorem **74**, 75, 295
\NG (Ŋ) 187
\ng (ŋ) 187
\nobreak 296
\nocite 189
\noindent 296
nominal m-width 32
nominal x-height 32
\nonstopmode 195
norm 4
\normalfont 29, 296
\normalsize 50, 296
\notag 296
note key 105
\notin ($\notin$) 5
\nsubseteq ($\nsubseteq$) 5
\nu (ν) 36
nullchapter 249
numberblanklines key 139
numbers key 139
numbersep key 139

O

\O (Ø) 187
\o (ø) 187
O'Connor, Flannery 145
obeytabs key 139
octothorp symbol (#) 80, 297
\OE (Œ) 187
\oe (œ) 187
\oint ($\oint$) 2
\Omega (Ω) 37
\omega (ω) **36**, 54
online in a bib file 184
operator macros 77
optional argument 297
origin key 108
orphan 298
orthogonal commands 298
\outputpenalty 298
\overbrace 253, 298
overfull (hbox) 299
Overleaf 15, 16, 26
\overline 4

P

package **11**, 16, 18, 25, 39, 155, 156, 299
- amsfonts 18
- amsmath 18, 77, 78, 253, 255
- amssymb 18
- amsthm 18
- apacite 255
- atbegshi 253

babel 251
biblatex 178, 180, 182, 249, 250, 254
bigdelim 250
bigstrut 250
centernot 251
cleveref 46
complete list for TNThesis 155
csquotes 249, 251
doc 211
enumitem 83
fontenc 250
fvextra 249
hyperref 156
inputenc 250
interval 51, 250
keyval 250
listings 135
listings 136, 250, 255
lmodern 250
luximono 249
mathrsfs 255
mathtools 77, 253, 255
microtype 146, 250
minted 135
minted 136, 137, 141, 249, 250
multirow 250
natbib 254
relsize 255
tikz 57
tikz 57, 59, 67, 250, 254
url 250

page key 110
page numbering 158
\pagestyle 300
pagination 300
Pakin, Scott 25
\par 79, 193, **300**
\parbox 300
\ParentheticalCite 254
\parfillskip 301
\parindent 301
\parshape 301
\parskip 302
\partial (∂) 2, 3
patent in a bib file 184
pc 32
penalty 302
percentage symbol **24**, 176
periodical in a bib file 184
Persian proverb 205
\phantom 303
\phantomsection 303
phdthesis in a bib file 184
\Phi (Φ) 37
\phi (ϕ) 36
\Pi (Π) 37
\pi (π) 36
pica 32
\pm ($\pm$) 5
pmatrix* 4
\pmod 5
point 32
postcaption key 105
pound symbol 80
\Pr 77
preamble 16, **19**, 74, 75, 77, 78, 146, 156, 193, 303
\pretolerance 304
\prime ($\prime$) 3
primitive 304
problem 74
proceedings in a bib file 184
proclamation 304
\prod ($\prod$) 35
product 35
programs, displaying 135
\projlim 77
proposition 74, 304
ProQuest (website) 1
\protect 305
proTeXt 11
protrusion 305
\providecommand 305
\Psi (Ψ) 37

\psi (ψ) 36
pt 32
Pygments
  language options 137
  website 137
Python (website) 136

Q

Q (quiet) 195
\QEDhere 75, 251, 253
\qquad 33
\quad **33**, 53
question 74
question mark 22, **191**, 194
  double 44

R

R (run) 195
\raggedleft **305**
\raggedright **306**
Rand, Ayn 191
\rangle ($\rangle$) 3, 48
\rar 59
\rceil ($\rceil$) 48
\rdelim **130**, 133
recto page 160, 306
\ref **44**, 46, 106, 194
reference 306
reference in a bib file 185
referencing 42–44, 194
  clever 46
\refstepcounter 307
\relax 307
remark 74
\renewcommand 307
report in a bib file 185
requirements
  floats 103
  Graduate School 1, 17, 51, 102, 151, 157, 161, 163, 170
\resizebox 308
\rfloor ($\rfloor$) 48
\rho (ρ) 36
\rightarrow ($\rightarrow$) 3
\rinterval 6
\rmfamily 29, 308
robust 308
Roman numerals 309
\rowcolor **130**, 132
\rule 309
rulecolor key 139
Russell, Bertrand 73

S

S (scroll) 195
sans serif font 309
scale key 109
scaled point 32, 310
\scriptsize 50
\scrollmode 195
\scshape 29, 310
\sec 77
section 310
  requirements 169
  title 171, 172
\section 171
semicolon (tikzpicture) 68
seriation 83
set 35
\setcounter 311
\setlength 311
\setmintedinline 142
\settowidth 311
\sffamily 29, 309, 312
-shell-escape 137
\shipout 312
shortcaption key 104
showspaces key 139
showtabs key 139
\Sigma (Σ) 37
\sigma (σ) 36
\sin 2, 77

\sinh 77
skip 312
\sloppy 312
\slshape 29, 313
\small 50, 313
\smashoperator **34**, 253, 313
Socrates 9
software in a bib file 185
sp 32, 310
spacecolor key 139
spacing **21**, 22, 79
  double 161
  horizontal 31, 193
  single 161
  vertical 31
\SpecialListItem 167, 256
spine margin 314
spurious 314
\sqrt[n] ($\sqrt[n]{\ }$) 4
\ss (ß) 187
Stack Exchange, TeX-LaTeX 25
Stack Overflow 26
standard in a bib file 185
\star ($\star$) 90
\stepcounter 314
stepnumber key 139
stripall key 139
\strut 314
sty file 13
style key 139
subscript **34**, 77, 78
  substack 34
subsection 314
  requirements 169
  title 172, 173
\subsection 173
\subset ($\subset$) 5
\subseteq ($\subseteq$) 5
\subsetneq ($\subsetneq$) 5
\substack 34
subsubsection 315
  requirements 169
  title 172, 175
\subsubsection 174
\sum ($\sum$) 11, **35**
summation 35
\sup 77
superscript 34
\SuperSize 50
suppbook in a bib file 183
suppcollection in a bib file 183
suppperiodical in a bib file 184
symbol list 25
synctex file 13

T

\t (o͡o) 187
tabcolor key 139
\Table **120**, 250
  table options 121
Table 251
table of contents **163**, 255
tables
  caption 112
  general keys 104
  label 113
  placement options 113
  requirements 103
tabsize key 139
tabwidth key 121
\tag 315
\tan 77
\tanh 77
\tau (τ) 36
techreport in a bib file 185
TEX 10, 315
  e-TEX 11
  LuaTEX 17
  pdfTEX 17
  XeTEX 17
tex file 13
TEX Users Group 25
TEX Live 11

TEXnicCenter 16, 200
TEXstudio 16, 201
\text 3, 6, **41**, 52, 78, 133, 316
\textasciitilde (~) 88, 318
\textasteriskcentered (∗) 86
\textbf 28, 154, 263
textblock 316
\textbullet (•) 85
\textdagger (†) 86
\textdaggerdbl (‡) 87
\textemdash (—) 89
\textgreater (>) 86
\textheight 110
\textit 19, 28, 147, 152, 154, 284
\textlnot (¬) 88
\textmd 28
\textnormal 28
\textopenbullet (◦) 87
\textperiodcentered (·) 90
\textreferencemark (※) 88
\textrightarrow (→) 89
\textrm 28
\textsc 28, 310
\textsection (§) 89
\textsf 28, 309, 312
\textsl 28, 313
\textsuperscript 106
\texttt 28, 318
\textup 28
\textwidth **110**
TEXworks **16**, 16, 24, 137, 202
\TH (Þ) 187
\th (þ) 187
\the 316
theorem 74
\theoremstyle 74, 317
thesis 317
thesis in a bib file 185
\ThesisAbstract 161
\ThesisAuthor 159
\ThesisContent 156, 251
\ThesisTitle 159
\Theta (Θ) 37
\theta (θ) 2, 36
\thickspace 33
\thinspace 33
tie (~) 317
tilde symbol (~) 317
\tiny 50
title case 172
title page **158**, 159, 162
TNTcontrol 255
TNTmain **156**, 159, 160, 162, 163, 165, 175, 176, 178, 180, 193, 209
\to (→) 2, 3
toc file 13
token 34, 318
\tolerance 318
\toprule **125**, 128–133
totalheight key 109
\tracing commands 195
\triangleright (▷) 90
trim key 107
\ttfamily 29, 318
TUG 25
Twain, Mark 39
type key 110

U

\u (ŏ) 187
\ulcorner (⌜) 48
\underbrace 253, 319
underfull (hbox) 319
\underline 28
union (∪) 5, **52**
units, length 31
unpublished in a bib file 185
\Uparrow (⇑) 48
\uparrow 133
\uparrow (↑) 48
\Updownarrow (⇕) 48
\updownarrow (↕) 48
upright font 319

\upshape 29
\Upsilon (Υ) 37
\upsilon (υ) 36
\urcorner ($\urcorner$) 48
\usepackage 18, 19

V

\v (ǒ) 187
\varDelta ($\varDelta$) 37
\varepsilon (ε) 36
\varGamma ($\varGamma$) 37
\varinjlim 77
\varkappa ($\varkappa$) 36
\varLambda ($\varLambda$) 37
\varliminf 77
\varlimsup 77
\varOmega ($\varOmega$) 37
\varPhi ($\varPhi$) 37
\varphi (φ) 36
\varPi ($\varPi$) 37
\varpi (ϖ) 36
\varprojlim 77
\varPsi ($\varPsi$) 37
\varrho (ϱ) 36
\varSigma ($\varSigma$) 37
\varsigma (ς) 36
\varTheta ($\varTheta$) 37
\vartheta (ϑ) 36
\varUpsilon ($\varUpsilon$) 37
\varXi ($\varXi$) 37
\vbox 320
\vdots ($\vdots$) 4
vector 3
Verne, Jules 101
verso page 320
vertex 68–70
vertical mode 14
\vfill 33
vita page 176, **190**, 320
\vline 321
vmatrix* 5
\vspace **31**, 194
\vss 321

W

website
- APA 7 style 208
- copyrights 162
- CTAN 25, 155
- LaTeX fonts 28
- LaTeX forum 25
- LaTeX symbols 25
- LyX 16
- MacTeX 11
- MikTeX 1, 11, 15
- Overleaf 16, 26
- ProQuest 1
- proTeXt 11
- Stack Exchange 25
- Stack Overflow 26
- TeX Live 11
- TeXnicCenter 16
- TeXstudio 16
- TNThesis 1
- TUG 25
- wikibooks 25
- WinEdt 16

widow 321
\widowpenalty 322
width key 108
Wilson, Peter 191
WinEdt 16
WinEdt 203
Word 10
writing
- Latin abbrev. 152
- let vs. put vs. choose 150
- royal we 152
- such that vs. so that 150
- suppose vs. assume 150
- theorem names 152
- transition words 149

trivially vs. obviously vs. clearly 150
www in a bib file 185

X

X (exit) 195
\xdef 322
\Xi (Ξ) 37
\xi (ξ) 36
\xleftarrow ($\leftarrow$) 322
\xrightarrow ($\rightarrow$) 3, 323

Z

\zeta (ζ) 36

Reference

Epigraphs

- *The only true wisdom is in knowing that you know nothing.* -Socrates
- *Sometimes it's necessary to go a long distance out of the way in order to come back a short distance correctly.* -Edward Albee
- *You know that there is a problem with the education system when you realize that out of the 3 R's, only one actually begins with an R.* -Dennis Miller
- *It ain't what a man knows that gets him into trouble. It's what he knows for sure that just ain't so.* -Mark Twain
- *Always remember that you are absolutely unique. Just like everyone else.* -Margaret Mead
- *Mathematics may be defined as the subject in which we never know what we are talking about, nor whether what we are saying is true.* -Bertrand Russell
- *Seeing much, studying much, suffering much are the three pillars of learning.* -Benjamin Disraeli
- “*Oh, figures!*” answered Ned. “*You can make figures do whatever you want.*” -Jules Verne
- *It is a capital mistake to theorize before one has data. Insensibly one begins to twist facts to suit theories, instead of theories to suit facts.* -Sir Arthur Conan Doyle
- *Programming is the art of telling another human being what one wants the computer to do.* -Donald Knuth
- *Everywhere I go I'm asked if I think the university stifles writers. My opinion is that they don't stifle enough of them. There's many a best-seller that could have been prevented by a good teacher.* -Flannery O'Connor
- *Every year, many, many stupid people graduate from college. And if they can do it, so can you.* -John Green
- *Our greatest weakness lies in giving up. The most certain way to succeed is always to try just one more time.* -Thomas Edison

- *People love answers, but only as long as they are the ones who came up with them.* -Criss Jami
- *If you don't know, the thing to do is not to get scared, but to learn.* -Ayn Rand
- *Success is stumbling from failure to failure with no loss of enthusiasm.* -Winston Churchill
- *It is nothing for one to know something unless another knows that you know it.* -Persian proverb
- *The eye sees only what the mind is prepared to comprehend.* -Robertson Davies
- *Nothing in this world can take the place of persistence.* -Calvin Coolidge
- *The highest form of ignorance is to reject something you know nothing about.* -Wayne W. Dyer