Do It Yourself:

One interesting feature that \LaTeX has is that you can define your own commands. So if you want to do something special you simply make a command to do it. The \TeX system also has ways for you to create your own fonts but we will not need to go quite that far. Creating commands, on the other hand can come in handy quite often. The general syntax for a new command is

\begin{verbatim}
\newcommand{name}[num]{definition}
\end{verbatim}

Where the name is the command name. Command names must start with a backslash \ and can not begin with \end. Furthermore, the command can not already exist. The num is the number of arguments the command can take and the definition is the way that the new command will act. For example, if we were to define,

\begin{verbatim}
\newcommand{\x}{$x$}
\end{verbatim}

Then \x would produce \textit{x}. \LaTeX simply takes \textit{$x$} and uses it in place of the \textit{\x}. This was an example of a command without any arguments. If we had used

\begin{verbatim}
\newcommand{\x}[1]{$x^{#1}$}
\end{verbatim}

Then \x{3} would produce \textit{x$^3$}. We can use as many arguments as we would like, simply place each in curly brackets. For example, with the new command,

\begin{verbatim}
\newcommand{\x}[2]{$x_{#1}^{#2}$}
\end{verbatim}

\x{1}{3} would produce \textit{x$_1^3$}. New commands need not be mathematical in nature. For example, the command

\begin{verbatim}
\newcommand{\names}[2]{#1 #2's name written formally would be #2, #1}
\end{verbatim}

would make \texttt{\names{Don}{Spickler} and \names{John}{Doe}.} produce, Don Spickler’s name written formally would be Spickler, Don and John Doe’s name written formally would be Doe, John. One command I create frequently is for binomial coefficients. The command,

\begin{verbatim}
\newcommand{\binom}[2]{\left( \begin{array}{cc} #1 \ #2 \ \end{array}\right)}
\end{verbatim}

will make $\binom{n}{t}$ produce \(\binom{n}{t}\) and

\begin{verbatim}
[ \\
\sum_{i = 0}^n \binom{n}{i} = 2^n \\
]\end{verbatim}

produce

\[\sum_{i = 0}^n \binom{n}{i} = 2^n\]

You can also redefine a command that already exists by using the redefine command. The general syntax for the redefine command is

\begin{verbatim}
\renewcommand{name}[num]{definition}
\end{verbatim}

where the name, num and definition are the same as with newcommand. For example,
Consider the variable $x$.

\renewcommand{$x$}{$X$}

Consider the variable $X$.

produces, Consider the variable $x$. Consider the variable $X$.

Be careful not to redefine a \LaTeX{} command unless you really want to. Most of the time you will place all of your new commands in the preamble to the document but technically they can go anywhere in the document.

You can also create your own environments with the \texttt{\newenvironment} and the \texttt{\renewenvironment} commands. The general syntax for these commands is,

\newenvironment{name}{[num]}{begdef}{enddef}
\renewenvironment{name}{[num]}{begdef}{enddef}

The name is the name of the environment, the num is the number of arguments, begdef is the stuff to do before the text that will be included in the environment and enddef is the stuff to do after the text that will be included in the environment. For example, the environment

\newenvironment{cards}{\vskip 10pt \noindent $\diamondsuit\clubsuit\heartsuit\spadesuit$ \hfill}{\hfill $\spadesuit\heartsuit\clubsuit\diamondsuit$ \vskip 10pt}

with

\begin{cards}
Pick a card, any card.
\end{cards}

will produce,

$\spadesuit\heartsuit\clubsuit\diamondsuit$ Pick a card, any card. $\spadesuit\heartsuit\clubsuit\diamondsuit$

The next several pages explain some of the more advanced features in \LaTeX{} when you are working with larger projects, such as reports or books that have multiple chapters or sections. When you are doing a bigger project you will need to use a different document class. So far all we have used is the article. That is, our first line to the \LaTeX{} document was something like,

\documentclass[10pt]{article}

For larger projects we need to use \texttt{report} or \texttt{book} instead of \texttt{article}. This will allow you to use the chapter, section, subsection and subsubsection features as well as the automatic title page, contents list and figure list.

You can also set up your own counters that can work either independently of the built-in counters or integrated with the built-in counters.

The following \LaTeX{} code created the table of contents and the first two sections of chapter 1. We will examine each of the features one at a time.

\tableofcontents

\chapter{More Advanced Features}

\section{You can count on \LaTeX{}}

\vskip 10pt

Actually we should say that \LaTeX{} can count for you. As you
saw in a previous \TeX{}nique, \LaTeX{} can automatically number lists, footnotes, equations, figures and pages. There are many more types of automatic numbering built into \LaTeX{}. Some of these include, chapter, section and subsection numbering as well as table, figure and paragraph numbering.

\begin{lemma}
There is always more than one way to prove a mathematical theorem.
\label{lemma:first}
\end{lemma}

\begin{lemma}
There are two or more ways to prove any mathematical theorem. See lemma \ref{lemma:first} on page \pageref{lemma:first}.
\end{lemma}

\begin{thm}
What can go wrong will go wrong.
\label{thm:Murphy}
\end{thm}

\begin{cor}
What possibly can go wrong will possibly go wrong. See Theorem \ref{thm:Murphy}
\end{cor}

\begin{dlaw}
It always takes twice the amount of time to complete a project as I think it will take.
\end{dlaw}

\begin{drule}
The weekend is a time to get caught up on work. So why am I more behind on Monday than I was the previous Friday?
People seldom remember how quickly you finish a job but they always remember how well you did the job.

The weekend is a time to relax and be rejuvenated for the next week. So why am I more tired on Monday than I was the previous Friday?
Contents

1 More Advanced Features 6
  1.1 You can count on \LaTeX ................................. 6
  1.2 Next Section ..................................... 6
  1.3 Let’s Look at What We Did to Create the Last Two Sections ................. 6
  1.4 Including External \LaTeX Files into a Document .......................... 8
Chapter 1

More Advanced Features

1.1 You can count on \LaTeX

Actually we should say that \LaTeX can count for you. As you saw in a previous \TeX\nique, \LaTeX can automatically number lists, footnotes, equations, figures and pages. There are many more types of automatic numbering built into \LaTeX. Some of these include, chapter, section and subsection numbering as well as table, figure and paragraph numbering.

Lemma 1.1.1 There is always more than one way to prove a mathematical theorem.

Lemma 1.1.2 There are two or more ways to prove any mathematical theorem. See lemma 1.1.1 on page 6.

Theorem 1.1 What can go wrong will go wrong.

Corollary 6.1 What possibly can go wrong will possibly go wrong. See Theorem 1.1

Don’s Law 1: It always takes twice the amount of time to complete a project as I think it will take.

Don’s Rule 1: The weekend is a time to get caught up on work. So why am I more behind on Monday than I was the previous Friday?

1.2 Next Section

Don’s Law 2: People seldom remember how quickly you finish a job but they always remember how well you did the job.

Don’s Rule 0: The weekend is a time to relax and be rejuvenated for the next week. So why am I more tired on Monday than I was the previous Friday?

1.3 Let’s Look at What We Did to Create the Last Two Sections

First the command \tableofcontents takes the chapter and section commands throughout the document and automatically produces a table of contents from them. The Contents title is produced automatically as is the headings and page number layout.

The command \chapter{More Advanced Features} produces a new chapter by skipping to the next page, placing the Chapter heading, number and title at the top of the page.

The command \section{You can count on \LaTeX} produces a new section. Notice that the first paragraph of the section is not indented. This is also done automatically since it is the current convention in printed materials.
The next command of interest is the \texttt{newtheorem} command. This command creates another environment similar to the \texttt{theorem} environment. That is, you can produce automatically numbered theorems, lemmas, corollaries, ... simply with a single command and a begin/end environment structure. For example, the command

\begin{verbatim}
\newtheorem{lemma}{Lemma}[section]
\end{verbatim}

creates an environment \texttt{lemma} that produces a bold first word of \texttt{Lemma} and is labeled with the Chapter/Section numbering before the lemma number. In general the syntax is

\begin{verbatim}
\newtheorem{env_name}{caption}[within]
\end{verbatim}

where \texttt{env_name} is the name of the environment, \texttt{caption} is the caption the is before the numbering and \texttt{within} is the numbering scheme that is before the number of the item. You can use any of the counters available in \LaTeX{} for the \texttt{within}, listed below, as well as any counter you create.

<table>
<thead>
<tr>
<th>part</th>
<th>paragraph</th>
<th>figure</th>
<th>enumi</th>
</tr>
</thead>
<tbody>
<tr>
<td>chapter</td>
<td>subparagraph</td>
<td>table</td>
<td>enumii</td>
</tr>
<tr>
<td>section</td>
<td>page</td>
<td>footnote</td>
<td>enumiii</td>
</tr>
<tr>
<td>subsection</td>
<td>equation</td>
<td>mpfootnote</td>
<td>enumiv</td>
</tr>
<tr>
<td>subsubsection</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is most common to use one of the following.

part
chapter
section
subsection
subsubsection

So with the lemma environment in our example, the environment,

\begin{verbatim}
\begin{lemma}
There is always more than one way to prove a mathematical theorem.
\end{lemma}
\end{verbatim}

produces

\textbf{Lemma 1.3.1} \textit{There is always more than one way to prove a mathematical theorem.}

The 1.3.1 tells us that it is in Chapter 1, section 3 and is the first lemma in that section.

Notice that the other theorem environments are numbered by the chapter and page.

\begin{verbatim}
\newtheorem{thm}{Theorem}[chapter]
\newtheorem{cor}{Corollary}[page]
\end{verbatim}

Notice the differences between the output of these environments and that of the lemma.

The next new topic that is in this document is cross-referencing. Notice the three commands

\begin{verbatim}
\label{lemma:first}, \ref{lemma:first} and \pageref{lemma:first}
\end{verbatim}

The label command has the general syntax \texttt{\label{labelref}} where \texttt{labelref} is any label you want. You could use lemma1, FundThmofFinAbelGroups, GreensThm or even George. I would recommend that you use something that is not easily duplicated and is representative of the theorem you are referencing. The general syntax of the \texttt{ref} command is \texttt{\ref{labelref}}. This command will print the current value of the number of the theorem that is referenced by \texttt{labelref}. Finally, the syntax for the \texttt{pageref} command is \texttt{\pageref{labelref}}. This command will print the current page number that the theorem that is referenced by \texttt{labelref} is on. You may think that this is a bit of overkill but this little extra work can save hours when you are editing a large work. For example, say you did not use these cross referencing features but instead typed in the theorem numbers and page numbers by hand. What would happen if you decided to move a
chapter, interchange a couple sections, combine two chapters into one or add a chapter or two in the middle of the document? What would happen is that you would have to read through each page of the document and edit all of the references. If you used the cross referencing features all you would need to do is recompose the document and all of the changes would be made automatically. If you are headed to graduate school you will be writing documents of this magnitude and be doing major editing to them.

Creating new counters is easy and useful, especially if you do not want to use the default numbering systems available with \LaTeX. To create a new counter simply use the command \texttt{\newcounter{counter}} or \texttt{\newcounter{counter}[numscheme]} where \texttt{counter} is the name of the new counter and \texttt{numscheme} is the numbering style that resets your counter when it changes. For example,

\begin{verbatim}
\newcounter{don_s_counter_1}
\end{verbatim}

creates a new counter named \texttt{don\_s\_counter\_1}. Since there is no \texttt{numscheme} the counter keeps incrementing throughout the document and does not reset, unless you reset it with the \texttt{setcounter} command. The command

\begin{verbatim}
\newcounter{don_s_counter_2}[section]
\end{verbatim}

creates a new counter named \texttt{don\_s\_counter\_2} and resets each time the section number is updated, that is, each time we move to a new section of the document.

You need to update your own counters manually. To update a counter use the \texttt{stepcounter} command. For example,

\begin{verbatim}
\stepcounter{don_s_counter_1}
\end{verbatim}

increments the \texttt{don\_s\_counter\_1} counter. To print the value of the counter at any time in the document use either the \texttt{\alph}, \texttt{\arabic}, \texttt{\roman} or \texttt{\fnsymbol} commands. For example, the code

\begin{verbatim}
\newenvironment{dlaw}
{\noindent{\bf Don’s Law} \arabic{don_s_counter_1}: } \}
{\vskip 10pt \stepcounter{don_s_counter_1}}
\end{verbatim}

created a new environment called \texttt{dlaw}. In the pretext portion of the definition the command

\begin{verbatim}
\arabic{don_s_counter_1}
\end{verbatim}

prints the counter number using Arabic numbers. In the posttext portion of the definition the command \texttt{\stepcounter{don_s_counter_1}} increments the counter. Note that counters all start at and are reset to 0.

Whenever you use any counting environment, like theorems, chapters or our own counters you must compose the document several times, either two or three. This is because when you have a counter or cross reference the first composition of the file will write information to the disk containing the reference numbers and page numbers that are needed. The second composition of the document will use the saved data to fill in the appropriate references in the document. There are some references that require a third composition but most of the time two will be sufficient.

\section*{1.4 Including External \LaTeX\ Files into a Document}

When your project gets really big, like a book, you will want to break the document up into smaller files and paste them together at the end. For example when you are writing a book you may want to place each chapter or section in its own file and then have the computer join them together into a book when you are finished. \LaTeX offers a very simple way to do this using the \texttt{\input} and the \texttt{\input} commands.

The \texttt{\input{file}} command causes the indicated \texttt{file} to be read and processed, exactly as if its contents had been inserted in the current file at that point. The file name may be a complete file name with extension or just a first name, in which case the file extension of .tex is used.

The \texttt{\include{file}} command is similar to the \texttt{\input{file}} command except that included files are automatically placed on the next page of the document. That is, the \texttt{\include{file}} command is equivalent to

\begin{verbatim}
\include{file}
\end{verbatim}
\clearpage \input{file} \clearpage

The \texttt{\include{file}} command can also be used in conjunction with the \texttt{\includeonly} command for selective inclusion of files. If file is one the file names in the file list of the \texttt{\includeonly} command the \texttt{\include} command is again equivalent to
\clearpage \input{file} \clearpage

except that if the file included file does not exist, then a warning message rather than an error is produced. If the file is not in the file list, the \texttt{\include} command is equivalent to \texttt{\clearpage}. Unlike the \texttt{\input} command, the \texttt{\include} command may not appear in the preamble or in a file read by another \texttt{\include} command.

The \texttt{\includeonly{file_list}} command controls which files will be read in by an \texttt{\include} command. The file list should be a comma-separated list of filenames. Each filename must match exactly a filename specified in a \texttt{\include} command. This command can only appear in the preamble of the document.

For example,
\begin{verbatim}
\input{section1_1.tex}
\end{verbatim}

will read the text from the \texttt{section1_1.tex} file and place it at the location of the input command. The command
\begin{verbatim}
\include{section1_1}
\end{verbatim}

will do the same thing except that the contents of the \texttt{section1_1.tex} file will begin on a new page. The code
\begin{verbatim}
\documentclass[10pt]{report}
\includeonly{section1,section3}
\begin{document}
\include{section1}
\include{section2}
\include{section3}
\end{document}
\end{verbatim}

will include only the files section1.tex and section3.tex.