Illinois Wesleyan's Introduction to LaTeX

A guide for students brought to you by the Department of Mathematics

Updated May 21, 2015

ILLINOIS WESLEYAN
UNIVERSITY

Contents

Acknowledgements

Much of the material in this document is modeled after *An Introduction to Using TEX in the Harvard Mathematics Department*, which can be found at http://www.math.harvard.edu/texman/. Many people were involved in the production of this guide. The original idea was presented by Melvyn Jeter, and the original project was carried out under the leadership of Zahia Drici during the 2014-15 academic year. Debbie Gorden designed the document, while Jacob Henry '16, Matt Kellett '16, and Stephen Lachowski '15 did a lot of the supporting technical work. Thanks also to Jinze Zheng '17 for contributing some updates. This is considered a dynamic document, and is overseen by Dan Roberts.

1 THE BEGINNING

We are excited about bringing more students alongside LaTeX and will be outlining basics here. This document will not begin to touch on all the "**TEX**" services that are available, but it will be a document of "helps" that will hopefully enable you to do class work in "**TEX**". At the very least these notes should enable you to typeset most coursework assignments. Feel free to ask others for help. Everyone has learned a trick or two and are typically willing to share what they have learned. If you are so inclined you may refer to books on LaTeX that describe fancier techniques. Also, online resources can be very helpful.

2 SOME HISTORY

During the summer of 1978, Donald Knuth developed his first version of "**TEX**." In 1989, he released new versions of "**TEX**" and METAFONT. The science of programming languages continues to grow rapidly. Since the source code of "**TEX**" is essentially in the public domain other programmers are allowed and encouraged to improve the system. This free access allows the source code to continuously evolve. However, each one is required to use another name to distribute the modified "**TEX**." Donald Knuth's original bare-bones "**TEX**", which is a very powerful engine, lacked many ready-made useful macros that make typesetting a lot easier for the beginner and the casual user. It is important to note that all the different "**TEXs**" have been built on top of Knuth's original.

3 PROGRAMS USED

In the Mathematics Department [2nd floor, east side in the Center for Natural Science] you will find computer labs E202, E204, and E210. In these labs the "$\mathbf{T_EX}$" programs called TeXWorks and TeXShop are used. TeXShop is by far the easier program to use.

4 DOWNLOADING LATEX

There are two parts to generating a LaTeX document: the front end (editor) and the language. The front end is how you interact with the language, i.e. typing it. The language is a set of programs and libraries that tell the computer how to generate a document from the code that you type into the front end editor. There are many LaTeX distributions available. Here are a few free options.

Program	**Notes**	**Obtaining**
TeXShop	Mac only	University of Oregon website
MiKTeX	PC only	miktex.org
TeXworks	Mac and PC	texworks webpage

5 FILE MAKING

At this point if you are not familiar with the preliminary makings of a file; i.e. create document, name it, and edit it you need to find some one who can teach you those steps.

"$\mathbf{T_EX}$" is not a word processor, but a formatter. This means that you create a file containing commands for "$\mathbf{T_EX}$" using a text editor and then you use the "$\mathbf{T_EX}$" program to convert this file of commands into a printed document. In TeXShop these two functions are knitted together in a very efficient manner. All "$\mathbf{T_EX}$" files that you create will need the extension "*.tex*" rather than "*.doc*".

6 TEMPLATE SELECTION

There are several types of templates that you can use in LaTeX. The beginning of your file begins with a command which tells LaTeX what kind of document to make. The most commonly used templates are: article, book, and letter and the command would look like this:

```
\documentclass{article}[14]
```

The 14 dictates the size of the fonts.

Some professors have provided standard templates to use for certain courses. These can be found on the Math Department website.

7 ADDING PACKAGES

Many features of LaTeX aren't included in the standard library. To use these features, we must include other packages. The following are some of the commonly-used packages, and are listed in a random order.

Command for Including Package	**Package Description**
`\usepackage{amsmath}`	The amsmath package is the most important package of the AMS collection. Introduces several useful math environments; for example, the `align` environment.
`\usepackage{graphicx}`	This package introduces the `includegraphics` command, which is needed for inserting and formatting figures. Works for different file types including `.pdf, .eps, .png, .jpg`.
`\usepackage{amsfonts}`	The bundled distribution of the AMSFonts collection. This includes many useful symbols and fonts, such as `mathbb`.
`\usepackage{epsfig}`	Also used to include `.eps` figures.

8 DOCUMENT PARTS

8.1 PREAMBLE

Every document begins with a preamble. The preamble is everything that comes before the command

`\begin{document}`

. This is where your document specifications are set, as well as declaring all packages, global commands, default font size, and environments you intend to use. One command that must be present in every LaTeX document is

`\documentclass{}`

. For example, below is the preamble for the document that you are reading right now.

```
\documentclass{article}[14, epsf]
\usepackage{amsfonts}
\usepackage{amsmath,amscd,amssymb}
\usepackage{amsthm}
\usepackage{epsfig}
\usepackage{hyperref}
\usepackage{listings}
\lstset{basicstyle=\small\ttfamily, columns=flexible, breaklines=true}

\newtheorem{theorem}{Theorem}
```

```
\newtheorem{definition}{Definition}
\newtheorem{axiom}{Axiom}
\newtheorem{proposition}{Proposition}
\newtheorem{example}{Example}
\newtheorem{exercise}{Exercise}
\newtheorem{lemma}{Lemma}
\newtheorem{corollary}{Corollary}
```

The

```
\newtheorem
```

command is used to define environments (with counters). For more information on this command see Section 12.

8.2 STRUCTURE

Even the most basic document has some kind of structure. Often this involves a title block or page, an abstract, numbered sections, and lettered appendices. LaTeX makes it easy to organize your document this way.

8.3 TITLE

For titled documents, your preamble needs the following lines in it.

```
\title{My Document: Will Anyone Read It?}
\author{Charlie Brown}
```

Then wherever you want the title block (probably right after `\begin{document}`) put `\maketitle`. This will insert the title, author and the current date. Suppose it is February 1 but your paper is due April 9, and you want the date on the paper to be the due date. Then add `\date{April 9, 2015}` to your preamble along with the title and author commands.

If you have more than one author, they can be separated by `\and`. You can add contact information either by including line breaks `\\` with each author. Examples:

```
\author{Charlie Brown \and Woodstock}
```

```
\author{Charlie Brown \\
  Department of Cartoons \\
  \and
Woodstock \\
  Department of Cartoons }
```

8.4 ABSTRACT

Wherever you want the abstract (usually right after `\maketitle`) put
`\begin{abstract} This article breaks new ground...\end{abstract}`

8.5 SECTIONS

There are plenty of commands to indicate sections of the document.

```
\chapter       <--Only available using document classes book or report
  \section      <--The remainder are available in all document classes
    \subsection
      \subsubsection
        \paragraph
          \subparagraph
```

These tell LaTeX the hierarchy of your document, which is useful when it automatically creates a table of contents, but none of these are necessary. For example, you do not need to have a section before you use a subsection. The format of any of these is just

```
\section{Title of the Section}
You can include a "short title" which will be used in the table of contents
and other places that refer to the part by putting the short title in square
brackets:
\section[Short Title]{Long Title}
These document parts are numbered automatically. If you do not want the part to be
numbered, use the "starred" version.
\section*{Title of the Section}
The * indicates suspension of numbering for this section.
```

8.6 APPENDICES

Right before the first appendix put `\appendix`. Then use the `\section` command for each appendix. Appendices will be lettered (A, B, C, ...) rather than numbered like the sections.

Section and Appendix Example.

```
\begin{document}
\section{Introduction}
This paper is important because ...

\section{Related Research}
Other people have studied this, but they are all wrong ...

\section{Theory}
We assume the cow is a sphere, then generalize to other shapes ...

\subsection{Basic Model}
  Suppose the cow is a sphere...

\subsection{Fancy Model}
  Now suppose we have a cow of irregular shape...
```

```
\subsubsection{Four Legs}  ...

\section{Conclusions} ...

\appendix

\section{Proofs of Results} ...
```

9 TEXT MODES

Basically you will work in two different modes: *math mode* and *text mode*. *Text mode* is the default. Most commands in the *text mode* change the appearance or layout of the text on the page (i.e., size, font, boldfacing, italicizing, spacing).

A dollar sign set is used to put letters, numbers, or words into *math mode*. The command would look like this: $know$. In this case the "**T$_{\text{E}}$Xed**" word would look like this, *know*.

For example, if you need to create a mathematical formula which looks like this; $M_2 + T^7$, it can be done by using the following code: `$M_2 + T^7$`.

"**T$_{\text{E}}$X**" commands are typically preceded with a backslash. For example if you want a word or words to be underlined then you would enter the following: `\underline{word or words to be underlined}`.

Double dollar signs are used to indicate "displayed math" mode, and will pull the mathematics outside of the paragraph and center it on its own line. Like this:

$$M_2 + T^7$$

Most commands work in only one of the "**T$_{\text{E}}$X**" modes. For instance, the commands which generate math symbols, are only valid in *math mode*, i.e. enclosed in dollar signs. Similarly, commands to center text, or to typeset a table or a bibliography, work only in *text mode*.

10 FONT STYLES

We have grown accustomed to utilizing many kinds of type for emphasizing things of importance in our documents and it is the same in LaTeX.

```
\textbf{quick brown fox} creates bold face lettering
```

i.e. **quick brown fox**

```
\textit{quick brown fox} creates italic lettering
```

i.e. *quick brown fox*

```
\textsf{quick brown fox} creates san serif lettering
```

i.e. quick brown fox

`\textsf{quick brown fox} creates slanted lettering`

i.e. *quick brown fox*

`\texttt{quick brown fox} creates typewriter lettering`

i.e. `quick brown fox`

`\emph{quick brown fox} creates emphasized lettering`

i.e. *quick brown fox*

To emphasize a few words within a sentence you would do this:

```
Please \textbf{ask questions} if you run into problems with
TeX.  It would create this:
```

Please **ask questions** if you run into problems with TeX.

11 FONT SIZES

Font size can also be modified. Examples follow:

`\tiny{size}`

i.e. size

`\scriptsize{size}`

i.e. size

`\footnotesize{size}`

i.e. size

`\small{size}`

i.e. size

`\large{size}`

i.e. size

`\Large{size}`

i.e. size

`\LARGE{size}`

i.e. size

12 SPECIAL ENVIRONMENTS

It is important to note that environments must have 'a beginning' and 'an ending' that enclose the text that needs to be different than the default. Centering is a commonly needed environment and can be created in two different ways. If you are using mostly text then:

```
\begin{center}  Learning to create in LaTeX is fun. \end{center}
```

is appropriate. Using these commands around this text would create this:

Learning to create in LaTeX is fun.

If you want to center mathematical symbols you could use these symbols:

```
$$a=b+c+d+e$$
```

and they would create this:

$$a = b + c + d + e$$

To have a numbered equation you use the commands

```
\begin{equation}  and  \end{equation}
```

and they would create

$$a = b + c + d + e \tag{1}$$

To create an enumeration environment you use:

```
\begin{enumerate}
\item First item
\item Second item
\item Third, and final, item
\end{enumerate}
```

which would create:

1. First item
2. Second item
3. Third, and final, item

the default of the enumerating environment. If you wanted letters instead of integers you would use these commands:

```
\begin{enumerate}
\item[a] First item
\item[b] Second item
\item[c] Third, and final, item \end{enumerate}
```

and it would look like this:

a First item

b Second item

c Third, and final, item

Whatever you put in the square brackets will be what is displayed for enumerating the list.

There are other environments that aren't included in the basic LaTeX library, namely theorems, propositions, examples, exercises, lemmas, and corollaries. These are created in a two-step process. The command `\newtheorem` followed by curly bracket sets create new environments when you include them in the header of the document. For example, if we want to create a "Theorem" environment, we would include the following in our preamble.

```
\newtheorem{theorem}{Theorem}
```

In the first set of brackets, we type what the environment name is. In the second set of brackets, we type what we want displayed when that environment is called. This would allow us to use the environment

```
\begin{theorem}
\end{theorem}
```

to generate the following heading.

Theorem 1.

We can create any environment we want, such as

Lemma 1.

Corollary 1.

The "proof" environment is predefined and included in the `amsthm` package. The open box placed at the end of the proof is included.

Proof. Proof goes here. □

13 BIBLIOGRAPHY

A bibliography is incorporated into a LaTeX document via the `thebibliography` environment. Here is an example of how to type a bibliography in code. The 99 in set brackets denotes the number of characters in the longest entry.

```
\begin{thebibliography}{99}
\bibitem{Lin and Shyu}Chiang Lin and Tay-Woei Shyu, A Necessary and
    Sufficient Condition for the Star Decomposition of Complete Graphs,
    \emph{Journal of Graph Theory}, 23, no. 4, 1996, 361 -- 364.
```

```
\bibitem{Tarsi 1}M. Tarsi, Decomposition of complete multigraphs into
    stars, \emph{Discrete Math.}, 26, 1979, 273 -- 278.
\bibitem{Yamamoto et al.}S. Yamamoto, H. Ikeda, S. Shige-eda, K. Ushio,
    and N. Hamada, On claw-decomposition of complete graphs and complete
     bigraphs, \emph{Hiroshima Math. J.}, 5, 1975, 33 -- 42.
\end{thebibliography}
```

References

[1] Chiang Lin and Tay-Woei Shyu, A Necessary and Sufficient Condition for the Star Decomposition of Complete Graphs, *Journal of Graph Theory*, 23, no. 4, 1996, 361 – 364.

[2] M. Tarsi, Decomposition of complete multigraphs into stars, *Discrete Math.*, 26, 1979, 273 – 278.

[3] S. Yamamoto, H. Ikeda, S. Shige-eda, K. Ushio, and N. Hamada, On claw-decomposition of complete graphs and complete bigraphs, *Hiroshima Math. J.*, 5, 1975, 33 – 42.

If you prefer choosing the labels, as opposed to numbers, for the entries in your bibliography, then you can use square brackets.

```
\begin{thebibliography}{[ER]}
\bibitem{[Z]}Y. Zhang, Bounded gaps between primes, \emph{Ann. of Math.
    (2)} 179 (2014), no. 3, 1121 -- 1174.
\bibitem{[ER]}P. Erd\"{o}s and R. Rado, A combinatorial theorem, \emph{J.
     London Math. Soc.} \textbf{25} (1950), 249 -- 255.
\end{thebibliography}
```

References

[Z] Y. Zhang, Bounded gaps between primes, *Ann. of Math. (2)* 179 (2014), no. 3, 1121 – 1174.

[ER] P. Erdös and R. Rado, A combinatorial theorem, *J. London Math. Soc.* **25** (1950), 249 – 255.

14 NUMBERED EQUATIONS

You may want to number an equation in order to refer to it later in the document. There are many ways to accomplish this, but perhaps the most straightforward way is by using the `equation` environment.

```
\begin{equation} \label{Euler Eq}
V-E+F=2
\end{equation}
```

produces the following.

$$V - E + F = 2 \tag{2}$$

The numbering is automatically kept by LaTeX. The `\label` command is attached so that we can refer to this equation later in the text. For example, we might want to display the following senctence. We can see from equation 2 that the number of vertices, edges, and faces of a polygon are intricately connected. To accomplish this, we would type the following.

```
We can see from equation \ref{Euler Eq} that the number of
vertices, edges, and faces of a polygon are intricately connected.
```

15 SUB- AND SUPER-SCRIPTING

A subscript is indicated by the underscore, while the superscript is indicated by the caret. Consider the following examples:

```
$\alpha_j$, $\alpha^j$, $\alpha_j^i$ and $\alpha_{j^i}$ yield
```

α_j, α^j, α_j^i and α_{j^i}, respectively.

This brings us to a point about using the curly brackets { }. They are used a lot in math mode to group things together. Many math commands in **TEX** are defined to operate on the next item which appears in the file: either a single letter, a command word, or a group delimitted by brace. For instance, the subscripting operator (the underscore) can be used to subscript a single letter or a single command. For example,

```
$a_b$ or $a_\beta$
```

But if you want a subscript containing something more complicated, say $a_{b+\beta}$, then you must use the braces as follows:

```
$a_{b+\beta}$
```

Another example of when the curly braces must be used is if you want the subscripts lower and smaller. Then, you would type

```
$\alpha_{_j}$
```

This applies the lowering operator twice to yield $\alpha_{_j}$.
Please note that

```
$\alpha__j$
```

will generate an error message.

16 TEXT IN MATH MODE

In math mode, TEX ignores all spaces. It has complicated built-in rules which it uses to determine how the equation should be spaced. (For instance, it knows to put a little extra space around + and = signs.) But sometimes you want to put words in the middle of an equation. For instance, you might want to generate

$$\{\phi \mid \phi \text{ is an additive character}\}$$

The way you do this is to use the `\text{ }` command to tell $LaTeX$ to go back into text mode briefly. In our example, you would type
`$$ \{\phi \mid \psi \text{ is an additive character}\} $$.`

This brings up a point about using math mode properly. In math mode, TEX assumes that any string of letters next to one another is a product, and spaces it accordingly. If, in fact, it was a word, it won't look good. So, for symbols which are actually parts of words, like Stab, Spec, Proj, etc., you should use `\operatorname{ }` to tell $LaTeX$ to space the letters in the symbol as text, not as a product. For instance, type `$$ X := \operatorname{Spec}A $$`
to get

$$X := \operatorname{Spec} A$$

Many common functions have a designated command to make them appear in regular typeface. For example, if we type `$cos(\alpha)$` then LaTeX will display $cos(\alpha)$. The easy way to fix this is by instead typing `$\cos(\alpha)$`, so that $\cos(\alpha)$ is displayed. Some other common functions that carry this option are `\sin, \tan, \sec, \csc, \cot, \ln`.

17 SEQUENCES OF EQUATIONS

To get a bunch of equations all lined up nicely, you can do it like this with `\eqnarray*`:

```
\begin{eqnarray*}
i_w \cdot \psi_v \cdot N_w &=& \psi_w \cdot i_w \cdot N_w\\
&=& \psi_w \cdot i_w \cdot \sum_{\sigma \in G_w} \sigma\\
&=& \psi_w \cdot \sum_{\sigma \in G_w} \sigma\\
&=& i_w \cdot N_w \cdot \psi_w.
\end{eqnarray*}
```

yields

$$\begin{aligned} i_w \cdot \psi_v \cdot N_w &= \psi_w \cdot i_w \cdot N_w \\ &= \psi_w \cdot i_w \cdot \sum_{\sigma \in G_w} \sigma \\ &= \psi_w \cdot \sum_{\sigma \in G_w} \sigma \\ &= i_w \cdot N_w \cdot \psi_w. \end{aligned}$$

A few things to notice are

1. `&` surrounds the symbol at which the equations are lined up–in this case, the = sign.
2. The lines end with `\\`, which denotes a carriage return.

Also, note that the $*$ means that the equations won't be numbered. So, if one wanted the equations to be numbered, s/he would use

```
\begin{eqnarray}
...
\end{eqnarray}
```

There are many other ways to format your equations. Too many to list here. Please note that they are available to learn online.

18 GETTING STUFF ON TOP OF OTHER STUFF

If you want to make a fraction, use the `\frac` command; `\frac` works on everything contained in a pair of curly brackets, or the entire equation if there is no enclosing pair of curly brackets. For example,

```
$$\lim_{\delta\rightarrow 0} \frac{f(x + \delta) - f(x)}{\delta}$$
```

gives

$$\lim_{\delta\rightarrow 0} \frac{f(x + \delta) - f(x)}{\delta}$$

19 BRACKETING

You can get all different kinds of sizes of parentheses, brackets, curly brackets, etc. to make your formulas look good. Typing

```
$$  \Biggl(  \biggl(  \Bigl(  \bigl(  ()  \bigr)  \Bigr)  \biggr)  \Biggr)  $$
```

gives you

$$\Biggl(\biggl(\Bigl(\bigl(()\bigr)\Bigr)\biggr)\Biggr)$$

Analogous commands work for [and `\{`. Note the `\` used to get curly braces.

You can also let TEX figure out how big a pair of parentheses should be to fit the contained expression. Use`\left(` and `\right)`, like this:

```
$$  \left( \sum a_i^2 \right)^{1/2} $$
```

gives

$$\left(\sum a_i^2 \right)^{1/2}$$

Of course, you can also use `\left[, \right], \left\{` and `\right\}`.

20 MATRICES

Generating matrices is similar to the process of constructing tables. New rows are made by typing two backslashes, `\\`, while new columns are made using the ampersand `(&)`. For example,

```
$$
\begin{matrix} 1 & 0 & 0\\
0 & 1 & 1\\
```

```
0 & 0 & 1
\end{matrix}
$$
```

yields

$$\begin{matrix}
1 & 0 & 0 \\
0 & 1 & 1 \\
0 & 0 & 1
\end{matrix}$$

Note, however, that no delimiters are present. Parentheses, square brackets, and absolute value bars may be added quickly by replacing "matrix" with "pmatrix," "bmatrix," or "vmatrix." Consider the following examples:

```
$$
\begin{matrix}
\begin{pmatrix}
1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{pmatrix}
&&& \begin{bmatrix}
1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{bmatrix}
&&&
\begin{vmatrix} 1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{vmatrix}}
\end{matrix}
$$
```

yields

$$\begin{matrix}
\begin{pmatrix}
1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{pmatrix}
&&& \begin{bmatrix}
1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{bmatrix}
&&&
\begin{vmatrix} 1 & 0 & 0\\
0 & 1 & 1\\
0 & 0 & 1
\end{vmatrix}}
\end{matrix}$$

Binomial coefficients: You might be tempted to think that this is the only way to generate certain expressions, such as the binomial coefficient. In this case, you can simply use the commands `\binom{n}{k}` or `{n\choose k}` (or `\dbinom{n}{k}` for display mode). Both commands generate $\binom{n}{k}$.

In some contexts, it is useful to denote high-order matrices by truncating some of the entries. If, for instance, you wanted to display the nth order identity matrix, you may wish to omit unnecessary lines of commands rather than type n lines of zeros and ones by using ellipses.

$$\begin{matrix}
1 & 0 & \cdots & 0\\
0 & 1 & \cdots & 0\\
\vdots & \vdots & \ddots & \vdots\\
0 & 0 & \cdots & 1
\end{matrix}$$

The above matrix was generated with the following code.

```
$$
\begin{matrix}
1 & 0 & \cdots & 0\\
0 & 1 & \cdots & 0\\
\vdots & \vdots & \ddots & \vdots\\
0 & 0 & \cdots & 1
\end{matrix}
$$
```

Another matrix environment you may wish to use gives the user the ability to put label the borders of a matrix, much like labeling rows and columns in a table.

21 CASE STATEMENTS

To get a case statement like:

$$x_\lambda = \begin{cases} x & \text{if } \lambda \text{ is an eigenvalue;}\\ -x & \text{if } -\lambda \text{ is an eigenvalue;}\\ 0 & \text{otherwise.} \end{cases}$$

type

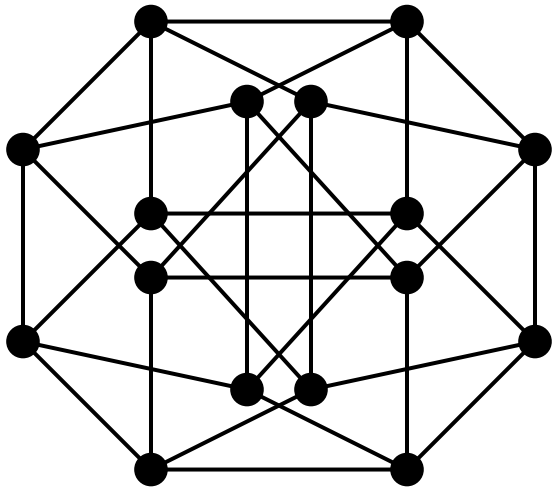
```
$$
x_ \lambda =
\begin{cases}
x & \text{if $\lambda$ is an eigenvalue;}\\
-x & \text{if $-\lambda$ is an eigenvalue;}\\
0 & \text{otherwise.}
\end{cases}
$$}
```

22 INCLUDING PICTURES

Many different file types can be accommodated by LaTeX. As mentioned in Section 7, the `graphicx` package allows for a wide variety of files to be included in your document. Assuming that we have included the command `\usepackage{graphicx}` in the preamble of the document, we can use the code

```
\begin{center}
\includegraphics[scale=0.75]{4cube.pdf}
\end{center}
```

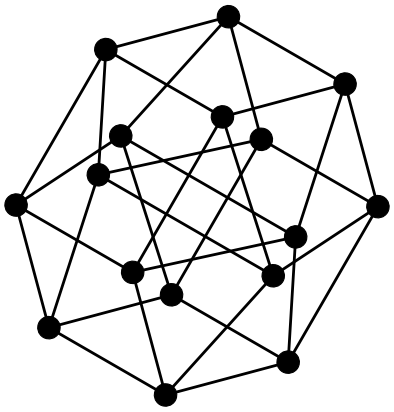
to include the file `4cube.pdf` displayed below. The option, `[scale=0.75]`, tells LaTeX to display the file at 75% of its original size.



We have a lot of control over the format of the display of the file. The general command `\includegraphics[a1=v1, a2=v2,...,an=vn]{filename}` tells LaTeX to display *filename* with the given attributes $a1, a2, \ldots, an$ set to values $v1, v2, \ldots, vn$, respectively. Mutable attributes include `scale, height, width, angle`, etc. For example, if we wish to display the same file as above, but fix the width to be 2 inches, and rotate the graph counter-clockwise by 60°, then we would enter the following code.

```
\begin{center}
\includegraphics[width=2in, angle=60]{4cube.pdf}
\end{center}
```

This would display



Everything mentioned above for the file extension .pdf will also work for the other filetypes that `graphicx` allows. See Section 7 for a list.

23 TABLES

To generate tables, use the `tabular` environment. For example,

```
\begin{center}
\begin{tabular}{c|ccccccc|}
$\lambda$& (5)& (4,1)& (3,2)& (3,1,1)& (2,2,1)& (2,1,1,1)&
(1,1,1,1,1)\\
\hline
$d_{_\lambda}$& 1& 4& 5& 6& 5& 4& 1\\
\end{tabular}
\end{center}
```

generates

λ	(5)	(4,1)	(3,2)	(3,1,1)	(2,2,1)	(2,1,1,1)	(1,1,1,1,1)
$d_{_\lambda}$	1	4	5	6	5	4	1

The string {c|ccccccc|} describes the layout of the table. The 8 `c`'s specify that the table has 8 columns, and the entries in each column are to be centered. You can also use `l` and `r` to make columns in which the entries are, respectively, left- and right-justified. The vertical bar characters, | , indicate that there are

to be vertical lines between the first and second columns, and after the eighth. The table is enclosed in a `center` environment to center the table as a whole on the page.

The text of the table itself appears after the string specifying the column layout. Entries in the table are separated by the ampersands &, and each row is ended with a carriage return command,\\ . The `\hline` command produces a horizontal line which runs the width of the table. The tabular environment and its relatives can be used to produce some pretty sophisticated tables; consult the internet for more information.