

# Scholarship Skills

Tim Sheard & Andrew Black  
Portland State University

L<sup>A</sup>T<sub>E</sub>X and B<sup>I</sup>B<sub>T</sub>E<sub>X</sub>

© 1996, 1997, 1999, 2000, 2018-19 David  
Maier  
© 2001 Todd Leen © 2006 Tim Sheard

# LaTeX

## LaTeX is a popular type-setting tool used by many computer scientists and mathematicians

- It has great support for type-setting mathematics (including a sophisticated macro system)
- It comes with supporting tools for managing bibliographic information – BibTeX
- Source is text and can be managed with Revision Control Systems such as SVN and CVS
- Since source is text, tools can easily create input for tables and figures.
- It's open-source, and free.
- Web-based versions available: ShareLaTeX, Overleaf



# Resources for LaTeX

- The most common version of LaTeX is pdf $\LaTeX$ .
- Online help for  $\LaTeX$ 
  - [http://www.emerson.emory.edu/services/latex/latex2e/latex2e\\_toc.html](http://www.emerson.emory.edu/services/latex/latex2e/latex2e_toc.html)
  - <http://tug.org/texlive/>
  - <http://www.giss.nasa.gov/tools/latex/>
  - <http://nwalsh.com/tex/texhelp/LaTeX.html>
  - <https://www.math.ucsd.edu/~wcheung/texforwindows.html>
    - TeX & LaTeX for mathematicians on windows!
- Free online book on LaTeX
  - “The **Not So** Short Introduction to LaTeX2e”
  - By Tobias Oetiker, Hubert Partl, Irene Hyna and Elisabeth Schlegl
  - <http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf>

# Markup Commands

- **LaTeX is a *markup language***

- Text and commands are interspersed in the same document
- Commands are alpha-strings preceded by a backslash
- Commands can have arguments and options
  - Arguments appear inside `{ }` . If a command has arguments you must supply them.
  - Options appear inside `[ ]` . Options need not be supplied (they have default values)

- Examples

```
\alpha
```

```
\begin{document}
```

```
\documentstyle[twoside]{report}
```

```
\begin{array}[t]{cl}
```

- Many commands come in pairs

```
\begin{centering}
```

```
\end{centering}
```

# Basic Setup

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

\begin{document}

\title{LaTeX for Scholarship Skills}
\author{
Tim Sheard\\
Computer Science Department\\
Maseeh College of Engineering and Computer
Science\\
Portland State University\\
}

\maketitle

\section{Introduction}

This document is an introduction to the use of
LaTeX for Scholarship Skills class members. In it I
will try and outline the basic operations
available in LaTeX.

\end{document}
```

LaTeX for Scholarship Skills

Tim Sheard  
Computer Science Department  
Maseeh College of Engineering and Computer Science  
Portland State University

February 9, 2006

## 1 Introduction

This document is an introduction to the use of LaTeX for Scholarship Skills class members. In it I will try and outline the basic operations available in LaTeX.

# Structure

Specifies what type of document and the default font size.

Every document begins with `\begin{document}` and ends with `\end{document}`

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

\begin{document}

\title{LaTeX for Scholarship Skills}
\author{
Tim Sheard\\
Computer Science Department\\
Maseeh College of Engineering and
Computer Science\\
Portland State University\\
}

\maketitle

\section{Introduction}

This document is an introduction to
the use of LaTeX for Scholarship
Skills class members. In it I will
try and outline the basic
operations available in LaTeX.

\end{document}
```

Title and author are specified with `\title{ ... }` and `\author{ ... }`

`\maketitle` tells where in the document the title should appear.

The `\section{ ... }` command introduces a new section whose name is ...

# Separating commands from text

In a LaTeX source file only certain characters are allowed. All other characters are created in the output by using commands.

## Allowed Characters

Upper Case Alpha: `ABCDEFGHIJKLMNOPQRSTUVWXYZ`

Lower Case Alpha: `abcdefghijklmnopqrstuvwxyz`

Digits: `0123456789`

Punctuation: `. : ; , ? ! ` ' ( ) [ ] - / * @`

Unless these characters are inside commands: if they are in the input, they will appear in the output.

## Special Characters

Used only inside LaTeX commands: `#$%&~_^ \{ }`

Used in Math Formulas: `+ = | < >`

# LaTeX Sources

```

\documentclass[12pt]{article}

\begin{document}

ABCDEFGHIJKLMNOPQRSTUVWXYZ

abcdefghijklmnopqrstuvwxyz

0123456789

.:; , ? ! ` ' ( ) [ ] - / * @

\# \ $ \% \& \_ \{ \}
\verb+~+
\verb+^+ \verb+\+

\end{document}

```

```

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789
.:; , ? ! ` ' ( ) [ ] - / * @
# $ % & _ { } ~ ^ \

```

Special characters  
require special  
commands to produce.



# Itemized lists

My favorite things (in no special order) include:

```
\begin{itemize}
\item Red Cats.
\item Blue Pajamas
\item Pink Elephants
\end{itemize}
```

But, if I had to place them in order, it would have to be:

```
\begin{enumerate}
\item Pink Elephants
\item Red Cats.
\item Blue Pajamas
\end{enumerate}
```

My favorite things (in no special order) include:

- Red Cats.
- Blue Pajamas
- Pink Elephants

But, if I had to place them in order, it would have to be:

1. Pink Elephants
2. Red Cats.
3. Blue Pajamas

# Labels and references

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

\begin{document}

\section{In the beginning}\label{alpha}
```

You have to start some where,  
otherwise you'll never get to the end.  
More about this in Section \ref{omega}.

```
\section{At the end}\label{omega}
```

When all is said and done, I'd rather be  
at the beginning (see Section  
\ref{alpha}) than at the end.

```
\end{document}
```

## 1 In the beginning

You have to start some where, otherwise you'll never get  
to the end. More about this in Section 2

## 2 At the end

When all is said and done, I'd rather be at the beginning  
(see Section 1) than at the end.

# Footnotes

```

\documentclass[14pt]{article}

\begin{document}

There are lots of thing I
never learned in Scholarship
Skills \footnote{But, using
LaTeX wasn't one of them.}
that I haven't used since. But
I do not regret taking the
course\footnote{I \emph{do}
regret losing my class
notes!}.

\end{document}

```

There are lots of things I never learned in Scholarship Skills<sup>1</sup> that I haven't used since. But I do not regret taking the course<sup>2</sup>.

---

<sup>1</sup>But, using LaTeX wasn't one of them.  
<sup>2</sup>I *do* regret losing my class notes!

# Sectioning

```
\section{The Biggest Stuff}\label{A}
Sections are the largest
parts of an article.
```

```
\subsection{The Next Stuff}\label{B}
Sub-sections are slightly
smaller.
```

```
\subsubsection{Down A Bit More}\label{C}
Sub-sub-sections really divide
the text.
```

```
\paragraph{About at the bottom.}
Only sentences\\ are smaller
than paragraphs.
```

```
\section{Discussion}
```

```
Biggest in Section \ref{A}.\\
Next in Subsection \ref{B}.\\
Down in Subsubsection \ref{C}.\\
```

## 1 The Biggest Stuff

Sections are the largest parts of an article.

### 1.1 The Next Stuff

Sub-sections are slightly smaller.

#### 1.1.1 Down A Bit More

Sub-sub-sections really divide the text.

**About at the bottom.** Only sentences are smaller than paragraphs.

## 2 Discussion

Biggest in Section 1.

Next in Subsection 1.1.

Down in Subsubsection 1.1.1.

# Tables

```
\begin{tabular}{l|c|r|} \hline
left & centered & right \\ \hline
big & little & small \\
Thomas & Richard & Harrison \\ \hline
\end{tabular}
```

left	centered	right
big	little	small
Thomas	Richard	Harrison

## Try it without vertical rules:

```
\begin{tabular}{lcr}
left & centered & right \\ \hline
big & little & small \\
Thomas & Richard & Harrison \\
\end{tabular}
```

left	centered	right
big	little	small
Thomas	Richard	Harrison

# Mathematics

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

\begin{document}
\[ x' + 2x^{2+y} =
  \frac{z_{i-1} * w^{j+1}}
  {\sqrt{3m}} \]
\end{document}
```

$$x' + 2x^{2+y} = \frac{z_{i-1} * w^{j+1}}{\sqrt{3m}}$$

# Inline Mathematics

Consider the coefficients  $a$ ,  
 $b$  and  $c$  in  
the equation  $ax^2 + bx + c$   
 $=$



Consider the coefficients  $a$ ,  $b$  and  $c$  in  
the equation  $ax^2 + bx + c =$

Don't use  $\textit{word}$  for italics. (Why?)



Begin typesetting math

\[

Exponentiation by  
superscripting $x' + 2x^{2+y} =$ 

The top part of a fraction

\frac{z\_{i-1} \* w^{j+1}}

subscripting

\{\sqrt{3m}\}

\]

The bottom part of a fraction

End typesetting math

$$x' + 2x^{2+y} = \frac{z_{i-1} * w^{j+1}}{\sqrt{3m}}$$

# Floating Figures

```

\section{Weather on Mars}
\begin{figure}
\hspace*{1in}
\begin{tabular}{|l||c|c|c|} \hline
& Today & Yesterday & Tomorrow \\ \hline
A & 356 & 22 & 18 \\ \hline
B & 851 & 456 & 129 \\ \hline
\end{tabular}
\caption{Temperature in degrees K, at sites A and B on Mars.}
\label{mars} \hrule
\end{figure}

```

In Figure `\ref{mars}` we report the temperature at A and B.

	Today	Yesterday	Tomorrow
A	356	22	18
B	851	456	129

Figure 1: Temperature in degrees K, at sites A and B on Mars.

## 1 Weather on Mars

In Figure 1 we report the temperature at A and B.

# Figures can be imported

`\section{An Alternative to Textual Error Messages}`

We have built a plugin for the Eclipse environment that addresses the problems with error messages that were revealed by the formative study.

The plugin is called Refactoring Annotations, ...

In general, Refactoring Annotations can be thought of as graphical error messages; specifically, the current plugin displays violated preconditions for the `\refacName{Extract Method}` refactoring.

```
\begin{figure}
  \centering
  \includegraphics[scale=\figureScale]
{annsOk}
  \caption{Refactoring Annotations overlaid
on program text.
```

The programmer has selected two lines (between the dotted lines) to extract. Refactoring Annotations show how the variables will be used:

`\texttt{front}` and `\texttt{rear}` will be parameters, as indicated by the arrows into the code to be extracted, and `\texttt{trued}` will be returned, as indicated by the arrow out of the code to be extracted.

```
\label{fig:annsOk}
\end{figure}
```

File in same directory as .tex file, or declare "graphicspath"

In file  
annsOk:

```
boolean areWheelsTrue(){
  Wheel front = bike.getFrontWheel();
  Wheel rear = bike.getRearWheel();
  boolean trued = isWheelTrue(front);
  trued = trued && isWheelTrue(rear);
  return trued;
}
```

violated preconditions, programmers need expressive, distinguishable, and understandable feedback that conveys the meaning of precondition violations; this is the focus of the remainder of this article.

### 3 AN ALTERNATIVE TO TEXTUAL ERROR MESSAGES

We have built a plugin for the Eclipse environment that addresses the problems with error messages that were revealed by the formative study. The plugin is called Refactoring Annotations, and can be downloaded from [http://multiview.cs.pdx.edu/refactoring/refactoring\\_annotations](http://multiview.cs.pdx.edu/refactoring/refactoring_annotations). In general, Refactoring Annotations can be thought of as graphical error messages; specifically, the current plugin displays violated preconditions for the EXTRACT METHOD refactoring.

The programmer starts using the Refactoring Annotations tool by selecting some program text. Refactoring Annotations overlay the program text to express control- and data-flow information about the programmer's selection. Each variable is assigned a distinct color, and each occurrence of the variable is highlighted, as shown in Figure 3. Across the top of the selection, an arrow points to the first use of a variable whose value that will have to be passed as an argument into the extracted method. Across the bottom, an arrow points from the last assignment of a variable whose value will have to be returned. L-values have black boxes around them, while r-values do not. An arrow to the left of the selection indicates that control flows from beginning to end.

These annotations are intended to be most useful when preconditions are violated, as shown in Figure 4. When the selection contains assignments to more than one variable, multiple arrows are drawn leaving the bottom, showing multiple return values (Figure 4, top). When a selection contains a conditional return, an arrow is drawn from the

```

boolean areWheelsTrue() {

    Wheel front = bike.getFrontWheel();
    Wheel rear = bike.getRearWheel();

    boolean trued = isWheelTrue(front);
    trued = trued && isWheelTrue(rear);

    return trued;
}

```

Fig. 3. Refactoring Annotations overlaid on program text. The programmer has selected two lines (between the dotted lines) to extract. Refactoring Annotations show how the variables will be used: `front` and `rear` will be parameters, as indicated by the arrows into the code to be extracted, and `trued` will be returned, as indicated by the arrow out of the code to be extracted.

```

void goOnVacation() {

    Bike roadBike = getRoadBike();
}

```

# Citations

```
\documentclass[14pt]{article}  
\begin{document}
```

```
We studied four papers in lecture. The first, by  
Scott\cite{Scott92}, is a book. Then second, by  
Cambers and Leavens\cite{Chambers95}, is a journal  
paper. The third, by Heiler and  
Rosenthal\cite{Heiler85}, is a paper in a  
proceedings. The last paper, by Dayal and  
Smith\cite{Dayal85}, is in a collection of papers.
```

```
\bibliographystyle{plain}  
\bibliography{myBib}  
  
\end{document}
```

# Result

We studied four papers in lecture. The first, by Scott[4], is a book. Then second, by Chambers and Leavens[1], is a journal paper. The third, by Heiler and Rosenthal[3], is a paper in a proceedings. The last paper, by Dayal and Smith[2], is in a collection of papers.

## References

- [1] Craig Chambers and Gary T. Leavens. Typechecking and modules for multimethods. *ACM Transactions on Programming Languages and Systems*, 17(6):805–843, November 1995.
- [2] Umeshwar Dayal and John Miles Smith. PROBE: A knowledge-oriented database management system. In *On Knowledge Base Management Systems (Islamorada)*, pages 227–257. Springer-Verlag, 1985.
- [3] S. Heiler and A. Rosenthal. G-whiz, a visual interface for the functional model with recursion. In *Proc. Int’l. Conf. on Very Large Data Bases*, page 209, Stockholm, Sweden, August 1985.
- [4] Marla Scott. *Effective Programming in C*. Addison-Wesley, 1992.

# References

References in LaTeX are kept in a .bib file.

Use BibTeX to create the text that goes in the reference section of the paper

**The Mantra is:**

latex paper

Creates the .aux file with a list of all citation keys

bibtex paper

Finds the references in .bib and creates text

latex paper

latex paper

Inserts text for references

Gets the cross references right, the second time.

ShareLaTeX and Overleaf manage this process.

# Bibliographies with LaTeX

The bibfile stores all the data about individual papers.

Every paper is given a *key*.

The key used in the `\cite` command. This appears in the text of the paper.

```
\cite{key}
```

**Black recommends the natbib package; natbib is a reimplementation of the LATEX `\cite` command, to work with both author-year and numerical citations. It is compatible with the standard bibliographic style files, such as `plain.bst`, as well as with those for `harvard`, `apalike`, `chicago`, `astron`, `authordate`.**

**Load with `\usepackage[options]{natbib}`.**



# `\cite*` commands

The natbib package has two basic citation commands, `\citet` and `\citep` for *textual and parenthetical citations, respectively*. There are also starred versions `\citet*` and `\citep*` that print the full author list, and not just the abbreviated one. All of these may take one or two optional arguments to add text before and after the citation.

<code>\citet{jon90}</code>	-->	Jones et al. (1990)
<code>\citet[chap. 2]{jon90}</code>	-->	Jones et al. (1990, chap. 2)
<code>\citep{jon90}</code>	-->	(Jones et al., 1990)
<code>\citep[chap. 2]{jon90}</code>	-->	(Jones et al., 1990, chap. 2)
<code>\citep[see][]{jon90}</code>	-->	(see Jones et al., 1990)
<code>\citep[see][chap. 2]{jon90}</code>	-->	(see Jones et al., 1990, chap. 2)
<code>\citet*{jon90}</code>	-->	Jones, Baker, and Williams (1990)
<code>\citep*{jon90}</code>	-->	(Jones, Baker, and Williams, 1990)

# The bib file

**The bib file stores all the data about individual papers.**

**Every paper is given a *key***, which is used in the `\cite{...}` command.

**There are many kinds of references**

**We will look at 4 common kinds**

Book, journal article, proceedings paper, collection paper

**Other interesting ones are**

web page, thesis and tech report. There are many others.

Not every style file implements all kinds of reference.

# Example bib file

```
%% This BibTeX bibliography file was created using BibDesk.
```

```
%% http://bibdesk.sourceforge.net/
```

```
@techreport{haines1993,
```

```
  Address = {Pittsburgh, Pennsylvania, United States},
```

```
  Author = {Nicholas Haines and Darrell Kindred and J. Gregory Morrisett and Scott M. Nettles and Jeannette M. Wing},
```

```
  Date-Added = {2011-05-02 08:46:26 -0700},
```

```
  Date-Modified = {2011-05-02 08:50:24 -0700},
```

```
  Institution = {School of Computer Science, CMU},
```

```
  Keywords = {transactions, threads, skeins, persistence, recovery, undoability, serializability, Standard ML, modules},
```

```
  Month = {December},
```

```
  Number = {CMU-CS-93-202},
```

```
  Title = {Tinkertoy Transactions},
```

```
  Year = {1993},
```

```
  Abstract = {We describe ... }}
```

```
@book{silber2005,
```

```
  Author = {Abraham Silberschatz and Peter Baer Galvin and Greg Gagne},
```

```
  Booktitle = {Operating System Concepts},
```

```
  Date-Added = {2011-03-22 17:04:57 -0700},
```

```
  Date-Modified = {2011-03-22 17:12:35 -0700},
```

```
  Edition = {Seventh Edition},
```

```
  Pages = {xv+886},
```

```
  Publisher = {Wiley},
```

```
  Title = {Operating System Concepts},
```

```
  Url = {http://www.cetlylive.com/wp-content/uploads/2010/11/Operating-System-Concepts-7-th-Edition.pdf},
```

```
  Year = {2005}}
```

```
...
```

## Book

```
@Book{Scott92,  
  author = "Marla Scott",  
  title = "Effective  
          Programming in {C}",  
  year = "1992",  
  publisher = "Addison-Wesley"  
}
```

# Journal Article

```
@Article{Chambers95,  
  author =      "Craig Chambers and Gary T. Leavens",  
  title =      "Typechecking and Modules for  
                Multimethods",  
  journal =    "ACM Transactions on  
                Programming Languages  
                and Systems",  
  volume =     "17",  
  number =     "6",  
  pages =     "805--843",  
  month =     nov,  
  year =      "1995"  
}
```

# Proceedings paper

```
@InProceedings{Heiler85,  
  author =      "S. Heiler and A. Rosenthal",  
  title =      "{G}-Whiz, a Visual Interface for  
                the Functional Model  
                with Recursion",  
  booktitle =  "Proc. Int'l. Conf. on Very Large  
                Data Bases",  
  pages =      "209",  
  address =    "Stockholm, Sweden",  
  month =     aug,  
  year =      "1985",  
  keywords =   "VLDB",  
}
```

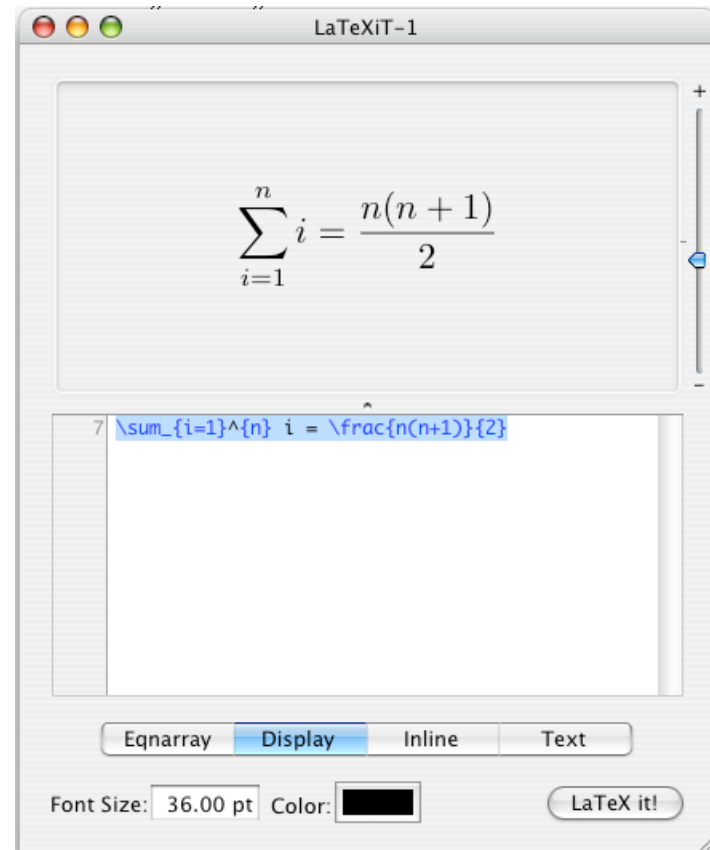
# Collections paper

```
@InCollection{Dayal85,  
  title =      "{PROBE}: {A} Knowledge-Oriented  
                Database Management System",  
  author =     "Umeshwar Dayal and John Miles Smith",  
  editor =     "Michael L Brodie and John Mylopoulous"  
  year =       "1986",  
  booktitle =  "On knowledge base management systems:  
                integrating artificial intelligence and  
                database technologies",  
  publisher =  "Springer-Verlag",  
  address =    "New York",  
  pages =      "227--257",  
}
```

Note that `@inbook` would not work, because it won't allow both author and editor. `@inbook` is for *chapters* of a book.

# LaTeX on the Macintosh

- TeXShop — freeware dual-view text editor and pdf previewer
- LaTeXiT — lets you type fragments of math, typeset them, and paste into another application, like Keynote or PowerPoint.





document.tex

435 second, to help programmers recover successfully from violated preconditions,  
 436 programmers need expressive, distinguishable, and understandable feedback that  
 437 conveys the meaning of precondition violations;  
 438 this is the focus of the remainder of this article.

439

440 `\section{An Alternative to Textual Error`  
 441 `Messages}\label{sec:basicRefAnnsDescription}`

442

443 We have built a plugin for the Eclipse environment that addresses the problems with  
 error messages that were revealed by the formative study.

444 The plugin is called Refactoring Annotations, and can be downloaded from  
 445 `\url{http://multiview.cs.pdx.edu/refactoring/refactoring_annotatons}`.  
 446 In general, Refactoring Annotations can be thought of  
 447 as graphical error messages;  
 specifically, the current plugin displays violated preconditions for the  
 448 `\refacName{Extract Method}` refactoring.

449

450 `\begin{figure}`  
 451 `\centering`  
 452 `\includegraphics[scale=\figureScale]{annsOk}`  
 453 `\caption{Refactoring Annotations overlaid on program text.}`  
 454 The programmer has selected two lines (between the dotted lines) to  
 455 extract.  
 456 Refactoring Annotations show how the variables will be used:  
 457 `\texttt{front}` and `\texttt{rear}` will be parameters, as indicated by the  
 458 arrows into the code to be extracted, and  
 459 `\texttt{trued}` will be returned, as indicated by the arrow out of the code  
 460 to be extracted. `\label{fig:annsOk}`  
 461 `\end{figure}`

462

463

464 The programmer starts using the Refactoring Annotations tool by selecting some  
 program text.

465 Refactoring Annotations overlay the program text to express control- and data-flow  
 466 information about the programmer's selection.

467 Each variable is assigned a distinct color, and each occurrence of the variable  
 468 is highlighted, as shown in Figure~\ref{fig:annsOk}.

469 Across the top of the selection, an arrow points to the first use of a variable  
 470 whose value  
 471 that will have to be passed as an argument into the extracted method.  
 472 Across the bottom, an arrow points from the last assignment of a variable  
 473 whose value  
 474 will have to be returned.

475 L-values have black boxes around them, while r-values do not.

476 An arrow to the left of the selection indicates that control flows from

document.pdf

Page 4 of 15 Scale 137

Previous Next LaTeX BibTeX Drawer Go to page Magnification

IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. X, NO. T, MONTH YEAR

```

boolean areWheelsTrue() {
    Wheel front = bike.getFrontWheel();
    Wheel rear = bike.getRearWheel();
    boolean trued = isWheelTrue(front);
    trued = trued && isWheelTrue(rear);
    return trued;
}

```

return statement to the  
 arrow (Figure 4, middle  
 branch (break or con  
 from the branch statem  
 ure 4, bottom). In each  
 arrows, indicating the l  
 When code violates  
 tions are intended to g  
 to correct the violation  
 or reduce the selection  
 Other solutions include  
 break and continue  
 refactoring.

Refactoring Annotat  
 amount of code to be  
 of tens or hundreds of  
 passed in or returned,  
 values are colored. In t  
 assigns to many variab  
 complex. However, we  
 more values that are pa  
 the extracted method. T  
 complex Refactoring A  
 EXTRACT METHOD p  
 has commented, Refact  
 complexity metric.

Refactoring Annotat  
 programmer in resolving p  
 First, because Refactor  
 ple precondition violat  
 give the programmer an  
 Correcting a condition  
 correcting a condition  
 ple assignments. Likev  
 likely to be easier than  
 Refactoring Annotatio  
 help programmers to d  
 Refactoring Annotat  
 visualizations. Our co  
 similar to Control Stru  
 Control Structure Diag  
 pend on the program

Fig. 3. Refactoring Annotations overlaid on program text. The programmer has selected two lines (between the dotted lines) to extract. Refactoring Annotations show how the variables will be used: front and rear will be parameters, as indicated by the arrows into the code to be extracted, and trued will be returned, as indicated by the arrow out of the code to be extracted.

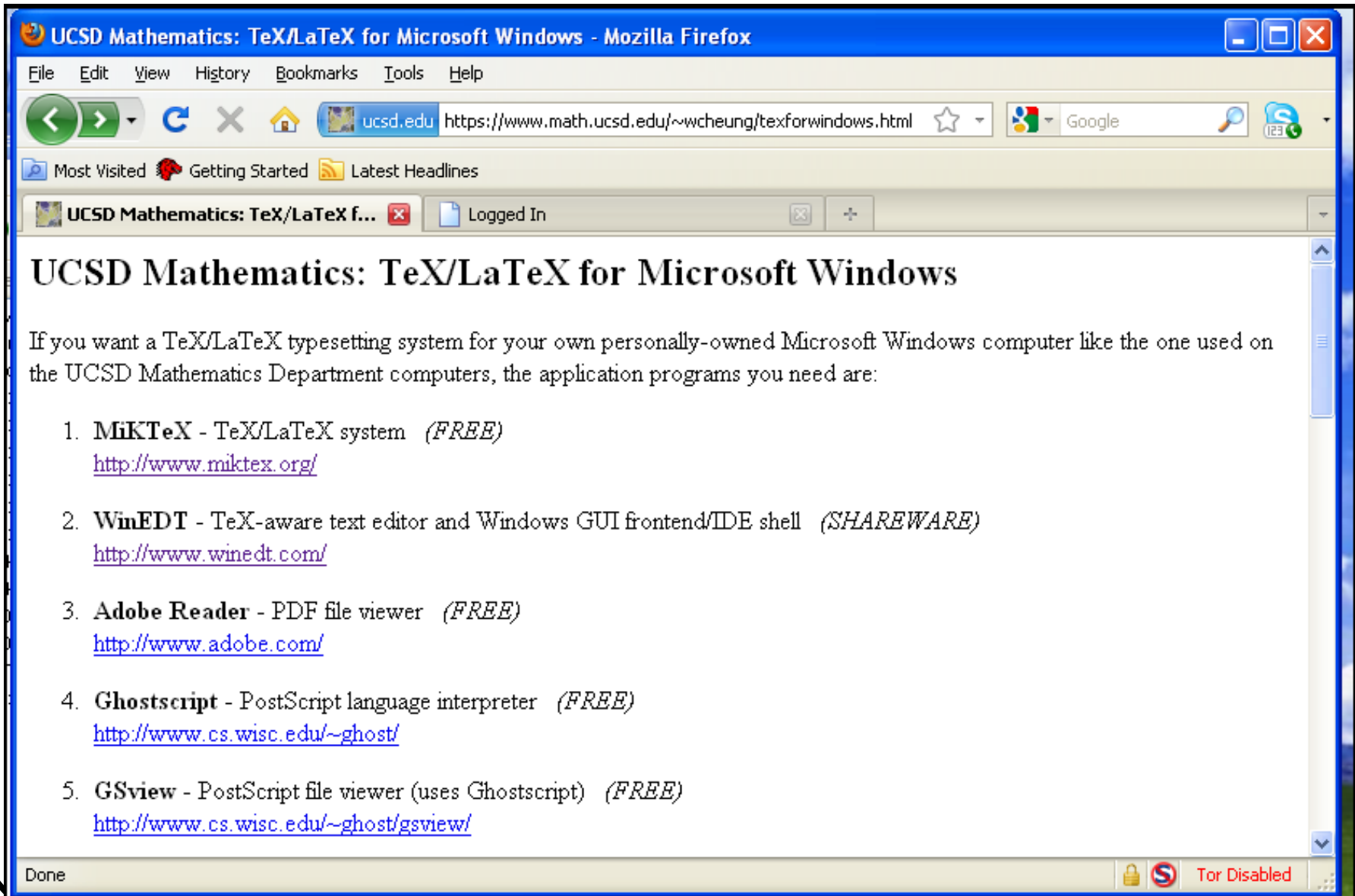
```

void goOnVacation() {
    Bike roadBike = getRoadBike();
    Bike mountainBike = getMtnBike();
    loadOnCar(roadBike, mountainBike);
}

boolean curbHop(int curbHeight) {
    int hopHeight = liftFrontWheel();
    if (hopHeight < curbHeight) {
        endo();
        return FAILURE;
    }
    liftRearWheel();
}

```

# Tex and LaTeX for windows



The screenshot shows a Mozilla Firefox browser window with the title "UCSD Mathematics: TeX/LaTeX for Microsoft Windows - Mozilla Firefox". The address bar shows the URL "https://www.math.ucsd.edu/~wcheung/texforwindows.html". The page content includes a heading "UCSD Mathematics: TeX/LaTeX for Microsoft Windows" and a paragraph stating: "If you want a TeX/LaTeX typesetting system for your own personally-owned Microsoft Windows computer like the one used on the UCSD Mathematics Department computers, the application programs you need are:". Below this, there is a numbered list of five items:

1. **MiKTeX** - TeX/LaTeX system (*FREE*)  
<http://www.miktex.org/>
2. **WinEDT** - TeX-aware text editor and Windows GUI frontend/IDE shell (*SHAREWARE*)  
<http://www.winedt.com/>
3. **Adobe Reader** - PDF file viewer (*FREE*)  
<http://www.adobe.com/>
4. **Ghostscript** - PostScript language interpreter (*FREE*)  
<http://www.cs.wisc.edu/~ghost/>
5. **GSview** - PostScript file viewer (uses Ghostscript) (*FREE*)  
<http://www.cs.wisc.edu/~ghost/gsview/>

The browser's status bar at the bottom shows "Done" and "Tor Disabled".