\textbf{Definitions in \LaTeX}

CSC 341, “Automata, Formal Languages, and Computational Complexity”
Department of Computer Science · Grinnell College
August 31, 2020

\LaTeX{} is an extensible markup language. The user can define new commands and environments, appropriate to particular documents, in terms of the predefined ones. The commands thus defined work like macros in C. A “call” to a defined command is in effect expanded into the replacement text that the definition associates with it. If there are arguments, they get plugged into the replacement text at points indicated by placeholders. \LaTeX{} then processes the result of these replacements as if it had formed part of the original source file.

Definitions of new commands can appear either in the preamble or inside the document environment, but it’s more usual to place them in the preamble. The control sequence that defines new commands is ‘\texttt{\newcommand}'. It takes the name of the new command as its first argument and the expansion text as its last argument. An optional argument between the two, enclosed in square brackets, specifies the number of arguments that the new command takes (the default is 0). In the expansion text, the placeholders indicating the points at which arguments will be plugged in are ‘\#1’ for the first argument, ‘\#2’ for the second, and so on.

Let’s look at a couple of examples. Here’s the definition of a command with no parameters:

\begin{verbatim}
\newcommand{\qed}{\hfill\rule{1.3 ex}{1.3 ex}}
\end{verbatim}

Once \LaTeX{} has encountered this command definition, the new, user-defined command, \texttt{\qed}, is available to the author whenever it is necessary to typeset a small black square at the right end of the current line. When \LaTeX{} subsequently encounters a use of the ‘\texttt{\qed}' control sequence, it replaces it with ‘\texttt{\hfill\rule{1.3ex}{1.3ex}}’ and sends that replacement text to the virtual typesetter. The typesetter executes the \texttt{\hfill} command (which inserts whitespace as padding) and then the \texttt{\rule} command (which prints a solid black rectangle of the width and height specified by its arguments; ‘1.3ex’ means “1.3 times the height of the lower-case letter ‘x’ in the current font”).

Here’s the definition of a command that takes one argument:

\begin{verbatim}
\newcommand{\prooftarget}[1]{To prove: \bf{#1}\medskip}
\end{verbatim}

\texttt{\prooftarget} might look something like this:

\begin{verbatim}
\prooftarget{There are infinitely many prime numbers.}
\end{verbatim}

\LaTeX{} expands this command into the text

\texttt{To prove: There are infinitely many prime numbers.}\medskip

It then proceeds with typesetting, just as if it had encountered this replacement text to begin with rather than the \texttt{\prooftarget} command. The result looks like this:

\texttt{To prove: There are infinitely many prime numbers.}

(The \texttt{\bf} command typesets its argument in boldface. The \texttt{\medskip} command terminates the current line and inserts some whitespace underneath that line.)
Users can also create new environments with the \texttt{\newenvironment} command. It relies on the same expansion mechanism as \texttt{\newcommand}, but in a “call” to the newly defined environment, the caller places the text to be treated specially between \texttt{\begin} and \texttt{\end} commands that take the environment name as argument.

The \texttt{\newenvironment} command itself takes three arguments: the name of the environment, a “begin text” comprising typesetting commands that are to be placed at the beginning of the expansion text (as the new environment is entered) and a similar “end text” to be placed at the end of the expansion text (at exit from the new environment). The begin and end texts respectively set up and tear down the special treatment that the environment imposes on the text it encloses. They often include \texttt{\begin} and \texttt{\end} commands for other environments.