Packages and Classes

1 Newcommands

Almost everything in LaTeX can be customized, including LaTeX commands themselves. To create a new command, place

```
\newcommand{\name}{definition}
```

in the preamble. Then, to call the command, use `\name` in the document. The compiler will complain if `\name` is a predefined LaTeX command. The `\newcommand` command has an optional parameter to include input. For instance,

```
\newcommand{\integral}[2]{$\int_{\mathbb{R}} #1 \, d#2$}
```

defines the command `\integral` which takes in 2 inputs, placed where the `#1` and `#2` appear in the definition. Calling `\integral{\sin x}{x}` now produces $\int_{\mathbb{R}} \sin x \, dx$.

For function names in math mode which are not predefined (such as $\ln x$, $\sin x$, $\arctan x$), use a command such as `\DeclareMathOperator{\dimension}{dim}` to define a command `\dimension`. This produces "dim", a math mode symbol.

To redefine a previously defined LaTeX command, use the syntax

```
\renewcommand{\old}{\new}
```

For instance, `\renewcommand{\phi}{\varphi}` changes the appearance of φ throughout the document. As another example, this command can be used to change the end-of-proof symbol in the **amsthm** proof environment into creating a black square pushed to the right of the line by placing this into the preamble:

```
\renewcommand{\qed}{\hfill \( \blacksquare \)}
```

Analogous to `\newcommand`, there is a `\newenvironment` command to create custom environments. The syntax to be placed in the preamble is

```
\newenvironment{name}{before}{after}
```

To call the command, use `\begin{name} .. \end{name}`. Then, commands in `before` are run before `..` and commands in `after` are run after `..`. Just like `\newcommand`, there is an option for up to 9 input variables.

Especially when reusing the same preamble for multiple documents, it may be convenient to store the preamble in a file with the `.sty` extension (the `.sty` stands for "style file"). In the first line of the style file, place the command

```
\ProvidesPackage{351Week5Package}
```

and then call the package just like any other package with the `\usepackage` command in the preamble of the main document.

2 Packages

Packages can also be used to extend the functionality of LaTeX in some way. The packages we have introduced in our course so far, listed below, are among the most frequently used LaTeX packages.

Package	Purpose
`amsmath`	Typesetting mathematics
`amssymb`	Math symbols and fonts
`amsthm`	Theorem and proof environments
`geometry`	Control page margins
`makeindex`	Create and index
`mathspec`	For including fonts (with XeLaTeX)

There are over 5000 more LaTeX packages! Other widely used packages include:

Package	Purpose
`hyperref`	Hyperlinks and clickable references
`enumitem`	Improved enumerate and itemize environment
`booktabs`	Improved tabular environment
`IEEEtrantools`	Improved multiline math (see page 64 of the text)
`fancyhdr`	Improved headers and footers
`babel`	Support for other languages
`listings`	For typesetting computer code
`graphicx`	To include outside graphics
`pgf`	To create graphics (TikZ)
`natbib`	An alternative to BibTeX
`microtype`	Micro-typographic extensions (only with pdfLaTeX)
`textpos`	Absolute positioning of text
`pgfornament`	Ornamental flourishes

Most of the packages listed here are shipped with many versions of LaTeX and probably can be accessed using `\usepackage{name}` in the preamble. If they are not already installed, these and many other packages can be downloaded from the "Comprehensive TeX Archive Network", online at `https://www.ctan.org`. This is also where you can find the documentation for the packages listed above. **The first step in using any of the above packages is to actually read the documentation!**

How to use a package that is not already installed:

1. Find the package at `https://www.ctan.org` or elsewhere.
2. **Read the readme or the documentation.**
3. See pages 89–90 of the text on how to install. Another good resource on how to install an extra package is at `https://en.wikibooks.org/wiki/LaTeX/Installing_Extra_Packages`. As a shortcut, if all that is needed is a `.sty` file, then you can try copying the `.sty` files into the folder which contains the `.tex` file.
4. Use by including `\usepackage{name}` in the preamble.

Some third party software packages automate this procedure, possibly doing it automatically as soon as a package is loaded with `\usepackage` in the preamble.

It is considered bad form to load many packages and then not use them. Loading obscure packages makes the `.tex` less portable and increases the chance that packages will conflict with one another. Packages also tend to become obsolete. As a general rule, use a minimum number of packages.

3 Classes

Class files are loaded by placing the `\documentclass{class}` command in the first line of the `.tex` file. Classes tend to have their own specialized commands; for example, the familiar `article` class provides commands such as `\section`, `\tableofcontents`, and `\author`.

Although so far we have only used the `article` class, there are many other class files. Some of the more popular packages are:

Class	Purpose
`amsart`	`article` alternative
`paper`	`article` alternative (used in this document)
`book`	Books
`memoir`	`book` alternative; a great choice for books/theses
`letter`	Formal letters
`scrlttr2`	`letter` alternative; one of many Koma-Script classes
`moderncv`	Curriculum vitae
`beamer`	Presentation slides
`tikzposter`	Conference posters
`exam`	Exams
`standalone`	cropped `.pdf` output (good for TikZ)

If not already present on your system, class files can be found on `https://www.ctan.org` and installed in a similar way that packages are installed. Sometimes it is possible to simply place the desired `.cls` file in the folder containing the `.tex` file.

Of course one should **read the documentation** and look at example files to learn how to use any particular class!