LaTeX Seminar: Defining Macros

Often with LaTeX, one finds that certain symbols are used over and over again. For example, any number theorist is bound to use the symbol **Q** (or $\mathbb{Q}$ if you prefer) for the rational numbers countless times in the course of their TeX career. It would be a major pain if every time one wanted to print this symbol, the commands `\mathbf{Q}` or `\mathbb{Q}` had to be issued. The real power of TeX lies in its ready customizability.

1. Defining Commands Without Argument

First, and importantly, all macros and user defined commands *must* appear before the `\begin{documant}` command but after the `\documentclass` and `\usepackage` commands. Two of the simplest and most useful commands for writing macros are `\newcommand` and `\renewcommand`. In both cases, the format is

```
\newcommand{\name}{command}.
```

Here, `name` is the name given to the command, and `command` is the long, explicit version of what you want `\name` to do every time it is issued. For example, we have written the macro

```
\newcommand{\Q}{\mathbf{Q}},
```

which prints **Q** every time `\Q` is written in math mode (observe that `\mathbf` is only valid in math mode). Sometimes, you will define a command that LaTeX has already defined. In general, you should find out what the predefined command does, and if it is something essential or really useful, you probably should think up a different name. If it is dispensible, though, use the command `\renewcommand` which redefines the symbol to your specifications. For example, after writing the macro

```
\renewcommand{\a}{\alpha},
```

we can easily print $\alpha, \alpha, \alpha, \alpha$ with two short character strokes.

2. Commands With Argument

Macros can also accept multiple arguments. The general command to write such macros is

```
\newcommand{\name}[narg]{command},
```

where `narg` is a positive integer indicating the number of arguments that `\name` is to take. When `\name` is called, the syntax is `\name{arg1}{arg2}...{argn}`. The variable `#j` is used for the j^{th} argument. Thus, for example, defining the command

```
\newcommand{\Hom}[3]{\mathrm{Hom}_{#1}\left(#2,#3\right)}
```

defines a new command `\Hom` with three arguments. Thus, when we issue the command `\Hom{k}{V}{W}` we get

$$\mathrm{Hom}_k\left(V, W\right).$$

3. Declaring Math Operator

Many times, it is convenient to be able to produce roman script operators such as sin, cos, log, lim in math mode. If these are just called by typing `$sin,cos,log,lim$` the result will be

$$sin, cos, log, lim$$

instead of

$$\sin, \cos, \log, \lim$$

as desired. It so happens that the commands `\sin,\cos,\log,\lim` are already defined in LaTeX and produce these symbols. If we want to define similar operators that will print in roman script, we can issue the command (again, in the preamble, where all the new commands are defined)

```
\DeclareMathOperator{\name}{operator},
```

where `\name` is the command that will be issued to produce `operator` in roman script. Thus, we have defined `\DeclareMathOperator{\GL}{GL}` so that we can print $\mathrm{GL}(V)$ without having to type `\mathrm{GL}(V)` every time.

One can also define new environments in LaTeX, but this is beyond the scope of the seminar, and an appropriate reference should be consulted. Play around with the file `comm.tex` and write some useful macros.