lot of powerful commands are left out here, so if you would like to be considered proficient with Linux, you will need to get a book and do some additional reading.

Here are some important commands in Ubuntu:

> Redirects the standard output into a text file with the specified name. In other words, if the output would usually appear on the screen, now it will be written to a text file. Examples:

```
grep include simulation_engine.h > include_statements.txt
ls -a > files.txt
```

cat Concatenate any number of text files and output the results to the screen. Use the **>** operator to redirect the output to a file (see the entry for **>** above).

```
cat text_file1.txt text_file2.txt
```

cd Change directory. Use this command to navigate through the computer's file structure. Note that **cd** with no arguments will send you to the home folder, which is represented by a **~**. If you use **cd /**, you will end up in the root directory, and **cd ..** moves you up one level in the directory hierarchy.

```
cd ~/sf_ushare/Gmatrix
```

chmod Modify the permissions for a file.

```
chmod u+x myscript.h
```

cp Copy a file to a new location.

```
cp ~/sf_ushare/test.sh ~/test_moved.sh
```

grep A powerful utility that prints lines from a text file matching a specified pattern. It is beyond the scope of this book, but combining **grep** with "regular expressions" can be a very powerful way to search and parse text files. If you are interested, use a search engine to find tutorials on "regular expressions".

```
grep Mean_of_means simrun_summary.csv
grep Mean_of_means *.csv
```

head Display the first ten lines of a text file. Use the argument **-n** to change the number of lines displayed.

```
head -n 50 text_file.txt
```

less View the contents of a text file.

```
less text_file.txt
```

`ls` List the files in a directory. Use the argument **`-a`** to show all the files in a directory, including hidden ones. Use the argument **`-l`** to show additional information about the files, including the permissions, owner, and size.

```
ls -a
```

`man` Show the help manual for the relevant program or command.

```
man grep
```

`mkdir` Create a new directory.

```
mkdir my_new_directory
```

`mv` Move a file to a new location. This command deletes the copy of the file in the old location. If you use move within the same directory and specify a new name, this command has the effect of renaming the file.

```
mv ~/test.sh ~/sf_ushare
mv ~/test.sh ~/sf_ushare/test_moved.sh
```

`nano` Open a text file with the **`nano`** command-line text editor.

```
nano test.sh
```

`pwd` Display the path of the current working directory.

```
pwd
```

`rm` Delete a file or directory. Be very careful with this one.

```
rm test.sh
```

`sudo` If you precede a command with **`sudo`**, you will execute the command as a super user, capable of making any change to the computer. Use this command with caution, as a super user can literally destroy your operating system (by, for example, deleting every file in the root directory accidentally).

```
sudo apt-get update
sudo apt-get upgrade
sudo apt-get install jedit
```

`tail` Display the final 10 lines from a text file. Use the argument **`-n`** to change the number of lines displayed.

```
tail -n 50 test.sh
```