Linux Basics

cd	change to home directory
cd ..	change to parent directory
cd directory	change to the directory called "directory"
cd ../directory	change to the directory called "directory" which is in the parent
ls	list everything in the current directory
pwd	print working directory (full path of current directory)
mkdir directory	make a new directory called "directory"
rmdir directory	remove an existing directory called "directory" (cannot be undone)
vim file	open a file called "file" or create it if it doesn't exist
vim file1 file2 etc.	open up as many files as you want simultaneously
rm file	remove a file called "file" (cannot be undone)
cp file1 file2	copy "file1" into a new file called "file2"
cp file **../**file2	copy "file" into the parent directory and name it "file2"
cp file directory/file2	copy "file" into directory "directory" and name it "file2"
mv file directory	move "file" into a directory called "directory"
mv file **../**	move "file" into parent directory

Note: if you create a new file but leave it blank, you must save and quit for it to remain. Else it will get deleted. Saving and quitting in vim is discussed on the next page.

Command Line Shortcuts

- *The "tab" key:* the tab key will auto-complete something you are writing in the command line. For example, say there was a file called "my_file.c" and no other file in the directory started with the letter 'm'. To open up the file, instead of having to manually type out "vim my_file.c", simply write "vim m" and then press the tab key. It will auto-complete the filename in the command line and jump from "vim m" immediately to "vim my_file.c". Therefore, you won't need to type out the whole file name.
    - When the file being opened could be multiple files, entering tab will auto-complete as far as it can go. For example, if in addition to "my_file.c" there was a file called "my_file.h", then pressing tab after entering "vim m" will auto-complete all the way to "vim my_file.". It stops there because this could either be "my_file.h" or "my_file.c". Simply then enter the 'c' or 'h' for the desired file.
- *The "up" and "down" arrows:* a history of all of the entries to the command line are saved, and the up and down arrow keys can be used to scroll through them. This way, if a recently entered command needs to be put into the command line again, it can be accessed via the arrow keys and does not have to be manually typed out. For example, say "vim my_file.c", "vim my_file.h", "vim text.txt", and "./my_program" were all entered into the command line in that order. To get "./my_program" into the command line again, enter the up arrow once. To get "vim text.txt" into the command line again, hit the up arrow twice. Etc. etc. This will save you a lot of time from having to constantly type in the same things over and over again in the command line.

Vim Basics

- Enter the escape key to enter **normal/command mode**. This is the mode where you enter commands and is the default mode you are in when you open a file.
- Enter "i" to enter **insert mode**. This is the text editing mode.

<u>The below commands are all while in normal mode</u>

Saving and Quitting

:q	quit if you haven't made any changes
:q!	quit without saving
:w	save
:wq	save and quit

Moving the Cursor

gg	go to the beginning of the file
G	go to the end of the file
nG	go the the "nth" line of the file
ctrl-d	go halfway down the screen
ctrl-u	go halfway up the screen
shift-4 aka **$**	go to the end of the line - Then, to enter insert mode and start inserting text immediately after the end of the line, enter "a" instead of "i".
0 (zero, not O)	go the the beginning of the line - Then, to enter insert mode and start inserting text immediately before the beginning of the line, enter "i".

Text Editing Commands

Single Line Shortcuts

yy	copy current line
dd	delete or cut current line

Highlighting Text

shift-v	**visual line mode:** highlight a line. Now use the up and down arrow keys or jump commands (described above) to highlight multiple lines of text.
ctrl-v	**visual block mode:** highlight blocks of text irregardless of rows and columns (i.e. blocks on the screen). Most useful for highlighting pieces of text on a line but not the whole line. Use the left and right arrow keys to highlight character by character on the line.

General Copying/Cutting/Pasting

y	copy currently highlighted text
d	delete/cut currently highlighted text
p	paste the copied or cut text on the line directly below the one the cursor is currently on

Navigating Multiple Open Files

:first	go to the first file
:last	go to the last file
:next	go to the next file
:previous	go to the previous file

- put "w" before any of these to save i.e. **:wnext** means save current file, go to next file

:nnext	go "n" files ahead
:nprevious	go "n" files backwards

- put "w" after the n to save i.e. **:2wnext** means save current file, go 2 files ahead

:args	display the list of files currently open for editing

Viewing 2 files simultaneously

:split file	split the screen with the file called "file"
Ctrl-w	navigate between the split screen. If you're on the top file, enter the down arrow after entering Ctrl-w to go to the file on the bottom. If you're on the bottom file, enter the up arrow after entering Ctrl-w to go to the file on the top.

- use the quit commands on page 2 to end the split screen by closing the file the cursor is currently on i.e. if the cursor is on the top file and you enter **:wq** that will save the top file and quit out of the split screen leaving the bottom file on the screen only.

Other Commands

u	undo
/text	search for all occurrences of "text" in the file. Use "n" to go to the next occurrence.
:%s/text1/text2/g	search and replace. Find all occurrences of "text1" and replace it with "text2".