Linux Command Line Basics

cd	go to home directory
cd ..	go to parent directory
cd ../..	go two directories up
ls	list all non-hidden files
ls -a	list hidden files
pwd	print working directory (full path of current directory)
mkdir dir	create a new directory name *dir*
rmdir dir	delete an existing directory named *dir*
vim file	open a file named *file* or create it if it doesn't exist
vim file1 file2 etc.	open up as many files as you want simultaneously
rm file	delete a file called *file*
cp file1 file2	copy *file1* into *file2* - *file2* doesn't have to exist yet
mv file dir	move *file* into the directory *dir*
mv file1 file2	rename *file1* to *file2*
mv dir1 dir2	rename directory *dir1* to *dir2*

For these commands you can use any combination of ../, directory names, and file names to get the exact directory and file paths desired. For example, **cp** file1 ../../dir1/dir2/file2 would copy *file1* and put it in a file named *file2* which is in a directory named *dir2* which is in a directory named *dir1* which is two directory levels above the current directory. Extrapolate that logic to the other commands.

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 will autocomplete something you are writing in the command line which will save you lots of typing. For example, say there was a file named *file.c* and no other file in the directory started with the letter f. To open up the file, instead of having to manually type out *vim file.c* simply write *vim f* and then press the tab key. It will auto-complete the filename in the command line and jump from *vim f* to *vim file.c*. For the case where multiple files start with the same letters it will autocomplete as far as it can go. For example, if *file.h* also exists then entering tab after entering *vim f* will expand to *vim file.* but stop there because it's not known if an h or c should come after the period.

A history of what you have entered is saved in the command line. Use the up and down arrow keys to navigate through these. This will save you from having to type the same things out over and over again.

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".
:set paste If you are copying/pasting code from your own computer (such as in visual studio for example) into vim, the code will not paste correctly in regards to indentation if you do not enter this command before pasting