Lynne Coblammers
EECS 678
Lab 1
2015.01.28

Using Vim Editor:

1. Open a file with some C source code. Copy the first 12 lines of text from this file. Create three new files named a.c, b.c, and c.c and paste this text at the top of each new file. Save each new file.

To open the file, type vim <filename> in the terminal window. Next, to copy the first 12 lines of text, navigate to the very top of the screen in command mode by using 'k' to move up and 'h' to move to the left. Type '12yy' which yanks, or copies, 12 lines of code. To open a new file in vim, type :e a.c, which will create the file (assuming it doesn't exist). Next, paste the 12 lines above the cursor by typing P. Save this file by typing :w. Repeat this process by opening b.c and c.c, pasting the lines, and saving. Finally, type :q to exit vim.

2. Open two different source files for editing, ensuring both are visible on screen simultaneously, and switch between editing each of these two files and issuing commands to a terminal you have open.

To open two source files for editing, type gvim <filename1> <filename2>. The g preceding vim allows you to open vim in a separate window from the terminal. Only one will initially be visible. To make both visible, first, the screen must be split. One way to do this is to split it vertically by typing <c-w>v. Now, the screen should be split vertically with each half showing the same source code file. To switch to the right half of the screen, type <c-w>w. Now, both files should be open in the buffer, so to access the second file, open the buffer explorer by typing \be, navigate to the second file by using k to move up or j to move down, end hit enter to open it. You can test that commands work by copying a line from the buffer you're currently in by typing 'yy', typing <c-w>w to switch to the other buffer, and typing 'P' to paste the line you just copied. Furthermore, you can still access the terminal by opening its window, which is separate from the vim window, and issuing commands as normal.

3. As you are reading the code for a large C program (with multiple source files spanned across multiple directories), you come across a call to an unknown function. Find the definition of this function. Go back to the calling context where you started.

Before opening the program in vim, you must have tags enabled. To do this, type the command ctags -R in each of the directories that code is contained in. Now, with the code open in vim, when you see the unknown function, move your cursor to the function name using j, k, h, and l. Once your cursor is on the function name, type <c-]>. This will jump to the declaration of the function (or give you options to select if that function name is used in multiple files). Once you have examined the function, you can easily return to where it was called by typing <c-t>.

4. Given a file with a million lines of text, remove the whitespace (spaces, tabs, and newlines) from the beginning of every line. That is, when you have finished, each line should start with a non-whitespace

character.

With the file open in vim, do a search and replace. To do this, type :%s /^\s\+//. The %s command substitutes throughout the current buffer. ^\s\+ is the pattern being substituted, which means you are looking at the beginning of lines (^) for any number (\+) of spaces (\s). This is being substituted with nothing, thus there is no replacement portion between the final two slashes (//). After executing this command, each line with begin with non-whitespace.

5. Find and replace every occurrence of the string 'Bill Self' in your source file with the string 'basketball genius Bill Self' (assume that case matters). After you're done, reformat your file so that each line adheres to an 80 character text width. If you are using vim, you may assume that your vimrc has the appropriate textwidth and formatoptions settings so that lines are formatted correctly when the formatting command is used.

With the source file open in vim, use the substitute command to replace the string. Do this by typing \ :%s /Bill Self/basketball genius Bill Self/g. The %s means it will search the entire buffer, and the /g at the end means it will do it for more than the first occurrence on the line. Next, to reformat the file, first navigate to the end of the file using <c-f> and j to find out the number of lines. Then, if, for example, there were 93 lines, type 93gqk, which will wrap the 93 lines above the cursor to the preset text width.