Adapted from:
http://courses.knox.edu/cs205/205Tutorials/viPrimer.html

---

vi Primer

This document is designed to introduce you to the standard UNIX™ screen editor, ***vi*** (short for "visual"). ***Vi*** can be used to enter or change any kind of text, letter, essay, report, or program. Together with the *nroff* text formatting program it forms a complete word processing system. This is only an introduction to ***vi***, so not all the details and commands of ***vi*** are included. However, it is hoped that all the commands necessary for you to do 98% of the editing for programs, documentation, and papers is included. If you want more information on vi you should consult the papers "An Introduction to Display Editing with vi" and "Ex Reference Manual", both in Volume 2A of the Unix Programmer's Manual.

To enter the visual screen editor, just type

*% **vi filename***

and hit RETURN when you see the standard Unix prompt "%". "Filename" is the name of the file you want to edit. If you want to start a new file, just leave the file name off.

If you're editing an existing file, the first 23 lines or so of the file will be displayed on the screen, the cursor will be put at the upper left hand corner of the screen (the HOME position), and you're ready to go. If you're starting a new file you'll get a "file does not exist" message and a bunch of tilde's ("~") at the start of the display - meaning that there's nothing in the file yet.

The ***vi*** editor has two "modes", a **command mode** where you can move around on the screen and issue commands to do things to your file, and an **insert mode** where you add text to your file. When you first enter ***vi*** you are in command mode and anything you type from the main keyboard is considered a command by the editor. You will not be able to use the numeric keypad for anything in ***vi*** so just forget about it.

Vi has a lot of commands; practically all of the keys on the main keyboard will do something. For example, to move around in your file you can either use the arrow keys or you can use ***h***, ***j***, ***k***, and ***l*** on the keyboard. *h* will move you one space left (arrow left), *j* will move you one line down (arrow down), *k* will move you one line up (arrow up), and *l* will move you one space right (arrow right). (The h, j, k, and l keys are the preferred way to move short distances.)

The *w* key will move you one "word" right, the "*$*" key will move you to the end of the current line, and the "*0*" (zero) key will move you to the start of the current line. We will go over many of the commands you can enter in the following pages.

To enter ***insert mode***, just type an "*i*". After this, all the keys on the typewriter keyboard are used for entering text into your file. They cannot be used for entering commands while you are in insert mode. To get out of insert mode and back to command mode you must hit the **<ESC>** key (usually found on the upper left part of the keyboard). In fact, if you are ever unsure about what you're doing

and want to make sure you are in some "sane" state, just hit the **<ESC>** key several times. Eventually the terminal will beep at you; when it does that you are back in command mode.

Finally, to get out of ***vi*** and save your work just type ***ZZ*** when you are in command mode.

The next several sections will describe many of the vi commands that you can execute in Command mode. They are broken up into several sections loosely organized as Moving Around, Inserting Text, Deleting and Changing Text, Searching for Patterns, and File Commands.

Be aware that ***vi*** (and Unix for that matter) is case-sensitive. For example, the "**j**" command will move you down one line, while the "**J**" command will join two lines together. So be careful.

MOVING AROUND

Open the file title “MyFile” found in your home directory.

We've already seen some of the moving around commands. Let's go over them again and add commands to move us larger distances:

h move one space to the left on the current line.

j move one line down, staying in the same column if we can.

k move one line up, staying in the same column if we can.

l move one space right on the current line.

w move one "word" forward (right) in the text. This command wraps around lines, and defines a word as any stretch of alphanumeric characters not including spaces or punctuation marks.

b move one "word" back (left) in the text.

e move to the end of the current word. Very handy for making

plurals.

G go to the end of the file. (Puts cursor at beginning of last

line.)

1G go to the first line in the file. (Similarly, nG goes to the

nth line in the file.)

0 go to the beginning of the current line.

$ go to the end of the current line.

^d (ctrl-d) move down 1/2 a page on the screen.

^f (ctrl-f) move forward one entire screen.

^u (ctrl-u) move up 1/2 a page on the screen.

^b (ctrl-b) move backward one entire screen.

H move to the upper left of the screen (Home).

M move to the Middle of the screen.

L move to the last line on the screen.

INSERTING TEXT

There are several ways to get into insert mode, not all of them obvious. So beware. Remember that the way to get out of insert mode is to hit the

<ESC> key.

i insert text to the left of the cursor.

a append text to the right of the cursor.

o open a line below the current one and insert text there.

I insert text at the very beginning of the current line.

A append text at the very end of the current line.

O Open a line above the current one and insert text there.

r replace the single character currently under the cursor with the next character you type. Notice that you enter insert mode for only one character when you use this command and you don't need to type **<ESC>** after you've replaced the character. (Good for changing lower case letters to upper case. To change an x to X, just type rX, or alternately, just a **~**, which is the *change case* command.)

R enter overstrike mode. This types over all the characters on the current line and Replaces then with what you've typed. This is close to how a typewriter would work. When you get to the next line, vi will open up a new line for you and you'll be in insert mode.

DELETING AND CHANGING TEXT

x delete the single character currently under the cursor.

d the general delete command. This command can be "modified" by specifying how much text you want to delete after you type the d. For example, if you type **d<SPACE>** it will delete one character (this is the same as x above).

dw deletes the rest of a word to the right of the cursor

db deletes the word to the left, and

dd deletes the entire line.

If you preface the d with a number n you will repeat the delete n times, as in

5dd will delete the current line and the next four lines for a total of 5 lines.

5dw will delete the next five words.

D delete from the cursor to the end of the current line. A synonym for **d$**.

u the *undo* command. This very handy command will undo the last delete, change or insertion of text. Repeating this command will continue to undo previous commands

SEARCHING FOR PATTERNS

Another way of moving around in your file is to search for a particular pattern of characters (for example looking for all occurrences of the name "fred"). **Vi** will do searches for you starting at the current cursor position and wrapping all the way around the file. If the pattern you are looking for is not found, then the message "Pattern not found" is displayed on the bottom of the screen.

/string To perform a search forward in the file you type a **/** (slash).

The slash is echoed on the bottom line of the screen and you may then type in the string you're looking for. When you're finished press **<RETURN>** and the search will begin from the current cursor position towards the end of the file, wrapping around to the top if it needs to. The cursor will stop at the first character of the found string. Note that this command is case-sensitive (a search for “fred” will not pick up the capitalized name “Fred”).

?string To perform a search backwards in the file you type a ? (question mark). The question mark is echoed on the bottom line of the screen and from then on the procedure is the same as above for forward searching.

n To find the next occurrence of the same string, just type an n.

N To reverse the direction of the search (forward to backwards or vice versa) and find the next occurrence just type an upper-case N.

Once you are done searching, you might notice that those words meeting the criteria of your last search command are still highlighted. To remove this effect, simply search for string of letters not found in your document (i.e. filnidhf)

FILE COMMANDS

To operate on the entire file generally requires that you drop into *ex command mode* (You see, vi is really a part of the ex editor. But who really cares anyway?) You do this by typing a **colon :**. This will be echoed on the bottom line of the screen and you can type in the file command down there.

:wq write out the file and quit the editor. This is the same as the **ZZ** command discussed earlier. If your file doesn't have a name yet, you must do this command in two steps:

:w filename and then :q

:r goes out and finds the file with name "filename" and inserts it into the middle of your text after the current line. Handy for putting programs into the middle of your documentation.

:q! quit the editor and don't save any of the changes made. Be sure you really want to do this, since you don't get a second chance.

:x a synonym for :wq above.

:!command lets you issue one Unix shell command from the editor.

UNIX will execute the shell command and then display the message "Press RETURN to continue" and you'll be back in the editor at the spot where you left off. Great for reading your mail without leaving the editor.

:e edit a new file. Beware! as this will destroy whatever was in the editing buffer before you executed the new edit. It won't destroy the original version of the file because vi always works with a copy, but any changes you made that you hadn't saved will be gone.

MISCELLANEOUS COMMANDS

. (period) This handy command will repeat the last text changing command you executed at the current cursor position. Very handy for inserting the same word several times in different places in the text. Just insert the word once, move to a new place and type a period. Viola! it will be inserted again.

p The put text command, in conjunction with the delete command allows you to do a Cut and Paste operation. Say you want to move 8 lines of text down to the bottom of your file. Just type 8dd to delete them (Cut) from their original position, type a G to get to the end of the file, and type a p to put them back after the last line of the file. (To put them before the last line type an upper-case P.)

Y The yank command lets you do a Copy and Paste, by just yanking a number of lines into a buffer instead of deleting them. You can then use the put command to paste them wherever you want them.

J The join command will take the current line and the next line and make them into one long line.

^L The repaint command will put your screen back to normal if it has been messed up somehow.
