10. VI – A UNIX Text Editor

(see also references)

Lab 10.1 - About vi

A text editor is a program (like Microsoft Word without the menu) that you can use to create and modify files. All a text editor does is allow you to edit text, and not format the layout of the text like a word processor does. Full-screen Unix text editors like vi, Emacs, and Pico enable you to create and edit text files, such as program code.

Lab 10.2 - Invoking vi

You can invoke vi in different ways to suit your purposes.

Creating a new file: Suppose you want to create a new text file, called file1, and then enter text into it. At your shell prompt, type:

```
vi file1
```

Vi detects that the file1 file does not exist. It creates a new empty file and opens a file buffer in which you can enter text. To indicate that the file is empty, vi displays tilde characters (~) down the left side of the file buffer like the diagram below.

Important Note: You must enter a filename after the vi command. Although it will still open a vi buffer such as the diagram on the next page, you will not be able to save your work.

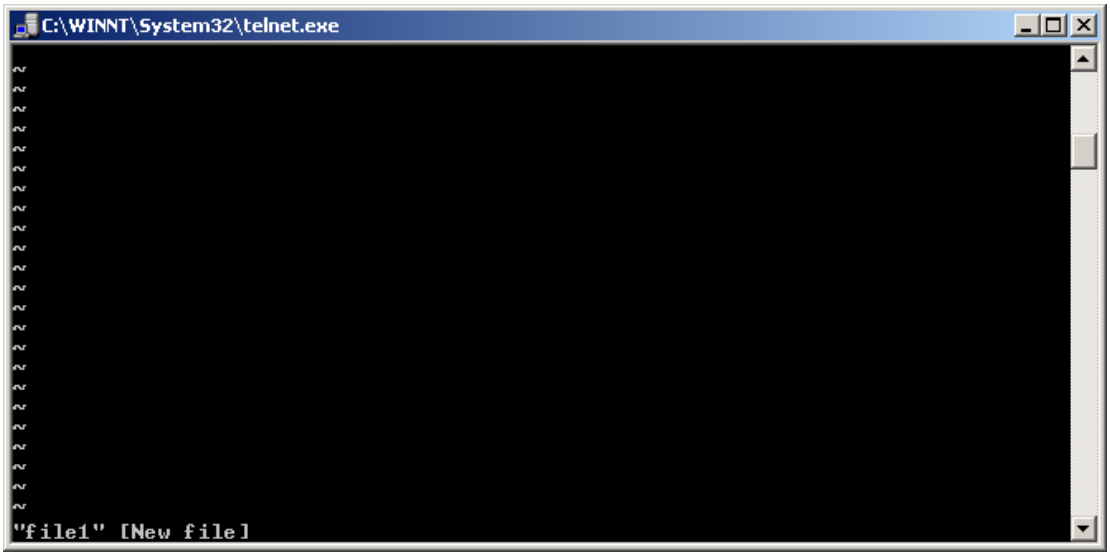

Figure 10.1: vi empty file buffer

Opening an existing file. Suppose you have already created and saved the file file1. Now you want to reopen and edit it. At your shell prompt, type:

vi file1 *it's the same command to open a new file as it is to open an existing file.*

Vi finds the file named file1and reads the contents of the file into a file buffer for you to edit.

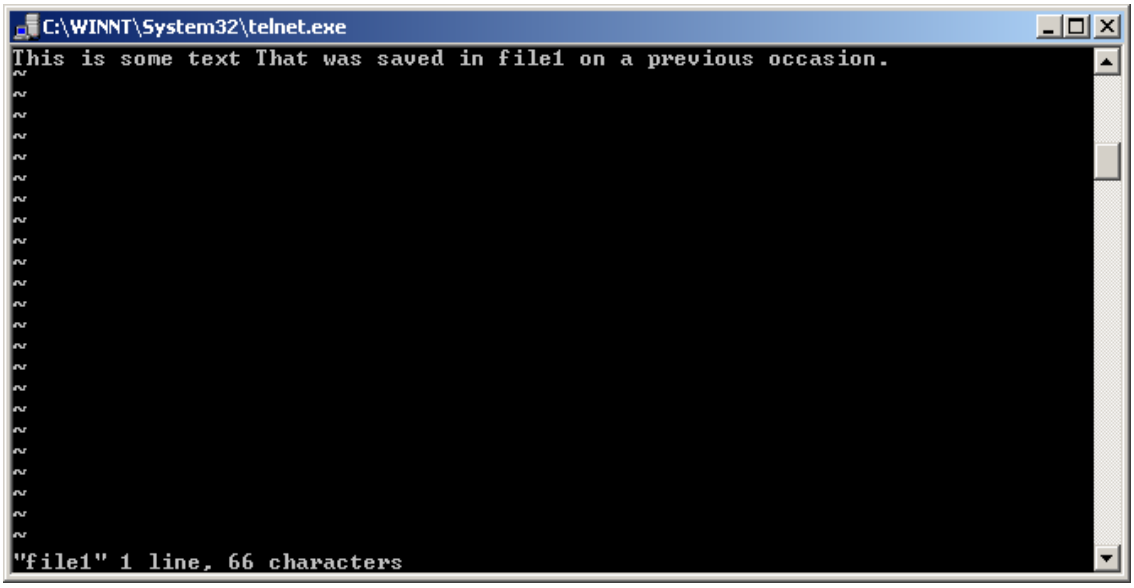

Figure 10.2: opening a previously made file in vi

Lab 10.3 - Using vi modes

Vi has three "modes": edit, insert, and colon.

Lab 10.3.1 - Edit / Command mode

Vi enters edit mode by default when it starts up. Edit mode allows you to move the cursor and edit the text buffer with commands we will discuss later. Whenever you press Esc in vi, it will return you to command mode, no matter what mode you are in.

Lab 10.3.2 - Insert mode

Insert mode is the mode that lets you enter text into the text file. It "drops" the cursor at a specific point in the buffer, allowing you to insert text. To enter insert mode, you position the cursor where you want to place text and press i. Press ENTER only after each paragraph. On Unix computers, vi has automatic text wrap. If you make a mistake, press BACKSPACE or DELETE to move the cursor backwards over the error. Then retype the line from that point. If the error is in a different line than the cursor, press ESC to return to edit mode and then reposition the cursor at the error, and press i to get back to insert mode.

Press i to enter insert mode and type the following line of text in your file1:

```
This is a line of text.
```

Now press Esc. This returns you to command mode and you are now in a mode to type commands rather than text.

Here are some of the ways to enter insert mode:

i Insert before cursor.

I Insert at start of current line.

a Append after cursor.

A Append at end of current line.

o Open new line below.

O Open new line above.

You should still be in command mode. If you are not sure then press Esc. Now position your cursor somewhere on the line you created and while still in command mode, try typing **a** which should move after the cursor and allow you to type text as if you had entered i. The list of commands above are the same as typing I to enter insert mode except they position where you insert the text.

Note: Capitals make a difference when entering commands in vi.

Lab 10.3.3 - Colon mode

Colon mode moves the cursor to the command line, allowing you to invoke program commands such as write / save to file (**:w**), quit (**:q**) or save and quit (**:wq**). You enter colon mode from command mode by typing a colon followed by a command such as those below. If you are in insert mode, it will simply type out a colon on the screen so make sure you are in command mode. Some useful commands are:

:w	Write buffer to the current filename.
:q!	Quit vi without saving buffer.
:wq	Write buffer to current filename and quit vi.
:w newname	Write buffer to file newname.
:r	Read the current filename into the buffer.
:r oldname	Read the file oldname into the buffer.
:e filename	Close current buffer and edit (open) filename.
:e #	Close current buffer and edit (open) previous file.

To save what you have done to date, enter colon mode and type w after the colon.

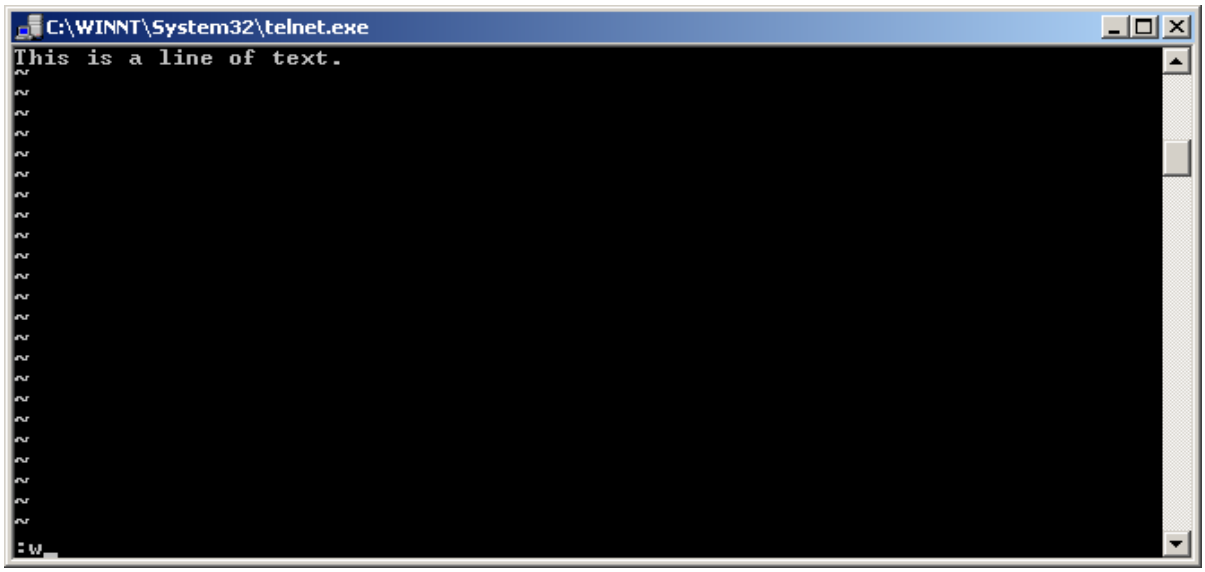

Figure 10.3: saving in colon mode

Quick quit (ZZ)

A fast way out without having to use colon mode is, while in command mode, press SHIFT/z twice to save your file and exit vi.

Cursor movement

In addition to the arrow keys, you can use these command mode commands to move the cursor:

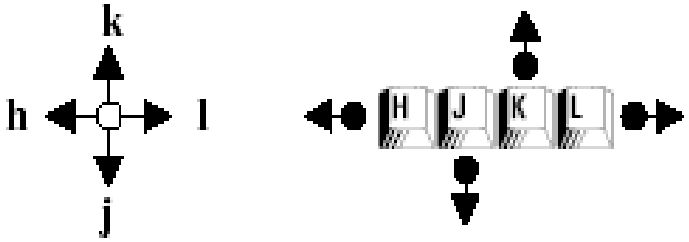

Figure 10.4: cursor movements using commands

h	Left one character.
j	Down one line.
k	Up one line.
l	Right one character.
b	Beginning of previous word.
e	End of next word.
w	Beginning of next word.
{	Backward one paragraph.
}	Forward one paragraph.
^	Go to first character of current line.
$	Go to last character of current line.
CTRL/d	Forward (down) one-half page.
CTRL/u	Backward (up) one-half page.
CTRL/f	Forward one page.
CTRL/b	Backward one page.
G	Go to the end of the file.
1G	Go to the first line of the file.
nG	Go to line number n.

You do not have to know all these commands as long as you can make you way around the text. They are shortcuts to make it quicker for you.

Lab 10.4 - More edit mode commands

Here are some additional edit mode commands and techniques that you can use to work efficiently. While in insert mode they act normally as simple letters, in command mode certain letters or combinations of letters have a function.

Lab 10.4.1 - Removing characters: x, X

To remove the character under the cursor, press x. To remove the character to the left of the cursor, press X. Position your cursor over the l in your file and press x. This will remove the l from the word line. It deletes whatever the cursor is positioned over when you type it.

Lab 10.4.2 - Restoring characters: u, U

To restore the last delete, press u. To undo all changes made to the current text line, press U. Type the command u and you should see the l returning.

Lab 10.4.3 - Replacing text: r, R

To replace the single character under the cursor, press r, then type the replacement character. Position the cursor over your **a** in your line of text and press r. Now type 1 and you will see that the "a" is replaced by "1".

To replace a portion of your current text line, position the cursor over the first character you want to change and press R. Vi is now in "strike-over" mode. It allows you to type over the existing text to the end of the current text line. Press ESC at any time to return to edit mode.

Lab 10.4.4 - Changing text: c

The change command lets you use the keyboard to mark a text block and replace it with new text.

Position the cursor on the first character of the text you want to replace. Type c, followed by a vi text string definition code. For example, to define to the end of the current word, press w. Vi replaces the final character in the defined string with a dollar sign ($). You can now type the new text string. If your new text string has more characters than the original, vi

enters insert mode when it reaches the dollar sign. If your new text string has fewer characters than the original, vi deletes any remaining characters when you return to edit mode. To return to edit mode, press ESC.

Lab 10.4.5 - Deleting text: d

The delete command lets you use the keyboard to define a text block and delete it from the file buffer. Position the cursor on the first character of the text you want to remove. Then press d, followed by a text string definition code. For example, to define to the end of the current word, press w.

Vi moves the text into the undo buffer. You can restore the deleted text at this point with the undo command (u). Or you can move the cursor and paste the contents of the undo buffer at a new location with the paste command (p) - (see next section).

Lab 10.5 - Cut, copy, and paste functions

Vi lets you delete (cut) or yank (copy) a section of text into the undo buffer. You can then reposition the cursor and paste the text from the undo buffer into the new location in your file buffer. The text you yank or delete remains in the undo buffer until you yank or delete again, overwriting the current contents.

Cutting text (d). See "Deleting text: (d)."

Yanking text (y). When you yank text, vi copies it into the undo buffer, leaving the original file buffer text unchanged. Position the cursor on the first character of the text block you want to copy. Then press y, followed by a vi text string definition code. For example, to define to the end of the current word, press w. Vi copies the text from your file buffer into the undo buffer.

Pasting text (p). You can use the paste command to insert the contents of the undo buffer into your text at the cursor location. To use the paste command, first delete or yank the text you wish to paste (see above). Next, move the cursor to the place where you want to paste the text. Type p. Vi inserts the contents of the undo buffer into your file buffer at the cursor location.

Lab 10.5.1 - Defining a string

The delete, yank, and change commands all follow the same general command syntax:

[n] command object

where n is an optional repeat count, command is the one-character command, and object is a code that defines the text to be
affected. The object codes and their meanings include:

w word
(from cursor to start of sentence
) from cursor to end of sentence
{ from cursor to start of paragraph
} from cursor to end of paragraph
^ from cursor to first non-space character of line
$ from cursor to end of line

In addition:

dd deletes the entire text line
yy yanks the entire text line
cc changes the entire text line

Here are some examples to help you learn vi editing command syntax:

dw Delete word.

2dw Delete two words.

3cw Change three words.

y(Yank from cursor to beginning of sentence.

c) Change from cursor to end of sentence.

d{ Delete from cursor to beginning of paragraph.

y} Yank from cursor to end of paragraph.

c^ Change from cursor to beginning of line.

d$ Delete from cursor to end of line.

5dd Delete the next five lines.

3yw Yank the next three words.

10X Delete 10 characters to the left of the cursor.

Note: Vi considers a sentence to be a text string ending in a period, question mark, or exclamation point and followed by a new
text line or two spaces. It considers a paragraph to be a series of text lines followed by a blank line.

Lab 10.5.2 - Searching for a text string

Vi lets you define a text string and search your file buffer for it. Suppose you want to search the file buffer for every occurrence of
the word "halyard."

In edit mode, press /. A cursor appears at the bottom of the screen. Now, type halyard and press ENTER. Vi searches the file
for the first occurrence of the string, placing the cursor on the first character when it finds it.

You can now press n to advance to the next occurrence of "halyard," or enter a new search string. Vi search commands include:

/x Search forward for x.

?x Search backward for x.

n Search forward for next occurrence of defined string.

N Search backward for next occurrence of defined string.

The following special characters act as "wild cards," which allow you more flexibility in searching for a text string:

. Match any character. For example, the search string m . d would match "mad," "mud," and "mod."

* Match any string. For example, the search string hoo* would match "hoosiers," "hoops," and "hook shot."

Lab 10.5.3 - Search and replace

You can use vi to search your file buffer for a text string and replace each occurrence with new text. For example, to replace
"halliard" with "halyard," in edit mode type:

:%/s/halyard/halliard/gc

Vi searches the file for "halyard." You'll see each occurrence in context. Press y and then ENTER to replace the word and
continue, or press n and then ENTER to continue searching without altering the text. (To stop a search in progress, press ESC.)
When done, vi prompts you to press ENTER. Vi now displays the file buffer with the changes in place.

You can also search and replace without confirmation. For example, in edit mode type:

:%/s/halyard/halliard/g

Vi searches and replaces each occurrence. However, it does not ask you to approve the changes.

Lab 10.5.4 - Getting help

The online manual contains a vi summary. To see it, at the Unix shell prompt, type man vi.

Exercise 10.1

Create a file called days and type the days of the week into it, one below the other starting with Monday.

Save the file and text and Create a file called words and copy the following:

ROY KEANE is set to declare his availability to play for the Republic of Ireland again.

The estranged former Irish captain is expected to confirm his willingness to play for his country following Mick McCarthy's resignation on Tuesday.

Highly-placed sources in the FAI have been given strong indications that Keane will shortly announce his return from self-imposed exile.

Go back into the days file and add a list of the Months on to it.

Add the contents of the days file to the words file.

Use the various vi commands on this file.