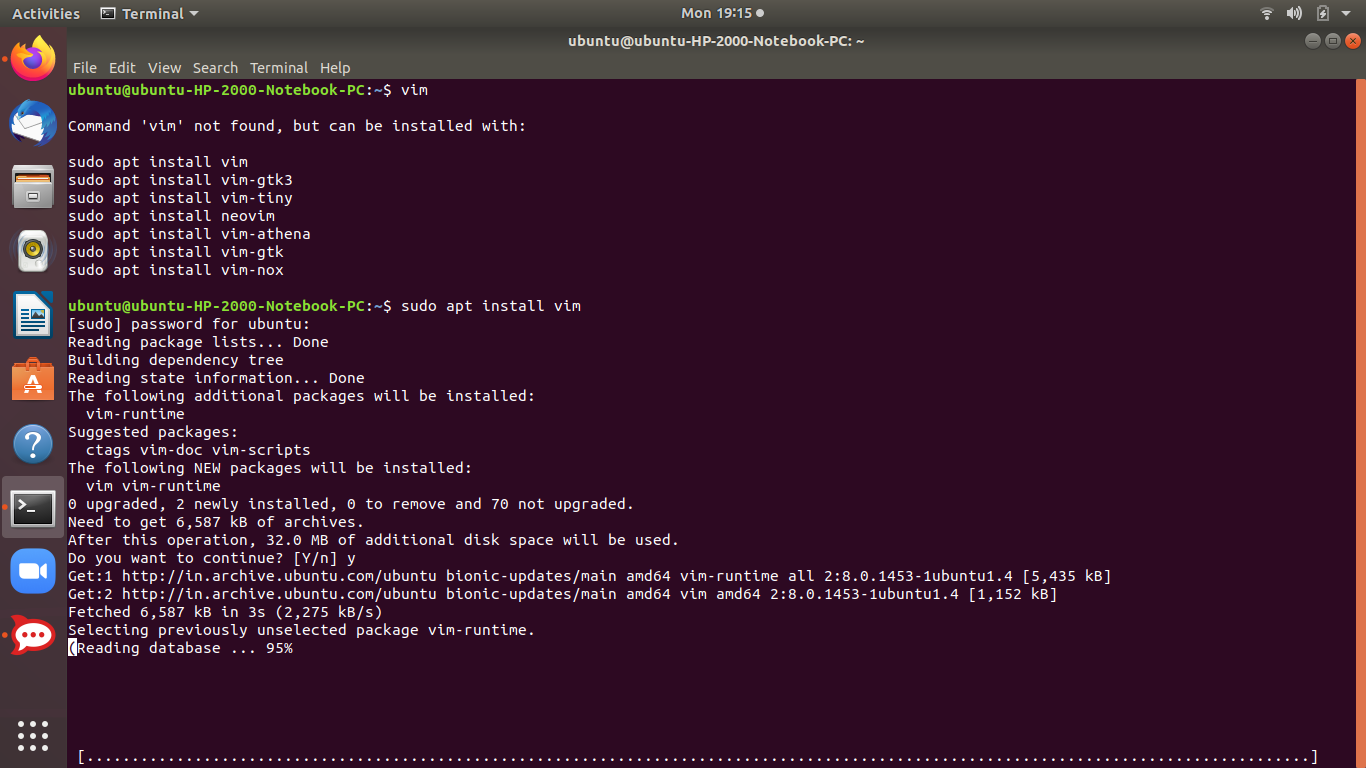
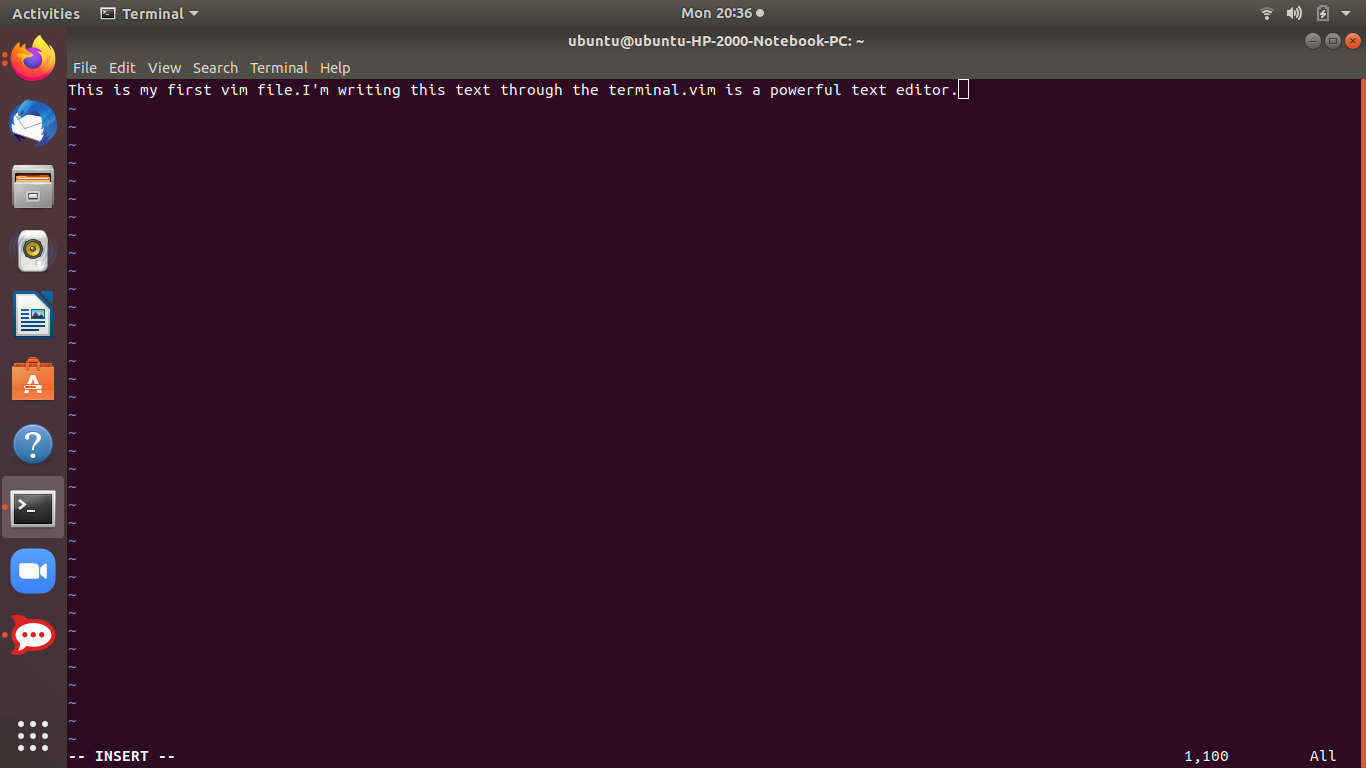
**VIM-editor In Linux**

**Vim** is an advanced and highly configurable text editor built to enable efficient text editing. Vim text editor is developed by Bram Moolenaar. It supports most file types and vim editor is also known as a programmer’s editor.

To install vim : $ sudo apt install vim

1. **To Create a file in vim:** vim filename.txt

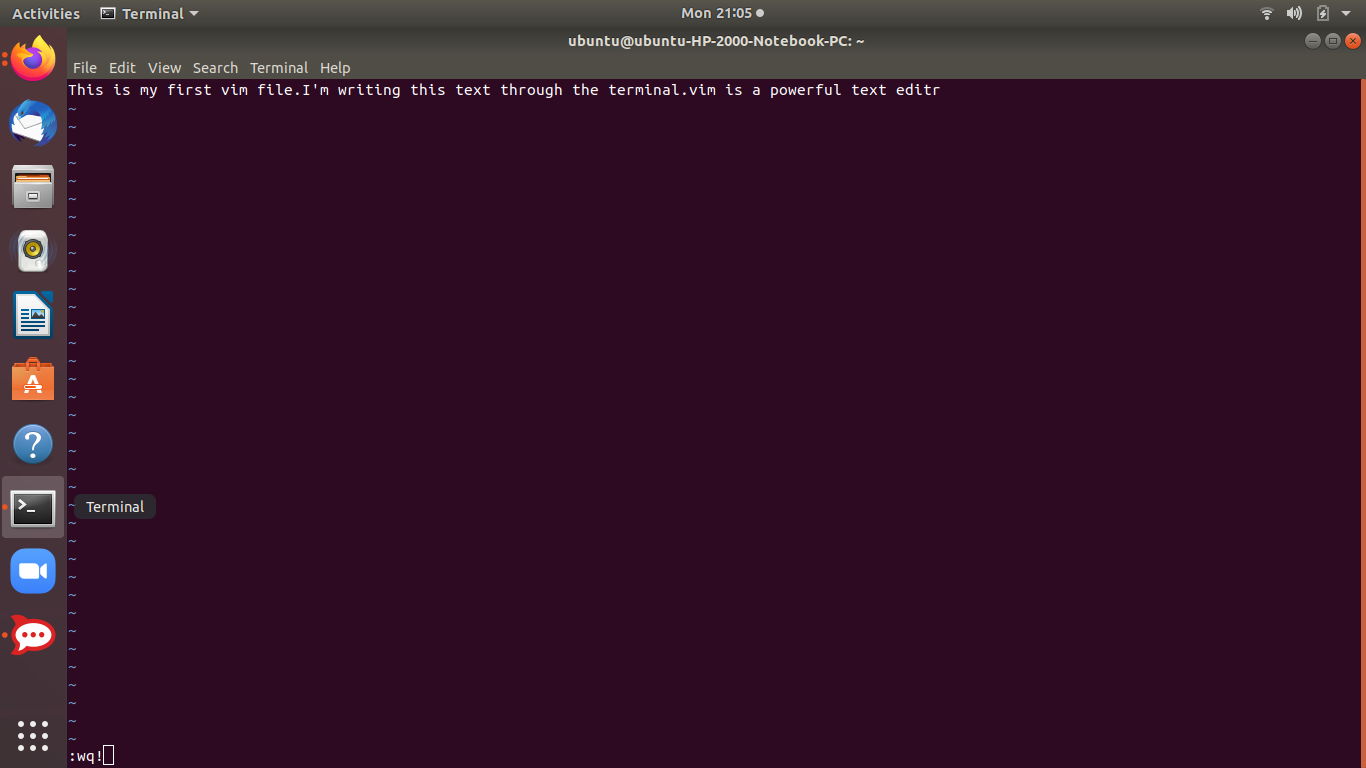
***Write into the file:***To insert text into this new vim file we need to go into the insert mode by pressing the key “i” and then enter our text.



**2) Exit and save:** to save and close the file to do that first exit from insert mode by pressing the Esc key.

To write a command first type semicolon( and then type the command wq! And then hit ENTER.

:wq!

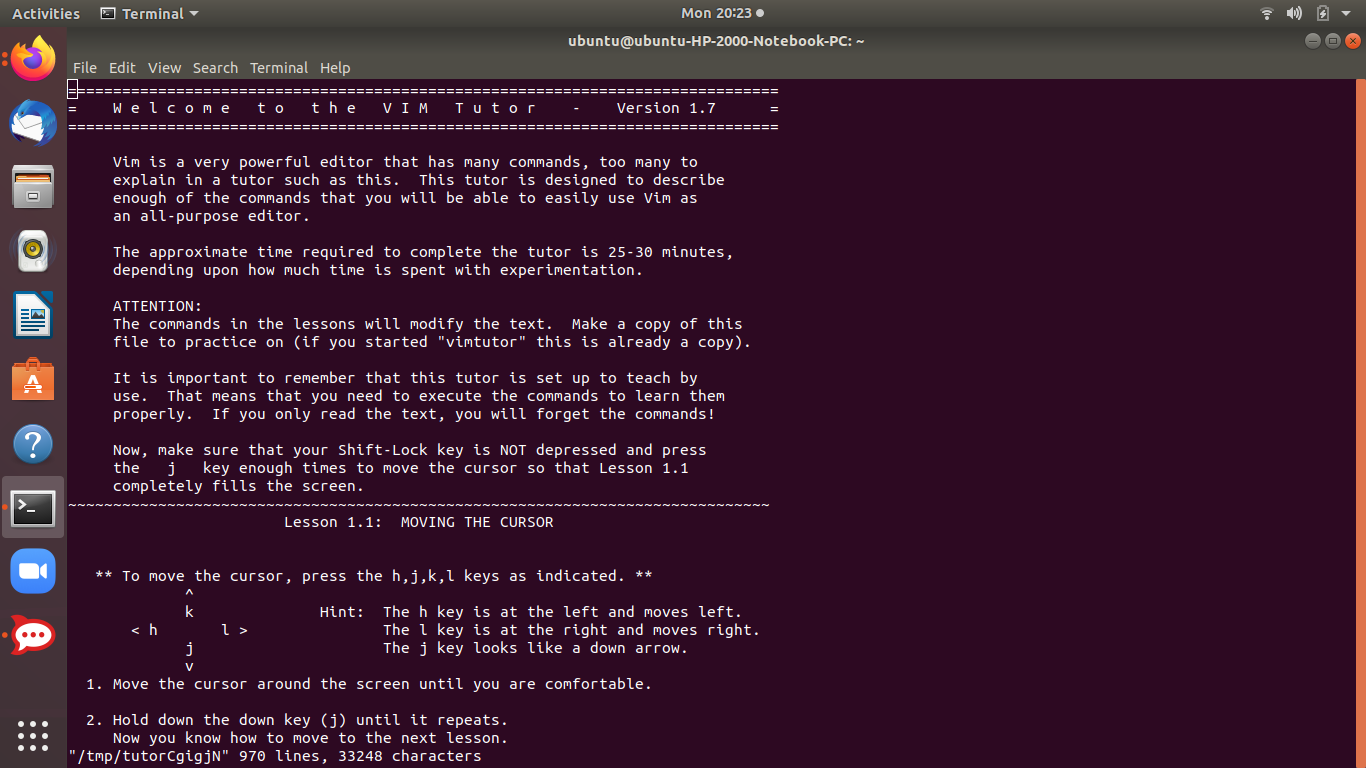


**Exit without saving the** **file:**

To exit from the file without saving the file just use the command q! As follows :q!

* **Vimtutor:** Vim comes with its own tutorial. You can see this tutorial by command vimtutor into the terminal .

$ vimtutor



* \*\* To move the cursor, press the h,j,k,l keys as indicated. \*\*

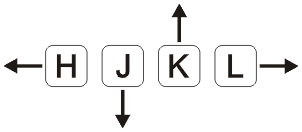
^

k Hint: The h key is at the left and moves left.

< h l > The l key is at the right and moves right.

j The j key looks like a down arrow.

V



To move the cursor at the start of the file use the gg command

:gg

To move the cursor at the bottom of the file use the G command

:G

To view the current cursor location in the file using the following command:

:ctrl+g

**Exiting Vim:**

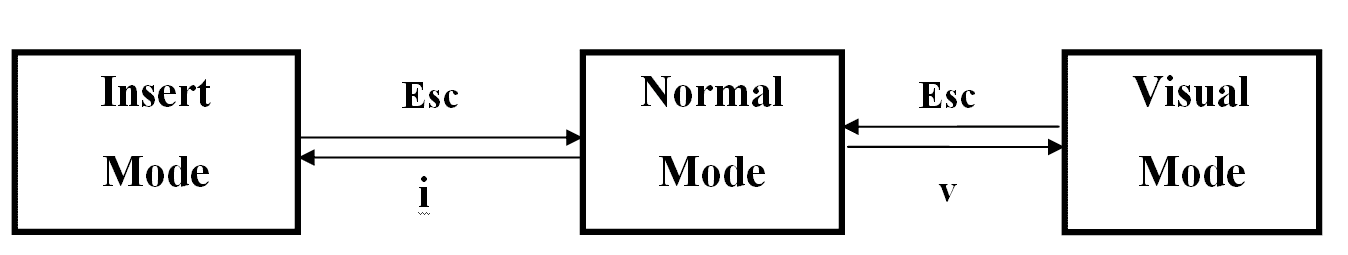
To exit without doing anything go into command mode by pressing the Esc key and type the following command. :q

To exit from vim without saving changes type the following command in vim :q!

To quit and save changes type the following command :wq

**Modes Of vim:** vim is a modal text editor.It has 3 modes:

1. **Normal Mode:** for editing through and simply navigating .
2. **Insert Mode :** to insert/write and edit text into the file
3. **Visual Mode:** to select and delete large texts.



*Reference link :* <https://youtu.be/ER5JYFKkYDg>

**Basic Commands:**

* i : insert
* Esc: to escape insert mode
* gg: returns the cursor to the top of the document
* Shift+g: returns the cursor to the last line of the document
* 0: (zero)to get on the first character of the text
* 2g: to get on the 2nd line of the text and so on..
* v: visual mode
* x: used to delete characters
* 2x: to delete 2 characters ,2x,4x..
* u: undo
* 2u: to undo 2 characters ,2u,4u..
* dd: sdelete a line

**How to navigate without H,J,K,L Keys:**

* w: goes to beginning character of the word
* e: goes to nd character of the word
* 3w: moves 3 words forward, so on..
* b: backwards
* 3b: move 3 words backwards

**Put Command:** to put a particular line on some other place

Eg: 17GP puts the selected text on a line after 17th i.e;18

**Replace Command:** if we mistyped a letter and we want to correct,and replace

It with other word.

Eg: ri replace letter i

To replace the line use c$ command: c$

**Change Command:** to change a word , ce

**Search Command:**To search the word After the cursor uses the backslash key and then write the word and press enter.

command :/word

Use n to move on next matching word :n

**Search and Replace:**

To replace the word in file use s/ command in vim like

:s/searchword/replaceword/

To do replace all occurrence of word use g

:s/searchword/replaceword/g

**Text Editing: Insertion**

We have edited some text files before by using the i key. There are Four keys used for insertion text. Just type the key into the normal mode in vim.

* i -> This key is used to put the cursor before the current position.
* a -> This key is used to put the cursor after the current position.
* o -> This key is used to put the cursor below the line.
* O -> This key is used to puts the cursor above the line.

**Motion:** Motions provide context to your Operators. These execute the action in a particular way.

Here is a list of some motions

* w - until the start of the next word, EXCLUDING its first character.
* e - to the end of the current word, INCLUDING the last character.
* $ - to the end of the line, INCLUDING the last character.

**Count:** Count is the number for which replete the motion for count number. Here is a demonstration of the use of count and motion

* To move courser 2 words forward use the following command
* 2w Here 2 is the number of counts and w is used for word
* To move cursor 4 line forward use the following command 4$

**Deletion Commands:** Always use the Esc key to go into normal mode and use the insertion, deletion keys, and other keys.

* dw -To **delete the word** move the cursor to the beginning of the word and use dw command in normal mode. The word under the cursor will be deleted.
* d2w To delete 2 words use the command
* d$ -To **delete the line** move cursor to the beginning of the line and use d$ command in normal mode. The line under the cursor will be deleted.

**Undo and Redo:**

As we are programmers most time we are using undo and redo .vim to provide these to both features in it. To undo press u key in normal mode

u

To redo use the ctrl+r key in normal mode in vim

Ctrl+r

## **Visual commands**

Type any of these while some text is selected to apply the action

* y - yank (copy) marked text
* d - delete marked text
* c - delete the marked text and go into insert mode (like c does above)

## **Cut and Paste**

* yy - yank (copy) a line
* p - put (paste) the clipboard after cursor
* P - put (paste) before cursor
* dd - delete (cut) a line
* x - delete (cut) current character
* X - delete previous character (like backspace)

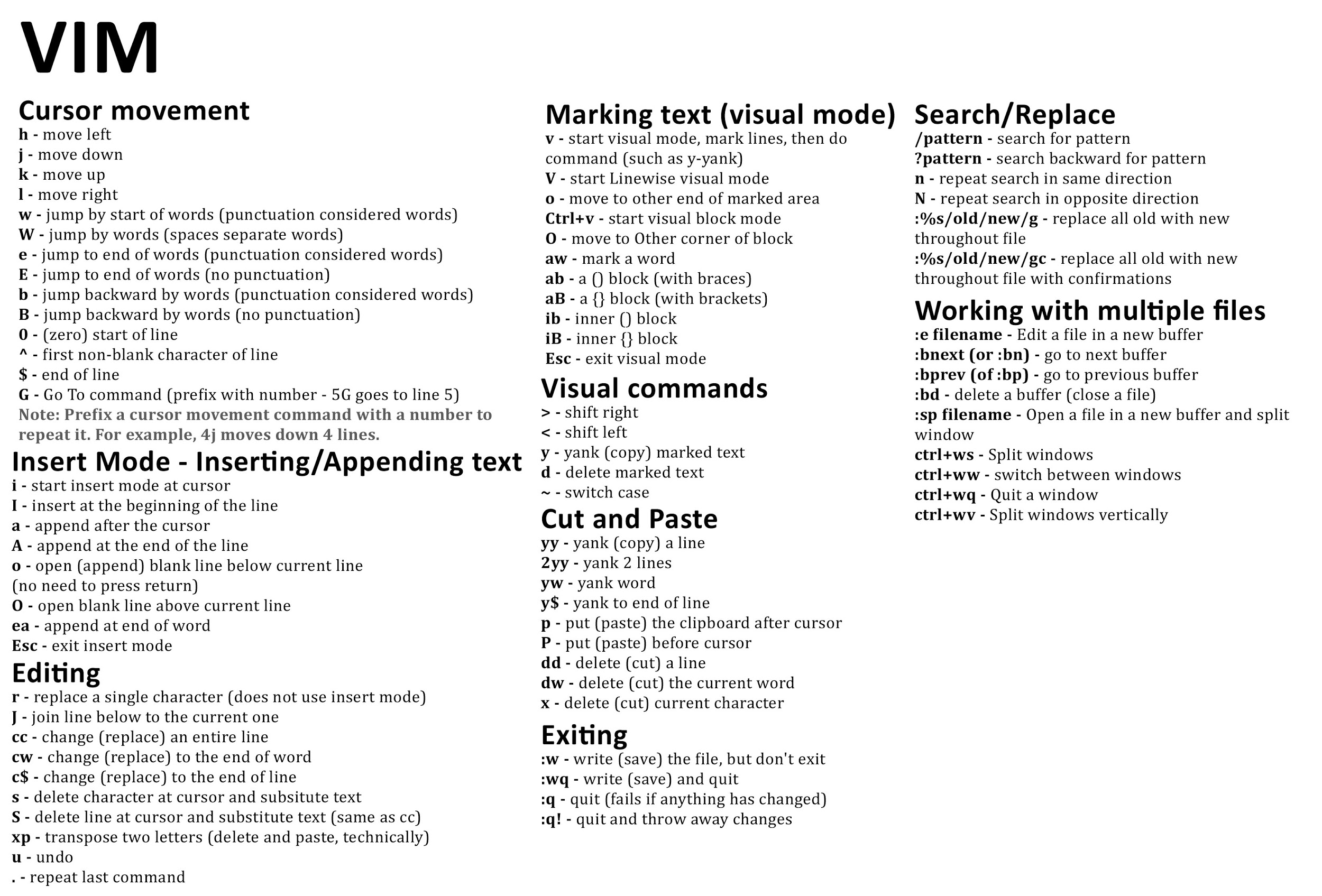
## **Marking text (visual mode)**

* v - starts visual mode
  + From here you can move around as in normal mode (h, j, k, l etc.) and can then do a command (such as y, d, or c)
* V - starts linewise visual mode
* Ctrl+v - start visual block mode
* Esc or Ctrl+[ - exit visual mode
* Advanced
  + O - move to other corner of block
  + o - move to other end of marked area

## **Working with multiple files**

* :e filename - Edit a file
* :tabe - Make a new tab
* gt - Go to the next tab
* gT - Go to the previous tab
* Advanced
  + :vsp - vertically split windows
  + ctrl+ws - Split windows horizontally
  + ctrl+wv - Split windows vertically
  + ctrl+ww - switch between windows
  + ctrl+wq - Quit a window

Other vim commands are available in the sheet below:

***Sheet containing vim commands:***

Reference links:

<https://www.geeksforgeeks.org/getting-started-with-vim-editor-in-linux/>

<https://vimsheet.com/>

# **Buffers**

Buffers in vim are the in-memory text of files. Your window is a viewport on a buffer. You can switch between open buffers, this is similar to tabs in other editors.

## **Opening Multiple Files**

You can specify opening multiple files on the command-line. For example, vim file1 file2 file3 each file opens in its own buffer.

## **Main Buffer Commands**

:buffers or :ls : Show open buffers

:b {bufname} : Use buffer name, supports partial and tab completion

:bd : Close current buffer

:bn : Switch to Next buffer

:bp : Switch to Previous buffer in list

:b# : Last buffer visited, actual # sign

:b1 : Switch to buffer #1

:bm : Switch to next modified buffer

## 

## 

## **Buffer Navigation**

Use the above commands to navigate and switch between buffers. The :b {bufname} might be the most useful. Type :b and then start typing the filename, partials work if it is unique, or use tab completion.

## **Split Buffers**

You can open buffers in a split window, using the follow. This opens a second window with the buffer loaded. If you want to close the split, and keep the buffer open, use :close or ctrl-w c and not :bd. Using :bd will close the buffer, and if it is in the only buffer open in a split window, it will also close the window.

:sb 1 : Open buffer #1 in a split

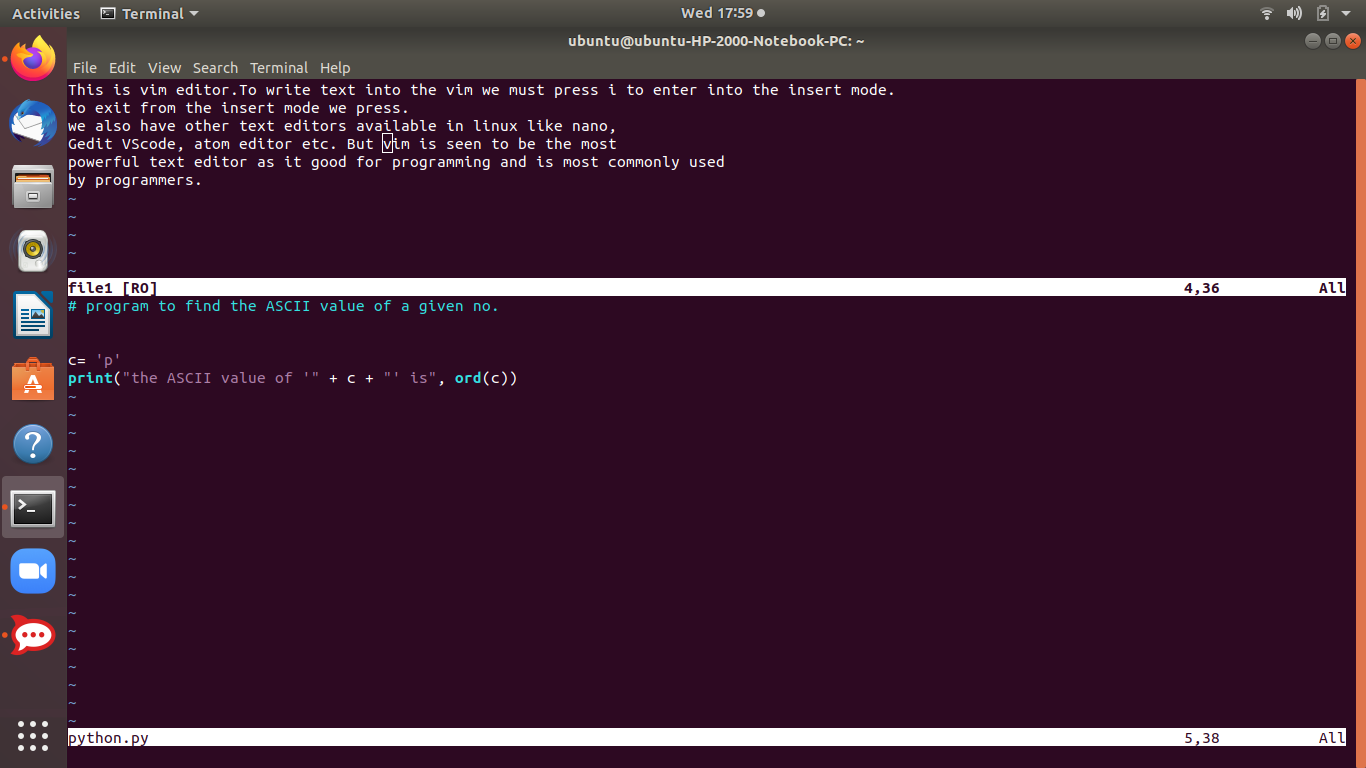
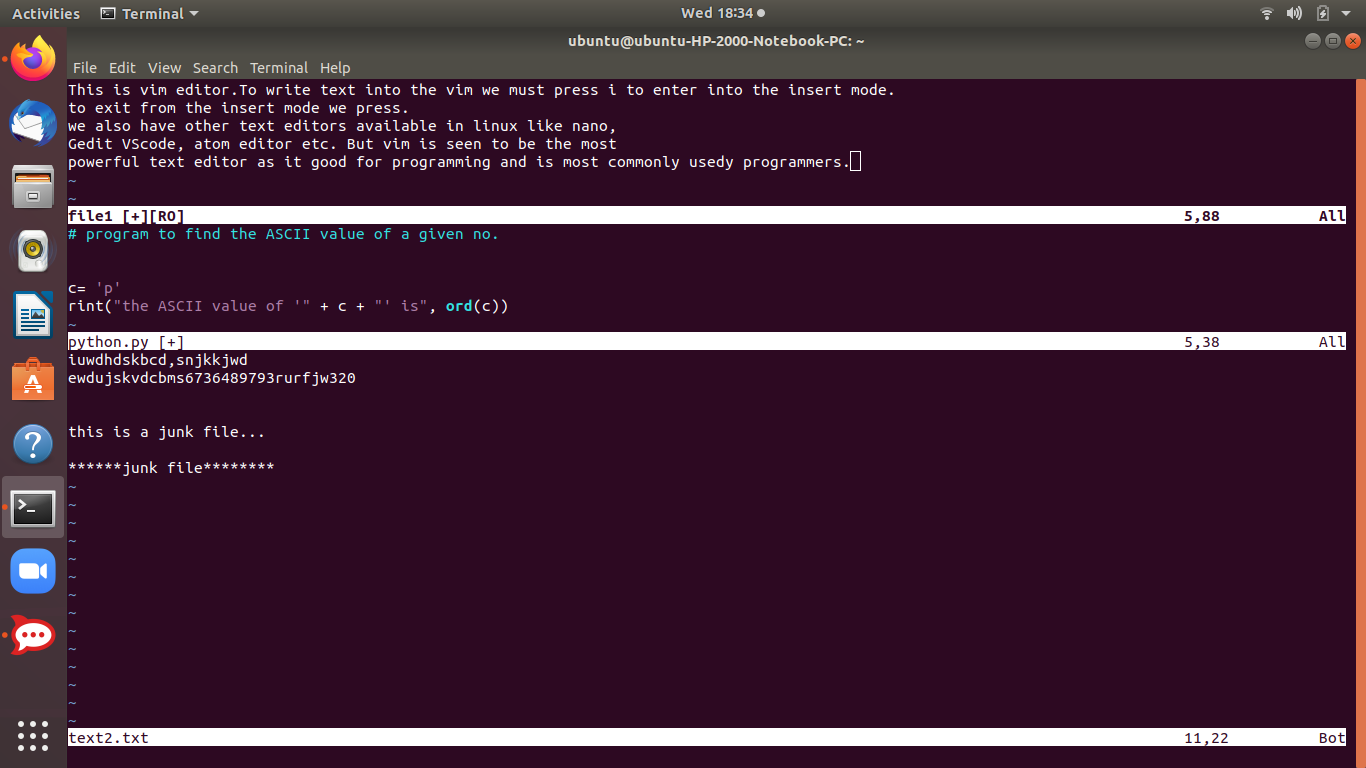
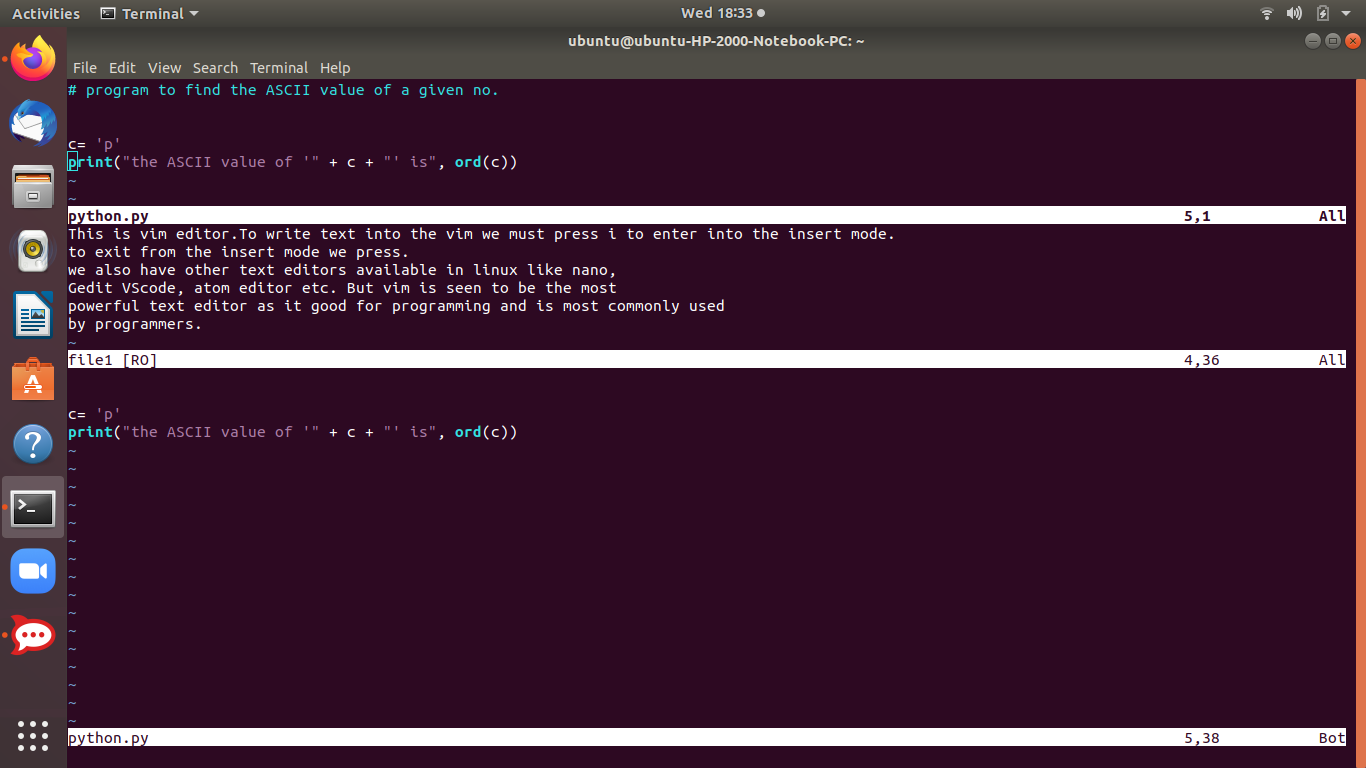
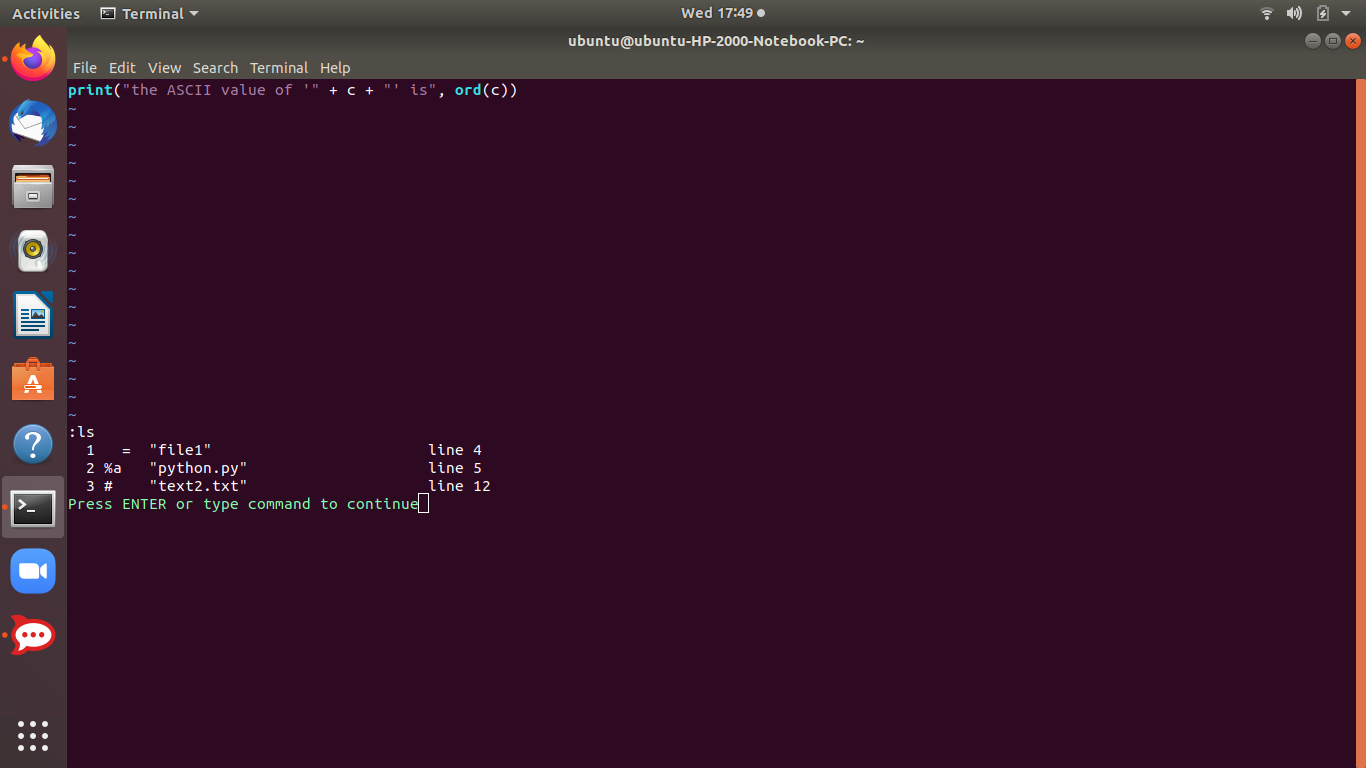
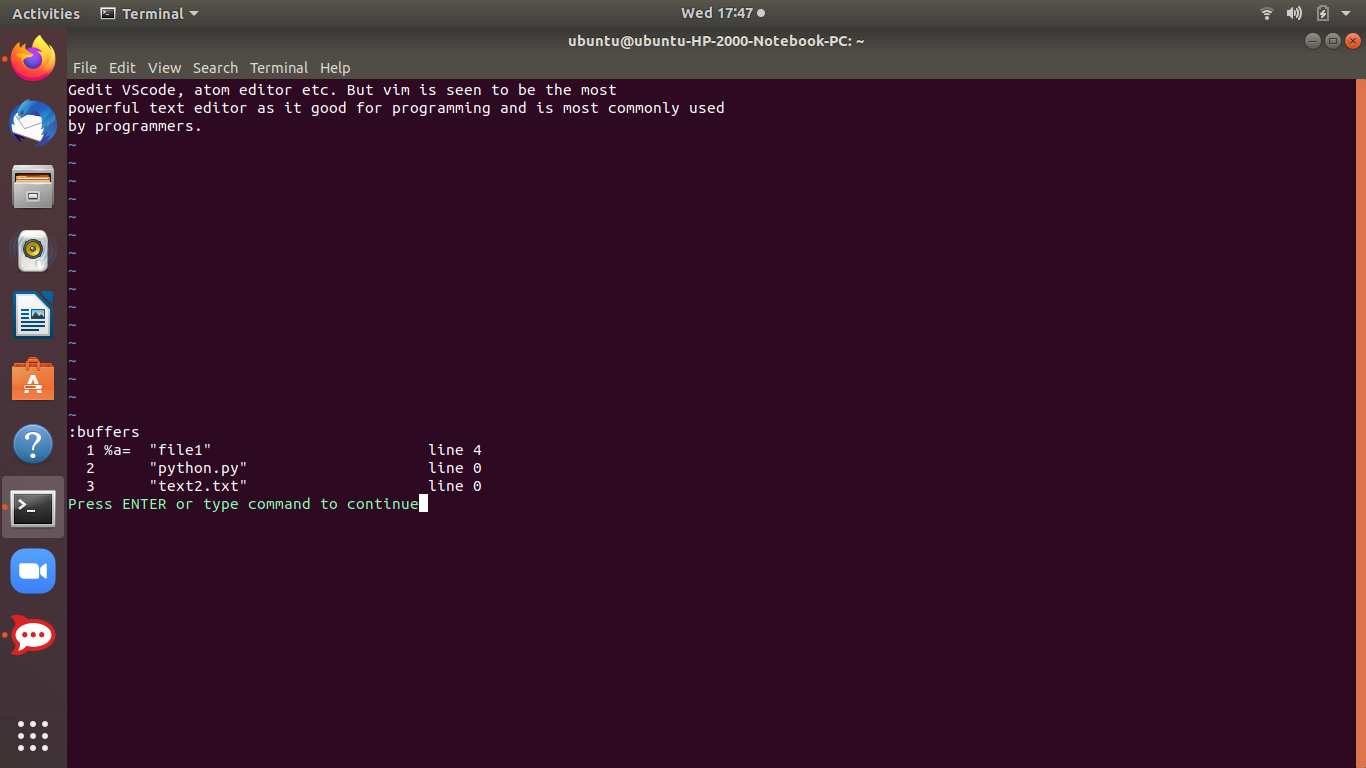
:sb {bufname} : Open buffer bufname in a split

:sbn : Open next buffer in a split

:sbp : Open previous buffer in a split

:sba : Open all buffers in split windows

* *Here # is the number sign and %a is the buffer sign .The file name against each buffer sign shows which file is currently open in the buffer.*



* The last 3 screenshots show the split command for the 3 files I’ve opened in vim.

Reference link:

<https://mkaz.blog/working-with-vim/buffers/>

**Tabs**

In Vim, each file is loaded into a buffer, which can be displayed in any number of windows, in any number of tabs.In each a new file can be opened and tabs make editing and working with multiple files easy.These are viewports or work spaces for the programmer.

**Opening a tab:**

* run the :tabnew command while in normal mode. This will open a new tab with an empty buffer.
* If you want to edit a file in the new tab, run :tabnew *filename*

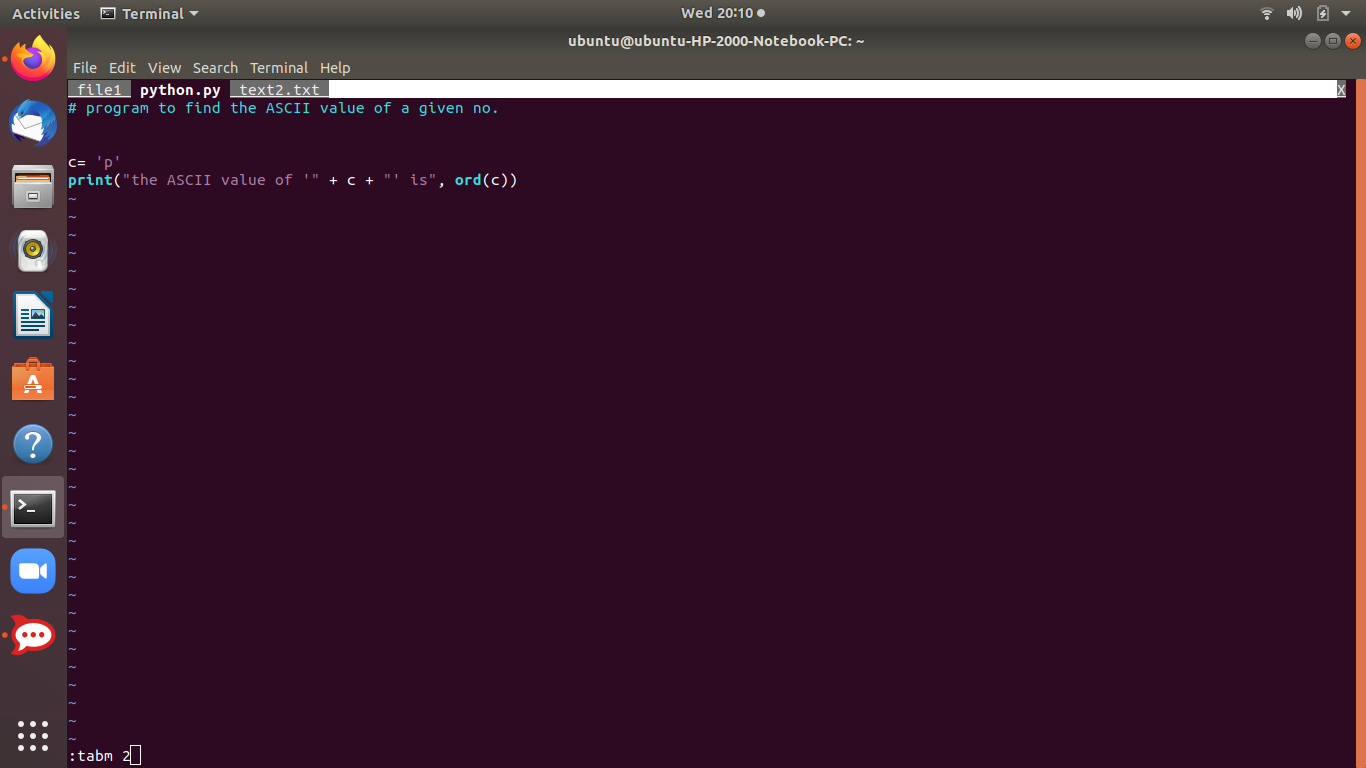
**Moving in between tabs:**

* switch between tabs using :tabn and :tabp, or you can use gt while you’re in normal mode.
* If you have a lot of tabs open, you can use :tabfirst, or just :tabfir, to jump to the first tab, and :tablast to jump to the last tab that’s open.

**Rearranging tabs**

* move the tabs to a specific spot in the tab order using :tabm *n*, where *n* is the position number that you want to use.
* If you don’t give the :tabm command an argument, then the current tab will be moved to the last spot.

*Numbering of tabs starts from 0*



*Reference link:*<https://www.linux.com/training-tutorials/vim-tips-using-tabs/>

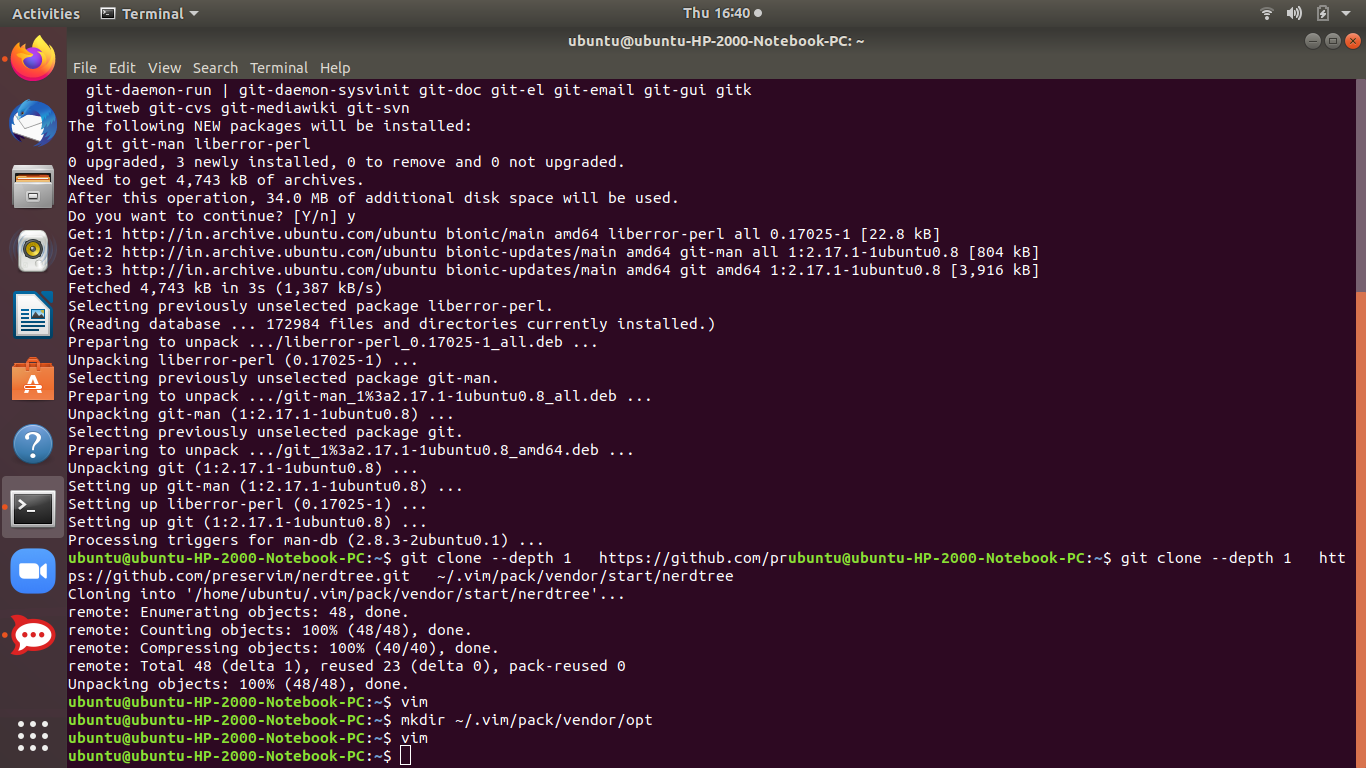
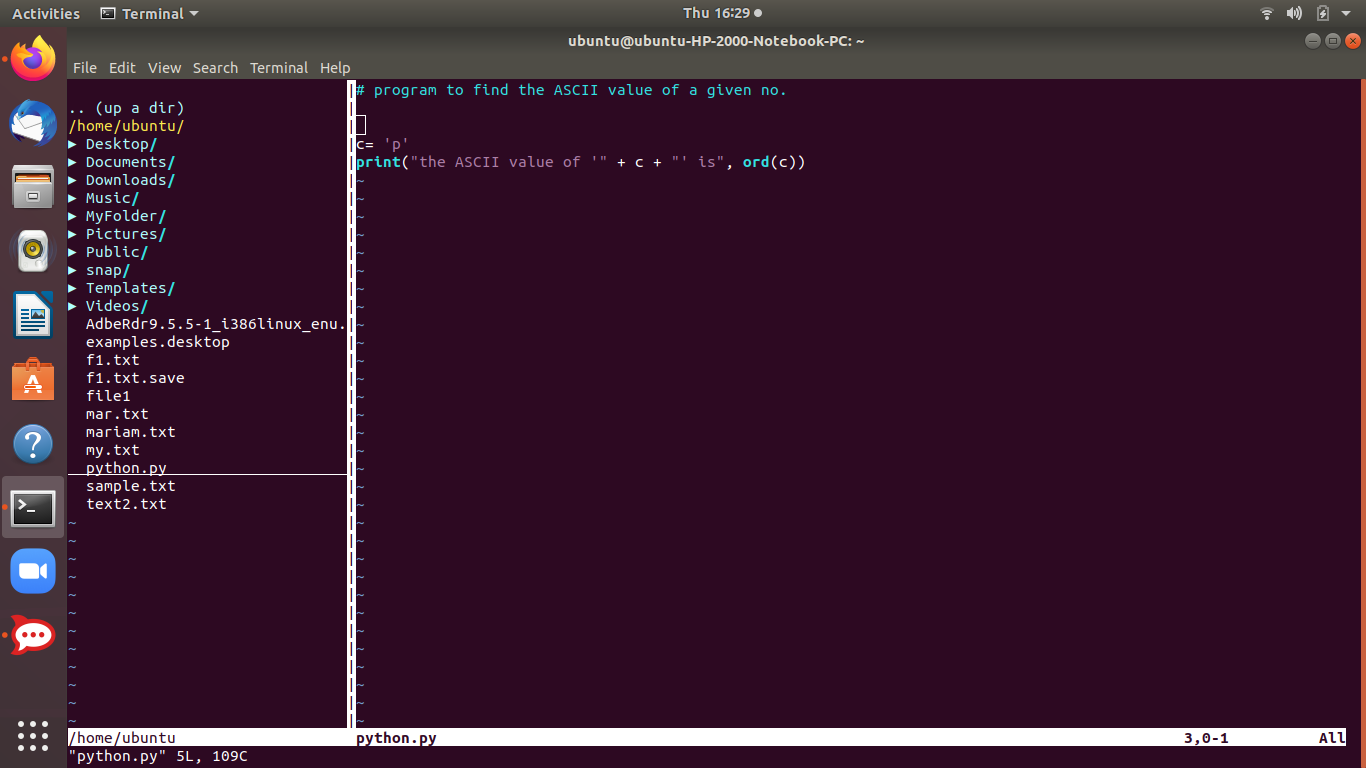
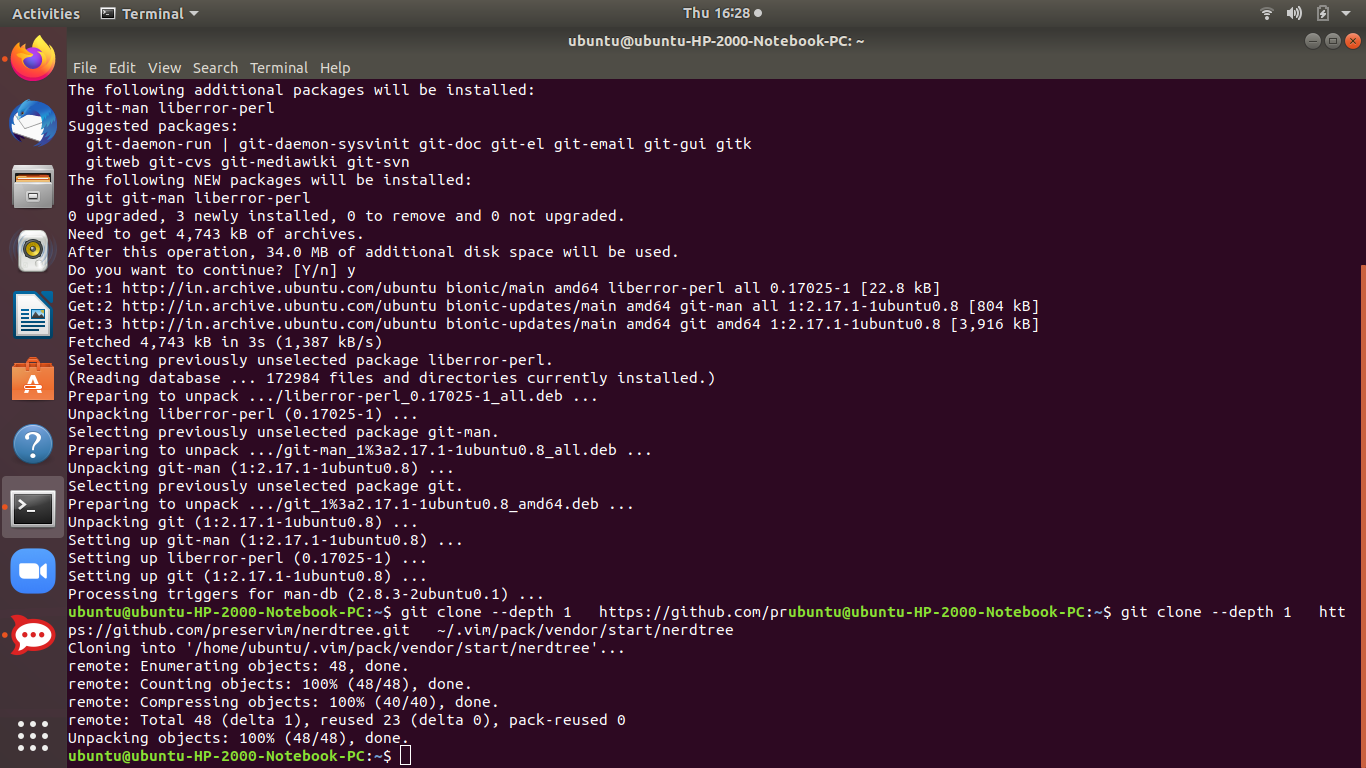
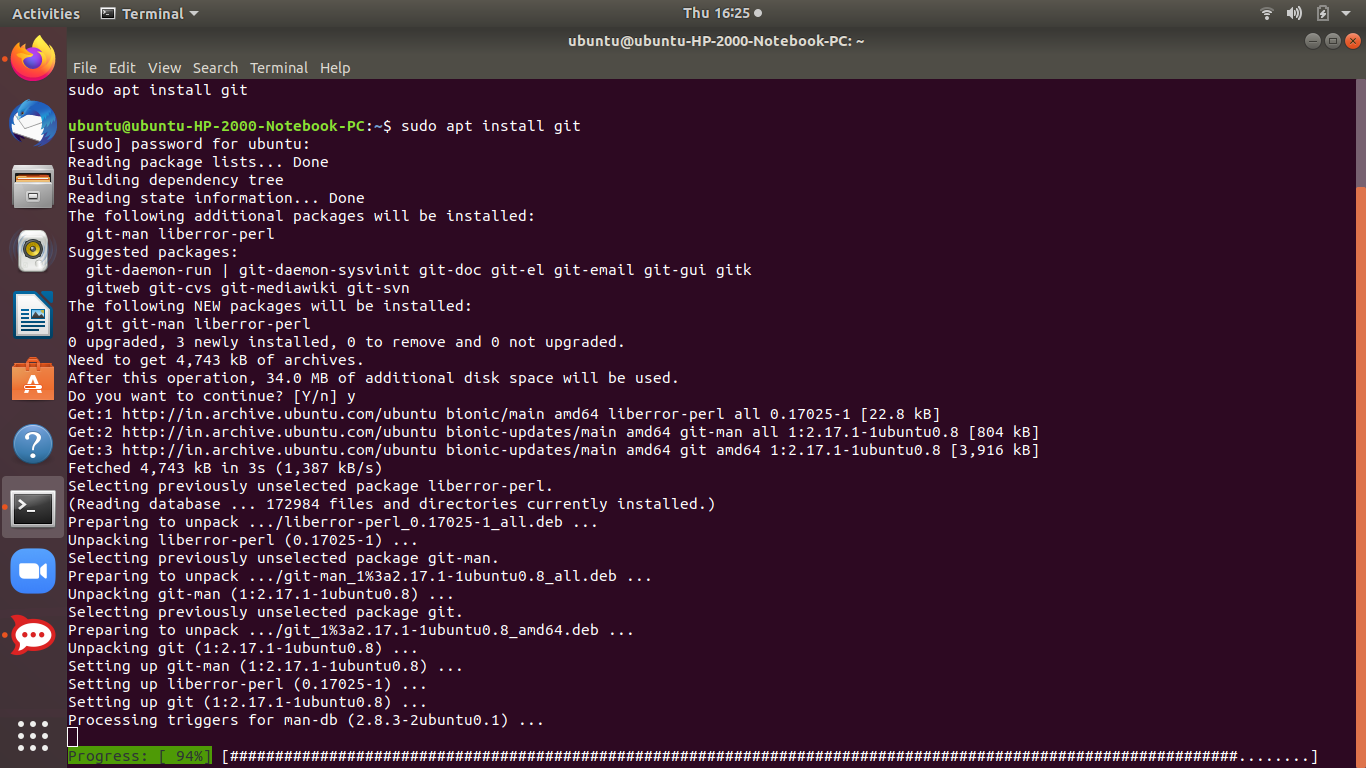
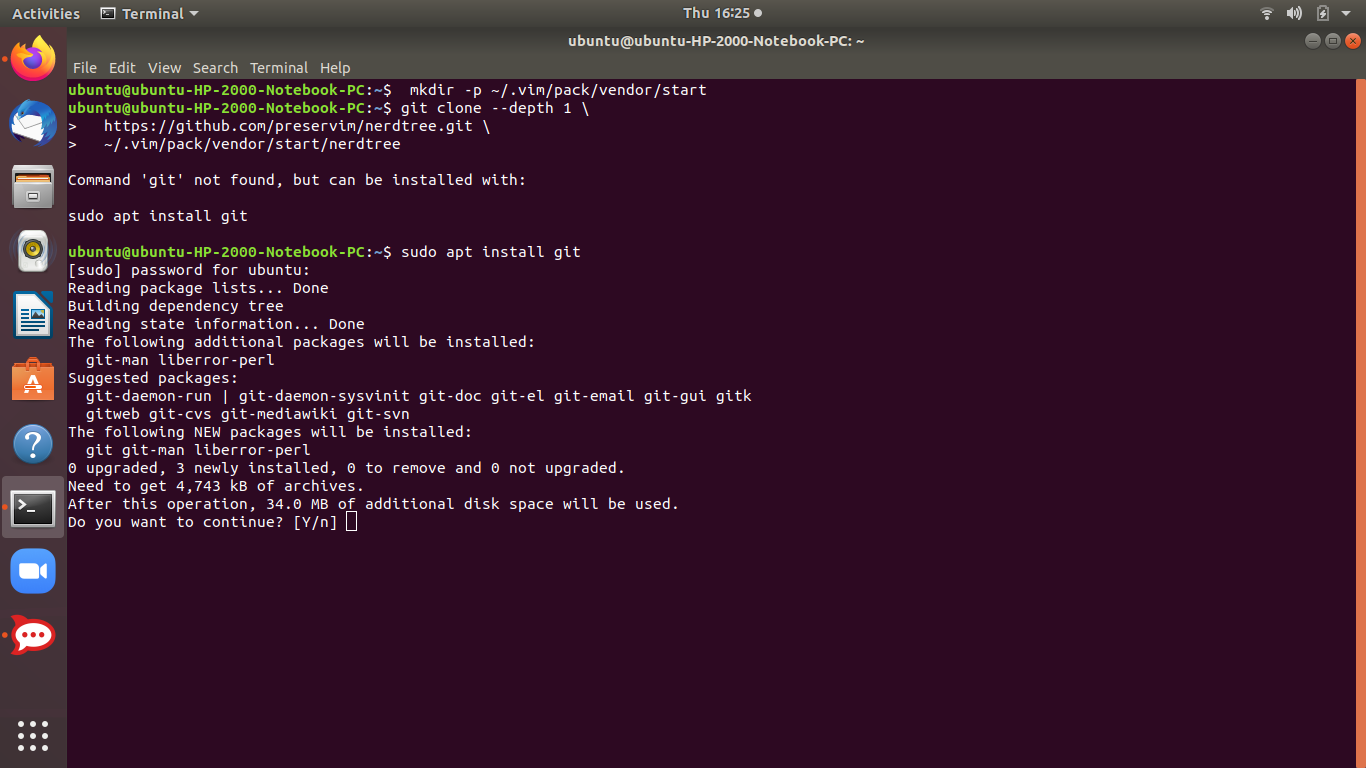
**Plugins in Vim**

Plug-ins are extensions or add-ons to an existing software that extend the functionality of that software.

Much like Notepad++ and Sublime, Vim is a text editor rather than full-fledged IDE. This means that almost all functionality is added via plugins.

Most commonly used plugins in vim are:

* *Vim Airline :gives a powerful tabline and status at top of vim*
* *NERDtree :visually explores directories in tree form*
* *Vim Gitgutter: helpful in using Git repositories(add,del,remove)*
* *Vim Fugitive: the main feature is :Git which calls any git command*
* *There are innumerable other plugins based on what we want …*



*In the above screenshots we have installed the plugin NERDTree manually .NERDTree gives a parent directory structure to our directories and files*

**We can install vim plugins in 2 ways:**

1. Manually
2. Through Plugin manager

**Reference links:**

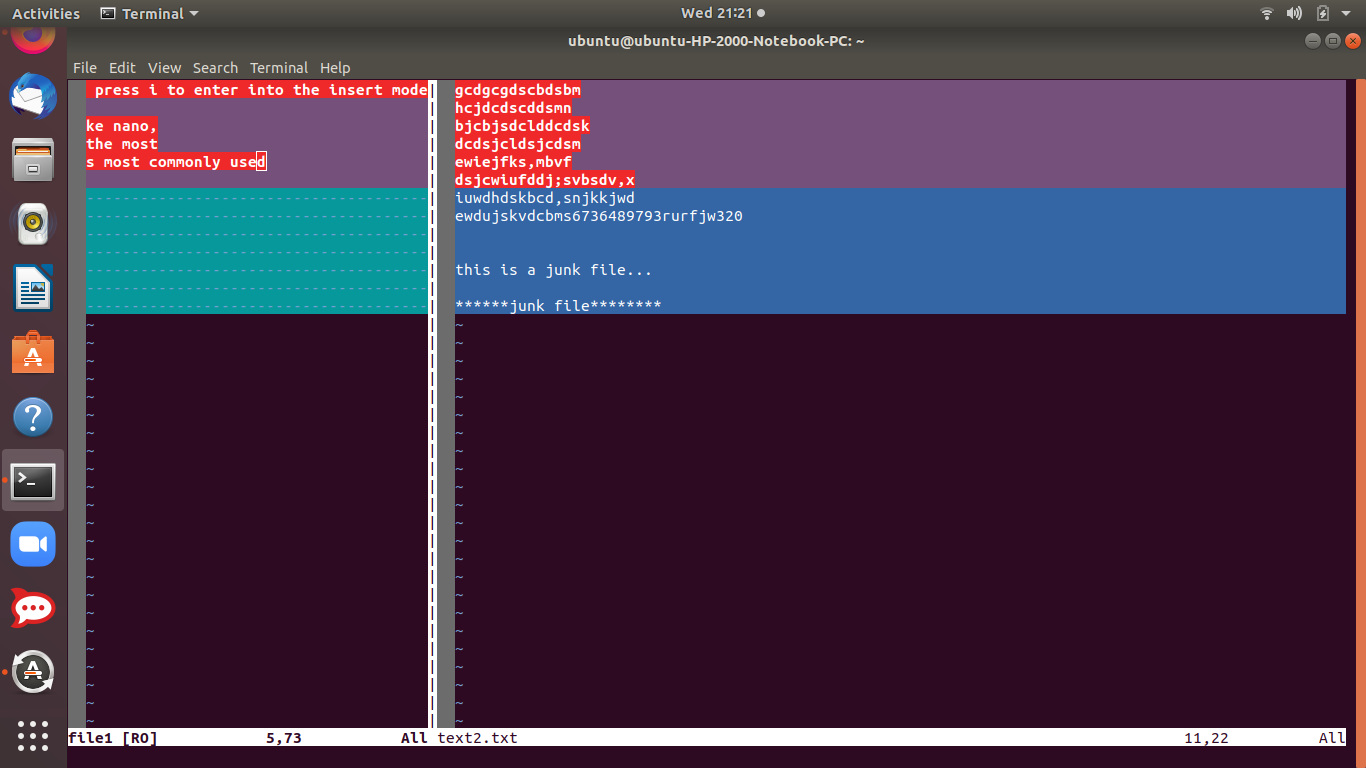
<https://opensource.com/article/20/2/how-install-vim-plugins>

<https://linuxhint.com/vim_install_plugins/>

**How to compare 2 files for similarity/difference**

* **Vimdiff command:** Vimdiff enables a user to edit uptill 4 different versions of a file while showing their differences.When we run it ,it opens 2,3 or 4 files using vim text editor.

$ vimdiff file1 file2 file3..



## 