Basic Guide: Using VIM

Introduction:

VI and VIM are in-terminal text editors that come standard with pretty much every UNIX operating system. Our Astro computers have both. These editors are an alternative to using Emacs for editing and writing programs in python. VIM is essentially an extended version of VI, with more features, so for the remainder of this discussion I will be simply referring to VIM. (But if you ever do research and need to ssh onto another campus's servers and they only have VI, 99% of what you know will still apply).

There are advantages and disadvantages to using VIM, just like with any text editors. The main disadvantage of VIM is that it has a very steep learning curve, because it is operated via many keyboard shortcuts with somewhat obscure names like dd, dw, d}, p, u, :q!, etc. In addition, although VIM will do syntax highlighting for Python (ie, color code things based on what type of thing they are), it doesn't have much intuitive support for writing long, extensive, and complex codes (though if you got comfortable enough you could conceivably do it). On the other hand, the advantage of VIM is once you learn how to use it, it is one of the most efficient ways of editing text. (Because of all the shortcuts, and efficient ways of opening and closing).

It is perfectly reasonable to use a dedicated program like emacs, sublime, canopy, etc., for creating your code, and learning VIM as a way to edit your code on the fly as you try to run it. Or, to give another example, when I did research involving running simulations on a supercomputer, once the fortran files (like .py files) were transferred to the super computer, the only way to actually edit them was through vi, as the supercomputer didn't have emacs, etc, installed. So it's handy to know.

*Note, for the purposes of this tutorial, ">>" indicates the end of your prompt, where you begin typing in terminal.

Tutorial:

VIM can be intimidating at first, but bear with me. You can create a new file by typing

```
>>VIM filename
```

The syntax is identical for opening an existing file. When you open a new file, what you see is a bunch of (~) going down the page. These are just place holders.

The basic rule of VIM is that there are two modes: insert mode, and edit mode. When you are in insert mode, VIM behaves like a normal text editor: you can type letters, backspace them, enter new lines, change variable values, etc. When VIM is in edit mode, your cursor will navigate around the document, but you won't be able to type anything. However, from here you can quickly search the document, delete lines and words, delete paragraphs (or parts of them), paste, undo, redo, delete the remainder of a line, and much more.

- To enter insert mode from edit mode, press "i" or the insert key on some pcs.
- To enter edit mode from insert mode, hit <esc>.

When you are in edit mode, typing letters will cause them to appear at the bottom of the screen in a small bar; this is where you can enter commands for things. I'm simply going to list those commands here, and let you experiment on how they work (try creating a document of gibberish, with multiple paragraphs and spaces between words, to mess around with).

Commands while in edit mode:

- 'i' –change to insert mode
- 'dd' –delete entire line (cursor can be anywhere in line)
- 'dw' –delete word (starting from character cursor is on), i.e. to delete the whole word you must be on the first letter
- 'd}' – delete up to the next paragraph (from current cursor position)
- 'D' – delete remainder of line (from current cursor position)
- 'p' – paste most recently deleted word/block of text
- 'u' – undo last change
- '.' – repeat last change
- ':q!' – "quit, don't save", closes document without saving any changes made.
- ':wq' – "write/quit," writes file to disk (saves it), then closes the document, leaving you back in terminal.
- ':n' – move to the nth line of the document (ex. :4)
- '/text' search document for string "text"
- 'G' – move to last line in file.