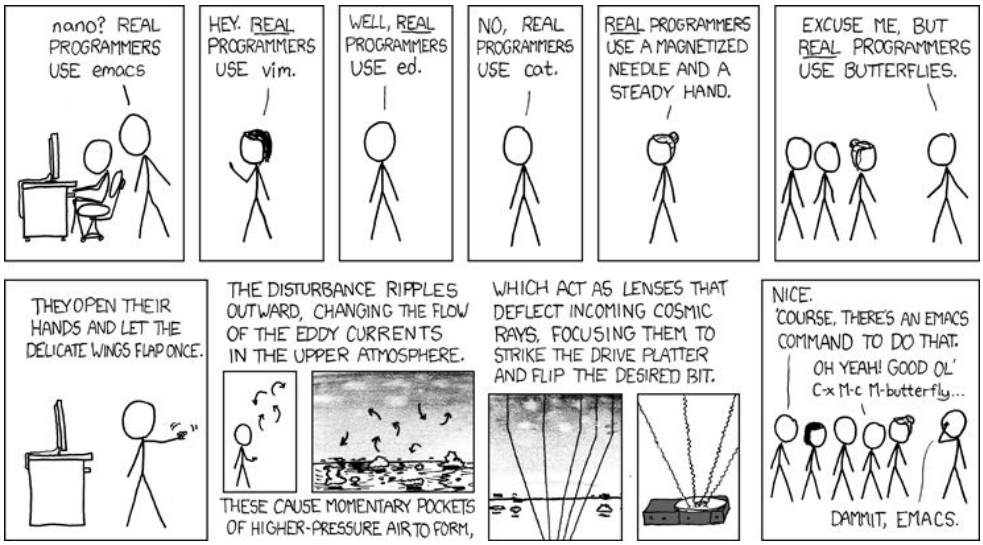
vim & .vimrc Cheat Sheet



What is vim?

- Vim is an in terminal text editor, it is useful when sshing into the lab to do work.
- There are also other in terminal text editors like Emacs that are just as good. It's really one half dozen or the other.

Starting Vim

- To start up vim in a terminal just run **`vi [file]`** or **`vim [file]`**
- When you first open a file in vim you will be in command mode. You don't start out being able to type into your file.
- Learning a few useful commands can make your vim life a lot easier

Commands

:q	• **quit** vim • **NOTE:** If you have not saved since your last edit you will get a warning and it won't let you exit. If you really want to exit without saving changes use :q!
h, j, k, l	• Will move your cursor around in your file. • Thankfully most of the lab's versions of vim support arrow key movement as well.
:i OR i	• Puts you into **insert** mode, will allow you to actually type into the file.
[Esc]	• Will take you back to command mode from insert mode.
:w	• **write** to a file (aka save) • Unfortunately vim doesn't natively support the [ctrl] + s we are all used to doing to save files.
:wq OR :x	• **write** and **quit** at the same time • Can be nice if you are trying to work quickly.

dd OR D	• Will delete the line of code that your cursor is currently living at. • Can also do [number]dd to delete the next [number] lines including the one you are on.
yy OR Y	• Will **yank** (copy) the line your cursor is currently on. • Can also do [number]yy to copy the next [number] lines including the one you are on.
p OR P	• Will **put** (paste) into the line your cursor is currently on. • p (lowercase) will put after the current line • P (uppercase) will put before the current line
u OR :u	• Will **undo** changes you may have made accidently. • u will under changes that are seen as a chunk • :u will undo one change at a time
:red	• Will **redo** one change at a time that you have undone.
%	• Jump between matching sets of parentheses, useful if you think you have made a mistake somewhere.
:[line number]	• Will jump your cursor to this line in the file. • Especially useful when you are getting an error at a particular line in a file.
/[some words]	• Will search your file for a word/ pattern. • Typing “n” will move you to the next match in the file and “N” will move you to the previous match. • **NOTE:** you can also search using regular expressions
:split	• Will split your screen horizontally within the same file, so you can see/edit 2 parts of the same file at once. Can be helpful with longer files. • To exit this view just do :q.
:vsplit	• Will split your screen vertically. • Again exit this view with :q.
Splits	• You can also split with another file by given a file name after the :split or :vsplit. • To switch between the halves of the screen just do [ctrl] + w.
Other Commands	• There are many other commands but those were the basics that everyone generally finds useful. • Google can help you find out what they are :)

.vimrc

What?	• A vimrc is basically a configuration file for vim. It lives in your root directory in a file named .vimrc. • **NOTE:** Anything in your vimrc can also be run as a

	command in vim when prepended by a ":"
Comments	• If you want to put a comment into your vimrc you start the line with ". • Can be helpful when using obscure looking configurations.
set mouse=a	• **THE BEST THING EVER** • This will allow clicking and scrolling in vim which is not natively supported.
set number	• Will display line numbers for the file you are editing which can be very helpful.
syntax on	• Will turn on syntax highlighting in vim, particularly useful if you have a particular color scheme you want to use.
colorscheme [name]	• Will set your color scheme to whatever scheme you have picked out, the internet is full of great ones. • **NOTE:** will have to download the color scheme file and put it in your .vim/colors directory
set hlsearch	• Will highlight matches when you search in a file, can be useful if you are a visual person.
set incsearch	• Will search for things incrementally as you type in your search query. • Super helpful if you also have hlsearch turned on.
set showmatch	• When cursor is on a parenthesis highlight its match, if you don't see a match know you have missed one somewhere.

imap jj <Esc>	• Will map jj to the [ESC] key which will save your wrists!
	• You can also map other commands to keys:
	  ○ imap for mappings in insert mode
	  ○ map for mappings in command mode

A note on mappings	• Try not to go to crazy with them (even though it is tempting). If you have too many hotkeys you will most likely end up confusing yourself and anyone you are pair programming with.
iab [abbr] [expansion]	• Will allow you to set an abbreviation and what is should be expanded to. To use just type the abbreviation in insert mode and then hit [tab]. • ex: iab i import ○ now I can just type i + [tab] and get import
Other Settings	• There are a lot of other cool settings you could use for vim these were just the basics. • Google whatever you find frustrating about vim and there should be something you can do about it in your vimrc.

Sample .vimrc contents

```
set number
set autoindent smartindent
set mouse=a
set tabstop=4
set shiftwidth=4
set expandtab
set ignorecase
set smartcase
set incsearch
set hlsearch
set showmatch
syntax on
imap jj <Esc>
```