Linux Notes 1

These notes are about basic commands, navigation, text editing, and environment variables

Definitely do the things in Blue Text yourself, and anything else you want to try.

Basic commands

- Linux commands are what you type at the command line in your terminal window to accomplish things. They are usually short, cryptic combinations of lower-case letters, followed by optional arguments strung out after a dash (-). You hit Return to get them to execute.
- For example: “ls” will give a listing to your screen of the contents of the current working directory.
- To find out more (often too much!) about a linux command type “man [command]”. ”man” is short for manual.
- Throughout this course, when I use “quotes” you don’t type them on your screen unless I specifically make an exception. Things in [square brackets] are where you substitute some suitable content – without the square brackets.

ls

- ls gives a directory listing
- To get more information from ls try:
- ls –lah
- Here I have added three useful options:
- The “l” gives a longer, more informative listing for each item in the directory.
- The “a” means that it also shows the hidden files in your directory – the ones that start with “.” like .bashrc or .bash_profile in your home directory.
- The “h” means “human readable” and it causes the file sizes to be reported in easier to understand units like “K” for kilobytes.

ls

- here is a typical result on my mac:

```
Parkers-MacBook-Pro:LiveOcean_roms pm7$ ls -lah
total 32
drwxr-xr-x   9 pm7  staff   288B May 13  2019 .
drwx------@ 57 pm7  staff   1.8K Feb 13 11:17 ..
-rw-r--r--@  1 pm7  staff    12K May 13  2019 .DS_Store
drwxr-xr-x  16 pm7  staff   512B Apr 20  2017 LO_ROMS
drwxr-xr-x  13 pm7  staff   416B Aug  6  2017 ROMS
drwxr-xr-x  13 pm7  staff   416B Jul  6  2017 ROMS_820
drwxr-xr-x  12 pm7  staff   384B May 28  2015 ROMS_WOAC
drwxr-xr-x  22 pm7  staff   704B Jan 13 09:16 makefiles
drwxr-xr-x   7 pm7  staff   224B Mar 23 15:58 output
```

- The mysterious code at the left is about permissions. If the first letter is “d” that means it is a directory. The following “rwx” means that the user (me) can Read, Write, and Execute the file. The “rwx” is repeated three times for user/group/anyone. When you write your own shell scripts later you will want to make sure they have an “x” so they can be executed.

Directory and file names, cd

- In linux it is easier, and traditional, to avoid spaces in naming things like files and directories.
- So it would be preferred to call a new directory "my_code" instead of "My Code"
- But you can still work with spaces by using quotes, for example to "cd" (change directory):
- cd "My Code"
- or even in a full path:
- cd /Users/pm7/Documents/"My EndNote Library.Data"
- TIP: you can also use "\ " to mean a space. The backslash (\) is a standard symbol for "the next thing is to be treated in a special way."
- CAUTION: if you cut and paste commands that have quotes, like the ones above, they often fail in linux because they include "smart quotes". You have to replace these by hand.
- Use "pwd" (print working directory) to find out where you are currently.
- Use "cd ~" to go to your "home" directory.
- "." is shorthand for the current directory
- ".." is shorthand for one directory up (the parent directory)

Environment, cp

- Use “cd ~” to go to your “home” directory. This is where login and environment information is kept.
- If your setup is like mine, you will find a file called “.bash_profile”. On your account on the remote linux machine it may be called ”.bashrc”
- You can edit this (!) to add shortcuts to your command line workflow.
- First make a copy in case it all goes bad:
- `cp .bash_profile .bash_profile_ORIG`
- “cp” is short for “copy” and you just follow it by two names, one that exists and the new one. If the new one exists it will just overwrite it, so be careful.

Make an alias, source

- If you have a text editor like Atom or Visual Studio Code, open up your .bash_profile and an add a line (above or below whatever is there – it does not matter):
- alias ipy='ipython --pylab'
- NOTE: if you are using VS Code it will not show hidden files like .bash_profile when you are looking for things in a folder. To get them to show you type "SHIFT+COMMAND+." (all three at once). This is on a mac – maybe different on a Windows machine.
- NOTE: the spaces are important, in particular no spaces around the "="
- You can add comment lines by starting them with "# "
- After you save the edited file, go to your terminal window in ~ and type "ipy"
- It should tell you "command not found"
- Now type "source .bash_profile" which basically re-runs the commands in that file. Now when you type "ipy" it should launch the ipython command line environment.
- You can do this for any common shell command. I use it a lot for getting to directories or remote machines I use commonly, for example I have:
- alias cdpm='cd $HOME/Documents'
- alias fjo='ssh parker@fjord.ocean.washington.edu'

Environment variables, env

- You may have noticed the $HOME in the "cdpm" alias on the last page. That is an environment variable that is the path to your home directory (~ is shorthand for the same place).
- You can see what that path is by typing “echo $HOME” or by doing “cd ~” or “cd $HOME” followed by “pwd”.
- You can see all the defined environment variables by typing “env” (try it)
- You’ll notice that the names in the listing don’t have the “$”, e.g. in mine it has the entry: “USER=pm7”
- You can make your own variables with commands like:
- “junk=77” (note no spaces, and no need for upper case)
- then typing “echo $junk” returns 77
- $junk will be gone the next time you open the command window (test this yourself).

Environment variables cont.

- To summarize: you can define a variable in bash with a command like "name=value" and then you access it using $name.
- I use such things to simplify commands in my .bash_profile:

```
# navigation
LOp=$HOME'/Documents'
alias cdlo='cd $LOp/LiveOcean'
alias cdloo='cd $LOp/LiveOcean_output'
```

cat, mkdir, mv, rm

- "cat [filename]" is good for displaying the contents of a text file (like .bash_profile) to your screen. You can also use "more" or "less".
- "head –n 5 [filename]" shows the first 5 lines of a file
- "tail –n 15 [filename]" shows the last 15 lines of a file
- "mkdir" is what you use to make a new directory
- "mv" is used to move a file from one place to another
- "rm" is used to remove files. Directories have to be empty before you can remove them.
- If you use "rm –rf [directory name]" it will force "f" the removal of even non-empty directories, and do so recursively "r" meaning for all sub-directories. This is a super-handy command for cleaning up, but BE CAREFUL!!! It removes things permanently, instead of putting them in the Trash on your Desktop.

mv, man

- A few tips on using mv...
- To move a file to a directory you would do:
- mv [filename] [path to directory]
- To rename it in the current directory you do:
- mv [filename] [new filename]
- This is a good point to try looking at the output of a man page. Type "man mv"...
- The results can be intimidating and confusing at first. You use the spacebar to page down, and type "q" to escape back to the shell.
- Try a few experiments: make a directory with some files in it. Make another empty directory. Move one of the files to the new directory. Move all of the files to a new directory that does not exist.

Text files and suffixes, >

- One quick way to make a text file in linux is to use "redirection" which is the ">" symbol. For example:
- `env > env.txt`
- takes the output of typing env and instead of sending it to your screen it sends it to (redirects it to) a new text file called env.txt (or any name you choose).
- Note about suffixes: on laptops we are used to things like .docx and so on being very important. In linux suffixes are only for your convenience - so I use .txt above to tell myself that this is an ordinary text file. Other common suffixes I use are .py for python code and .sh for bash shell scripts.

history

- To see a listing of the commands you have issued recently type "history". The last few lines of what I get are:

```
497  say words
498  nc towel.blinkenlights.nl 23
499  man nc
500  man ls
501  history
```

- Then if you type, e.g. "!500" it will rerun that command.

Resources

- https://files.fosswire.com/2007/08/fwunixref.pdf
- https://btiplantbioinfocourse.files.wordpress.com/2014/02/unix_command_sheet_2014.pdf
- More favorites from the class?