Advanced UNIX

UNIX Cheat sheets

- TreeBear
- Linux Cheat Sheet
- FOSS Linux Cheat Sheet
- tips for working fast in UNIX
- MSU unix class

Logging into biocluster

- `ssh -X YOURNAME@biocluster.ucr.edu` will log you in
- You'll be required to enter the password that was emailed to you from the system administrator
- To change your password, once you login type `passwd` to run the change program
- The Biocluster handbook provide info on the system. http://facility.bioinformatics.ucr.edu/
- `ssh` is a secure login program so that your password and all information sent from the laptop to the biocluster computers is encrypted.

Graphical displays

- This is not required for most things we will work on in the class. However, if you want to run graphical emacs or graphical displays you need to get this to work.
- the `-X` setups your display so graphical output on the server can be displayed. This requries the X or X11 tool to be running on your own computer. On Mac this is easy (Click on the Magnifying glass and enter X11).
- Or install the XQuartz program on your mac.
- Windows is more complicated, Try install Xming

Running a few more programs

- removing files: rm
- head, tail
- echo
- wc, grep
- sort, uniq
- awk

Removing files

- Didn't cover how to delete a file last week
- `rm FILENAME` will remove a file. Do this carefully. Default behavior may be to prompt you if you want to remove it so you may have to enter 'y'.
- If you need to remove a whole directory you would type `rm -rf DIRECTORY`. This will NOT prompt you to ask you if you are sure you want to do this!
- BE CAREFUL. UNIX will not warn you and the files, once deleted, are not really retrievable. These are not trashbins like you see on Mac or Windows.

Heads and Tails

- See the first lines (10) of a file: `head filename`
- See the first 20 lines of a file: `head -n 20 filename`
- See last lines (10) of a file: `tail filename`
- See last lines (20) of a file: `tail -n 20 filename`

Echo

- `echo` just prints what you give it in a string
- `echo "hello world"` will print this out
- we use echo sometimes as examples in the class, but it can be useful if you want to write out a message in a log file

Redirecting output

- To redirect output to a file instead of the screen use `>`
- `ls > myfiles`
- you can append to a file with `ls >> myfiles`
- You can pass output from one program to another with `|` pipe
- `ls -1 /shared/gen220/data_files/sequences/ | more` will show you the list of what is in a folder
- There are two output streams 'STDOUT' and 'STDERR'
- The `>` will write STDOUT, but any error messages will still get printed
- `ls >& out` - using the `&` at the end will

Word counting

- `wc` will calculate the number of lines, words, and characters
- `wc -l` will report just the number of lines
- Useful for example if you want to tally a result up and find out how many results there are.

Sorting

- use `sort` to sort a list
- `sort -n` will sort numerically `-r` will reverse the order
- `sort -k1` will sort by column 1 of delimited dataset
- `sort -k2 -t, -n` will sort by column2 where the separator is , and then sort by numberic

Try sorting it

- Here is a file that represents cell cycle data. So a microarray was used to sample gene expression among 20+ timepoints. The rows are genes, and the columns are the timepoints sampled.
- Sort the `/shared/gen220/data_files/expression/Spellman.csv` numerically, by the 60 minute time point.
- What are the top highest expressed genes? What are the least?

Unique lists

- use `uniq` to generate a uniq list from a (possibly) redundant one
- `uniq -c` will print the unique list but show you how many times a string comes up
- the list must be sorted that you pass to uniq - it is only comparing adjacent lines to redundancy
- Here's a collection of 100 random numbers `/shared/gen220/data_files/misc/rand.txt` - let's look at the uniq list of how many times each occurs
- `sort -n /shared/gen220/data_files/misc/rand.txt | uniq -c`

awk

- awk - a very simple scripting language
- Just focusing on one feature, it is useful for extracting a column of data.
- `awk '{print $1}' filename` will print column one, it assumes whitespace delimited
- `awk -F, '{print $1}' filename` will now separate columns by commas
- `awk '{print $3}' /shared/gen220/data_files/misc/heroes_and_villans.txt`

Putting it together

- How many Heroes and how many Villans are there in the `/shared/gen220/data_files/misc/heroes_and_villans.txt` file?
- The following is output of a BLAST/FASTA sequence search report. We will talk more about this format but can you determine how many unique genes were searched in the query file (represented by Column 1). `/shared/gen220/data_files/reports/K12-vs-O157.FASTA.tab`

Permissions

- chmod, chgrp
- To set permission so that everyone can read a file: `chmod a+r filename`
- To recursively set it for a folder: `chmod -R a+r`
- chgrp lets you set the group ownership. By default this will be 'Users' but you are also in the 'gen220' group so if you want to share files, and only with people in this class, you would change the group for the file with `chgrp gen220 FILENAME`.
- Then you would want to set the permissions so that only people in the group can view it. `chmod a-r FILENAME` followed by `chmod g+r FILENAME`

Find

- useful tool to find files by name, size, date and other attributes
- `find . -type d` - find all the directories starting in the current directory
- `find /shared/gen220 -name '*.gbk'` - find all the files that end in .gbk in the shared folder
- `find . -name '
- Some more tips on how to use find are here

Compression

- Files you get may be compressed. This can save time in transferring and save disk space if you compress things you aren't going to use for a while.
- Compressing files with gzip, uncompress with gunzip
- Another compression program is bzip2 (bunzip2 to uncompress). Makes smaller files but can take longer to run
- We can even read from a compressed file in our programs so that we can leave it compressed.
- Try `zmore /shared/gen220/data_files/reports/K12-vs-O157.FASTA.gz`
- Will work an example like this when get to programming.

Running in the background

- UNIX is intended to be a shared system
- You can run more than one program at a time
- Starting up a program with a `&` at the end will put it in the background
- `sort file > file2 &` will run in the background
- If you are running a program and didn't already put it in the background you can suspend it with 'Control-Z'
- Now it is suspended, you can put it in the background by typing `bg`
- a program in the background can also be brought to the foreground by typing `fg`

Your PATH

- the path controls the programs you have available to run on the command line. This is done through a variable called $PATH.
- you can see your path and all environment variables by typing `env`
- You also just pring out the PATH variable with `echo $PATH`
- You can see if a program is available by using the `which` command
- `which grep` shows the path to grep. `which blastall` shows the path to the BLAST application
- On biocluster not all applications are immediately available in your path. We use the modules system to load or unload them
- Try to see where the blastp is `which blastp` -- this shows something that is in fact and old version of the program we want.
- `module load ncbi-blast` then `which blastp` to see it is the other version
- `module list` shows the modules you have installed
- `module unload ncbi-blast` will unload an installed path

scripts with the shell

- All of this has been programming in a way
- The shell is a programming language
- So the steps you write down on the command line could also be stored in a script with each command on a line
- To execute a script you either run the interpreter (bash) when you specify the script: `bash myscript.sh`
- Or you set the permission bit for execute on with `chmod +x FILENAME` and the header of the script look like this:

```
#!/bin/bash
```

- Later when we do Perl the same thing will apply except the top of the script will be `#!/usr/bin/perl`

More on Biocluster

- Several hosts we can log into. You will start on biocluster
- When we run analyses will need to run these on `owl`
- `ssh YOURNAME@owl.ucr.edu`
- for even long running jobs we will need to submit these to a job queue
- This requires writing your steps as a bash script like discussed before
- Then using the command `qsub`. Lots more info on this on the biocluster FAQ on the Facilities page.