Unix 101 Part 1 - files and directories

Concept: Terminal lets you access the Unix system that lies beneath the fancy Mac GUI.

Before we begin

1. Download the file https://research.nhgri.nih.gov/Training/unix/downloads/UnixClass_120516.tar.gz
2. Double-click on the file to uncompress it. This will create a directory (folder) called *UnixClass*
3. Drag the *UnixClass* directory to the Desktop folder on your Mac.

List directory contents and change working directory	
Run Terminal.app	Find using Spotlight (/Applications/Utilities)
`ls`	what's here?
`pwd`	where am I?
`cd Desktop`	change directory
`ls`	
`cd UnixClass`	tab to complete names
`ls`	
`pwd`	
`cd ..`	change to parent directory
`pwd`	
`ls /usr/bin`	many of the utilities we're using live here
`pwd`	we haven't moved
`ls UnixClass`	list files in directory
`ls -l UnixClass`	up arrow to modify last command
`cd !$`	useful shortcut: last word of previous cmd
`ls -ltr`	list files in order from oldest to newest; note we can often combine single-letter options
`man ls`	manual (help) pages for most Unix commands

Note that most punctuation characters have a specific meaning and purpose in the Unix shell. For this reason, it is generally a good idea to avoid using punctuation, including spaces, in filenames. The "safe" characters are letters, numbers, and _ (underscore), - (dash) and . (period). Anything else will likely create a conflict with one program or another.

Also note that Unix doesn't care about the last few characters of your file name. Whether you name a file "mydata.txt" or "mydata.csv" has no effect on how the programs work. You must tell programs to do the right thing with them, by specifying the column separator, etc. In general, Unix programs are set up to work well with tab-delimited data, as we will use in the examples in this class.

Unix 101 Part 2 - viewing, counting and searching text files

Concepts: Redirecting output, piping the output of one command to the input of another.
Tasks: See what's inside files, even very large ones, and start to be able to summarize and search through text files.

View file contents	
`cat hgmd_brca1.txt`	dump file contents to screen
`less hgmd_brca1.txt`	show one screen at a time
`<down> or <enter>`	line forward
`<up>`	line back
`f or <space>`	page forward
`b`	page back
`g`	go to start
`G`	go to end
`q`	quit
`less -S hgmd_brca1.txt`	show without wrapping text
`<right>`	page right
`<left>`	page left
`q`	Note content: some Ovarian cancer, some Breast cancer, some both
Interrupt a running program	
`cat`	no file; sits and waits
`ctrl-C`	to interrupt (cancel) running command
View first or last lines of file	
`head hgmd_brca1.txt`	first 10 lines
`head -3 hgmd_brca1.txt`	first 3 lines
`tail hgmd_brca1.txt`	last 10 lines
`tail -1 hgmd_brca1.txt`	last line
Count characters, words and lines	
`wc hgmd_brca1.txt`	line, word, character count
`wc -l hgmd_brca1.txt`	just count lines
Search for patterns, redirect output to a file	
`grep Ovar hgmd_brca1.txt`	gets some of them (how many?)
`grep Ovar hgmd_brca1.txt > ovarian.txt`	redirect output into file
`wc -l ovarian.txt`	seems too small.
`less -S ovarian.txt`	What's wrong?
`grep ovar hgmd_brca1.txt > ovarian.txt`	gets the others (grep is case-sensitive)
`wc -l ovarian.txt`	note that we overwrote file, no warning!
`grep -i ovar hgmd_brca1.txt > ovarian.txt`	gets all (-i option makes case-insensitive)
`wc -l ovarian.txt`	

`grep -v -i ovar hgmd_brca1.txt > no_ovarian.txt`	gets inverse: lines not containing ovar
`wc -l no_ovarian.txt`	we're creating a lot of files…
Pipe output from one program to input of another	
`grep -i ovar hgmd_brca1.txt | wc -l`	pipeline sends output of first command as input to second
`grep "Hum Mut" hgmd_brca1.txt | wc`	quotes needed when search with spaces or other punctuation used by the Unix shell
`grep X hgmd_brca1.txt | less -S`	pipe to less lets us see what we're getting
`grep X hgmd_brca1.txt | grep -i ovar | wc -l`	pipe grep to grep to create "and" queries; can have many pipes

Hands-On Exercises #1

Note: Answer exercises 1.1 and 1.2 using the file `hgmd_brca1.txt`.

1.1. Use head and tail to extract lines 2-4 from hgmd_brca1.txt
1.2. How many BRCA1 variants were published in 2003? How many of these involve both ovarian and breast cancer?
1.3. Use man to learn what the -w option does in the grep command. Try it.

Unix 101 Part 3 - Wildcards in file names and in grep

Concepts: Wildcards let you apply the same command to multiple files at the same time. Wildcards in grep are different and allow you to find matches to patterns, not just specific strings.

Wildcards for file names	
`ls *.txt`	show all files that end with '.txt'
`ls r*`	show all files that begin with 'r'
`wc -l *.txt`	count lines in all files that end with '.txt'
Wildcards in grep patterns	
`grep W...X hgmd_brca1.txt | less -S`	in grep, '.' is wildcard for single character
`grep "W.*X" hgmd_brca1.txt | less -S`	use '*' to indicate "zero or more". Need quotes to prevent * from being interpreted by Unix shell.
`grep "W[0-9]*X" hgmd_brca1.txt | less -S`	square brackets define a "character class", a set of characters that will match a given position. This is more precise than previous command.

Unix 101 Part 4 - sorting and more counting

Concepts: One can sort alphabetically or numerically. Sorting not only puts things in the desired order; it can also identify unique combinations of values.

Sort alphabetically	
`less wgEncodeRegTfbs.txt`	new BED file from UCSC; transcription factor binding sites
`sort wgEncodeRegTfbs.txt | less -S`	
`wc -l wgEncodeRegTfbs.txt`	it's a big file
`head -1000 wgEncodeRegTfbs.txt > tfbs.txt`	let's take a smaller sample for today
`sort tfbs.txt | less`	now it doesn't take too long to sort
`sort tfbs.txt | head`	use 'head' instead of 'less' to keep a view of recent output in the terminal window
`sort -k5,5 tfbs.txt | head`	can sort on the score column; but this is still an alphabetical sort
Sort numerically	
`sort -n -k5,5 tfbs.txt | head`	-n sorts numerically
`sort -r -n -k5,5 tfbs.txt | head`	and -r reverses the order of the sorting
`sort -k5,5nr tfbs.txt | head`	another way to say that
Sort on multiple columns	
`sort -k1,1 -k2,2n tfbs.txt | head`	can sort on multiple columns; -n and -r can be applied to individual columns
Count unique (distinct) values	
	how many different TFs are in there? how many sites for each?
`cut -f4 tfbs.txt | less`	extract single column
`cut -f4 tfbs.txt | sort -u | less`	sort -u gives only unique (distinct) values
`cut -f4 tfbs.txt | sort -u | wc -l`	count them
`cut -f4 tfbs.txt | sort | uniq | wc -l`	this is just like sort -u
`cut -f4 tfbs.txt | uniq | wc -l`	**don't do this**: sort is needed before uniq
Count number of repetitions of each value	
`cut -f4 tfbs.txt | sort | uniq -c | less`	uniq -c counts the number of sequential repetitions
`cut -f4 tfbs.txt | sort | uniq -c | sort -nr | head`	show which TFs have the most binding sites
Count unique combinations of values, make subtotals	
	which TFs have sites on the fewest different chromosomes?
`cut -f4,1 tfbs.txt | less`	can extract multiple columns, but can't specify order
`cut -f1,4 tfbs.txt | head`	keep the first 10 lines of output for reference
`cut -f1,4 tfbs.txt | sort -u | head`	First, get distinct chrom/TF pairs…

`cut -f1,4 tfbs.txt | sort -u | cut -f2 | head`	...extract column that we want to "subtotal" on
`cut -f1,4 tfbs.txt | sort -u | cut -f2 | sort | uniq -c | head`	...then re-sort to lump TFs together and count number of chroms for each TF
`cut -f1,4 tfbs.txt | sort -u | cut -f2 | sort | uniq -c | sort -n | head`	sort again to find TFs with fewest chroms

Unix 101 Part 5 - Working with compressed (.gz) files

Concepts: compression is very useful for large files, and many files that you download from the internet are compressed, to save download time as well as space. You can uncompress the files, or use pipes to work with them while they are compressed.

`ls -l refFlat.txt`	check out file sizes
`gzip refFlat.txt`	compress file; replaces original file
`ls -l refFlat.txt.gz`	this file is compressed
`less refFlat.txt.gz`	can't work with compressed data directly
`gunzip refFlat.txt.gz`	uncompress file
`ls -l`	compressed version is gone
`gzip refFlat.txt`	compress file again
`gunzip -c refFlat.txt.gz | less`	can uncompress to pipe, saving disk space
`zless refFlat.txt.gz`	shorthand for the same thing

Unix 101 Part 6 – Working with text files

We caution against using Word or Excel to edit text files that you later want to manipulate on the command line using the Unix commands that you've learned in this class. While both programs provide options to save files in various text formats, the resulting .txt or .csv files are not always compatible with Unix. A better option is to use a text editor like BBedit (Mac) to edit any files that will be also accessed under Unix. A list of recommended free text editors is provided at the end of this document.

Hands-On Exercises #2

Note: Answer the following exercises using the file `tfbs.txt`.

2.1. Are all TF binding sites on the "+" (forward) strand?
2.2. What are the highest scoring binding sites for c-Myc? the lowest scoring?
2.3. How many binding sites for NFKB have the highest score? the second highest?
2.4. Going back to hgmd_brca1.txt, count how many variants for each type of cancer.

Unix 101 Extra Topics

These are some basic concepts that didn't fit in the three-hour session. But you can explore on your own, now that you are more comfortable working at the command line.

Biowulf HPC Cluster at NIH

Biowulf support
https://hpc.nih.gov/
Everything you need to know about the NIH Biowulf HPC Cluster, including how to get an account, user guides, training, list of installed applications, etc.

Online class: Introduction to Biowulf
https://hpc.nih.gov/training/intro_biowulf/
The 'Introduction to Biowulf' class is an online, self-paced class. New Biowulf users are encouraged to work through the entire class, and experienced Biowulf users can view specific videos to brush up on a particular section.

Copy, move, rename and delete files and directories

Tasks: list, copy, move and rename files and directories, just like you can do with the Finder.
So why bother? These work on any Unix machine, including remote servers and clusters.

`cp hgmd_brca1.txt my.txt`	copy file
`ls -l`	
`mv my.txt brca1.txt`	rename (move) file
`rm brca1.txt`	delete (remove) file **Note: there is no undo, and no "recycle bin"**
`mkdir brca1`	create directory
`ls -l`	notice 'd' for directory, permission flags
`cp hgmd_brca1.txt brca1`	copy file into different dir
`ls -l brca1`	notice: same file name, new timestamp
`rm brca1`	not right command for directory
`rmdir brca1`	can't do this; dir not empty
`rm -r brca1`	remove recursively (all contents) **Warning: this can be quite dangerous; you can permanently delete a large number of files. USE WITH CAUTION.**
`mkdir brca1`	
`cp hgmd_brca1.txt brca1/hgmd`	copy using different name
`mv brca1/hgmd .`	move file here (current directory)
`rmdir brca1`	remove empty dir
`rm hgmd`	
`rm "refFlat 2.txt"`	must use quotes if filename includes spaces or other punctuation

Accessing Remote Computers

Concepts: Unix commands work exactly the same on Linux servers and HPC clusters (like biowulf) as they do on your Mac. To access these computers over the network, we use ssh (secure shell) and scp (secure copy).

Log in to a remote computer	
`ssh studXX@biowulf.nih.gov`	"studXX" is your username on the remote computer. If username is the same on local and remote, can leave out "username@"
`Password:`	Enter password (case sensitive)
`pwd`	All commands that we know work here, e.g. Where are we?
`ls –l`	What's here?
`exit`	Log out
Copy files to remote server	
`scp UnixClass.tar.gz studXX@biowulf.nih.gov:`	Note colon after hostname.
`Password:`	Asks for password on remote (won't mention this again, happens every time we connect)
`ssh studXX@biowulf.nih.gov`	
`ls`	The file is here, in your home directory, because we didn't provide a destination directory name
`tar –xzf UnixClass.tar.gz`	Untar gzipped archive
`ls –l`	created a new directory
`ls –l UnixClass`	Familiar files
`exit`	
Copy files from remote server	
`scp studXX@biowulf.nih.gov:UnixClass/hgmd_brca1.txt .`	Note dot to indicate that the destination is the current directory; always have to provide a destination for scp

Text editors

If you need to edit text files on your desktop machine, we recommend the following:

BBedit (Mac)
https://www.barebones.com/products/bbedit/
Notepad (Windows)
Installed on NHGRI Windows machines
Notepad++ (Windows)
http://notepad-plus-plus.org/
Visual Studio Code (Mac and Windows)
https://code.visualstudio.com/