Linux/Bash Quick Reference

HPC Shell software carpentry lesson chapters 3-6
http://www.hpc-carpentry.org/hpc-shell/

Discovering Your Environment

pwd	print working directory
hostname	get the name of the current host computer

Listing Files and Directories

ls	list short - list files and directories
ls -l -a -h	flags for longer list, hidden files, and human readable
ls —help or man ls	to get more help on any command
alias ll='ls -l'	assign **ls -l** to **ll**. Must be quoted. Linux is very customizable.

commands are programs, case sensitive, you can combine flags like **ls -lah**

Creating and Removing Directories

mkdir test_dir	make directory - spaces separate arguments, use underscores
mkdir -p test_dir	**-p** protects against errors if directory already exists
rmdir test_dir	remove a directory, must be empty

Navigating Through the Directory Tree

cd test_dir	change directory
cd ~/test_dir	absolute path from home directory
cd /homes/daveturner	absolute path starting at root **/**
cd ../dir	relative path with **..** meaning go back one directory
cd	go back to your home directory, same as **cd ~**

Creating and Manipulating Files

touch file.txt	create an empty file. Suffixes can be meaningful.
mv file.txt test_dir	move a file to a different location
mv file.txt file2.txt	rename the file instead
cp file2.txt file.txt	copy the file and its contents
cp -rp dir dir2	copy the entire directory and maintain modification dates
rm file2.txt	remove a file
rm -fr dir2	recursively remove a directory and contents

Editting Files

nano file.txt	edit a file using **nano** - control characters at the bottom
vi file.txt	edit a file using **vi** or **vim** - **:wq** to save and quit
	https://www.tutorialspoint.com/unix/unix-vi-editor.htm
MobaXTerm	edit files on **Windows** system and sync with Linux system

Downloading and Uncompressing Files

wget **http://www.hpc-carpentry.org/hpc-shell/files/bash-lesson.tar.gz**
curl -O **http://www.hpc-carpentry.org/hpc-shell/files/bash-lesson.tar.gz**

tar -xvzf bash-lesson.tar.gz	decompress and expand the archive
gzip/gunzip	compress/expand data with **.gz** suffix
compress/uncompress	compress/expand data with **.Z** suffix
zip/unzip	compress/expand data with **.zip** suffix
scp file.txt username@hpc.edu	use **scp** to transfer files from local system to Linux system

Displaying File Contents

cat/more/less file.txt	display the contents of a file to the terminal
head/tail -1 file.txt	display the first/last line of the file
wc -l file.txt	count the number of lines (**-l**), words (**-w**) or characters (**-c**)

Wildcards

(test wildcards with **ls -l** first to make sure you are choosing the desired files)

wildcard *	matches any number of type of characters (anything)
wildcard ?	matches any single character
wildcard [a-e,w,z]	matches any single character to a list or range
ls -l *_1.fastq	will list all files ending with _**1.fastq**
wc -l S*[89]_?.fastq	count lines in files with 8 or 9 before the underscore
grep Act5 dmel-all*	display only lines with Act5 in them

Redirecting Output

echo “Hello there”	echo a string to the terminal
echo “Hello there” > file.txt	**>** will redirect the stdout output to a file
host=`hostname`	run **hostname** and store in variable $host
echo “from $host” >> file.txt	**>>** to append, **2>** for stderr, **&>** for both
grep Act5 dmel-all* | wc -l	pipes link output of one command to input of the next

https://www.gnu.org/software/bash/manual/html_node/Bash-Features.html
https://www.geeksforgeeks.org/bash-scripting-for-loop/

Writing Bash Scripts

#!/bin/bash	in the first line it defines what shell the script will run in follow by any other linux commands applications to run
chmod u+x script.sh	add user execute permission to the bash script if needed
ls -l	shows read, write, execute permissions for user, group, others
./script.sh	run a script from the current directory
file=genome.fastq	set the variable ***file*** to the ***genome.fastq*** name
wc -l $file	use the variable $file
echo “$file”	double quoted variables are evaluated, not within single quotes

Bash Loops

```
files=$(ls *.fastq)
for file in $files
do
        wc -l $file
done
```

get a list of .fastq files
set variable file successively to 3 different names
word count each file

Bash Conditionals

```
if [[ $num -eq 1 ]]
then
        echo “it was 1”
elif [[ $num -ne 0 ]]
then
        echo $num
else
        echo “It was 0”
fi
```

numbers -eq -ne -lt -le -gt -ge
string comparisons -z (empty) -n (not empty)
== (equal) != (not equal)
files -e (exists) -d (directory)

Example bash script

```
#!/bin/bash -l
# shebang line identifying the shell type 'bash'
# Other lines starting with '#' are comments

# Examples of setting variables (they don't need to be all caps)
HOST=$(hostname)     # execute the 'hostname' command and put results in HOST
NCORES=$(getconf _NPROCESSORS_ONLN)

# Try using the variable.
# Use double quotes here as variables are not interpretted inside single quotes
echo "The host name is $HOST and has $NCORES cores"

# Create a simple loop to echo the files in the current directory
for FILE in $( ls )
do
   echo "File $FILE is in this directory"
done

# Use a C-like loop to set the index I from 1 to 5
for (( i=1; i<=5; i++ ))
do
   echo "Index I is $i"
done

# Check for a specific file and echo an error if it isn't there
if [ -f /homes/daveturner/myfile.txt ]
then
   echo "File myfile.txt exists in /homes/daveturner"
else
   echo "myfile.txt was not found in /homes/daveturner"
fi
```