Terminal Commands Summary:

Google term + “linux man” for more documentation

Ctrl+c to exit a bad bash statement (i.e., when you have a “>” for a prompt)

pwd: prints the full name of the current working directory

mkdir: create directory(ies) if they do not already exist. Enter spaces between directory names to create multiple

```
mkdir [OPTION]... DIRECTORY...
```

touch: change file timestamps. Useful for creating a file as well.

```
touch [OPTION]... FILE...
```

ls: list directory contents

```
ls[OPTION]... FILE…
```

cp: copy files and directories

```
cp file-from-directory file-to-directory
```

- To copy the file `MyBestPicture.jpg` from current folder into the /vacation directory

```
cp MyBestPicture.jpg  ./vacation
```

- Recursively copy (-r) all jpg files into jpg folder. Do not overwrite existing files (-n)

```
cp -nr **/*.jpg jpg
```

mv: move or rename files

```
mv file-from-directory file-to-directory
```

-To move the file `MyList1.txt` from current directory into the /vacation` directory:

```
mv MyBestPicture.jpg  ./vacation
```

rm: removes files or directories

```
rm PictureCounts.md (to remove a file called PictureCounts)
```

tee: Allows for redirecting standard input to standard output

- The output from the `ls` command is redirected to the file parts.txt

```
ls | tee parts.txt
```

head: output the first part of files. Default is first 10 lines, can use options to output more (below outputs 50 lines).

```
head [OPTION]... [FILE]...
head -50 frank_1.txt
```

tail: output the last part of files. Default is the last 10 lines, can use options to output more (below outputs 50 lines).

```
tail [OPTION]... [FILE]...
tail -50 frank_1.txt
```

less: to preview a file.

```
less frank_1.txt
```

cat: concatenate files and print on standard output. Can be used with one file to print file to output

cat [*OPTION*]... [*FILE*]...

```
cat 1_call_to_worship.txt 3_weird_machines.txt
```

- Concatenate 3 files into all2.txt

```
cat carrie-b.todos.1 carrie-b.todos.2 carrie-b.todos.3 > all2.txt
```

- Concatenate all .todo files into all3.txt

```
cat ./*.todos* > all3.txt
```

find: search for files in a directory hierarchy. Find by itself will find all files and directories in the current tree.

- Find all start files (omit start to find all files; -iname for case insensitive):

```
find . -type f -name start.txt
```

- Find all end folders (omit end to find all directories; -iname for case insensitive):

```
find . -type d -name End
```

- Find all files with a given name and wildcard

```
find . -type f -name flag*
```

- Find files that meet either of two conditions

```
find . -type f -name flag1 -o -name flag2
```

- Find files that meet both of two conditions

```
find . -type f -name *book* -a -name *dangerous*
```

- Type Searches

```
find . -type f -name *.txt
find . -type f -name *.pdf
```

- Time Searches

```
#Find all files and folders created at least 2 minutes ago
find . -cmin +2

# Find all files and folders created between 2 and 6 minutes ago
find . -cmin +2 -cmin -6
```

- Size Searches

```
# Find all files and folders at least 5 kilobytes in size
find . -size +5k

# Find all files and folders less than 50 kilobytes in size
find . -size -50k
```

exec: Execute a command

- Find and immediately copy all files that begin with flag to My_Flags directory

```
find . -type f -iname flag* -exec cp {} My_Flags \;
```

- Find and copy all the files that are not pngs into the NoPNGsAllowed folder.

```
find . -type f ! -iname *.png -exec cp {} NoPNGsAllowed \;
```

- Recursively (-r) copy .csv files into AllRecords folder without overwriting existing files (-n)

```
find . -type f -iname *.csv* -exec cp -nr {} ./AllRecords \;
```

grep: searches named input files for lines containing a given pattern

```
grep [OPTIONS] PATTERN [FILE...]
grep [OPTIONS] [-e PATTERN | -f FILE] [FILE...]
```

- Find the days for which power2all was active (searching log files)

```
grep -rli "power2all" .
```

- Find the times for which glanzmann logged on and off

```
grep -i "glanzmann has" *
```

- Copy every log for which power2all appears into its respective folder.

```
find . -type f -exec grep -rli "power2all"  {} \; -exec cp {} power2all \;
```

- Output lines of files that include both Michael and Davis to michael_davis_orders_output.output. Include file name and line numbers (-n). Do not include directories in search (just files)

```
grep -n --exclude-dir=* Michael.*Davis * > michael_davis_orders.output
```

- Find “done,” ignoring case (-i), in Carrie/all.txt and output lines to done.txt

```
grep -i done ./Carrie/all.txt > ./Carrie/done.txt
```

- Find items not “done,” (-iv) ignoring case, in Carrie/all.txt and output lines to unfinished.txt

```
grep -iv done ./Carrie/all.txt > ./Carrie/unfinished.txt
```

wc: Print newline, word, and byte counts for each FILE, and a total line if more than one FILE is specified.

```
wc [OPTION]... [FILE]...
wc [OPTION]... --files0-from=F
```

- | conveys that we are piping the results from our find and grep commands into the next command
- wc -l conveys that we are looking to count the number of lines retrieved
- |wc -l in conjunction with find and grep retrieves the record count

- From within a log folder, count the total number of log files

```
find . -type f | wc -l
```

- Count the number of instances in which the user "glanzmann" appeared

```
grep -i "glanzmann has joined" * | wc -l
```

- Count the number of instances in which the user "glanzmann" spoke

```
grep -i "<glanzmann>" * | wc -l
```

- Count the total number of days for which "E1ven" logged on

```
grep -rli "E1ven" . | wc -l
```

- Count the number of files of type jpeg and append results to PictureCounts.md

```
ls jpg | wc -l >> PictureCounts.md
find . -type f -iname "*.jpg" | wc -l >> PictureCounts.md
```

- Count line items in ./John/done.txt and output to ProductivityReport.txt

```
wc -l < ./John/done.txt >> ProductivityReport.txt
```

tar: archiving utility

- Create a tar archive file tecmint-14-09-12.tar for a directory /home/tecmint in the current working directory

```
tar -cvf tecmint-14-09-12.tar /home/tecmint
```

- Use -cvzf to create a compressed gzip archive file (can also just use .tgz instead of tar.tgz

```
tar -cvzf tecmint-14-09-12.tar.tgz /home/tecmint
```

- Create a Tar Archive of IRC_Logs

```
tar -cvf IRCLogs.tar IRC_Logs/
```

- Unarchive a Zip Folder

```
tar -xvf Gibberish_Folder.tar
```

echo: displays a line of text

- Create a new file and fill it with a line of text

```
echo "Hey there! This is my sentence" > MyFile.txt
```

- Clear Pride.txt and insert replacement text

```
echo "Nope. Not a fan of Pride and Prejudice" > Pride.txt
```

- Add a line to the bottom of Alice.txt

```
echo "AhmedWuzHere." >> Alice.txt
```

sed: stream editor for filtering and transforming text

sed [*OPTION*]... *{script-only-if-no-other-script}* [*input-file*]...

```
sed 's/clank/clink/g' othello.txt > othello_new.txt
```

- sed is the first part that runs the program.
- s is 'substitute' option for sed, telling it that specific substitutions are coming next.
- /clank/is the pattern that sed will search for.
- /clink/is the string that will be substituted whenever the first pattern is matched.
- g means to apply this substitution 'globally'

```
sed 's/[a-z]/Z/g' sed.txt - changes lower case letters to Z
sed 's/[A-Z]/z/g' sed.txt - changes upper case letters to z
sed 's/[0-9]/#/g' sed.txt - changes numbers to #
```

awk: programing language designed specifically for processing text which allows quick and useful tasks in the command line

- Print only the first field of the 17-18-Breaches.txt. Items are tab-delimited.

```
awk -F"\t" '{print $1}' 17-18-Breaches.txt
```

- Print only the breaches from 'web' companies.

```
awk '/web/' 17-18-Breaches.txt
```

- Out of the web companies that were breached, print only the company name (tab-delimited)

```
awk -F"\t" '/web/{print $1}' 17-18-Breaches.txt
```

- Print all the breaches from 2017

```
awk '/2017/' 17-18-Breaches.txt
```

- For the companies that had breaches in 2017, print only the company name and the number of records lost (tab-delimited)

```
awk -F"\t" '/2017/{print $1, $3}' 17-18-Breaches.txt
```

- For the companies that had breaches in 2018, save the company name, Company type and number of breaches to a new file named 2018Breaches.txt

```
awk -F"\t" '/2018/{print $1, $4, $3}' 17-18-Breaches.txt > 2018Breaches.txt
```

- Calculate values from column 5 of michael_davis_orders.output and append sum to VIPCustomerDetails.md. Use comma as a delimiter.

```
awk -F',' '{sum+=$5;} END{print sum;}' michael_davis_orders.output  >> VIPCustomerDetails.md
```

- Does Directory exist?

```
if [ -d directoryName]
then...
else….
fi
```

- Does File exist?

```
if [ -f fileName]
then…
fi
```