awk

- Description: a scripting language used for processing and displaying text. Formula: `awk+ options + awkcommand + file+ file to save`
- Examples: Print the first column of every line of a file `awk '{print$1}' ~/Documents/Csv/chips.csv` Print first field of /etc/passwd file `awk -F '{print $1} /etc/passwd` Print the first and 3 field with line numbers `awk -F: '{print NR,$1,$3} /etc/passwd`

cat

- Description: used for displaying the content of a file.
- Formula: `cat + option + files to display`
- Examples: Displays the content of the cookies file `cat cookies.txt` Displays the content with line numbers `cat -n /Documents/fold.sh` Displays the content of a file while suppressing repeating empty lines to a single empty line `cat -s hello.py`

cp

- Description: copies files/directories from a source to a destination.
- Formula: `cp + files to copy + destination` *Examples: Copy a file `cp Downloads/hi.txt Documents/` Copy the contents of a directory to another `cp Documents/Pictures/* ~/Downloads` Copy multiple files `cp hello.sh book.txt store.py`

cut

- Description: used to extract a specfic section of each line of a file and display it.
- Formula: `cut + option + files` Examples: Displays all of the users in your system `cut -d ':' -f1 /etc/passwd` Cut a file excluding a given field `cut -d ',' --complement -s -f3 passwds.txt` Cut a file with a delimiter then change it in the output `cut -d ';' -f1,8 --outputdelimiter=, =>, /Documents/cookies.txt`

grep

- Description: used to search text in a given file.
- Formula: `grep + option + search criteria + files`
- Examples: Search any line that contains the word chips in the given file `grep 'chips' ~/Downloads/store.txt` Search for all lines that do not contain the word hello `grep -'hello' ~/Downloads/greeting.txt` Search and match only the word `grep -o 'flat' Downloads/flat.txt`

head

- Description: displays the top number of lines given in a file.
- Formula: `head + option + file`
- Examples: Display the first 10 lines of a file `head file.txt` Display the first line of a file `head -1 file.txt` Display the first 5 lines `head -5 file.txt`

ls

- Description: used for displaying the files in a directory
- Formula `ls + option + directory to list`
- Examples: List the content of your current working directory `ls` long list all the files including hidden files `ls -la Documents/cis106` List the files in a given directory `ls lab2`

man

- Description: pages that describe linux shell commands,executable programs,system calls,special files and so forth.
- Formula: `man + command you want to look`
- Examples: shows manual for the ls command `man ls` shows manual page 2 for mkdir command `man 2 mkdir` shows manual page that has the word update `man -k update`

mkdir

- Description: used to create directories.
- Formula: `mkdir + the name of the directory`
- Examples: Create a directory in the present working directory `mkdir chips` Create multiple directories `mkdir chips/brand chips/color chips/price` Create a directory using absolute path `mkdir ~/water/lake`

mv

- Description: moves and renames directories and files.
- Formula: `mv + source + destination` to move files/directories `mv + file/directory to rename + new name` to rename files/directories.
- Examples: Moving a file from a directory to another `mv Documents/labs Downloads/` Renaming a file `mv paper.docx project.docx` Moving and renaming a file `mv Documents/homework.docx Downloads/test.docx`

tac

- Description: used for displaying content of a file in reverse order.
- Formula: `tac+ option + files to display`
- Examples: Displays the content of a file in reverse order `tac chips.txt`

tail

- Description: displays the last number of lines in a given file.
- Formula: `tail + option + file`
- Examples: Displays the last line of a file `tail -1 book.txt` Displays the last 10 lines of a file `tail book.txt`

touch

- Description: used to create files
- Formula: `touch + file to make`

- Examples: Creates a file `touch file` Creates multiple files `touch file1,file2,file3` Create a file in a directory `touch ~/wallpapers/paper.png`

tr

- Description: used for translating or deleting characters from standard output.
- Formula: `Standard output | tr + option + set + set`
- Examples: Translate periods to colons `cat file.txt |tr '.'':'` Translates white space into tabs `cat chips.txt |tr "[:space:]" ''`

tree

- Description:used to print a recursive directory listing the directory,subdirectories and files inside of the directory
- Formula: `tree + directory`
- Examples: Shows a tree for the chips directory `tree chips` Shows all files in directory including hidden `tree -a chips` Shows only directories `tree -d chips`

How to work with multiple terminals open?

to work with multiple terminals you can press ctrl + n to open a new terminal and use the `windowskey + left or right arrow keys` to set up your terminals side by side.

How to work with manual pages?

to work with man pages you must type man and the command you want to look at the options for then you can use the arrow keys to navigate or the e key to go forward one line and the y key to go backward one line.

How to parse (search) for specific words in the manual page?

in order to search for a word in a man page you must type / then the word you want to type a example would be /linux this would search for the word linux.

How to redirect output (> and |)

to use > to redirect output you must have it at the end of a command a example would be `ls -lh >output` this will take the results of the ls command and put them into a file called output. The | or pipe is used to seperate commands to be able to use multiple in one line a example would be `cat book.txt| grep 'ring'` this will search for ring in the contents of book.txt.

How to append the output of a command to a file

to append the output of a command to a file it has to be put at the end so like this `cat hello.sh |grep 'fruit'>output.txt`.

How to use wildcards

The * wildcard matches zero or more characters in a file name and can be used before or after if you type this wildcard like this ls `*.txt` it will only display text files however if you do `rings*` it will print all files that begin with rings. The ? wildcard matches one character so if you type `*.???` it will display all files that have a 3 letter file extension. The [] wildcard matches a single character in a range so if you do `ls d[a-z]*` it will match all files that have a range of letters after d. Digits also apply so it can be used like so `ls *[0-9]*.` this will look for files with any digit in its name at any point before the . or file extension.

For copying and moving multiple files at the same time

In order to copy multiple files you must use cp and just type the files out with there directories so `cp Documents/chips.txt Downloads/rings.py Pictures/water.png` and to move multiple files as well you must use mv and type it in the same way `mv Documents/hello.sh Documents/ice.png Pictures/`.

How to use brace expansion

Brace expansion is used with {} and can be put alongside commands in order to create multiple files or to create subdirectories a example using files would be `touch files{1..5}`.txt this creates 5 files all named files with there number next to it so like files1. When using it creating directories this is how you would do it `mkdir -p store/{brand,color}/{lays,red}/labels{1..3}` this creates a parent directory store then creates 2 sub directories lays,red and then gives lays and red 3 label files.