ls /

learner@host01:~$ ls /

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

ls /etc/perl

learner@host01:~$ ls /etc/perl

CPAN Net

//to know the present working directory

pwd

learner@host01:~$ pwd

/home/learner

//Making New Directory

mkdir learning\_unix

ls

learner@host01:~$ mkdir learning\_unix

learner@host01:~$ ls

a\_directory another\_directory learning\_unix script.sh

//Change Directory

cd learning\_unix

learner@host01:~$ cd learning\_unix

learner@host01:~/learning\_unix$

//Creating Two directories

mkdir outer

cd outer

learner@host01:~/learning\_unix$ mkdir outer

learner@host01:~/learning\_unix$ cd outer

learner@host01:~/learning\_unix/outer$

mkdir inner

cd inner

learner@host01:~/learning\_unix/outer$ mkdir inner

learner@host01:~/learning\_unix/outer$ cd inner

learner@host01:~/learning\_unix/outer/inner$

// Creating Two directories in single step

mkdir -p outer/inner

//TRAVERSING THROUGH THE DIRECTORIES

cd /

cd home

cd learner/

pwd

learner@host01:~/learning\_unix/outer/inner$ cd /

learner@host01:/$ cd home

learner@host01:/home$ cd learner/

learner@host01:~$

learner@host01:~$ pwd

/home/learner

//TRAVERSING THROUGH THE DIRECTORIES—Using single step

cd /home/learner/

//Diff between

cd /home/learner/ AND cd home/learner/

The first command says go the learner directory that is beneath the home directory that is at the top level (the root) of the file system. There can only be one /home/learner directory on any Unix system.

The second command says go to the learner directory that is beneath the home directory that is located wherever I am right now. There can potentially be many home/learner directories on a Unix system (though this is unlikely).

//Navigating Upwards and Downwards

cd ~/learning\_unix/

cd ..

cd ..

learner@host01:~$ cd ~/learning\_unix/

learner@host01:~/learning\_unix$ cd ..

learner@host01:~$ cd ..

learner@host01:/home$

What if you wanted to navigate up *two* levels in the file system in one go? It's very simple, just use two sets of the .. operator, separated by a forward slash:

cd ../..

#### //Absolute and relative paths

learner@host01:~/learning\_unix$ cd ../../../tmp

or...

learner@host01:~/learning\_unix$ cd /tmp

#### //Finding your way back home

cd / pwd

cd ~ pwd

Hopefully, you should find that cd and cd ~ do the same thing, i.e. they take you back to your home directory (from wherever you were). You will frequently want to jump straight back to your home directory, and typing cd is a very quick way to get there

cd /

cd ~/learning\_unix pwd

learner@host01:~$ cd /

learner@host01:/$ cd ~/learning\_unix

learner@host01:~/learning\_unix$ pwd

/home/learner/learning\_unix

//USING ls Command

ls -l –long listing

ls -R -complete route in every directory

ls -l -t -r --

ls -lh

//MAN option

Manual to know what a command does

//REMOVE DIRECTORY

**You have to be outside a directory before you can remove it with *rmdir***

//use of TAB bar

* Used to save time by just clicking the unique letters and the remaining work will be done by Tab

//History command gives all the commands that are being typed in a session

//TOUCH command –creates empty files

//MOVE command---create a directory and move the files top the same//also used in renaming

mv \*.txt temp/

mv \*t temp/

mv \*ea\* temp/

//COPY cp –command

learner@:learning\_unix$ touch ~/file3

learner@:learning\_unix$ ls ~

command\_line\_course file3 learning\_unix linux\_bootcamp learner@:learning\_unix$ cp ~/file3 .

learner@:learning\_unix$ ls file1 file2 file3

//ECHO command

On its own, echo isn't a very exciting Unix command. It just echoes text back to the screen. But we can redirect that text into an output file by using the > symbol. This allows for something called file [redirection](https://en.wikipedia.org/wiki/Redirection_(Unix)).

***Careful when using file redirection (>), it will overwrite any existing file of the same name***

//CAT command to view content in the file

//WC command to count lines,words and characters

//TEXT EDITORS like nano,vi etc to edit the text in a file

//$ PATH variable to show the path

//grep command –

* show lines that match a specified pattern
* ignore case when matching (-i)
* only match whole words (-w)
* show lines that don't match a pattern (-v)
* Use wildcard characters and other patterns to allow for alternatives (\*, ., and [])

learner@host01:~$ cat naga.txt

w is the winter of our discontent.

All children, except one, grow up.

The Galactic Empire was dying.

In a hole in the ground there lived a hobbit.

It was a pleasure to burn.

It was a bright, cold day in April, and the clocks were striking thirteen.

It was love at first sight.

I am an invisible man.

It was the day my grandmother exploded.

When he was nearly thirteen, my brother Jem got his arm badly broken at the elbow.

Marley was dead, to begin with.

* learner@host01:~$ grep it naga.txt

In a hole in the ground there lived a hobbit.

Marley was dead, to begin with.

* learner@host01:~$ grep -i it naga.txt

In a hole in the ground there lived a hobbit.

It was a pleasure to burn.

It was a bright, cold day in April, and the clocks were striking thirteen.

It was love at first sight.

It was the day my grandmother exploded.

Marley was dead, to begin with.

* learner@host01:~$ grep -v it naga.txt

w is the winter of our discontent.

All children, except one, grow up.

The Galactic Empire was dying.

It was a pleasure to burn.

It was a bright, cold day in April, and the clocks were striking thirteen.

It was love at first sight.

I am an invisible man.

It was the day my grandmother exploded.

When he was nearly thirteen, my brother Jem got his arm badly broken at the elbow.

* [ ]: Matches any one of a set characters
* [ ] with hyphen: Matches any one of a range characters
* ^: The pattern following it must occur at the beginning of each line
* ^ with [ ] : The pattern must not contain any character in the set specified
* $: The pattern preceding it must occur at the end of each line
* . (dot): Matches any one character
* \ (backslash): Ignores the special meaning of the character following it
* \*: zero or more occurrences of the previous character
* (dot).\*: Nothing or any numbers of characters.

→ nagsa@601YRK3:~/Documents$ grep [anr] naga.txt

there is a village in andhra pradesh state

where the ancient civilization flourished

people are hungry

many are starving

labour are fighting for minimum wages

farmers for msp

//DEALING COLUMNS

Curl command is used to extract info from any url and gives STDOUT

* curl https://raw.githubusercontent.com/Blahah/command\_line\_bootcamp/master/testfiles/grades.txt > grades.txt less grades.txt

learner@host01:~$ cat grades.txt

Class Leia Luke Kirk Spock Arthur Ford Malcom Kaylee

Maths 95 70 40 100 30 80 50 85

English 99 60 90 100 90 20 50 60

Biology 85 40 50 100 10 20 50 60

P.E 80 150 100 100 20 30 50 50

* cut command used to cut specific column/columns from the table

learner@host01:~$ cut -f 5 grades.txt

Spock

100

100

100

100

* learner@host01:~$ cut -f -2,4-7,9 grades.txt > grades\_no\_cheaters.txt
* learner@host01:~$ cat grades\_no\_cheaters.txt

Class Leia Kirk Spock Arthur Ford Kaylee

Maths 95 40 100 30 80 85

English 99 90 100 90 20 60

Biology 85 50 100 10 20 60

P.E 80 100 100 20 30 50

* learner@host01:~$ cut -f 3,8 grades.txt | paste grades\_no\_cheaters.txt -> sorted\_grades.txt
* learner@host01:~$ cat sorted\_grades.txt

Class Leia Kirk Spock Arthur Ford Kaylee Luke Malcom

Maths 95 40 100 30 80 85 70 50

English 99 90 100 90 20 60 60 50

Biology 85 50 100 10 20 60 40 50

P.E 80 100 100 20 30 50 150 50

//USE of pipe ( | )command

* grep was opening\_lines.txt | wc -c

The first use of grep searches the specified file for lines matching 'was', it sends the lines that match through a pipe to the wc program. We use the -c option to just count characters in the matching lines (316).

* grep was opening\_lines.txt | sort | head -n 3 | wc -c

The second example first sends the output of grep to the Unix sort command. This sorts a file alphanumerically by default. The sorted output is sent to the head command which by default shows the first 10 lines of a file. We use the -n option of this command to only show 3 lines. These 3 lines are then sent to the wc command as before.

//miscellaneous commands

head- by default displays first 10 lines of the text file

tail -n – gives last n lines of a file

* grep -c '[bc]at' file.txt

gives count of lines in a file contain the words 'cat' or 'bat' (-c option of grep counts lines)

**FILE PERMISSIONS**

localhost:~# cat naga

 there is boy who can fly

localhost:~# ls -l naga

-rw-r--r--    1 root     root            27 Jan  3 15:15 naga

localhost:~# touch file1 file2 file3 file4

localhost:~# ls -l

total 20

-rw-r--r--    1 root     root             0 Jan  3 15:19 file1

-rw-r--r--    1 root     root             0 Jan  3 15:19 file2

-rw-r--r--    1 root     root             0 Jan  3 15:19 file3

-rw-r--r--    1 root     root             0 Jan  3 15:19 file4

Add owner execute bit:

localhost:~# chmod u+x file1

localhost:~# ls -l file1

-rwxr--r--    1 root     root             0 Jan  3 15:19 file1

Add other write & execute bit:

localhost:~# chmod o+wx file2

localhost:~# ls -l file2

-rw-r--rwx    1 root     root             0 Jan  3 15:19 file2

Remove group read bit:

localhost:~# chmod g-r file3

localhost:~# ls -l file3

-rw----r--    1 root     root             0 Jan  3 15:19 file3

Add read, write and execute to everyone:

localhost:~# chmod ugo+rwx file4

localhost:~# ls -l file4

-rwxrwxrwx    1 root     root             0 Jan  3 15:19 file4

**WITH NUMBERS**

4-read

2-write

1-execute

localhost:~# touch file1 file2 file3 file 4

localhost:~# ls -l

-rw-r--r--    1 root     root             0 Jan  3 15:30 file

-rw-r--r--    1 root     root             0 Jan  3 15:30 file1

-rw-r--r--    1 root     root             0 Jan  3 15:30 file2

-rw-r--r--    1 root     root             0 Jan  3 15:30 file3

Add owner execute bit:

localhost:~# chmod 744 file

localhost:~# ls -l file

-rwxr--r--    1 root     root             0 Jan  3 15:30 file

Add other write & execute bit:

localhost:~# chmod 647 file1

localhost:~# ls -l file1

-rw-r--rwx    1 root     root             0 Jan  3 15:30 file1

Remove group read bit:

localhost:~# chmod 604 file2

localhost:~# ls -l file2

-rw----r--    1 root     root             0 Jan  3 15:30 file2

Add read, write and execute to everyone:

localhost:~# chmod 777 file3

localhost:~# ls -l file3

-rwxrwxrwx    1 root     root             0 Jan  3 15:30 file3

### **chmod with sudo**

Changing permissions on files that you do not have ownership

RECURSIVE PERMISSION CHANGES