Introduction to Linux & Terminal commands

* Shell → The shell is the Linux command line interpreter that provides a command line interface, which interpret our commands and tell the OS what to do. e.g → bash (Bourne Again SHell),

* Terminal Emulators → Terminal Emulator is a program that will let us use the terminal in a graphical way.

* Command Prompt → It is a part of terminal where we write commands.

Commands

pwd → Print name of current working directory

ls → List directory contents

ls -a → List all the files and subdirectories of current working directory even hidden one also.

ls -l → Displays the current directory contents in long format.

cd → changes the working directory to your home directory

cd .. → move one back from the current working directory.

cd - → changes the working directory to the previous working directory.

ls -R → list directory tree recursively

** cat - Concatenate files

cat <file-name> → Display the content of file

cat > <file-name> → To create a file

cat <file-1> <file-2> > <file-3> → It merge the content of file-1 & file-2 and Put into file-3

↑ redirection operator

* echo → display a line of text.

echo "Hello world" → It display Hello world

* man [command] → Display help pages (manual) of <command>.

** Pipe

pipe operator |

Using pipe, the output of one command can be piped into the input of another.

command1 | command2

cat file.txt
Hello world

pipe

cat file.txt | tr a-z A-Z > upper.txt
↳ It translate all the small cases to upper case and store it in upper.txt file

cat upper.txt
O/P → HELLO WORLD

mkdir <directory_name> ⟶ create a directory

mkdir -p dir1/dir2/dir3 ⟶ to create directories inside directory

touch <file_name> ⟶ create a file

** cp - copy files & directories

cp file1 file2 ⟶ copy file1 to file2. If file2 exists, it will overwrite with file1 content.

cp -r dir1 dir2 ⟶ Recursively copies dir1 to dir2. dir2 is created if it doesn't exist.

** mv - move files/dir and Rename files/dir

mv file1 file2 ⟶ move file1 to file2
mv dir1 dir2 ⟶ move dir1 to dir2

mv <myfile> <newfile> ⟶ rename file to newfile

rm <file-name> → remove a file

rm -f <file-name> → forcefully remove the file

rm -rf <directory> → remove a directory

** sudo command : super user do

The sudo command allow you to run programs with the security privileges of another user (by default, as the superuser).

sudo <command>

It prompts for password.

df → It shows free disk space

df -h → shows in human-readable format

df -g → shows in gigabyte

du → display disk usage statistics

** head : It shows first few lines of files

** tail : It shows last few lines of files

head -n 4 <file-name> → Display first 4 lines of the file

tail -n 5 <file-name> → Display last 5 lines of the file.

note:- If we don't specify no. of line (-n) by default it display first/last 10 lines of respective file.

**# diff → compare files line by line

* diff <file1> <file2> → Compare file1 & file2 and display the lines which doesn't match

locate "*.txt" → Display all the files location that ends with .txt

** find : walk a file hierarchy

find <folder-name> → shows all stuff i.e files, hidden files, folder.

find <dir-name> -type d → It only list the folder

find <dir-name> -type f → It only list the files.

find <dir-name> -type f -name "*.txt"
↳ display all the .txt file inside that directory

find <dir-name> -type f -mmin -20
↳ display all the files modify less than 20 min

find <dir-name> -type f -mmin +15
↳ display all the files that were modify in more than 15 min

* Check manual of find command for more details.

```
# find <dir-name> -size +1k
```

↳ display the files & dir having size more than 1kb

```
# find <dir-name> -perm 777
```

↳ display the files having read, write and execute permissions.

* There are 3 types of permissions : Read, write and Execute. And there are 3 type of people who uses the computer, owner, group & other.

- r → allows a file/directory to be opened & read.
- w → allows a file or files within a directory to be created, deleted & renamed.
- x → allows a file to be treated as a program & executed or allow a directory to be entered.

** File Attributes	Meaning
→ -rwx------	file is readable, writable & executable by file's owner. No one else has any access.
→ -rw-r--r--	File is readable, writable by file's owner Members of the file's owner group may read the file and everybody else can also read the file.
→ -rw-rw----	File is readable and writable by file's owner and group members only.

* To change the mode (permissions) of a file or dir, the chmod command is used.

# ~~ls~~ chmod u=rwx, g=rx, o=r <file-name>

↳ file is readable, writable & executable to users(owner), readable and executable to group members and only readable to others

OR

# chmod $\underset{u}{7}\underset{g}{7}\underset{o}{7}$ <file-name> → gives read, write and execute permission to all.

** File Modes in Octal

Octal	File Mode
0	- - -
1	- - x
2	- w -
3	- wx
4	r - -
5	r - x
6	rw -
7	rwx

r → read
w → write
x → execute

# whoami → display username of current user

# chown → Change a file's owner

# sudo chown root <file-name>

↳ This will give root permission

** grep command : searches a file for a particular pattern of characters.

grep -V → to see version

grep <pattern" <file-name> → to search a particular pattern in file

Options:

-w : match whole word
-i : ignores case-sensitive
-n : display the mached lines & their line numbers.
-A n : prints searched line & n lines after the result.
-B n : prints searched line & n line before the result
-r : searches recursively
-l : displays list of file that contains specified string.
-c : prints the file with the count of the lines that match a pattern.

* history → shows history of commands usage

clear → used to clear the terminal screen

** Cursor Movement

⇒ ctrl -a ⟶ Move the cursor to beginning of line

⇒ ctrl -e ⟶ Move cursor to the end of the line.

⇒ ctrl -k ⟶ remove text from the cursor location to the end of line.

⇒ ctrl -u ⟶ remove text from the cursor location to the beginning of the line

⇒ "tab key" → used for auto completion while typing a command

⇒ ctrl -l ⟶ clear the screen and move the cursor to top-left corner same as clear command.

⇒ !<comman-no-from-history> → repeat history list item number

⇒ ctrl -r ⟶ This search incrementally from the current cmd line up the history list.

* Sort → sort lines of text

sort <file.txt> ⟶ sort a file

sort -r file.txt ⟶ sort in reverse order

jobs ⟶ list the jobs running in background & foreground.

ping → checks for network connectivity between two nodes.

```
$ ping google.com
```

⇒ we can also use IP address to ping directly.

wget → used to dowload content from internet

wget <url>

* top → displays all the processes running on the machine, and shows available resources.

kill <pid> → to kill that process
↑ process id   or stop

uname → show kernel name

zip file.zip file.txt → create compressed file

unzip file.zip → to unzip the file

hostname → show DNS name & set hostname
hostname -i → display the ip

useradd <user-name> → to add a user
passwd <user-name> → to set password to the user
userdel <user-name> → to delete the user

uname -a → shows the type of kernel
uname -m → prints machine's hardware name
uname -r → print kernel release

cat /etc/os-release → display all the info about the os
lscpu → to get CPU details

*# free → display free and used memory
free -m → display the result in megabytes
free -h → display in human-readable format.

vmstat → to see stat of virtual memory

id → displays groups
getent → checks if a user exist
lsof → display all the open files

nslookup → display IP address of the domain

netstat → display network related info such as
routing table, interface statics, etc

cut ⇒ cut out selected portions of each line
of a file.

cut -c 1-3 <file.txt>
↳ print first three characters of each line from
the file

ps aux → shows processes running status

** "&" operator → used to combine the commands

ping google.com & ping facebook.com

⇒ The function of '&' is to make the command run
in background.

** "&&" operator → The command following "&&"
(AND) operator will execute only if the
command preceding this operator has
been successfully executed.

Example

```
# echo "first" && echo "second"
o/p → first
      second
```

** OR operator (||)

Execute second command only if execution of first command fails. This operator is like an "else" statement in programming.

** NOT operator (!)

This is much like an 'except' statement. This will execute all except the condition provides.

Ex

```
# rm -r !(*.html)
```

→ delete all the files except 'html' files all at once.