UNIT 3

UNIX LOGIN RELATED COMMANDS

1) **login :**
   - This command is used to create new session with system.
   - It's asked for password if required.

   SYNTAX:

   login username
2) **logout :**
   - For logout `**exit**` OR `**logout**` OR `**bye**` command is used.

   SYNTAX:

   exit / logout / bye
3) **passwd:**
   - This command is used for change the password
   - First we have to provide **Current Password** after that is we provide **New Password** and **Confirm New Password**

   SYNTAX:

   passwd
4) **who :**
   - This Command display all user who is currently logged in .
   - Also we can use Option with this commands like:

OPTION	DETAILS
-a	**Display All Users**
-b , --boot	**Display The Time of Last System Boot**
-d , --dead	**Display Dead Process**
--help	**Display Help Message With Exit**
--version	**Display version information and Exit**

   - Also we can use `**am i**` with **who** commands .which is **Display Current User.**
5) **clear :**
   - This commands is used to clear screen.

   SYNTAX:

   clear
6) **uname:**
   - Display about the system.

   SYNTAX:

   uname

OPTION	DETAILS
-a	Kernel name, network node hostname, kernel release date, kernel version, machine hardware name, hardware platform, operating system
-s	Kernel name
-n	hostname of the network node(current computer).
-r	Kernal release name
-v	Version of kernel
-m	Machine hardware name
-p	Types of processor
-i	Platform of hardware
-o	Operating system name

FILE AND DIRECTORY RELATED COMMANDS

1) ls :

- This Commands is used to display list of file and folders (Directory).
- We can use some option with this commands like :

OPTION	DETAILS
-l long list	Display the long list of file and directories with file permission , owner , size & last modification date & time.
-a hidden files	Display list of file including hidden files & directories.
-x multicolumn	Display output in multi column.
-f	Display the directories & Sub Directories list separated by /
-s size	Used with **-l** option . gives file size in blocks instead of bytes.
-t time stamp	Sort output by time stamp instead of name. Use **-u** option to sort output by access time.
-r reverse	Display output in reverse alphabetic order.
-R subdirectory	Display list of file & directories including sub directories.
-lx	Display Files & Directories extension wise.
--block-size	Allows User to display list of files & directories in block of size such as k , m , t , z , p , y

2) cat :

- cat stands for catenate.
- Its Read data from the files and output their contents.
- We can create file and store data into file using following commands.

SYNTAX	EXAMPLE
cat ><file_name>	cat >txt1
Type text / data	This is Demo
CTRL + D	**CTRL + D**

- Read Data from file.

  EXAMPLE:

  cat h1

- Append data into file.

  EXAMPLE:

  cat >>h1

  New line ADDED.

  CTRL + D

3) cd:

- This command is used to change directory.
- You use it to move around within the hierarchy of your file system.
- By using this command we can access the directory.

OPTION	DETAILS
. -CURRENT DIRECTORY	This option change us into the directory we're already in.
.. -PARENT DIRECTORY	Used to move one step upward from directory list.
~ -HOME DIRECTORY	Used to move users home directory which is **/home/**
/ -ROOT DIRECTORY	Used to move users root directory.
../../	Allow user to move two step upward from current directory.

4) pwd:

- print the name of working Directory.
- We ca use following option with pwd:

OPTION	DETAILS
-P	Prints the path name which is not contain any symbolic link.
-L	Prints the path name which may contain any symbolic link.
--help	Print help message and exit.

5) rmdir:

- rmdir is used to remove directory.
- We can also remove multiple directory using this command.

OPTION	DETAILS
-P	Remove the directory ,directory name and its parent directory which is become empty.
--help	Print help message and exit

6) mv :

- This command is used to move file/directory or rename file/directory.
- Move is different than cp command . This command completely remove file from one source and move to the specified location.
- **RENAME THE FILE:**
  SYNTAX:
  mv file1.txt nm.txt
- **MOVE THE FILE:**
  SYNTAX:
  mv file1.txt /bscit

7) cp:

- This Function is used to copies file and Directories.
- You can copy any file in same directory or any other directory using this command.
- **COPY FILE IN SAME DIRECTORY WITH ANOTHER NAME:**
  SYNTAX:
  cp file1.txt demo.txt
- **COPY FILE TO OTHER LOCATION:**
  SYNTAX:
  cp file1.txt /bscit

8) ln:

- ln creates a link to file TARGET with the name LINK NAME.
- if LINK NAME is omitted , a link to TARGET is created in the current directory , using the name of TARGET as the LINK NAME.
- we can use following commands with ln :

OPTION	DETAILS
-f force	Overwrite file if exist.
-i interactive	Prompt the user before overwrite destination files.
-r relative	Create Symbolic link relative to link location.
-s symbolic	Make symbolic link instead of hard link.
--help	Display help message and exit.

9) rm :

- This command is used to delete file.
  SYNTAX:
  rm text1.txt

10) mkdir :

- This Command is used to make / create directory.
  SYNTAX:
  mkdir college
- Also we can **make multiple directory** using following command:
  mkdir bca bscit

11) uptime:

- This Command tell how long the system has been running.
- This command return the information about the current time , how many users are currently logged on and the system load average for the past 1,5 or 15 minutes.

12) chmod :

- This command is used to change mode of access file.
- Add Permission for r = read , w = write , x = execute.
- Reference for u = owner , g = group , o = other , a = All (Owner , Group , Other).
- Add Read , Write , Execute permission to owner.

  chmod u+rwx text.txt
- Remove write permission for the group and other.

  chmod go-w text.txt
- Read and Write for owner , for other and group read only.

  chmod u+rw,go+r text.txt

13) chown:

- To Change owner

  chown master file1.txt
- Change Group

  chown :group1 file1.txt
- To Change both Owner and Group

  chown master:group1 file1.txt

14) chgrp:

- To Change Group ownership.

  sudo chgrp group2 file1.txt

15) find:

- This commands is used to find the files.

  Find specific file using name:

  find ./ -name text.txt

  Find file using extension:

  find ./ -name *.txt

  Find and delete file with confirmation.

  find ./ -name text.txt -exec rm -i {} \;

Find Empty files of directory

find ./ -empty

Find file with entered permission (read write /read write / read)

find ./ -perm 664

Find Directory and Sub Directory

find . -type d

16) more:

- display the text on screen at a time.
- If File is too large for its content to fit in one screen , it will scroll off your screen when you view it with cat.
- This sometimes happen so fast that , before you hit <CTRL-> to stop it , quite a bit of output would have scrolled off.

  EXAMPLE:

  more std1.txt

17) less :

- less is a program similar to the more , but which allows backward movement in the file as well as forward movement.
- less does not have to read the entire input file before starting, so with large input files it starts up faster than text editors like vi.

  EXAMPLE:

  less std1.txt

18) head:

- This command is used to display the beginning lines of the file.
- Number is specified the maximum number of lines that you want to display.

EXAMPLE:

```
head    std1.txt
head   -4    std1.txt
```

19) tail:

- This command is used to display the ending lines of the file.
- By Default tail command prints last 10 lines.
- Number is specified the maximum number of lines that you want to display.
- Filename is specified the name of file.

OPTION	DETAILS
-l	Specify units of line
-c	Specify units of bytes
-p	Specify units of block

20) wc:

- wc stands for word count.
- This command is used to count the number of bytes whitespace , separate word & total no of lines of files.
- If specify the more than one file name that it will return the lines of the file
- This command will display the output in the order of lines, words & bytes.

OPTION	DETAILS
-c	total no of characters of the file
-l line	total no of line including blank line of the line
-w word	total no of word of the file.
--help	Display help & exit.

EXAMPLE:

```
wc    std1.txt
```

21) touch:

- This command is used to update the access & modification time of each file to the current time.
- We can modify the date & time of file creation.
- Default this command update date & time.

OPTION	DETAILS
-a	Update only the access time.
-m	Update the modification time & date only.
-r	Use the file times.
-d	Use date String to change date.

EXAMPLE:

```
touch   -d   '1 MAY 2023 11:11'  std1.txt
```

22) stat:

- stat is command which is gives information about the file and filesystem.
- Gives information such as the size of the file, access permission and user ID and group ID.

EXAMPLE:

```
stat    std1.txt
```

23) alias:

- alias command instructs the shell to replace one String with another string while executing the commands.

EXAMPLE:

```
alias  d = "cd Desktop"
alias -p
```

24) type :

- type command is used to get type of the command means it's return type of the provided command.

EXAMPLE:

```
type    clear
type     ls
```

REDIRECTION OPERATORS

- By default input is begin with the keyboard , and output is displayed on your screen.
- In UNIX or LINUX your keyboard is your standard input (**stdin**) device , and the screen or particular terminal window is the standard output (**stdout**) device.
- **Output redirection with >**
    - Sometimes you will want to put output of a command in a file , or you may want to issue another command on the output of one command .
    - This is known as redirection output.
    - Redirection is done using " **>** " .

    EXAMPLE:

    cat >h1.txt

    This is demo.
- **Input redirection**
    - In other case you may want a file to be the input for a command that normally wouldn't accept a file as an option.
    - This redirection of Input is done using the "**<**" operator.
- **The >> Operator**
    - Instead of Overwriting file data you can append text into the file using **>>** Symbol.

      EXAMPLE:

      cat >> h1.txt

      This is New Data….
- **The << Operator**
    - Here document is used to redirect input into an interactive shell script or program.
    - We can run an interactive program within a shell script without user action by supplying the required input for the interactive program, or interactive shell script.
    - Here the shell interprets the << operator as an instruction to read input until it finds a line containing the specified delimiter.
    - All the input lines up to the line containing the delimiter are then fed into the standard input of the command.
    - The delimiter tells the shell that the here document has completed.
    - Without it, the shell continues to read input forever.
    - The delimiter must be a single word that does not contain spaces or tabs.

      EXAMPLE:

      cat >> h1.txt
- **PIPELINES AND I/O REDIRECTIONS:**
    - | is the UNIX pipe symbol which is used in Command line.
    - The pipe is different than > because it is used to pass the output of a command to another command, not a file.

      EXAMPLE:

      cat h1.txt | wc

FINDING PATTERNS

1. grep :
   - GREP stands for **G**lobal **R**egular **E**xpression **P**rint.
   - This command print lines based on matching patterns.

OPTION	DESCRIPTION
-c COUNT	Count No of lines that match pattern.
-i IGNORE	Ignore case during matching pattern.
-v VERSES	Print all lines except those conation pattern.
-n NUMBER	Display line number that found pattern.
-h	Prevent name of file at the beginning of line.

EXAMPLE:

grep "TEST" file1.txt
grep "TEST" file1.txt file2.txt
grep -i "TEST" file1.txt
grep -c "TEST" file1.txt

2. fgrep:
   - fgrep stands for **F**ast **G**lobal **R**egular **E**xpression **P**rint.

EXAMPLE:

GET COUNT OF WORD.

fgrep -c "data" h1.txt

DISPLAY MATCHING LINES.

fgrep -h "data" h1.txt

DISPLAY TEXT WITH IGNORE CASE.

fgrep -i "Test" h1.txt

DISPLAY LINE NUMBER.

fgrep -n "Test" h1.txt

3. egrep :
   - egrep stands for **E**xtended **G**lobal **R**egular **E**xpression **P**rint.
   - It uses full regular expression match pattern.
   - If no files are specified , egrep assume standard input.

DISPLAY MATCHING LINES.

egrep Hello hello.txt

DISPLAY LINE COUNT.

egrep -c Hello hello.txt

DISPLAY LINE WHICH IS NOT MATCH.

egrep -i Hello hello.txt

DISPLAY LINE NUMBER WITH MATCH.

egrep -n Hello hello.txt

WORKING WITH COLUMN AND FIELDS

1. **cut :**
   - This Command is used to cut the column or no of character from file.
   - By using this commands we can separate columns well as character.

OPTION	DETAILS
-f --fields	Used to cut or remove the fields. Specify field no
-d --delimiter	Used to cut column if the column separated by a particular sign
-c --character	Specified the starting position of the character & ending position of character to cut the column.
--help	Print message and exit.
--version	Display Version and exit.

EXAMPLE:

```
cut -f 1 std1.txt
cut -c 1-10 std1.txt
```

2. **paste:**
   - The paste command is used to merge file vertically.
   - This command merge file column wise.

EXAMPLE:

```
paste -d ":" std1 rs
paste std1 rs
```

3. **join:**
   - This command is used to join line of two different files on a common field
   - For that we must have one common field on both file.

OPTION	DETAILS
-i --ignore	Ignore case during join
-e --empty	replace missing input fields with EMPTY.
1 FIELD	join on this FIELD of file 1.
-2 FIELD	join on this FIELD of file 2.
--help	Print message & exit.
--version	Display version & exit.

EXAMPLE:

```
join std1 rs
```

TOOLS FOR SORTING

1. **sort:**
   - sort command is used to sort the line of text file.
   - It is useful in database file.

OPTION	DETAILS
-u	**Display sorted text remove duplicate lines**
-r	**Reverse Order**
-d	**Uses Dictionary sort Order**
-f	**Ignore caps during sorting**
-b	**Ignore before space at the beginning of line during sorting.**

EXAMPLE:

sort -r std1.txt

2. **uniq:**
   - This command used to filters out repeated lines in a file.
   - UNIQUE command prints the unique lines in sorted files & returns only one of matching line.
   - It can show only the line that appears exactly once or more lines appear more than once.

OPTION	DETAILS
-u unique	**Only prints unique line**
-d repeated	**Only print duplicate line**
-c count	**Number representing how many times they occur.**
-i ignore	**Ignore the case sensitivity**

EXAMPLE:

cut –d “” –f 4 rs | sort | uniq

FILE COMPARING COMMANDS

1. **cmp:**
   - This command is used to compare two file bytes by bytes.

OPTION	DETAILS
-b print bytes	**Print differing bytes.**
-l --verbose	**Output byte number and differing byte value.**
-s --quit --silent	**Suppress normal output**

EXAMPLE:

cmp std1 rs

2. **comm:**
   - This command is used to find common things of files.
   - We can compare two file line by line & bytes by bytes.

OPTION	DETAILS
-1	**Unique to left file**
-2	**Unique to Right file**
-3	**Unique that both appear in both file.**

EXMAPLE:

comm rs rs1

3. **diff:**
   - Difference between two files.
   - Unlike comm and cmp command this command also tells you which line in one file have to changed to make the two file identical.

EXAMPLE:

diff rs rs1

CHANGING INFORMATION IN FILE

1. **tr :**
   - copies the standard input to standard output.
   - Translates , squeeze or delete characters from standard input & writing to standard output.

OPTION	DETAILS
-c	**Complement set of value in string1**
-s	**Replace each input sequence of repeated character that is listed in list**
-d	**Specify units of block**

EXAMPLE :

std1.txt | tr '[a-z]' '[A-Z]'

2. **sed:**
   - sed is stream editor in UNIX like operating system.
   - It can perform lot's of function on files like insertion , deletion , replace.

OPTION	DETAILS
-p	**Print each line of a file two times**
-d	**Delete specified line from file**
-g	**Global replacement replacing all occurrence string**
-s	**Replacing substring**
&	**Enclosed pattern**

EXAMPLE:

REPLACE UNIX WORD TO LINUX.

sed 's/UNIX/LINUX/' file1

DELETE 3rd line.

sed '3 d' file1

put UNIX word in [ ].

sed 's/UNIX/[&]/' file1

EXAMINING FILE CONTENT

1. **od:**
   - This command is used to convert the content of input in different formats with octal format as the default format.

   EXAMPLE:

   od -c std1.txt

MATHEMATICAL CALCULATION

1. **bc:**
   - bc commands stands for basic calculator.
   - This command is used for perform arithmetic calculation.
   - For exit CTRL + D.
   - Default bc return the integer value.
   - By default scale is 0 , to change the scale specifying decimal value for scale.
   - bc allows you to set base for calculation.
   - You can set binary , octal or hexadecimal base for input or output by using ibase & obase keyword.

   EXAMPLE:

   ```
   bc
       5+5
   ```

2. **factor :**
   - This command print the prime factor of the number.
   - To use factor without an argument and then you enter first number you want to factor.
   - Factor display the prime factor of that number.

   EXAMPLE:

   ```
   factor  12
   ```

MONITORING INPUT OUTPUT

1. **tee :**
   - This command just as a plumber uses a tee fitting in a line to send output in two directions simultaneously / at the same time.
   - Tee actually act as splitter, so you may have one input & two or more output.

-a	Append output to existing files.

   EXAMPLE:

   ```
   who | tee user.txt
   ```

2. **script :**
   - Linux is used to make typescript or record all the terminal activities.
   - script is used to take a copy of everything which is output to the terminal and place it in a log file.
   - It should be followed by the name of the file to place the log in.
   - The exit command should be used to stop logging and close the file.
   - Everything between the script and the exit command is logged to the file.
   - This includes the confirmation messages from script itself.

   EXAMPLE:

   ```
   script
   ….
   ….
   endScript
   ```

DATE & TIME RELATED COMMAND

1. **cal :**
   - This command is used to display a simple calendar.
   - By default cal command display the calendar of current month.
   - We can display a complete year calendar by specifying year as an argument.

   EXAMPLE:

   DISPLAY CALENDAR OF 2022.

   cal 2022

   DISPLAY CURRENT MONTH CALENDAR

   cal

2. **date :**
   - This command is used to print or set the system date & time.
   - You can use format specifier with which is always used with "+" & followed by "%" sign.

OPTION	DETAILS
%a	Return abbreviated weekday name
%A	Return full weekday name
%b	Return abbreviated month name
%B	Return abbreviated full month name
%d	Return day of month
%D	Return date in mm/dd/yy format
%H	Return hour (00..23)
%M	Return minutes (00..59)
%m	Return month number (1..12)
%S	Return second (00..60)
%y	Return year in two digit
%Y	Return year in four digit
%I	Return hour (01..12)
%T	Return time in HH:MM:SS
%P	Return equivalent of either AM/PM
%p	Return equivalent of either AM/PM in lower case
%r	Return 12 hour clock time
%R	Return 24 hour and minutes same as %H and %M
%w	Return the day of week (0..6 – 0 Sunday,1 Monday...)
%u	Return number of year , with as a first day of week Sunday(00..53)
%Z	Return alphabetic time zone.

EXAMPLE:

1) **display date**

   date

2) **display weekday**

   date +"%A"

3) **display date in mm/dd/yy**

   date + "%D"

COMMUNICATION COMMAND

1) **telnet :**
   - connect the remote system using telnet protocol.
   - When this command invoked it enters the command mode & display telnet prompt.
   - In this mode it accept the commands to be executed.
   - When TELNET is used with ip address of the remote system, it asked for usename & password to enter the system.

2) **wall :**
   - This command is connect It facilitates the administrator to communicate with all the all users who are connected to the UNIX server.
   - It addresses all the users simultaneously.
   - The message passed as an argument to the wall command or it can be send by using standard input from terminal.
   - The message that we sent using wall will contect all broadcast information as header such as: current date and time, user etc.

     EXAMPLE:

     wall

     Submit Your UNIX Assignment Tomorrow.

3) **write :**
   - This command allows communicating with each other by coping the lines form one terminal to another.
   - When you run the write command, the user you are writing to gets a message of the format: Message from yourname@yourhost on yourtty at hh:mm ...

     EXAMPLE:

     write stud
   - On above example write a message to the user “stud”
   - After entering this command, you will be placed on a blank line, where everything you type will be sent to the other user (line by line).
   - Typing the interrupt character (CTRL-C, by default) will return you to the command prompt, and end the write session.

4) **mail :**
   - This command allows sending or receiving mail.
   - Mail servers mail in the directory /var/spool/mail.
   - When you gives mail command it will open mail server & display the first message in the mail box.
   - Use following options with mail command.

OPTION	DETAILS
+	Print next Message
-	Print previous Message
N	Print Message with numbered N
H	Print Header of all Message
d N	Deleted Message N
u N	Undeleted Message N
Q	Quit
s file name	Save Current Message with Header in Filename
m user	Forward mail to user
!cmd	Return to cmd mode

5) **finger :**
   - This command looks up and displays information about system users.
   - When we used this command without using any options it display the list of currently logged in users.
   - It displays the login, full name, terminal type, idle time.
   - Login time is displayed as month, day, hours, and minutes.

   EXAMPLE:

   finger stud

6) **mesg :**
   - control write access to your terminal

   EXAMPLE

   mesg

   mesg [y/n]

7) **ping :**
   - This command is used to check internet connectivity.
   - Using this command we can get response from the provided site or ip.

   EXAMPLE:

   ping www.google.com

   ping 216.58.203.4

   ping localhost

PROCESS RELATED COMMANDS

- A process , in simple terms, is an instance of a running program.
- Every process is identified by a number , known as process id or pid.
- Each process has :
  - Unique Process Identifier Number (PID)
  - Virtual Address Space
  - Security Context
  - Open devices/handles , executes code
  - Environment Variables , priority and more
- When you log in to UNIX system , a process is immediately set up by kernel.
- Every command that you type is actually the shell process.
- This Process alive until you log out.

1) ps :

- ps command used to display the attributes of process.
- It is one of the few commands of UNIX system that has knowledge of kernel built into it.
- Output of ps id depending on the version of UNIX as well as the hardware used.
- Like WHO command ps also generate header information , each line shows PID , the terminal which the process is associated , cumulative processor time and process name.

OPTION	DETAILS
-f	Display full Options
-u	Specified Usernames.
-a	Information about other user's processes as well as your own.
-e	Display all the processes
--forest	Will construct as ascii art style tree view of the process hierarchy
-C	Search the processes by their name or command use
-p	By Process ID

EXAMPLE: ps

2) nice :

- This command used to modify or print the scheduling priority of the job.
- When we use nice command it display the current scheduling priority.
- nice runs command COMMAND with an adjusted "niceness"
- A Process with a lower niceness value is given higher priority and more CPU time.
- A Process with a higher niceness value (a "nicer" process) is given a lower priority and less CPU time , freeing up resources for processes which are more demanding.
- Niceness values range from -20 (most favourable to the process) to 19 (least favourable to the process).
- If increment is not specified than default 10.
- Command is the name of command that is to be invoked.
- Argument is a string that invoking command.

EXAMPLE: nice

3) kill :

- This command used to terminate process.
- This command use one or more process PID as its argument.
- Kill by default use the single number 15 to terminate process.

OPTION	DETAILS
Pid	List of process that kill command should send a signal
-s --signal	Send the specified signal to the process
-l --list	List all the available signals.

EXAMPLE:

kill 2160

4) **at :**
   - This command used to schedules a command to be run once at a particular time.
   - By using this command it is also possible to run the jobs when the cpu overhead is less.
   - at allows complex time specified in form of HH:MM to load the job at specified time.
   - You can also schedule job to execute specified date

   EXAMPLE:

   1. To Execute some task Tuesday at two hours later then current time.
      at tuesday +2 hours
   2. Set time 12:15
      at 12:15

5) **batch :**
   - batch command is used to read command from standard input or a specified file and execute them when system load levels permit.

   EXAMPLE:

   batch

6) **wait :**
   - This command pauses until execution of a background process has ended.
   - We have to specify the pid or job ID of a currently executing background process (job) as an argument.
   - If we don't specify pid(n) the command waits until all jobs known to the invoking shell have terminated.
   - wait normally returns the exit status of the last job which terminated.

7) **sleep :**
   - This command used to delay for a specified amount of time.
   - It is used to suspends execution for time second then execute command after specified time.

   S= for second (the default)
   m=for minutes.
   h= for hours.
   d= for days.

   EXAMPLE:

   - **Create file 'test.sh' and run with `sh`**

     echo "display Message After 10Second"
     sleep 10
     echo "Welcome"

8) **top :**
   - This command is used to show the active linux Processes.

   EXAMPLE:

   top

9) **jobs :**
   - jobs command is used to list the jobs that you are running in the background and in the foreground .
   - if the prompt is returned with no information means no jobs are present.
   - All shell are not capable of running this command.
   - This command is only available in the csh , bash ,tcsh , and ksh shells.

   EXAMPLE:

   jobs

Concept of Mounting a File System

1) mount :

- In Unix all data arranged in files and all files are connected with a directory.
- The directory structure begin with “/”.
- The mount command mount a storage device or file system.
- So that it can be accessible and used with an operating system.
- The default form of mount is --- mount -t device dir.
- This command tell kernel to attach the file system found on device at the directory dir.
- The previous content an owner mode of this directory becomes invisible.

OPTION	DETAILS
-v	Output Version
-h	Print Help Message
-a	Mount all file system
-r	Mount the file system read only
-w	Mount the file system read and write only
-l	Mount the partition that has specified label.

Concept of Demounting a File System

1) unmount :

- By using unmount command we can detached the file system mounted from the hierarchy.
- A file system is specified by giving directory where it has been mounted .
- The file system cannot unmounted when it is busy.
- When there are open files on it or when some process has working directory there.

OPTION	DETAILS
-v	Output version
-h	Print Help Message
-n	Unmount without writing in /etc/mtab
-r	In case unmounting fail then try remount read only
-a	Demount all the file system of /etc/mtab

UNIT 4

➢ MODES IN VI EDITOR

- There are three basic mode:
  1. Command mode
  2. Insert / input mode
  3. Ex mode / line mode

1) Command mode :

- It's a default mode of vi editor.
- In this mode most letters or sort sequence of letter that we type will be considered as command without pressing enter key.
- You will have to be in this mode to copy or delete text
- When you press ESC key terminal give you beep sound.

2) Insert / input mode :

- It's a default mode of vi editor.
- This mode enables you to insert text into the file.
- Type "i" letter to enter in insert mode from command mode.
- Press ESC key to end insert mode & transfer into command mode again.

3) Ex mode / line mode :

- This mode used to handle files & perform substitutions.
- Use EX mode to enter line oriented commands.
- To enter this mode type ":" symbol, when you press ":" your cursor moves to the bottom of the screen.
- Type EX mode command & press enter to execute your command.
- All the commands of EX mode begin with ":" symbol

➢ Cursor movement

COMMAND	DETAILS
A	Move the cursor forward to the beginning of the word
E	Move cursor at the end or word
{	Move cursor at previous paragraph
}	Move cursor at next paragraph
K	Moves the cursor up one line.
J	Moves the cursor down one line.
H	Moves the cursor to the left one character position
L	Moves the cursor to the right one character position
0 or |	Position cursor at beginning of line.
$	Position cursor at end of line .
W	Position cursor to the next word
B	Position cursor to the previous word
(	Position cursor to beginning of current sentence
)	Position cursor to beginning of next sentence
+	Move cursor to beginning of next line
-	Move cursor to beginning of previous line

➢ SCREEN CONTROL COMMANDS

COMMAND	DETAILS
CTRL + F	Scroll whole Screen Down
CTRL + B	Screen whole Screen Up
CTRL + D	Move Down One Half Screen
CTRL + U	Move up(back) Half Screen (12 Lines Up)
CTRL + L	Redraw the Screen
CTRL + R	Redraw The Screen , Removing the deleted Lines

➢ COMMANDS REGARDING EDITING OF TEXT

COMMAND	DETAILS
COMMANDS FOR COPY	
Yy	Copy current line
Yw	Copy current word
COMMANDS FOR CUT	
Cc	Cut current line
Cw	Cut current word
COMMANDS FOR PASTE	
P	Paste text before current cursor position
p	Paste text after current cursor position
COMMANDS FOR DELETING TEXT	
X	Delete current character
Dd	Delete current line
Dw	Delete current word
Dc	Delete character from cursor position to end of the word
Db	Delete character from cursor position to beginning of the word
Dy	Delete character from cursor position to end of the paragraph
d$	Delete character from cursor position to end of the file

➢ INTRODUCTION OF NANO EDITOR:

- Nano is a simple , display-oriented text editor for UNIX.
- Nano editor is a user friendly , simple and WYSIWYG (**What You See Is What You Get**) text editor .
- It has an easy GUI which allows user to interact directly with the text in spite of switching between the mode as in vim editor.
- If nano is not installed write following command
  sudo apt install nano
- Create new file
  nano filename
- Save the file
  CTRL + O
- Cut Paste in file
  CTRL + K (CUT)
  CTRL + U (PASTE)
- Search / Find
  CTRL + W
- Exit
  CTRL + X

➢ SHELL KEYWORDS :

- Keyword are the word that have specific meaning for particular language.
- UNIX shell script are not having collection of commands but also have feature of programming like condition , loop etc.
- Keywords are also referred as reserved words.

echo	**read**	**set**	**unset**
readonly	**shift**	**exports**	**if**
else	**while**	**fi**	**case**
esac	**do**	**done**	**for**
break	**exit**	**wail**	**eval**
umask	**continue**	**until**	**return**

➢ SHELL VARIABLES :

- A variable is a character string to which we assign a value.
- The value assigned could be a number , text , filename , device or any other types of data.
- A variable is nothing more than a pointer to the data.
- The shell enables you to create , assign , and delete variables.
- The name of a variable can contain only letters (a to z or A to Z) , number (0 to 9) or the underscore character.

➢ SYSTEM VARIABLES :

1. **PS2:**
   - The characters that the shell displays as your command prompt are stored in the variable.
   - You can change this variable to be anything you want.

     EXAMPLE:

     PS2 = “second prompt ->”
     echo “This is
     second prompt->DEMO”
2. **PATH:**
   - Indicates the search path for commands. It is a colon-separated list of directories in which the shell looks for commands.
3. **HOME:**
   - Indicates the home directory of the current user: the default argument for the cd **built-in** command.
4. **LOGNAME:**
   - print user's login name.
5. **MAIL:**
   - Linux **mail** command is a command-line utility that allows us to send emails from the command line.
   - The mail command can be used directly by the terminal as well as the Shell script.
6. **IFS:**
   - Indicates the **Internal Field Separator** that is used by the parser for word splitting after expansion.
7. **TERM:**
   - Refers to the display type.