

Ex No: 1(b) BASIC LINUX COMMANDS

DATE: 22.01.2025

1.1 GENERAL PURPOSE COMMANDS

1. The date command

Description: Displays the current date and time.

Syntax:

\$ date

Input:

\$ date

Output:

Sat Apr 12 10:23:45 IST 2025

Other Formats:

Format Purpose Input Output

+%m Display month (numeric) \$ date +%m 04

+%h Display month (name) \$ date +%h Apr

+%d Display day of the month \$ date +%d 12

+%y Last two digits of year \$ date +%y 25

+%H Display hour \$ date +%H 10

+%M Display minutes \$ date +%M 23

+%S Display seconds \$ date +%S 45

2. The echo command

Description: Prints a message to the terminal.

Syntax:

\$ echo "your message"

Input:

\$ echo "God is Great"

Output:

God is Great

3. The cal command

Description: Displays calendar of specified month/year. **Syntax:**

\$ cal [month] [year]

Input:

\$ cal Jan 2012

Output:

```
January 2012
Su Mo Tu We Th Fr Sa
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
```

4. The bc command

Description: Launches a basic calculator.

Syntax:

\$ bc

Input:

\$ bc -l

16/4

5/2

Output:

4

2

5. The who command

Description: Shows users currently logged in. **Syntax:**

\$ who

Input:

\$ who

Output:

kaviya tty1 2025-04-12 09:00

6. The who am i command

Description: Shows info about current session user.

Syntax:

\$ who am i

Input:

\$ who am i

Output:

kaviya pts/0 2025-04-12 09:10

7. The id command

Description: Displays UID, GID, and groups of user.

Syntax:

\$ id

Input:

\$ id

Output:

uid=1000(kaviya) gid=1000(kaviya) groups=1000(kaviya),10(wheel)

8. The tt command

Description: Displays terminal name.

Syntax:

\$ tty

Input:

\$ tty

Output:

/dev/pts/0

9. The clear command

Description: Clearsthe terminal screen.

Syntax:

\$ clear

Input:

\$ clear

Output:

(Terminal screen gets cleared)

10. The man command

Description: Shows manual page for commands. **Syntax:**

\$ man [command]

Input:

\$ man date

Output:

(Manual page opens for the date command. Press q to quit.)

11. The ps command

Description: Shows running processes.

Syntax:

\$ ps

Input:

\$ ps

Output:

PID TTY TIME CMD

1234 pts/0 00:00:00 bash

1278 pts/0 00:00:00 ps

12. The uname command

Description: Shows system details.

Syntax:

\$ uname [option]

Input:

```
$ uname -a
```

Output:

```
Linux fedora 6.5.9-300.fc39.x86_64 #1 SMP x86_64 GNU/Linux
```

1.2 DIRECTORY COMMANDS

1. The pwd command

Description: Displays current directory path.

Syntax:

```
$ pwd
```

Input:

```
$ pwd
```

Output:

```
/home/kaviya
```

2. The mkdir command

Description: Creates a new directory.

Syntax:

```
$ mkdir dirname
```

Input:

```
$ mkdir receee
```

Output:

(A directory named receee is created)

3. The rmdir command

Description: Deletes an empty directory.

Syntax:

```
$ rmdir dirname
```

Input:

```
$ rmdir receee
```

Output:

(The receee directory is removed if empty)

4. The cd command

Description: Changes the current directory. **Syntax:**

```
$ cd dirname
```

Input:

```
$ cd receee
```

Output:

(You are now inside the receee directory)

5. The ls command

Description: Lists contents of the directory. **Syntax:**

```
$ ls
```

Input:

```
$ ls
```

Output:

```
file1.txt file2.sh receee
```

Input (long listing):

```
$ ls -l
```

Output:

```
-rw-rw-r-- 1 kaviya kaviya 0 Apr 12 10:24
```

```
file1.txt Input (including hidden files):
```

```
$ ls -a
```

Output:

```
. .. .bashrc file1.txt receee
```

1.3 3 FILE HANDLING COMMANDS

1. The 'cat' command

Purpose: Used to create a file.

SYNTAX:

```
$ cat > filename
```

EXAMPLE:

```
$ cat > rec
```

Arun

Kaviya

^D # (Press Ctrl + D to save and exit)

2. Display contents of a file

SYNTAX:

\$ cat filename

EXAMPLE:

\$ cat rec

Output:

Arun

Kaviya

3. The 'cp' command

Purpose: Copy contents from one file to another.

SYNTAX:

\$ cp oldfile newfile

EXAMPLE:

\$ cp rec cse

\$ cat cse

Output:

Arun

Kaviya

4. The 'rm' command

Purpose: Delete a file.

SYNTAX:

\$ rm filename

EXAMPLES:

\$ rm rec

\$ rm -f rec

\$ rm -fr directory_name # Deletes folder recursively

5. The 'mv' command

Purpose: Move or rename a file. **SYNTAX:**

\$ mv oldfile newfile

EXAMPLE:

```
$ mv cse eee
```

```
$ ls
```

Output: eee

6. The 'file' command

Purpose: Determine file type.

SYNTAX:

```
$ file filename
```

EXAMPLE:

```
$ file eee
```

Output: eee: ASCII text

7. The 'wc' command

Purpose: Word, line, and character count. **SYNTAX:**

```
$ wc filename
```

EXAMPLE:

```
$ wc eee
```

Output: 2 2 12 eee

8. Directing output to a file

Purpose: Save command output to a file. **SYNTAX:**

```
$ ls > filename
```

EXAMPLE:

```
$ ls > list.txt
```

```
$ cat list.txt
```

Output:

eee

list.txt

9. Pipes

Purpose: Use output of one command as input to another.

SYNTAX:

```
$ command1 | command2
```


EXAMPLE:

```
$ who | wc -l
```

Output: 3 # (Displays number of logged-in users)

10. The 'tee' command

Purpose: Save output in middle of a pipe.

SYNTAX:

```
$ command | tee filename
```

EXAMPLE:

```
$ who | tee sample | wc -l
```

Output: 3

```
$ cat sample
```

Output: list of logged-in users

11. Metacharacters in Unix

Purpose: Pattern matching with special characters.

Symbol Meaning

* Matches any number of characters

? Matches a single character

[] Matches any character in the set

[!] Negates the set

EXAMPLES:

```
$ ls r* # Files starting with r
```

```
$ ls ?kkk # Files like "rkkk", "skkk"
```

```
$ ls [a-m]* # Files starting with a-m
```

```
$ ls [!a-m]* # Files NOT starting with a-m
```

13. File Permissions

Each file has:

- Owner
- Group
- Others

Each with:

- r (read) = 4

- **w (write)** = 2

- **x (execute)** = 1

EXAMPLE:

\$ ls -l college

-rwxr-xr-- 1 Lak std 1525 Jan 10 12:10

college • **rw**x: Owner has read, write,

execute • **r-x**: Group has read and
execute

• **r--**: Others have only read

13. The 'chmod' command

SYNTAX:

\$ chmod category operation permission

filename **EXAMPLES:**

\$ chmod u-wx college

(Remove write & execute for user)

\$ chmod u+rw, g+rw college

(Add read & write to user & group)

\$ chmod g=wx college

(Set write & execute to group only)

14. Octal Notation

SYNTAX:

\$ chmod 761 college

Explanation:

- 7 (owner) = rwx

- 6 (group) = rw-

- 1 (others) = --x

1.4 GROUPING COMMANDS

1. Semicolon (;)

Executes multiple commands sequentially.

EXAMPLE:

```
$ who; date
```

Output:

(list of users)

Sat Apr 12 10:45:00 IST 2025

2. Logical AND (&&)

Executes next only if previous is successful.

EXAMPLE:

```
$ ls && date
```

Output:

(file list)

Sat Apr 12 10:45:00 IST 2025

3. Logical OR (||)

Executes next only if previous fails.

EXAMPLE:

```
$ ls nofile || date
```

Output:

ls: cannot access 'nofile': No such file or directory

Sat Apr 12 10:45:00 IST 2025

1.5 5 FILTERS

1. head

SYNTAX:

```
$ head filename
```

EXAMPLE:

```
$ head college  
(Shows top 10 lines)
```

```
$ head -5 college
```

(Shows top 5 lines)

2. tail

SYNTAX:

```
$ tail filename
```

EXAMPLE:

```
$ tail college
```

(Shows bottom 10 lines)

```
$ tail -5 college
```

(Shows bottom 5

lines) **3. more**

Used for paging large outputs. **SYNTAX:**

```
$ ls -l | more
```

4. grep

Search for patterns.

SYNTAX:

```
$ grep "pattern"
```

filename **EXAMPLE:**

```
$ cat > student
```

Arun cse

Ram ece

Kani cse

^D

```
$ grep "cse" student
```

Output:

Arun cse

Kani cse

5. sort

Sorts lines.

SYNTAX:

\$ sort filename

EXAMPLES:

\$ sort college # Sort alphabetically

\$ sort -r college # Reverse order

\$ sort -n numbers.txt # Numeric sort

\$ sort -u college # Remove duplicates

6. nl

Adds line numbers.

SYNTAX:

\$ nl filename

EXAMPLE:

\$ nl college

1 Arun

2 Kaviya

7. cut

Extracts specific character positions.

SYNTAX:

\$ cut -c1-4 filename

EXAMPLE:

\$ cut -c1-3 college

Output:

Aru

Kav

1.5 OTHER ESSENTIAL COMMANDS

1. free

Description: Displays the amount of free and used physical and swap memory in the system. • **Synopsis:** free [options]

• **Example:**

Input:

```
[root@localhost ~]# free -t
```

Output:

```
total used free shared buff/cache available
Mem: 4044380 605464 2045080 148820 1393836 3226708 Swap:
2621436 0 2621436
Total: 6665816 605464 4666516
```

2. top

Description: Provides a dynamic real-time view of processes in the system. • **Synopsis:** top [options]

• **Example:**

Input:

```
[root@localhost ~]# top
```

Output:

```
top - 08:07:28 up 24 min, 2 users, load average: 0.01, 0.06, 0.23
Tasks: 211 total, 1 running, 210 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.8 us, 0.3 sy, 0.0 ni, 98.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 4044380 total, 2052960 free, 600452 used, 1390968
buff/cache KiB Swap: 2621436 total, 2621436 free, 0 used. 3234820 avail
Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND 1105 root
20 0 175008 75700 51264 S 1.7 1.9 0:20.46 Xorg
2529 root 20 0 80444 32640 24796 S 1.0 0.8 0:02.47 gnome-term
```

3. ps

Description: Reports a snapshot of current processes.

• **Synopsis:** ps [options]

• **Example:**

Input:

```
[root@localhost ~]# ps -e
```

Output:

```
PID TTY TIME CMD
1 ? 00:00:03 systemd
2 ? 00:00:00 kthreadd
3 ? 00:00:00 ksoftirqd/0
```

4. vmstat

Description: Reports virtual memory statistics.

- **Synopsis:** vmstat [options]
- **Example:**

Input:

```
[root@localhost ~]# vmstat
```

Output:

```
procs -----memory----- ---swap-- ----io---- -system-- -----cpu-----
r b swpd free buff cache si so bi bo in cs us sy id wa st
0 0 0 1879368 1604 1487116 0 0 64 7 72 140 1 0 97 1 0
```

5. df

Description: Displays the amount of disk space available on the file system.

- **Synopsis:** df [options]
- **Example:**

Input:

```
[root@localhost ~]# df
```

Output:

```
Filesystem 1K-blocks Used Available Use% Mounted on
devtmpfs 2010800 0 2010800 0% /dev
tmpfs 2022188 148 2022040 1% /dev/shm
tmpfs 2022188 1404 2020784 1% /run
/dev/sda6 487652 168276 289680 37% /boot
```

6. ping

Description: Verifies whether a device can communicate with another over a network. • **Synopsis:** ping [options] destination

• **Example:**

Input:

```
[root@localhost ~]# ping 172.16.4.1
```

Output:

```
PING 172.16.4.1 (172.16.4.1) 56(84) bytes of data.
```

```
64 bytes from 172.16.4.1: icmp_seq=1 ttl=64 time=0.328 ms
```

```
64 bytes from 172.16.4.1: icmp_seq=2 ttl=64 time=0.228 ms
```

```
64 bytes from 172.16.4.1: icmp_seq=3 ttl=64 time=0.264 ms
```

```
64 bytes from 172.16.4.1: icmp_seq=4 ttl=64 time=0.312 ms
```

```
^C
```

```
--- 172.16.4.1 ping statistics ---
```

```
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
```

```
rtt min/avg/max/mdev = 0.228/0.283/0.328/0.039 ms
```

7. ifconfig

Description: Used to configure and display network interface parameters. • **Synopsis:** ifconfig [options]

• **Example:**

Input:

```
[root@localhost ~]# ifconfig
```

Output:

```
enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
```

```
    inet 172.16.6.102 netmask 255.255.252.0 broadcast 172.16.7.255
```

```
    inet6 fe80::4a0f:cfff:fe6d:6057 prefixlen 64 scopeid 0x20<link>
```

```
    ether 48:0f:cf:6d:60:57 txqueuelen 1000 (Ethernet)
```

```
    RX packets 23216 bytes 2483338 (2.3 MiB)
```

```
    RX errors 0 dropped 5 overruns 0 frame 0
```

```
    TX packets 1077 bytes 107740 (105.2 KiB)
```


TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

8. traceroute

Description: Tracks the route that a packet takes to reach the destination.

- **Synopsis:** traceroute [options] destination

- **Example:**

Input:

```
[root@localhost ~]# traceroute www.rajalakshmi.org
```

Output:

```
traceroute to www.rajalakshmi.org (220.227.30.51), 30 hops max, 60 byte
```

```
packets 1 gateway (172.16.4.1) 0.299 ms 0.297 ms 0.327 ms
```

```
2 220.225.219.38 (220.225.219.38) 6.185 ms 6.203 ms 6.189 ms
```