

RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105



**RAJALAKSHMI
ENGINEERING COLLEGE**

**CS23431
OPERATING SYSTEMS LAB**

Laboratory Observation NoteBook

Name : VISHWAK S

Year/Branch/Section : II/CSE/D

Register No. : 230701385

Semester : IV

Academic Year: 2024-25

INDEX

EXP.NO	Date	Title	Page No
1a	25.01.2025	Installation and Configuration of Linux	4
1b	01.02.2025	Basic Linux Commands	7
2	01.02.2025	Study of Unix editors : sed,vi,emacs	27
3 a)	07.02.2025	Shell script	32
		a) Arithmetic Operation -using expr command	
		b) Check leap year using if-else	34
3 b)	08.02.2025	a) Reverse the number using while loop	36
		b) Fibonacci series using for loop	38
4	14.02.2025	Text processing using Awk script	
		a) Employee average pay	40
		b) Results of an examination	43
5	15.02.2025	System calls –fork(), exec(), getpid(),opendir(), readdir()	46
6a	21.02.2025	FCFS	51

6b	21.02.2025	SJF	54
6c	28.02.2025	Priority	57
6d	28.02.2025	Round Robin	60
7.	28.03.2025	Inter-process Communication using Shared Memory	64
8	29.03.2025	Producer Consumer using Semaphores	69
9	05.04.2025	Bankers Deadlock Avoidance algorithms	75

10 a	11.04.2025	Best Fit	79
10 b	11.04.2025	First Fit	85
11a	12.04.2025	FIFO	87
11b	18.04.2025	LRU	91
11c	18.04.2025	Optimal	95
12	19.04.2025	File Organization Technique- single and Two level directory	99

Ex. No. : 1a

Date : 25.01.2025

Register No. : 230701385

Name : VISHWAK S

INSTALLATION AND CONFIGURATION OF LINUX

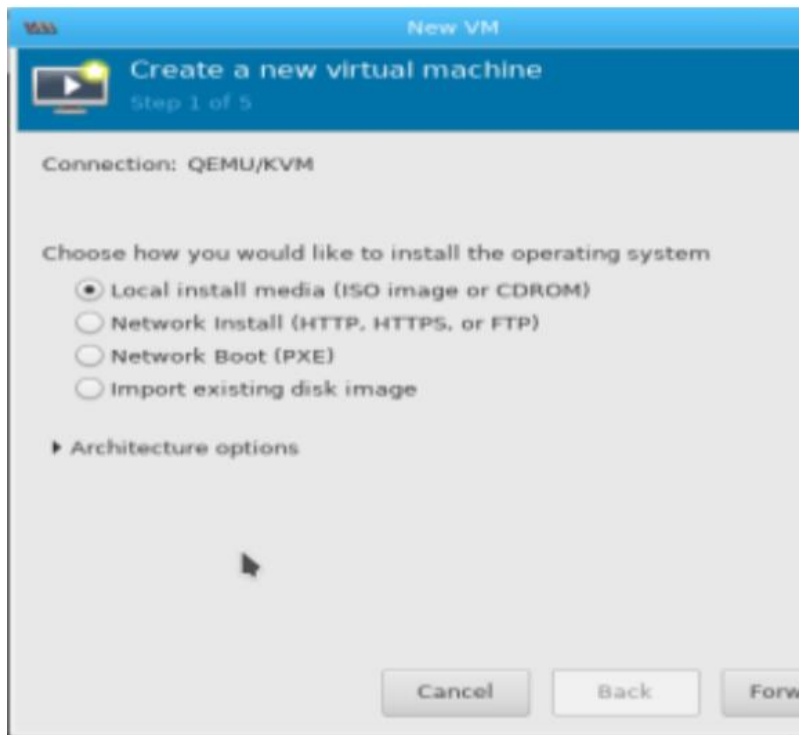
Aim:

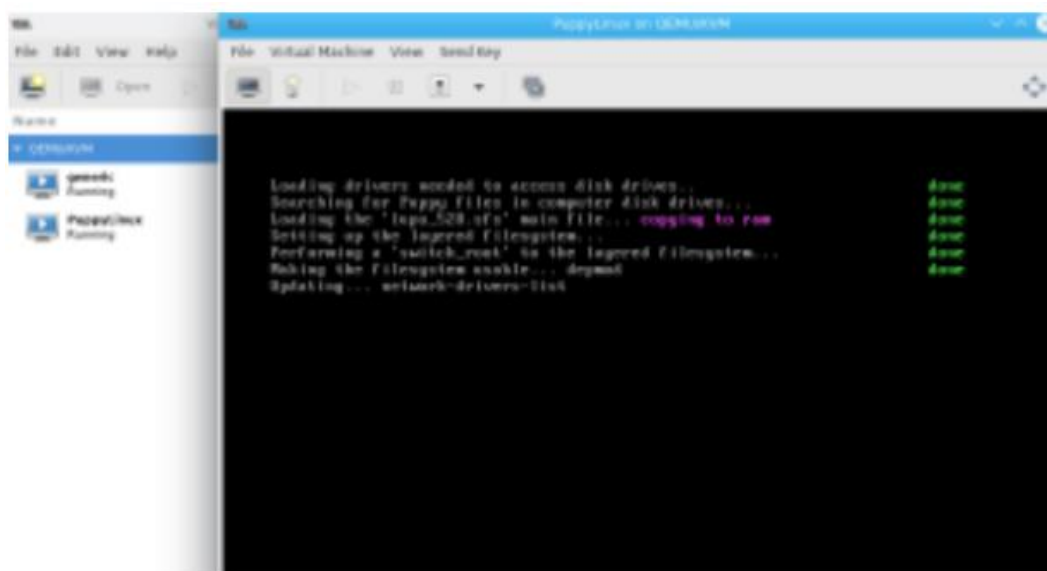
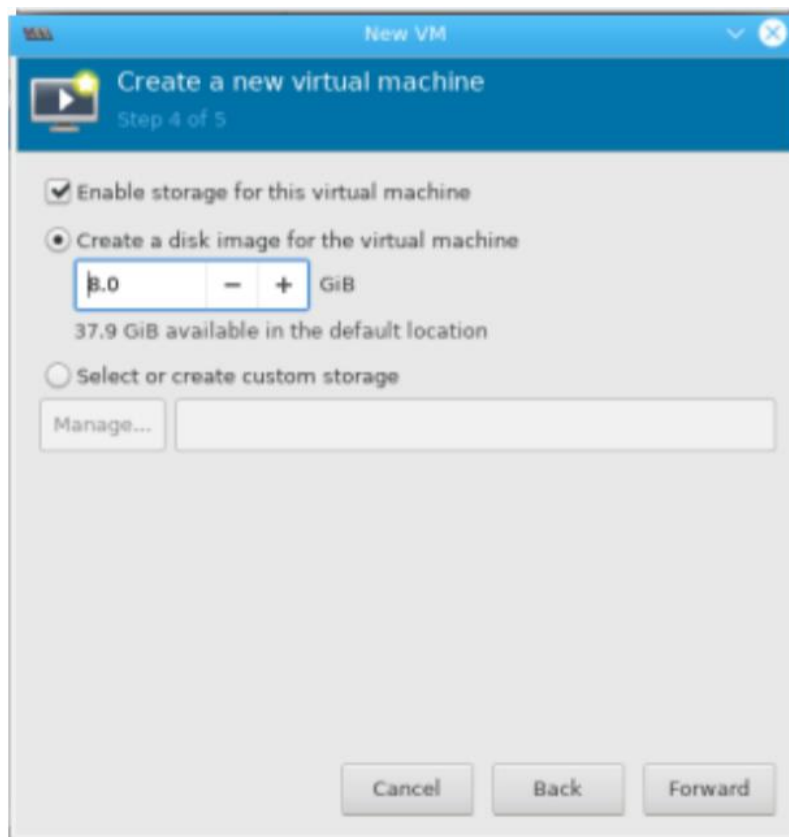
To install and configure Linux operating systems in a Virtual Machine.

Installation/Configuration Steps:

1. Install the required packages for virtualization
`dnf install xen virt-manager qemu libvirt`
2. Configure xend to startup on boot
`systemctl enable virt-manager.service`
3. Reboot the machine.
4. Create a virtual machine by first running virt-manager.
5. Click on File and then click to connect to localhost.
6. In the base menu, right click on to localhost(QEMU) to create new VM
7. Select Linux IOS image.
8. Choose puppy-linux ios and then kernel version
9. Select CPU and RAM limits
10. Create default disk to 8GB
11. Click create for the new VM with PuppyLinux

Output:





Result:

Hence the Linux installation has been studied successfully.

Ex. No. : 1b

Date : 01.02.2025

Register No. : 230701385

Name : VISHWAK S

BASIC LINUX COMMANDS

1.1 GENERAL PURPOSE COMMANDS

1. The 'date' command:

The date command displays the current date with day of week, month, day, time (24 hours clock) and the year.

SYNTAX: \$ date

The date command can also be used with the following format:

Format	Purpose	Example
+ %m	To display only month	\$ date + %m
+ %h	To display month name	\$ date + %h
+ %d	To display day of month	\$ date + %d
+ %y	To display last two digits of the year	\$ date + %y
+ %H	To display Hours	\$ date + %H
+ %M	To display Minutes	\$ date + %M
+ %S	To display Seconds	\$ date + %S

```
student@localhost ~$ date +%m
02
student@localhost ~$ date +%h
Feb
student@localhost ~$ date +%d
14
student@localhost ~$ date +%y
25
student@localhost ~$ date +%H
14
student@localhost ~$ date +%M
26
student@localhost ~$ date +%s
1739523407
student@localhost ~$ date +%S
51
```

2. The echo command:

The echo command is used to print the message on the screen. SYNTAX: \$ echo

EXAMPLE: \$ echo "God is Great"

A terminal window titled 'student : bash — Konsole' with a menu bar (File, Edit, View, Bookmarks, Settings, Help). The prompt is '[student@localhost ~]\$' and the command 'echo God is Great' has been entered. The output 'God is Great' is displayed on the next line.

```
[student@localhost ~]$ echo God is Great
God is Great
```

3. The 'cal' command:

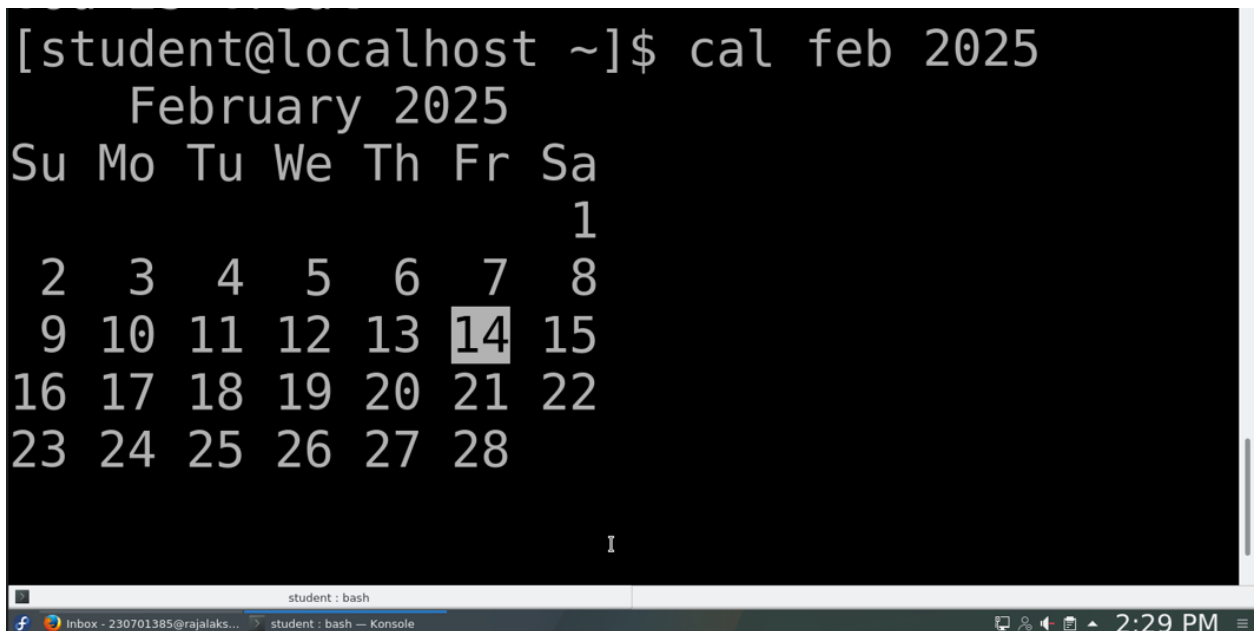
The cal command displays the specified month or year calendar.

SYNTAX: \$ cal

[month] [year]

EXAMPLE: \$ cal Jan

2012

A terminal window titled 'student : bash' with a taskbar at the bottom showing an inbox icon and the time 2:29 PM. The prompt is '[student@localhost ~]\$' and the command 'cal feb 2025' has been entered. The output shows a calendar for February 2025 with days of the week as headers and the 14th highlighted.

```
[student@localhost ~]$ cal feb 2025
February 2025
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

I
```


4. The 'bc' command:

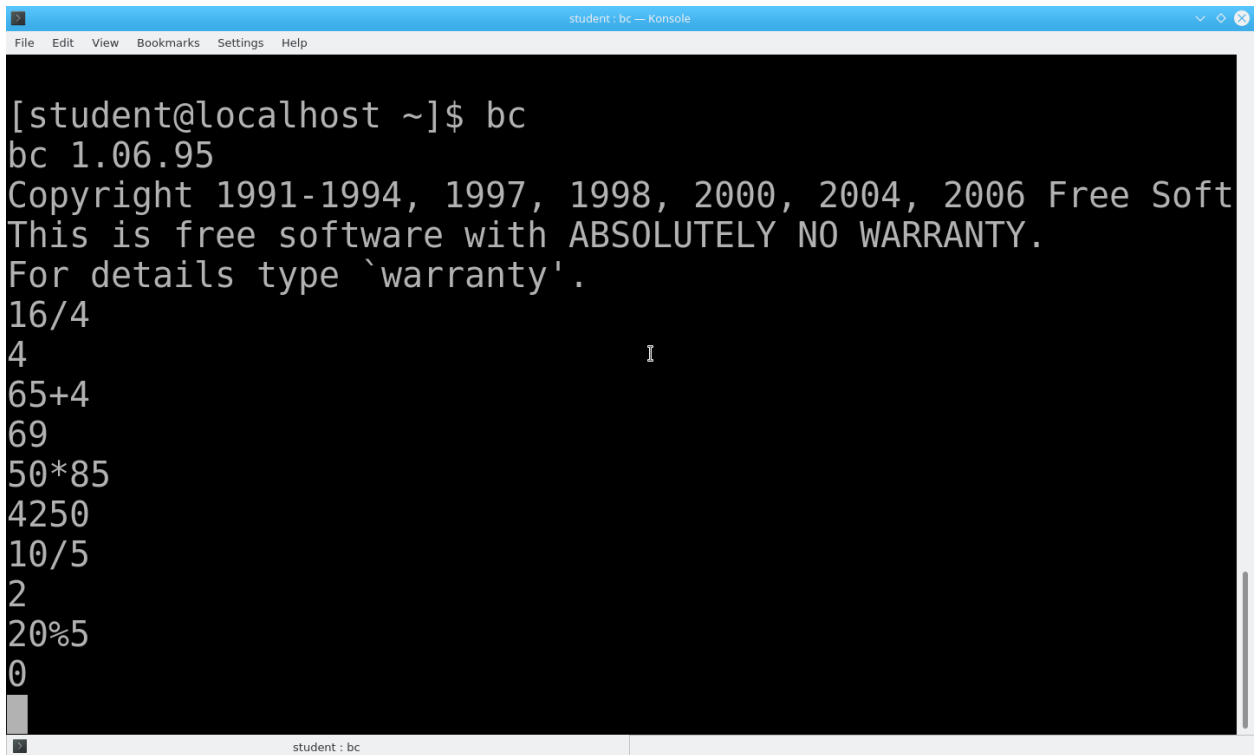
Unix offers an online calculator and can be invoked by the command bc.

SYNTAX: \$ bc

EXAMPLE: bc -l

16/4

5/2



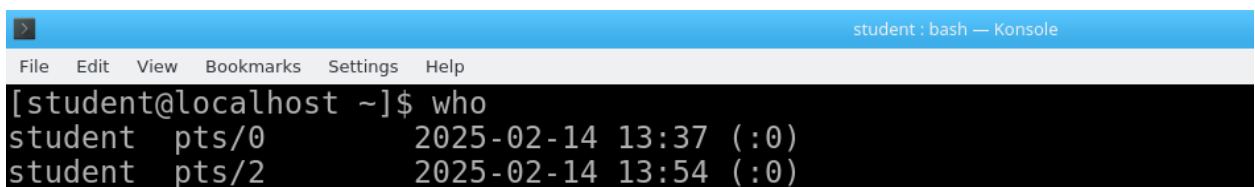
```
student : bc — Konsole
File Edit View Bookmarks Settings Help

[student@localhost ~]$ bc
bc 1.06.95
Copyright 1991-1994, 1997, 1998, 2000, 2004, 2006 Free Soft
This is free software with ABSOLUTELY NO WARRANTY.
For details type `warranty'.
16/4
4
65+4
69
50*85
4250
10/5
2
20%5
0
```

5. The 'who' command

The who command is used to display the data about all the users who are currently logged into the system.

SYNTAX: \$ who



```
student : bash — Konsole
File Edit View Bookmarks Settings Help

[student@localhost ~]$ who
student pts/0      2025-02-14 13:37 (:0)
student pts/2      2025-02-14 13:54 (:0)
```

6. The 'who am i' command

The who am i command displays data about login details of the user.

SYNTAX: \$ who am i

```
[student@localhost ~]$ who am i
student pts/2      2025-02-14 13:54 (:0)
```

7. The 'id' command

The id command displays the numerical value corresponding to your login.

SYNTAX: \$ id

```
[student@localhost ~]$ id
uid=1000(student) gid=1000(student) groups=1000(student) context=unconfined_u:unconfined_r:unconfined
t:s0-s0:c0.c1023
```

8. The 'tty' command

The tty (teletype) command is used to know the terminal name that we are using.

SYNTAX: \$ tty

```
[student@localhost ~]$ tty
/dev/pts/2
```

9. The 'ps' command

The ps command is used to the process currently alive in the machine with the 'ps' (process status) command, which displays information about process that are alive when you run the command. 'ps;' produces a snapshot of machine activity.

SYNTAX: \$ ps

EXAMPLE: \$ ps \$ ps -e \$ps -aux

```
[student@localhost ~]$ ps
  PID TTY          TIME CMD
 1711 pts/2        00:00:00 bash
 2004 pts/2        00:00:00 ps
```

10. The 'uname' command

The uname command is used to display relevant details about the operating system on the standard output.

-m -> Displays the machine id (i.e., name of the system hardware)

-n -> Displays the name of the network node. (host name)

-r -> Displays the release number of the operating system.

-s -> Displays the name of the operating system (i.e.. system name)

-v -> Displays the version of the operating system.

-a -> Displays the details of all the above five options.

SYNTAX: \$ uname

[option] EXAMPLE:

\$ uname -a

```
[student@localhost ~]$ uname  
Linux
```

1.2 DIRECTORY COMMANDS

1. The 'pwd' command:

The pwd (print working directory) command displays the current working directory.

SYNTAX: \$ pwd

2. The 'mkdir' command:

The mkdir is used to create an empty directory in a disk.

SYNTAX: \$ mkdir

dirname

EXAMPLE: \$ mkdir receee

3. The 'rmdir' command:

The rmdir is used to remove a directory from the disk. Before removing a directory, the directory must be empty (no files and directories).

SYNTAX: `$ rmdir dirname`

EXAMPLE: `$ rmdir receee`

4. The 'cd' command:

The cd command is used to move from one directory to another.

SYNTAX: `$ cd dirname`

EXAMPLE: `$ cd receee`

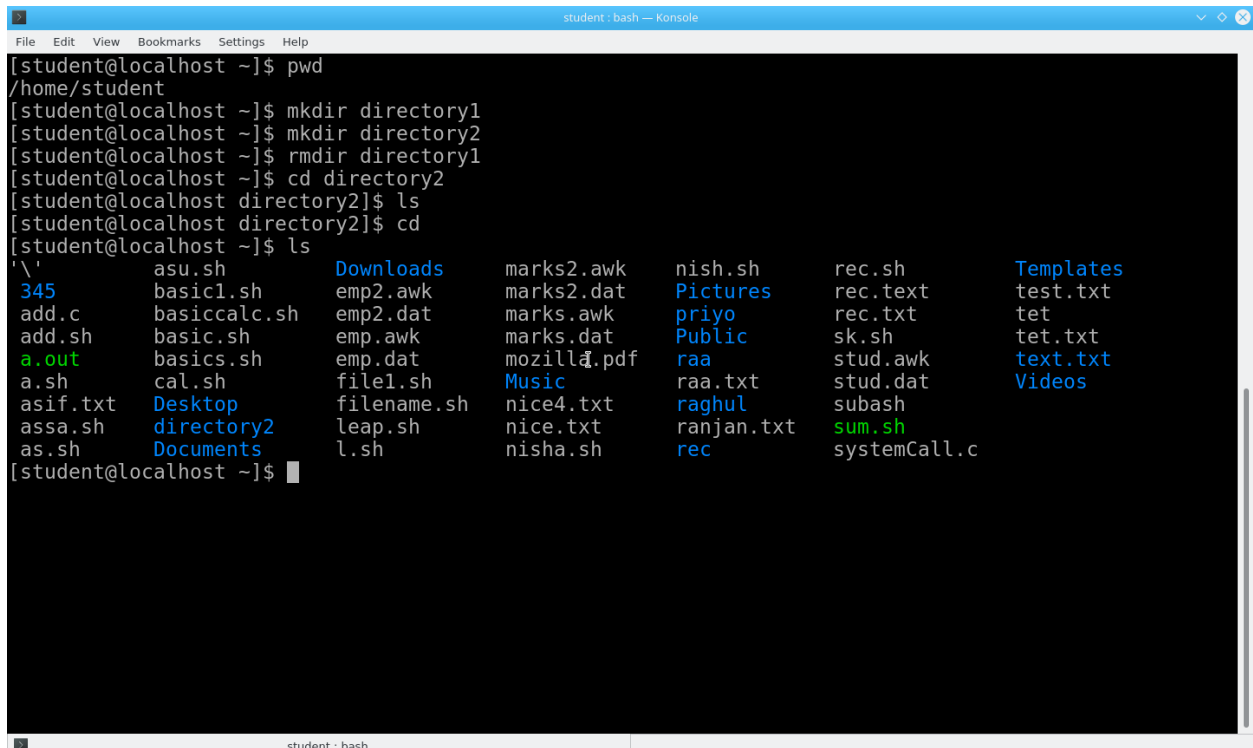
5. The 'ls' command:

The ls command displays the list of files in the current working directory.

SYNTAX: `$ ls` EXAMPLE: `$ ls`

`$ ls -l`

`$ ls -a`



```
student : bash -- Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ pwd
/home/student
[student@localhost ~]$ mkdir directory1
[student@localhost ~]$ mkdir directory2
[student@localhost ~]$ rmdir directory1
[student@localhost ~]$ cd directory2
[student@localhost directory2]$ ls
[student@localhost directory2]$ cd
[student@localhost ~]$ ls
'\          asu.sh          Downloads  marks2.awk  nish.sh     rec.sh      Templates
345        basic1.sh       emp2.awk  marks2.dat  Pictures    rec.text    test.txt
add.c      basiccalc.sh  emp2.dat  marks.awk   priyo       rec.txt     tet
add.sh     basic.sh      emp.awk   marks.dat   Public      sk.sh       tet.txt
a.out      basics.sh     emp.dat   mozill4.pdf raa         stud.awk    text.txt
a.sh       cal.sh        file1.sh  Music       raa.txt     stud.dat    Videos
asif.txt   Desktop      filename.sh nice4.txt   raghul      subash
assa.sh    directory2   leap.sh   nice.txt    ranjan.txt  sum.sh
as.sh      Documents    l.sh      nisha.sh   rec         systemCall.c
```

1.3 FILE HANDLING COMMANDS

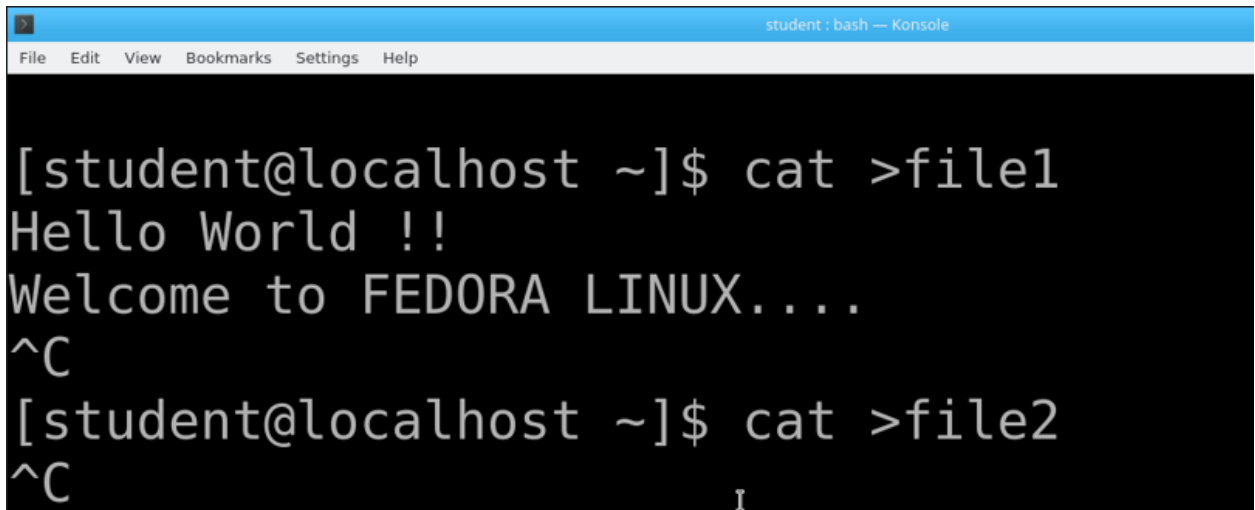
1. The 'cat' command:

The cat command is used to create a file.

SYNTAX: `$ cat >`

filename EXAMPLE:

`$ cat > rec`



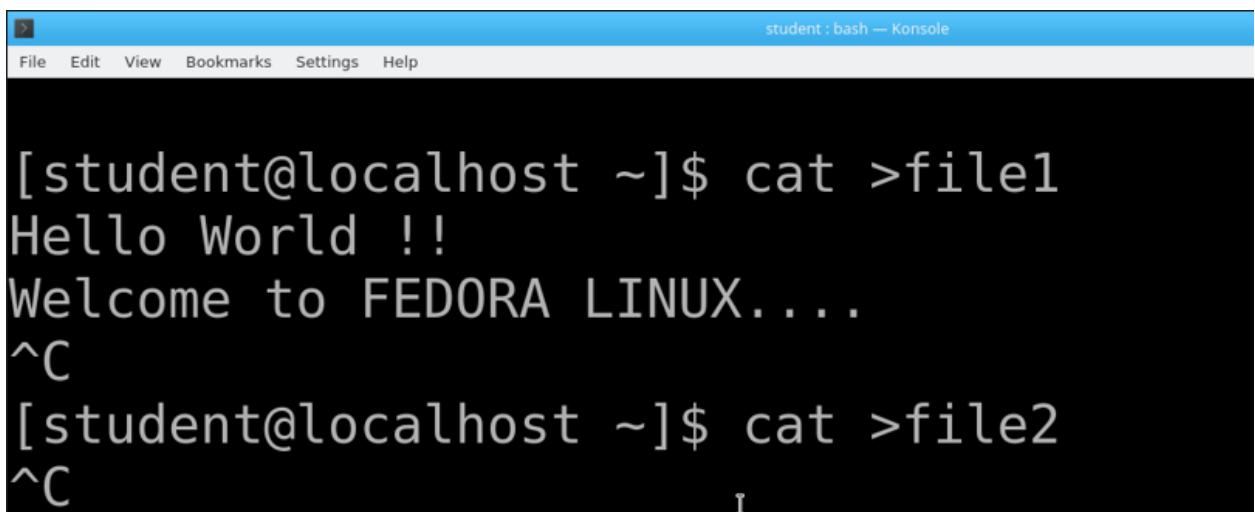
```
student : bash — Konsole
File Edit View Bookmarks Settings Help

[student@localhost ~]$ cat >file1
Hello World !!
Welcome to FEDORA LINUX....
^C
[student@localhost ~]$ cat >file2
^C
```

2. The 'Display contents of a file' command:

The cat command is also used to view the contents of a specified file.

SYNTAX: `$ cat filename`



```
student : bash — Konsole
File Edit View Bookmarks Settings Help

[student@localhost ~]$ cat >file1
Hello World !!
Welcome to FEDORA LINUX....
^C
[student@localhost ~]$ cat >file2
^C
```

3. The 'cp' command:

The cp command is used to copy the contents of one file to another and copies the file from one place to another.

SYNTAX: \$ cp oldfile

newfile EXAMPLE: \$

cp cse ece

```
[student@localhost ~]$ cp file1 file2
[student@localhost ~]$ cat file2
Hello World !!
Welcome to FEDORA LINUX....
```

4. The 'rm' command:

The rm command is used to remove or erase an existing file

SYNTAX: \$ rm filename

EXAMPLE: \$ rm rec

\$ rm -f rec

Use option -fr to delete recursively the contents of the directory and its subdirectories.

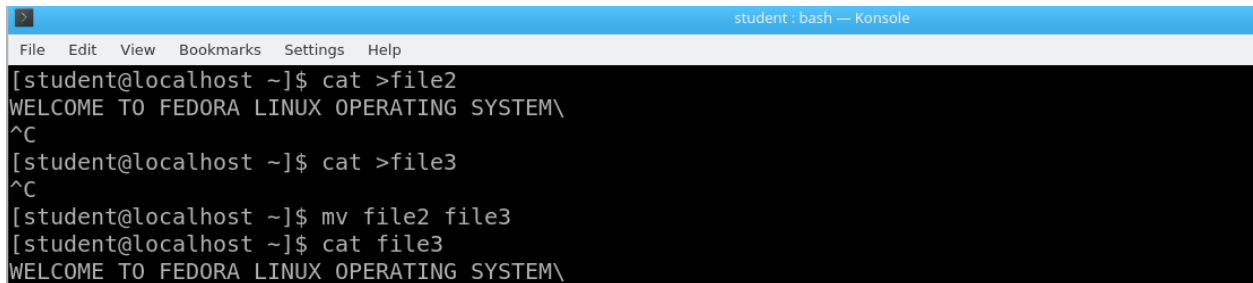
```
[student@localhost ~]$ rm file1
[student@localhost ~]$ cat file1
cat: file1: No such file or directory
```

5. The 'mv' command:

The mv command is used to move a file from one place to another. It removes a specified file from its original location and places it in a specified location.

SYNTAX: `$ mv oldfile newfile`

EXAMPLE: `$ mv cse eee`

A terminal window titled 'student : bash — Konsole' with a menu bar (File, Edit, View, Bookmarks, Settings, Help). The terminal shows the following commands and output:

```
[student@localhost ~]$ cat >file2
WELCOME TO FEDORA LINUX OPERATING SYSTEM\
^C
[student@localhost ~]$ cat >file3
^C
[student@localhost ~]$ mv file2 file3
[student@localhost ~]$ cat file3
WELCOME TO FEDORA LINUX OPERATING SYSTEM\
```

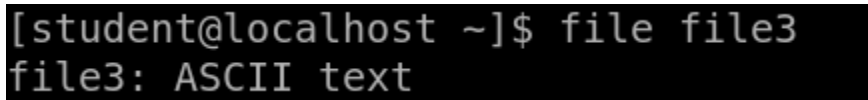
6. The 'file' command:

The file command is used to determine the type of file.

SYNTAX: `$ file`

filename EXAMPLE:

`$ file receee`

A terminal window showing the command and output:

```
[student@localhost ~]$ file file3
file3: ASCII text
```

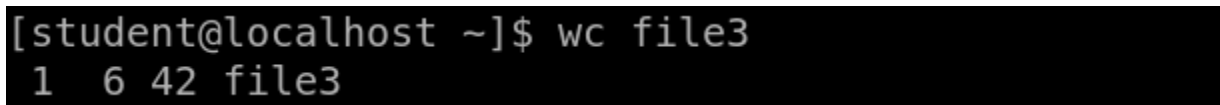
7. The 'wc' command:

The wc command is used to count the number of words, lines and characters in a file.

SYNTAX: `$ wc`

filename EXAMPLE:

`$ wc receee`

A terminal window showing the command and output:

```
[student@localhost ~]$ wc file3
1 6 42 file3
```

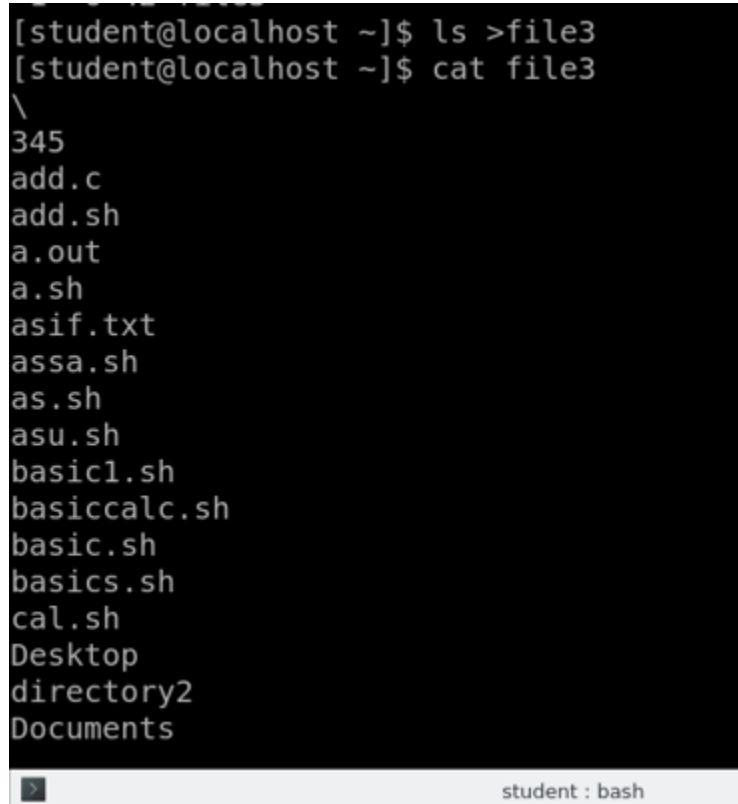
8. The 'Directing output to a file' command:

The `ls` command lists the files on the terminal (screen). Using the redirection operator '`>`' we can send the output to the file instead of showing it on the screen.

SYNTAX: `$ ls >`

filename EXAMPLE:

`$ ls > cseeee`

A terminal window with a black background and white text. The prompt is [student@localhost ~]. The first command is \$ ls >file3. The second command is \$ cat file3. The output of the cat command is a list of files and directories: \, 345, add.c, add.sh, a.out, a.sh, asif.txt, assa.sh, as.sh, asu.sh, basic1.sh, basiccalc.sh, basic.sh, basics.sh, cal.sh, Desktop, directory2, and Documents. At the bottom of the terminal window, there is a status bar that says "student : bash".

```
[student@localhost ~]$ ls >file3
[student@localhost ~]$ cat file3
\
345
add.c
add.sh
a.out
a.sh
asif.txt
assa.sh
as.sh
asu.sh
basic1.sh
basiccalc.sh
basic.sh
basics.sh
cal.sh
Desktop
directory2
Documents
```

9. The 'pipes' command:

The Unix allows us to connect two commands together using these pipes. A pipe (`|`) is a mechanism by which the output of one command can be channeled into the input of another command.

SYNTAX: `$`

command1 `|`

command2

EXAMPLE: `$ who |`

`wc -l`

10. The 'tee' command:

While using pipes, we have not seen any output from a command that gets piped into another command. To save the output, which is produced in the middle of a pipe, the tee command is very useful. SYNTAX: `$ command | tee filename`

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

11. The 'Metacharacters of unix' command:

Metacharacters are special characters that are at a higher and abstract level compared to most other characters in Unix. The shell understands and interprets these metacharacters in a special way.

* - Specifies number of characters

?- Specifies a single character

[]- used to match a whole set of file names at a command line.

! – Used to Specify Not

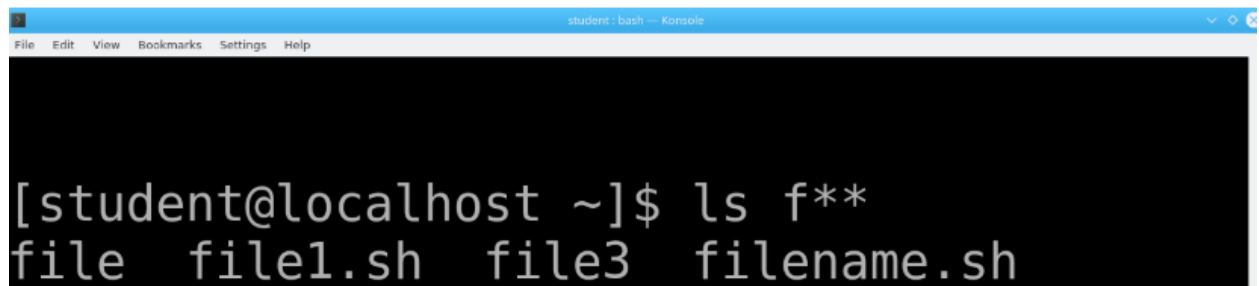
EXAMPLE:

`$ ls r**` - Displays all the files whose name begins with 'r'

`$ ls ?kkk` - Displays the files which are having 'kkk', from the second character irrespective of the first character.

`$ ls [a-m]` – Lists the files whose names begins alphabets from 'a' to 'm'

`$ ls [!a-m]` – Lists all files other than files whose names begin with alphabets from 'a' to 'm' 12.



```
student : bash -- Konsole
File Edit View Bookmarks Settings Help

[student@localhost ~]$ ls f**
file  file1.sh  file3  filename.sh
```

The 'File permissions' command:

File permission is the way of controlling the accessibility of files for each of three users namely Users, Groups and Others.

There are three types of file permissions are available, they are

r-read

w-write

x-execute

The permissions for each file can be divided into three parts of three bits each:

First three bits	Owner of the file
Next three bits	Group to which owner of the file belongs
Last three bits	Others

EXAMPLE: \$ ls college

-rwxr-xr-- 1 Lak std 1525 jan10 12:10 college

-rwx The file is readable, writable and executable by the owner of the file.

Lak Specifies Owner of the file.

r-x Indicates the absence of the write permission by the Group owner of the file. Std Is the Group Owner of the file.

r-- Indicates read permissions for others.

```
[student@localhost ~]$ ls >file3
[student@localhost ~]$ cat file3
\
345
add.c
add.sh
a.out
a.sh
asif.txt
assa.sh
as.sh
asu.sh
basic1.sh
basiccalc.sh
```

13. The 'chmod' command:

The chmod command is used to set the read, write and execute permissions for all categories of users for file.

SYNTAX: `$ chmod category operation permission file`

EXAMPLE:

`$ chmod u -wx college`

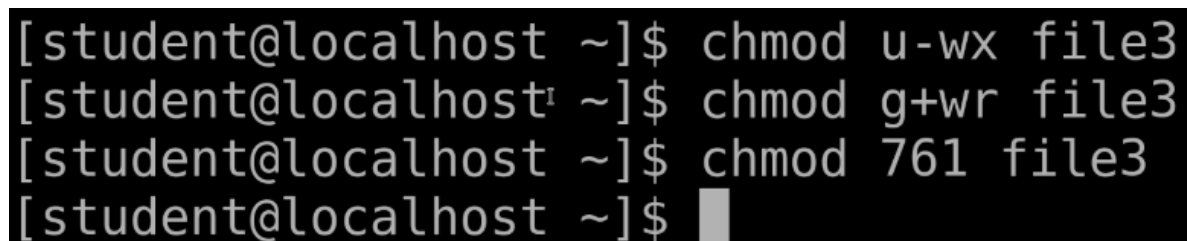
Removes write & execute permission for users for 'college' file.

`$ chmod u +rw, g+rw college`

Assigns read & write permission for users and groups for 'college' file.

`$ chmod g=wx college`

Assigns absolute permission for groups of all read, write and execute permissions for 'college' file.

A terminal window with a black background and white text. It shows four lines of commands being executed by a user named 'student' on a 'localhost' machine. The first line is '[student@localhost ~]\$ chmod u-wx file3'. The second line is '[student@localhost ~]\$ chmod g+wr file3'. The third line is '[student@localhost ~]\$ chmod 761 file3'. The fourth line is '[student@localhost ~]\$' followed by a cursor. The prompt character is a dollar sign (\$) and the tilde (~) represents the home directory.

```
[student@localhost ~]$ chmod u-wx file3
[student@localhost ~]$ chmod g+wr file3
[student@localhost ~]$ chmod 761 file3
[student@localhost ~]$
```

14. The 'Octal Notations' command:

The file permissions can be changed using octal notations also. The octal notations for file permission are

EXAMPLE:

`$ chmod 761 college`

Assigns all permission to the owner, read and write permissions to the group and only executable permission to the others for the 'college' file.

1.4 GROUPING COMMANDS

1. The 'semicolon' command:

The semicolon(;) command is used to separate multiple commands at the command line.

SYNTAX:

\$command1;command2;command3..... ;commandn

EXAMPLE: \$ who;date

2. The '&&' operator:

The '&&' operator signifies the logical AND operation in between two or more valid Unix commands.It means that only if the first command is successfully executed, then the next command will executed.

SYNTAX:

\$command && command && command3..... && commandn

EXAMPLE: \$ who && date

3. The '||' operator:

The '||' operator signifies the logical OR operation in between two or more valid Unix commands.It means, that only if the first command will happen to be un successfully,it will continue to execute next commands.

SYNTAX:

\$command1||command||command3..... ||commandn

EXAMPLE: \$ who || date

```
student : bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ who && date
student pts/0      2025-03-08 13:37 (:0)
student pts/1      2025-03-08 13:38 (:0)
Sat Mar  8 13:59:56 IST 2025
[student@localhost ~]$ who am i; date +%m
student pts/1      2025-03-08 13:38 (:0)
03
[student@localhost ~]$ date || who
Sat Mar  8 14:00:21 IST 2025
[student@localhost ~]$
```

1.5 FILTERS

1. The head filter

It displays the first ten lines of a file.

SYNTAX: `$ head filename`

EXAMPLE: `$ head college` Display the top ten lines.

`$ head -5 college` Display the top five lines.

```
student : bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ head company
TCS
Tech Mahindra
Infosys
IBM
Google
Microsoft
Oracle
Accenture
SAP
Cisco
```

2. The tail filter

It displays ten lines of a file from the end of the file.

SYNTAX: `$ tail filename`

EXAMPLE: `$ tail college` Display the last ten lines.

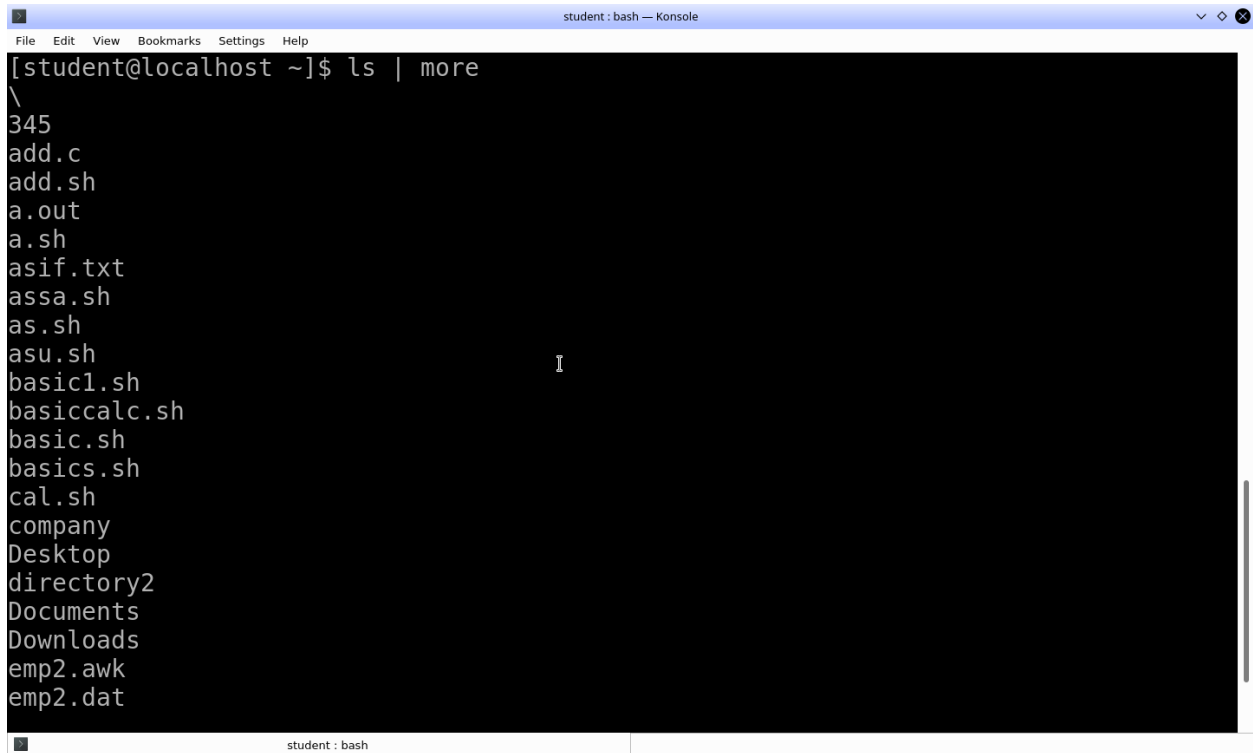
`$tail -5 college` Display the last five lines.

```
[student@localhost ~]$ tail company
Microsoft
Oracle
Accenture
SAP
Cisco
Adobe
Wipro
Zoho
Amazaon
Walmart
```

3. The more filter:

The pg command shows the file page by page.

SYNTAX: `$ ls -l | more`



```
student : bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ ls | more
\
345
add.c
add.sh
a.out
a.sh
asif.txt
assa.sh
as.sh
asu.sh
basic1.sh
basiccalc.sh
basic.sh
basics.sh
cal.sh
company
Desktop
directory2
Documents
Downloads
emp2.awk
emp2.dat
```

4. The 'grep' command:

This command is used to search for a particular pattern from a file or from the standard input and display those lines on the standard output. "Grep" stands for "global search for regular expression."

SYNTAX: `$ grep [pattern] [file_name]`

EXAMPLE: `$ cat > student`

Arun cse

Ram ece

Kani cse

`$ grep "cse" student`

Arun cse

Kani cse

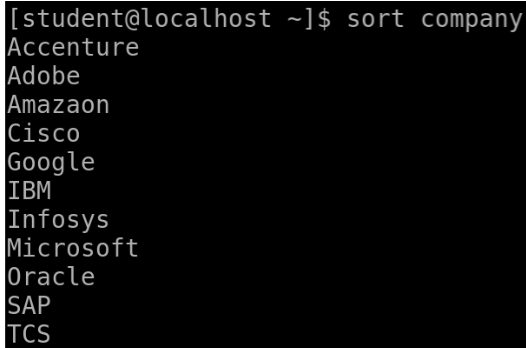


```
student: bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ cat > rec1
Ram cse
Arun ece
Venkat mech
Babu cse
Smith cse
^C
[student@localhost ~]$ grep "cse" rec1
Ram cse
Babu cse
Smith cse
```


5. The 'sort' command:

The sort command is used to sort the contents of a file. The sort command reports only to the screen, the actual file remains unchanged.

SYNTAX: `$ sort filename`



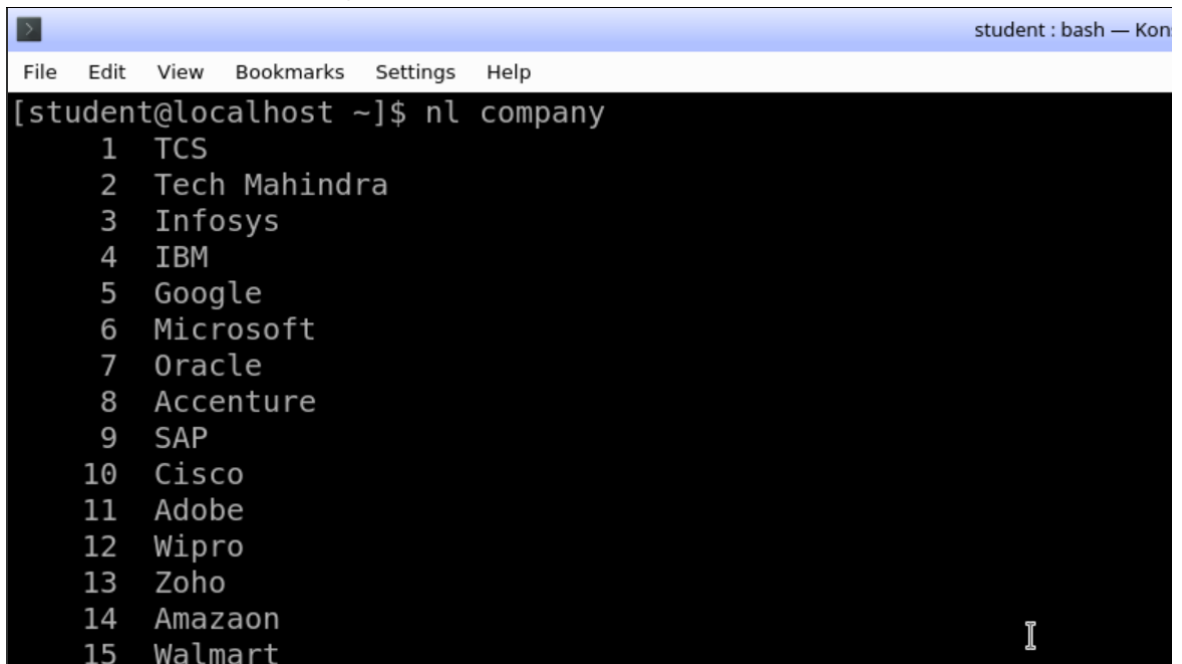
```
[student@localhost ~]$ sort company
Accenture
Adobe
Amazaon
Cisco
Google
IBM
Infosys
Microsoft
Oracle
SAP
TCS
```

6. The 'nl' command:

The nl filter adds line numbers to a file and it displays the file and not provides a to edit but simply displays the contents on the screen.

SYNTAX: `$ nl filename`

EXAMPLE: `$ nl college`



```
student : bash — Kon
File Edit View Bookmarks Settings Help
[student@localhost ~]$ nl company
 1 TCS
 2 Tech Mahindra
 3 Infosys
 4 IBM
 5 Google
 6 Microsoft
 7 Oracle
 8 Accenture
 9 SAP
10 Cisco
11 Adobe
12 Wipro
13 Zoho
14 Amazaon
15 Walmart
```

7. The 'cut' command:

We can select specific fields from a line of text using the cut command.

SYNTAX: `$ cut -c filename`

EXAMPLE: `$ cut -c college`

OPTION:

`-c` – Option cut on the specified character position from each line.

```
[student@localhost ~]$ cut -c -3 company
TCS
Tec
Inf
IBM
Goo
Mic
Ora
Acc
SAP
Cis
Ado
Wip
Zoh
Ama
Wal
```

1.6 OTHER ESSENTIAL COMMANDS

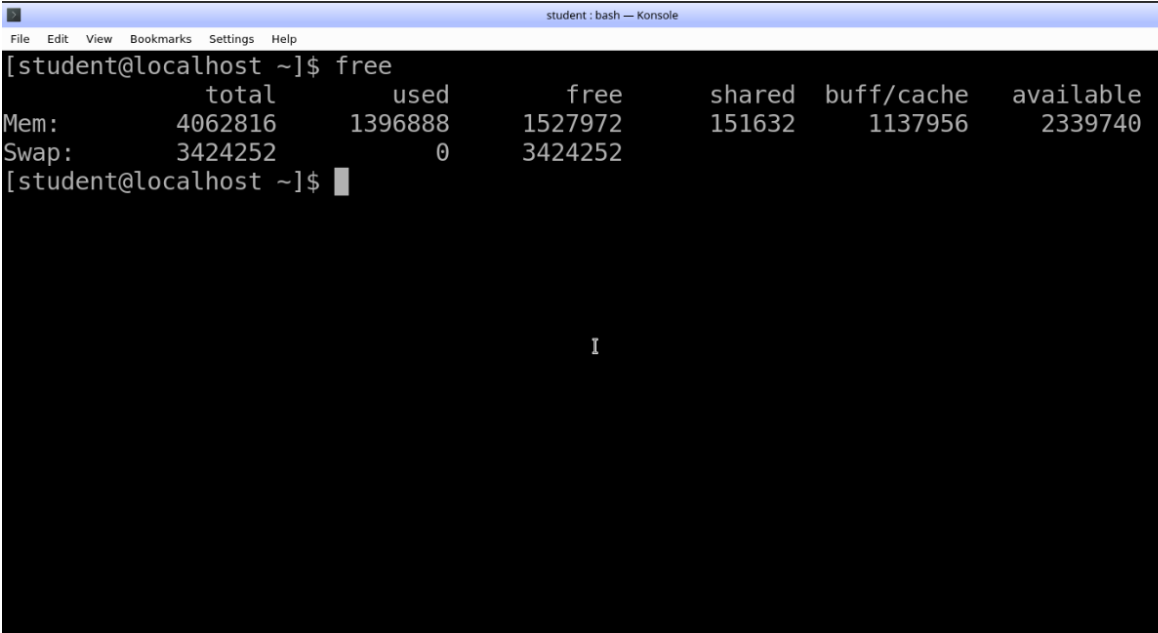
1. free

Display amount of free and used physical and swapped memory system. syno

free [options]

example

[root@localhost ~]# free -t



The screenshot shows a terminal window titled "student : bash — Konsole". The user is logged in as "student" at "localhost". The command "free" has been executed, displaying memory usage statistics. The output is as follows:

```
[student@localhost ~]$ free
```

	total	used	free	shared	buff/cache	available
Mem:	4062816	1396888	1527972	151632	1137956	2339740
Swap:	3424252	0	3424252			

The prompt returns to [student@localhost ~]\$.

2. top

It provides a dynamic real-time view of processes in the system.

synopsis- top [options]

example

[root@localhost ~]# top

```
student: top — Konsole
File Edit View Bookmarks Settings Help
top - 14:10:42 up 33 min, 2 users, load average: 0.43, 0.56, 0.45
Tasks: 166 total, 1 running, 165 sleeping, 0 stopped, 0 zombie
%Cpu(s): 6.9 us, 1.3 sy, 0.0 ni, 91.2 id, 0.0 wa, 0.3 hi, 0.2 si, 0.0 st
KiB Mem : 4062816 total, 1475260 free, 1438212 used, 1149344 buff/cache
KiB Swap: 3424252 total, 3424252 free, 0 used, 2287016 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 1114 student    20   0 360500  99984  64464 S   5.0   2.5   0:17.04 kwin x11
   867 root        20   0 162620  62308  37352 S   2.7   1.5   0:21.00 Xorg
 1749 student    20   0 1560112 859552 138016 S   2.7  21.2   5:57.47 Web Content
 1925 student    20   0 160968  59000  46376 S   2.3   1.5   0:03.94 spectacle
 1572 student    20   0 173784  57528  49612 S   2.0   1.4   0:05.43 konsole
 1143 student    20   0 949604 158504  90148 S   1.7   3.9   0:12.23 plasmashell
 1641 student    20   0 931280 303332 125812 S   0.7   7.5   2:11.43 firefox
 1140 student    20   0 202152  63636  51308 S   0.3   1.6   0:01.11 krunner
 1144 student    20   0 156576  34880  31896 S   0.3   0.9   0:00.24 polkit-kde-auth
 1235 student    20   0 174732  35064  31776 S   0.3   0.9   0:00.30 kactivitymanage
 1259 student    20   0 30076   6656   6164 S   0.3   0.2   0:00.23 at-spi2-registr
 1391 student    20   0 345264  66472  55028 S   0.3   1.6   0:00.64 akonadi_archive
 1395 student    20   0 151556  39432  36128 S   0.3   1.0   0:00.37 akonadi_ical_re
2013 student    20   0 16940   4004  3620 R   0.3   0.1   0:00.04 top
    1 root        20   0 32268  10412  8156 S   0.0   0.3   0:01.31 systemd
    2 root        20   0      0      0      0 S   0.0   0.0   0:00.00 kthreadd
    4 root        0 -20      0      0      0 S   0.0   0.0   0:00.00 kworker/0:0H
    6 root        0 -20      0      0      0 S   0.0   0.0   0:00.00 mm_percpu_wq
    7 root        20   0      0      0      0 S   0.0   0.0   0:00.02 ksoftirqd/0
    8 root        20   0      0      0      0 S   0.0   0.0   0:00.48 rcu_sched
    9 root        20   0      0      0      0 S   0.0   0.0   0:00.00 rcu_bh
   10 root        rt   0      0      0      0 S   0.0   0.0   0:00.00 migration/0
   11 root        rt   0      0      0      0 S   0.0   0.0   0:00.00 watchdog/0
   12 root        20   0      0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
   13 root        20   0      0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
   14 root        rt   0      0      0      0 S   0.0   0.0   0:00.00 watchdog/1
   15 root        rt   0      0      0      0 S   0.0   0.0   0:00.08 migration/1
   16 root        20   0      0      0      0 S   0.0   0.0   0:00.03 ksoftirqd/1
   18 root        0 -20      0      0      0 S   0.0   0.0   0:00.00 kworker/1:0H
   19 root        20   0      0      0      0 S   0.0   0.0   0:00.00 kdevtmpfs
   20 root        0 -20      0      0      0 S   0.0   0.0   0:00.00 netns
   23 root        20   0      0      0      0 S   0.0   0.0   0:00.00 oom_reaper
   24 root        0 -20      0      0      0 S   0.0   0.0   0:00.00 writeback
   25 root        20   0      0      0      0 S   0.0   0.0   0:00.00 kcompactd0

student: top
```

3. ps

It reports the snapshot of current processes.

synopsis- ps [options]

example

[root@localhost ~]# ps -e

```
student : bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ ps
  PID TTY          TIME CMD
 1576 pts/1    00:00:00 bash
 2018 pts/1    00:00:00 ps
```

4. vmstat

It reports virtual memory statistics.

synopsis- vmstat [options] example

[root@localhost ~]# vmstat

```
[student@localhost ~]$ vmstat
procs -----memory----- --swap-- -----io---- -system-- -----cpu-----
 r  b   swpd   free   buff   cache   si   so    bi   bo    in   cs us sy id wa st
 1   0       0 1481292 76780 1082632    0    0   194   66   630  961 14  2 84  0  0
```

5. df

It displays the amount of disk space available in file-system.

Synopsis- df [options]

example [root@localhost ~]# df

```
[student@localhost ~]$ df
Filesystem            1K-blocks    Used Available Use% Mounted on
devtmpfs               2020420         0   2020420   0% /dev
tmpfs                  2031408    41176   1990232   3% /dev/shm
tmpfs                  2031408     1208   2030200   1% /run
tmpfs                  2031408         0   2031408   0% /sys/fs/cgroup
/dev/mapper/fedora-root 44186572 5164500 36747768 13% /
tmpfs                  2031408        12   2031396   1% /tmp
/dev/sda6              999320    151196   779312   17% /boot
/dev/mapper/fedora-home 21567312 830380  19618316   5% /home
tmpfs                  406280         20    406260   1% /run/user/1000
```

6. ping

It is used to verify that a device can communicate with another on the network

PING stands for Packet Internet Groper. synopsis- ping [options]

[root@localhost ~]# ping 172.16.4.1

```
[student@localhost ~]$ ping
Usage: ping [-aAbBdDfhLnOqrRUvV64] [-c count] [-i interval] [-I interface]
        [-m mark] [-M pmtudisc_option] [-l preload] [-p pattern] [-Q tos]
        [-s packetsize] [-S sndbuf] [-t ttl] [-T timestamp_option]
        [-w deadline] [-W timeout] [hop1 ...] destination
Usage: ping -6 [-aAbBdDfhLnOqrRUvV] [-c count] [-i interval] [-I interface]
        [-l preload] [-m mark] [-M pmtudisc_option]
        [-N nodeinfo_option] [-p pattern] [-Q tclass] [-s packetsize]
        [-S sndbuf] [-t ttl] [-T timestamp_option] [-w deadline]
        [-W timeout] destination
```

7. ifconfig

It is used to configure network interfaces.

synopsis- ifconfig [options]

Example: [root@localhost ~]# ifconfig

```
[student@localhost ~]$ ifconfig
enp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.9.2 netmask 255.255.252.0 broadcast 172.16.11.255
    inet6 fe80::ed62:ec73:ec9f:3ad9 prefixlen 64 scopeid 0x20<link>
    ether 00:27:0e:13:ea:36 txqueuelen 1000 (Ethernet)
    RX packets 181394 bytes 112770604 (107.5 MiB)
    RX errors 0 dropped 43 overruns 0 frame 0
    TX packets 39784 bytes 21829577 (20.8 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
```

> student : bash

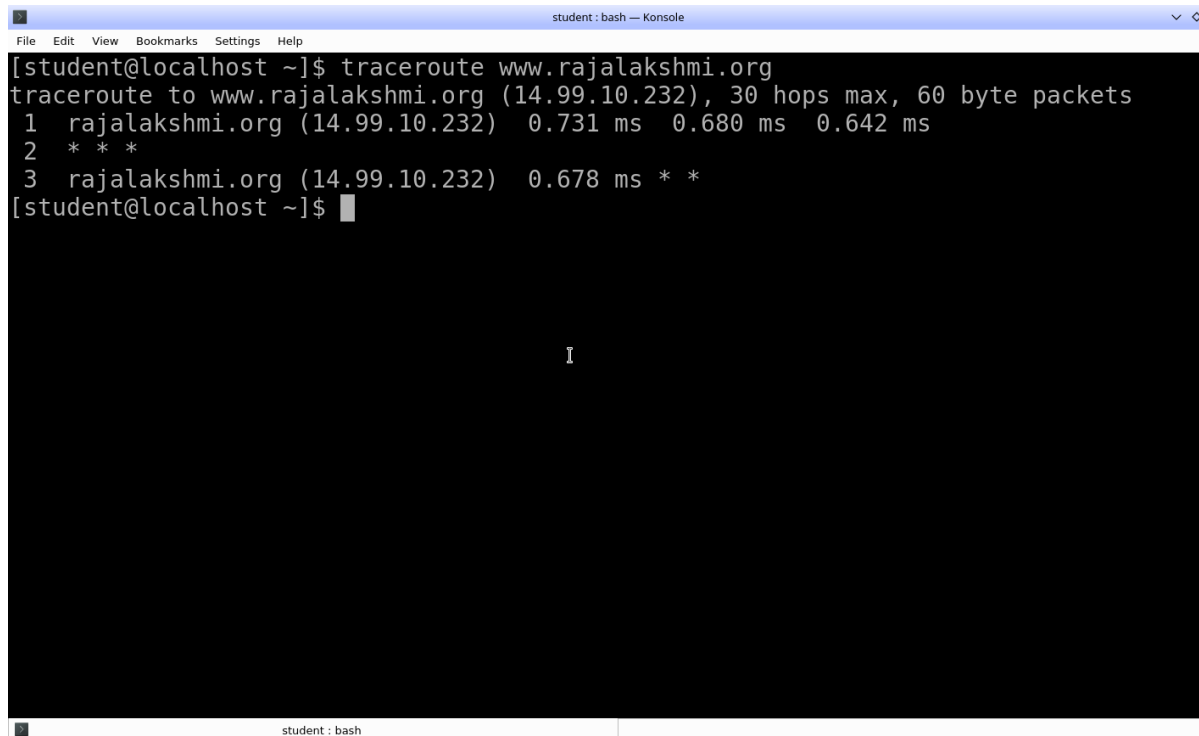
8.traceroute

It tracks the route the packet takes to reach the destination.

synopsis- traceroute [options]

Example

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



```
student : bash — Konsole
File Edit View Bookmarks Settings Help
[student@localhost ~]$ traceroute www.rajalakshmi.org
traceroute to www.rajalakshmi.org (14.99.10.232), 30 hops max, 60 byte packets
 1  rajalakshmi.org (14.99.10.232)  0.731 ms  0.680 ms  0.642 ms
 2  * * *
 3  rajalakshmi.org (14.99.10.232)  0.678 ms  * *
[student@localhost ~]$
```

Result:

Hence the basic LINUX commands, directory commands, file handling commands, grouping commands, filters and other essential commands have been studied and executed successfully.