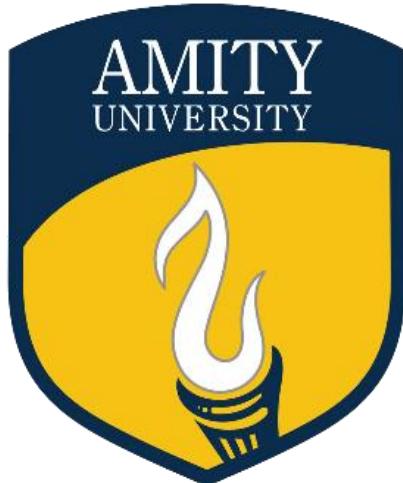


AMITY UNIVERSITY UTTAR PRADESH NOIDA
AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING



LINUX FOR DEVICES
[CSE 438]
PRACTICAL FILE

Submitted By:
Mihir Dadwal
A2305218162
B. TECH(CSE)
BATCH 2018-2022
7CSE-3X

Submitted To:
Ms Vibha Nehra

INDEX-

S No.	Name of Program	Date of Conduct	Maximum Marks	Marks Obtained	Signature of faculty
1.	Installation of Linux Operating System.	19/07/2021			
2.	Study of Unix/Linux general purpose utility command list obtained from (man, cat, cd, cp, ps, ls, mv, rm, mkdir, rmdir, date, time, chmod, pwd, cal) commands.	26/07/2021			
3.	Study of vi editor, Study of bash shell, bourne shell and C shell in Unix/Linux operating system.	02/08/2021			
4.	Write a shell script program to display “HELLO WORLD”,	09/08/2021			
5.	Write a shell script program to develop a scientific calculator.	09/08/2021			
6.	Install Docker and run : Start, stop, push, pull, log, docker ps, docker ps -a, create account, docker compose, docker build, docker run	16/08/2021			
7.	Write a shell script program to illustrate the implementation of following: a) Integer Comparison b) String comparison c) Logical operators d) File tests e) Conditional control structure f) Loop control structures	07/09/2021			

8.	<p>Write a shell script to</p> <ul style="list-style-type: none">a) print a number in reverse order.b)to reverse the string and reverse each string further in the listc)find the sum of all numbers in a file in Linuxd) validate password strength. Here are a few assumptions for the password string.Length – minimum of 8 characters,Contain both alphabet and number,Include both the small and capital case letters.	21/09/2021			
9.	Design and develop a “Birthday Reminder” that can automatically send birthday wishes with a personalized message via email.	05/10/2021			
10.	Check whether strings in csv are palindrome.	12/10/2021			
11.	Implement NIC Bonding and Teaming in Linux	19/10/2021			

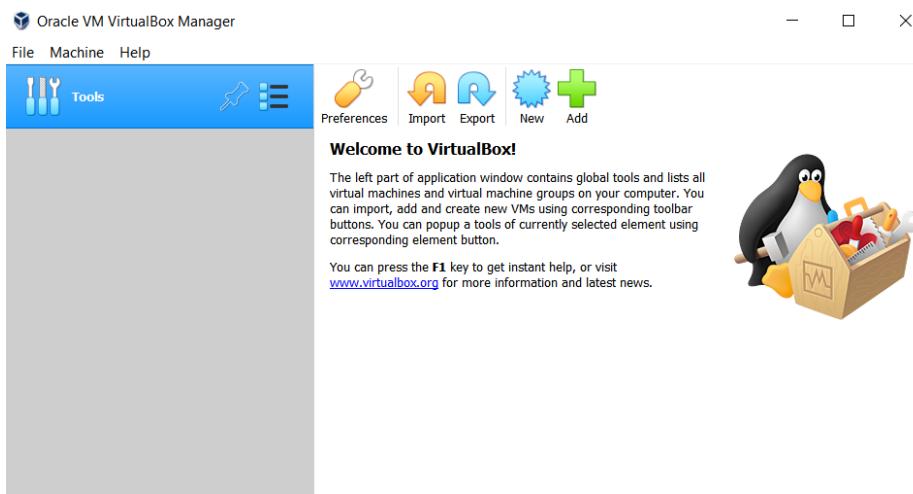
Practical-1

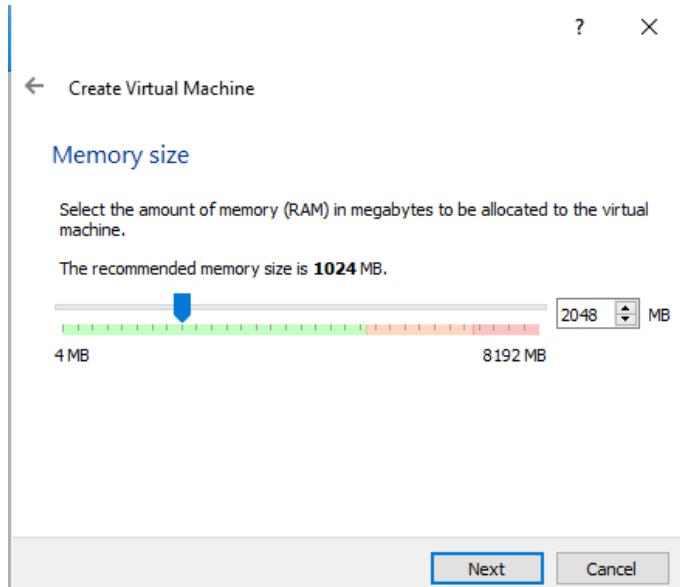
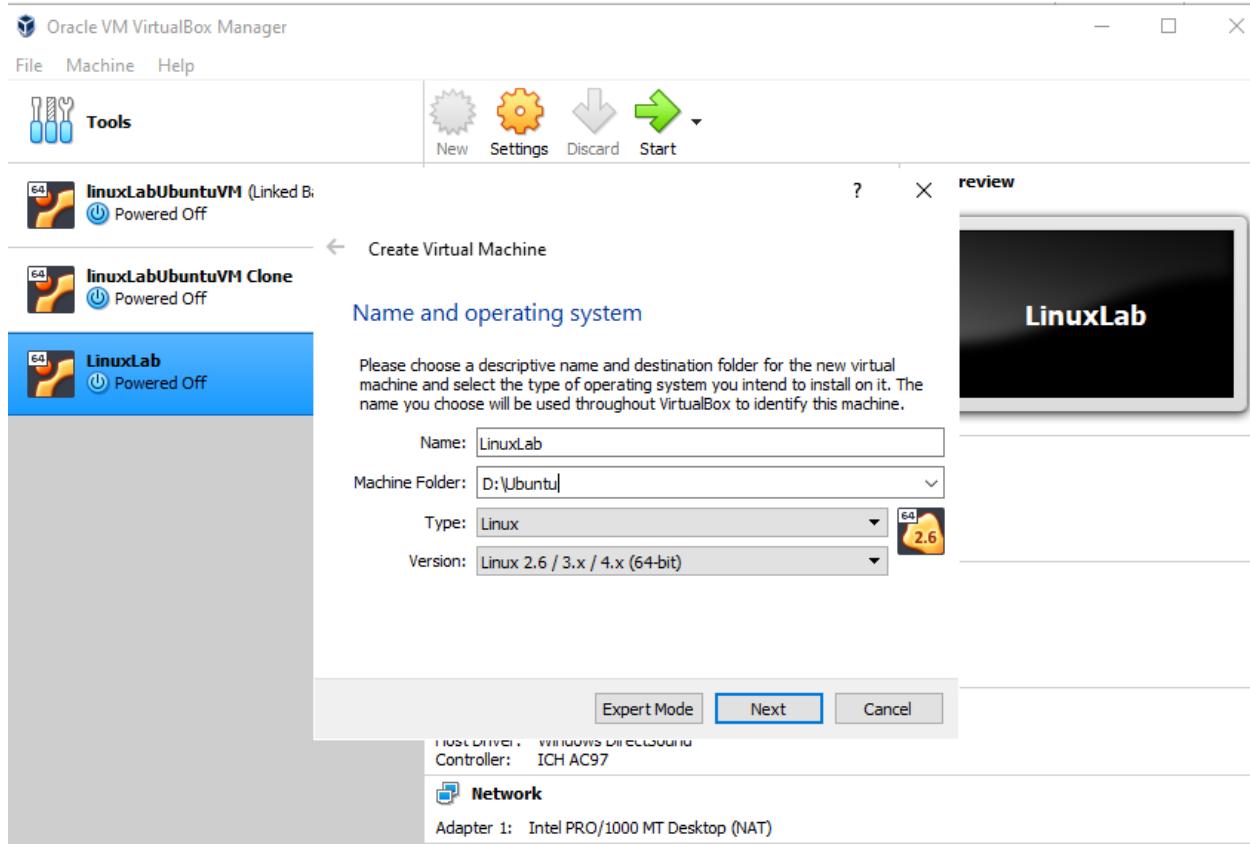
AIM- Installation of Linux Operating System.

THEORY-

- Linux is an open source operating system (OS). Linux was designed to be similar to UNIX, but has evolved to run on a wide variety of hardware from phones to supercomputers.
- Every Linux-based OS involves the Linux kernel—which manages hardware resources—and a set of software packages that make up the rest of the operating system. Linux makes very efficient **use** of the system's resources.
- Linux runs on a range of hardware, right from supercomputers to watches. You can give new life to your old and slow Windows system by installing a lightweight Linux system, or even run a NAS or media streamer using a particular distribution of Linux.
- Linux is perfect for everyday tasks like browsing, emailing, photo management, financial management, and much more.

PROCEDURE-





← Create Virtual Machine

Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **8.00 GB**.

- Do not add a virtual hard disk
- Create a virtual hard disk now
- Use an existing virtual hard disk file

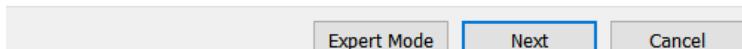


← Create Virtual Hard Disk

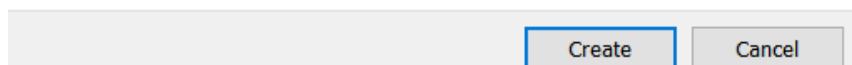
Hard disk file type

Please choose the type of file that you would like to use for the new virtual hard disk. If you do not need to use it with other virtualization software you can leave this setting unchanged.

- VDI (VirtualBox Disk Image)
- VHD (Virtual Hard Disk)
- VMDK (Virtual Machine Disk)



Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.



The screenshot shows the Oracle VM VirtualBox Manager interface. On the left, there is a list of existing virtual machines: "linuxLabUbuntuVM (Linked ...)" (Powered Off), "linuxLabUbuntuVM Clone" (Powered Off), and "LinuxLab" (Powered Off). The main pane displays the configuration for "linuxLabUbuntuVM".

General

- Name: linuxLabUbuntuVM
- Operating System: Ubuntu (64-bit)

System

- Base Memory: 2048 MB
- Processors: 5
- Boot Order: Floppy, Optical, Hard Disk
- Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization

Preview

A small window titled "Preview" showing a black screen with the white text "linuxLabUbuntuVM".

Display

- Video Memory: 16 MB
- Graphics Controller: VMSVGA
- Remote Desktop Server: Disabled
- Recording: Disabled

Storage

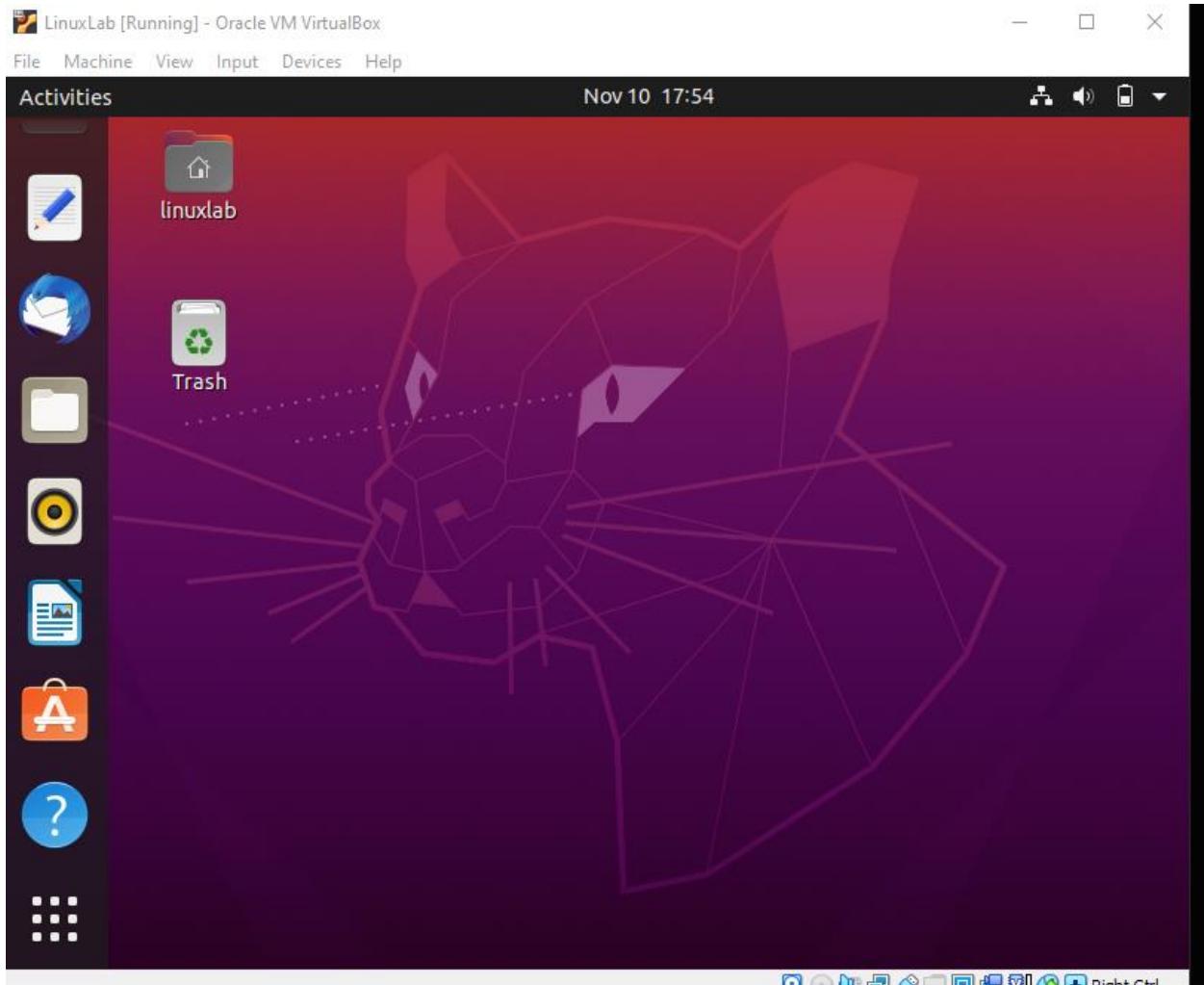
- Controller: IDE
- IDE Secondary Device 0: [Optical Drive] Empty
- Controller: SATA
- SATA Port 0: MihirUbuntuVM.vdi (Normal, 24.49 GB)

Audio

- Host Driver: Windows DirectSound
- Controller: ICH AC97

Network

- Adapter 1: Intel PRO/1000 MT Desktop (NAT)



CONCLUSION- Ubuntu has been successfully installed on my device.

Practical-2

AIM: Study of Unix/Linux general purpose utility command list obtained from (man, who, cat, cd, cp, ps, ls, mv, rm, mkdir, rmdir, echo, more, date, time, kill, history, chmod, chown, finger, pwd, cal, logout, shutdown) commands.

PROCEDURE-

1.man command

Used to display the user manual of any command that we can run on the terminal.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ man
What manual page do you want?
For example, try 'man man'.
linuxlab@linuxlab-VirtualBox:~/Desktop$ man echo
linuxlab@linuxlab-VirtualBox:~/Desktop$ echo hello
hello
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

2.pwd command

Used to find current path

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ pwd
/home/linuxlab/Desktop
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

3.ls command

It is used to list files in current directory.

```
linuxlab@linuxlab-VirtualBox:~$ ls
dead.letter  Documents  Music      Public        Templates  VIfile.txt
Desktop      Downloads  Pictures   shellfile.sh  Videos
linuxlab@linuxlab-VirtualBox:~$ █
```

4.ls -ltr command

It is used to list files by time in reverse order with long listing.

```
linuxlab@linuxlab-VirtualBox:~$ ls -ltr
total 44
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov  9 17:39 Templates
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov  9 17:39 Public
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 11:24 Documents
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 11:25 Downloads
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 11:25 Music
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 11:25 Pictures
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 11:25 Videos
-rw-rw-r-- 1 linuxlab linuxlab    21 Nov 10 11:40 Vfile.txt
-rw-rw-r-- 1 linuxlab linuxlab    18 Nov 10 11:46 shellfile.sh
-rw-rw-r-- 1 linuxlab mail      372 Nov 10 13:07 dead.letter
drwxr-xr-x 2 linuxlab linuxlab 4096 Nov 10 17:47 Desktop
linuxlab@linuxlab-VirtualBox:~$
```

5.history command

This command displays all the commands that were previously being executed by the user.

```
linuxlab@linuxlab-VirtualBox:~$ history
1 docker
2 sudo apt install docker.io
3 docker --version
4 sudo systemctl status docker
5 sudo apt-get update
6 sudo docker run hello-world
7 docker images
8 sudo docker images
9 docker ps
10 sudo docker ps
11 sudo docker ps -a
12 sudo chmod compare.sh
13 cat > compare.sh
14 bash compare.sh
15 cat > string.sh
16 bash string.shj
17 bash string.sh
18 cat >logical.sh
```

6.ping command

The ping command(packet internet groper) checks connectivity status between host to server.Ping uses ICMP(Internet Control Message protocol) and sends an ICMP echo to the server.It takes an input of an IP address or URL.

```
ss -t -w 1000
linuxlab@linuxlab-VirtualBox:~$ ping google.com
PING google.com (142.250.192.110) 56(84) bytes of data.
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=1 ttl=58 time=26.4 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=2 ttl=58 time=28.2 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=3 ttl=58 time=26.6 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=4 ttl=58 time=27.5 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=5 ttl=58 time=27.4 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=6 ttl=58 time=27.4 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=7 ttl=58 time=27.6 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=8 ttl=58 time=27.0 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=9 ttl=58 time=27.9 ms
64 bytes from bom12s17-in-f14.1e100.net (142.250.192.110): icmp_seq=10 ttl=58 time=27.4 ms
```

7.df command-The df command is used to display the disk space used in the file system. It displays the output as in the number of used blocks, available blocks, and the mounted directory.

```
linuxlab@linuxlab-VirtualBox:~$ df
Filesystem      1K-blocks    Used Available Use% Mounted on
udev              723808      0   723808   0% /dev
tmpfs             151200   1352   149848   1% /run
/dev/sda5     9736500 8242036  980160  90% /
tmpfs             755984      0   755984   0% /dev/shm
tmpfs               5120      4    5116   1% /run/lock
tmpfs             755984      0   755984   0% /sys/fs/cgroup
/dev/loop5            128    128      0 100% /snap/bare/5
/dev/loop0        224256 224256      0 100% /snap/gnome-3-34-1804/66
/dev/loop1        66432  66432      0 100% /snap/gtk-common-themes/1514
/dev/loop3        31872  31872      0 100% /snap/snapd/11036
/dev/loop4        52352  52352      0 100% /snap/snap-store/518
/dev/loop7        224256 224256      0 100% /snap/gnome-3-34-1804/72
/dev/loop6        66816  66816      0 100% /snap/gtk-common-themes/1519
/dev/loop2        56832  56832      0 100% /snap/core18/1988
/dev/loop8        52224  52224      0 100% /snap/snap-store/547
/dev/loop10       43264  43264      0 100% /snap/snapd/13831
/dev/loop9        56832  56832      0 100% /snap/core18/2246
/dev/sda1        523248      4  523244   1% /boot/efi
tmpfs             151196     20  151176   1% /run/user/1000
linuxlab@linuxlab-VirtualBox:~$
```



8.cd command

The “cd” command is used to change the current directory.

```
linuxlab@linuxlab-VirtualBox:~$ cd Desktop
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

9.cat command

The “cat” command is a multi-purpose utility in the Linux system. It can be used to create a file, display content of the file, copy the content of one file to another file, and more.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > sample.txt
abc
def
aa
ads
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

10 .ps command

To view the processes that you're running, use ps command.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ps
  PID TTY      TIME CMD
 2160 pts/0    00:00:00 bash
 2318 pts/0    00:00:00 ps
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

11.top command

Command used to view all the processes that are running.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ top

top - 18:15:12 up 21 min,  1 user,  load average: 0.01, 0.07, 0.09
Tasks: 181 total,   1 running, 180 sleeping,   0 stopped,   0 zombie
%Cpu(s):  9.9 us,  1.4 sy,  0.0 ni, 88.4 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0 st
MiB Mem : 1476.5 total,     86.9 free,    723.3 used,    666.4 buff/cache
MiB Swap:  448.5 total,     447.5 free,      1.0 used.    613.6 avail Mem

          PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
 1290 linuxlab  20   0 3439632 326104 118224 S  6.0 21.6  0:33.75 gnome ++
  732 linuxlab  20   0 529888 58988 38268 S  3.0  3.9  0:07.77 Xorg
 2152 linuxlab  20   0 823280 50748 38280 S  2.0  3.4  0:02.33 gnome ++
  12 root      20   0      0      0      0 S  0.3  0.0  0:00.17 ksofti +
 1621 linuxlab  20   0 912320 32408 21688 S  0.3  2.1  0:00.38 gsd-me +
2321 linuxlab  20   0 20488  3688  3176 R  0.3  0.2  0:00.03 top
  1 root      20   0 167776 11584  8436 S  0.0  0.8  0:02.17 systemd
  2 root      20   0      0      0      0 S  0.0  0.0  0:00.00 kthrea +
  3 root      0 -20      0      0      0 I  0.0  0.0  0:00.00 rcu_gp
  4 root      0 -20      0      0      0 I  0.0  0.0  0:00.00 rcu_pa +
  6 root      0 -20      0      0      0 I  0.0  0.0  0:00.00 kworker +
  9 root      0 -20      0      0      0 I  0.0  0.0  0:00.00 mm_per +
 10 root     20   0      0      0      0 S  0.0  0.0  0:00.00 rcu_ta +
 11 root     20   0      0      0      0 S  0.0  0.0  0:00.00 rcu_ta +
 13 root     20   0      0      0      0 I  0.0  0.0  0:00.85 rcu_sc +
 14 root     rt   0      0      0      0 S  0.0  0.0  0:00.02 migrat +
 15 root    -51   0      0      0      0 S  0.0  0.0  0:00.00 idle_i +
 16 root     20   0      0      0      0 S  0.0  0.0  0:00.00 cpuhp/0
 17 root     20   0      0      0      0 S  0.0  0.0  0:00.00 kdevtm +
```

12.mkdir command

To create a new directory, use “mkdir”.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ mkdir linux
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls
linux sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

13.rmdir command

To remove an empty directory use “rmdir”.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ rmdir linux
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls
sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

14.head command

“head” command displays the top part of a file. It displays the first 10 lines in a file.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ head sample.txt
abc
def
aa
ads
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

15.head command for n lines

Command used to display the n number of lines in a file.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ head -n 2 sample.txt
abc
def
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

16.tail command

“tail” command displays the last part of a file. It displays the last 10 lines in a file.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ tail sample.txt
abc
def
aa
ads
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

17.tail command for n lines

Command used to display the n number of lines in a file.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ tail -n 2 sample.txt
aa
ads
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

18.cp command

The "cp" command is used to copy a file or directory.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat abcd.txt
abc
def
aa
ads
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

20.id command

The "id" command is used to display the user ID (UID) and group ID (GID).

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ id
uid=1000(linuxlab) gid=1000(linuxlab) groups=1000(linuxlab),4(adm),24(cdrom),27
(sudo),30(dip),46(plugdev),120(lpadmin),131(lxd),132(sambashare)
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

21.wc command

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

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ wc sample.txt
4 4 15 sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$ █
```

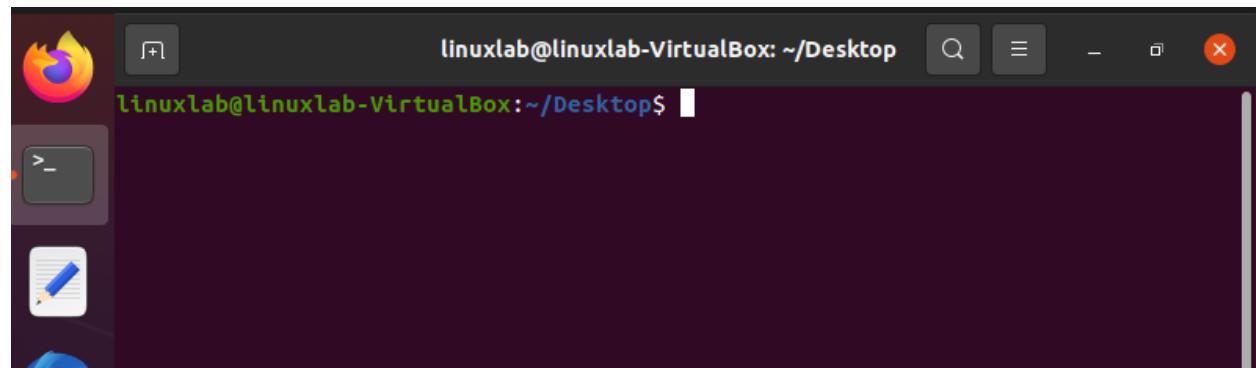
22.host command

The “host” command is used to display the IP address for a given domain name and vice versa. It performs the DNS lookups for the DNS Query.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ host google.com
google.com has address 142.250.192.142
google.com has IPv6 address 2404:6800:4009:825::200e
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 30 alt2.aspmx.l.google.com.
google.com mail is handled by 50 alt4.aspmx.l.google.com.
google.com mail is handled by 40 alt3.aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

23.clear command

Linux **clear** command is used to clear the terminal screen.



24.less command

“less” displays a file, allowing forward/backward movement within it.

```
abc
def
aa
ads
sample.txt (END)
```

25.ls -l command

Command used to display the files in long list format.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls -l
total 8
-rw-rw-r-- 1 linuxlab linuxlab 15 Nov 10 18:19 abcd.txt
-rw-rw-r-- 1 linuxlab linuxlab 15 Nov 10 18:13 sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

26.ls -t command

Command used to display the files in sorting format of time modification.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls -t
abcd.txt  sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

27.ls -h command

Command used to display the file sizes in human readable format.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls -h
abcd.txt  sample.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

28.ls -r command

Command used to display the files in reverse order format.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ls -r
sample.txt  abcd.txt
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

29.ip command-

: Linux "ip" command is an updated version of the ipconfig command. It is used to assign an IP address, initialize an interface, disable an interface.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ ip
Usage: ip [ OPTIONS ] OBJECT { COMMAND | help }
      ip [ -force ] -batch filename
where  OBJECT := { link | address | addrlabel | route | rule | neigh | ntable |
                  tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm
                  |
                  netns | l2tp | fou | macsec | tcp_metrics | token | netconf
| ila |
                  vrf | sr | nexthop }
OPTIONS := { -V[ersion] | -s[tatistics] | -d[etails] | -r[esolve] |
             -h[uman-readable] | -iec | -j[son] | -p[retty] |
             -family] { inet | inet6 | mpls | bridge | link } |
             -4 | -6 | -I | -D | -M | -B | -O |
             -loops] { maximum-addr-flush-attempts } | -br[ief] |
             -o[neline] | -t[imestamp] | -ts[hort] | -b[atch] [filename]
             |
             -rc[vbuf] [size] | -n[etns] name | -N[umeric] | -a[ll] |
             -c[olor]}}
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

30.exit command

Linux **exit** command is used to exit from the current shell.

LAB-3

AIM- Study of Vi Editor, Study of Bash shell, Bourne shell and C shell in Unix/Linux operating system.

THEORY-

vi Editor is used to edit files in Unix. It is done using the screen-oriented text editor, vi is one of the best ways. This editor enables you to edit lines in context with other lines in the file.

An improved version of the vi editor which is called the VIM has also been made available now. Here, VIM stands for vi improved.

- vi is generally considered the de facto standard in Unix editors because –
- It's usually available on all the flavors of Unix system.
- Its implementations are very similar across the board.
- It requires very few resources.
- It is more user-friendly than other editors such as the ed or the ex.

You can use the vi editor to edit an existing file or to create a new file from scratch. You can also use this editor to just read a text file.

In Unix, there are two major types of shells –

1. **Bourne shell** – If you are using a Bourne-type shell, the \$ character is the default prompt.
2. **C shell** – If you are using a C-type shell, the % character is the default prompt.

The Bourne Shell has the following subcategories: Bourne shell (sh), Korn shell (ksh), Bourne Again shell (bash), POSIX shell (sh).

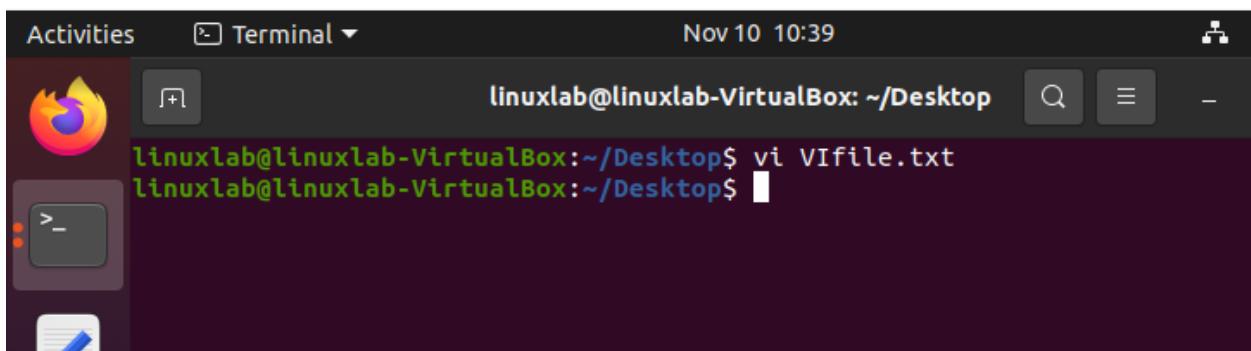
The different C-type shells follow: C shell (csh), TENEX/TOPS C shell (tcsh).

The original Unix shell was written in the mid-1970s by Stephen R. Bourne while he was at the AT&T Bell Labs in New Jersey.

Bourne shell was the first shell to appear on Unix systems, thus it is referred to as "the shell". Bourne shell is usually installed as **/bin/sh** on most versions of Unix. For this reason, it is the shell of choice for writing scripts that can be used on different versions of Unix.

PROCEDURE:

1. **vi filename command:** Creates a new file if it already does not exist, otherwise opens an existing file.



A screenshot of a Linux desktop environment showing a terminal window. The terminal window is titled "Terminal" and shows the command "vi VIfile.txt" being run. The output of the command is visible in the terminal window. The desktop environment includes a dock with icons for a browser, file manager, and terminal, and a system tray with icons for network, battery, and system status.

```
linuxlab@linuxlab-VirtualBox: ~/Desktop$ vi VIfile.txt
```

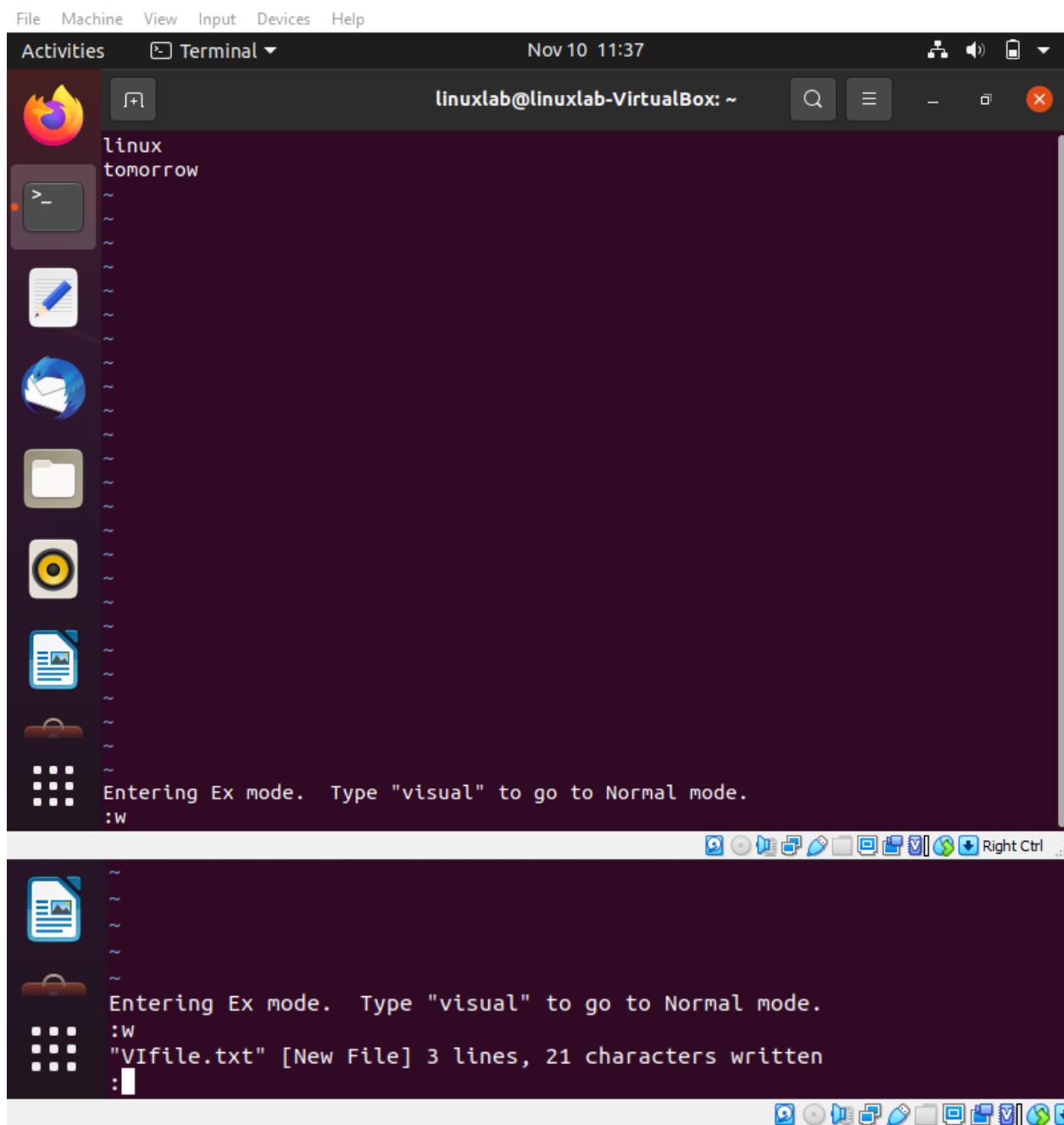
2. Inserting in vi: use 'I' to enter insertion mode

The screenshot shows a Linux desktop environment with a dark theme. At the top is a header bar with the terminal title "linuxlab@linuxlab-VirtualBox: ~/Desktop". The desktop background is black. A vertical dock on the left contains icons for various applications: a browser (Firefox), a terminal window titled "hello", a text editor, an email client, a file manager, a system tray, a file viewer, a file manager, a terminal window titled "recording @W", and a terminal window titled "recording @W". A docked application, likely a media player or video editor, is visible at the bottom.

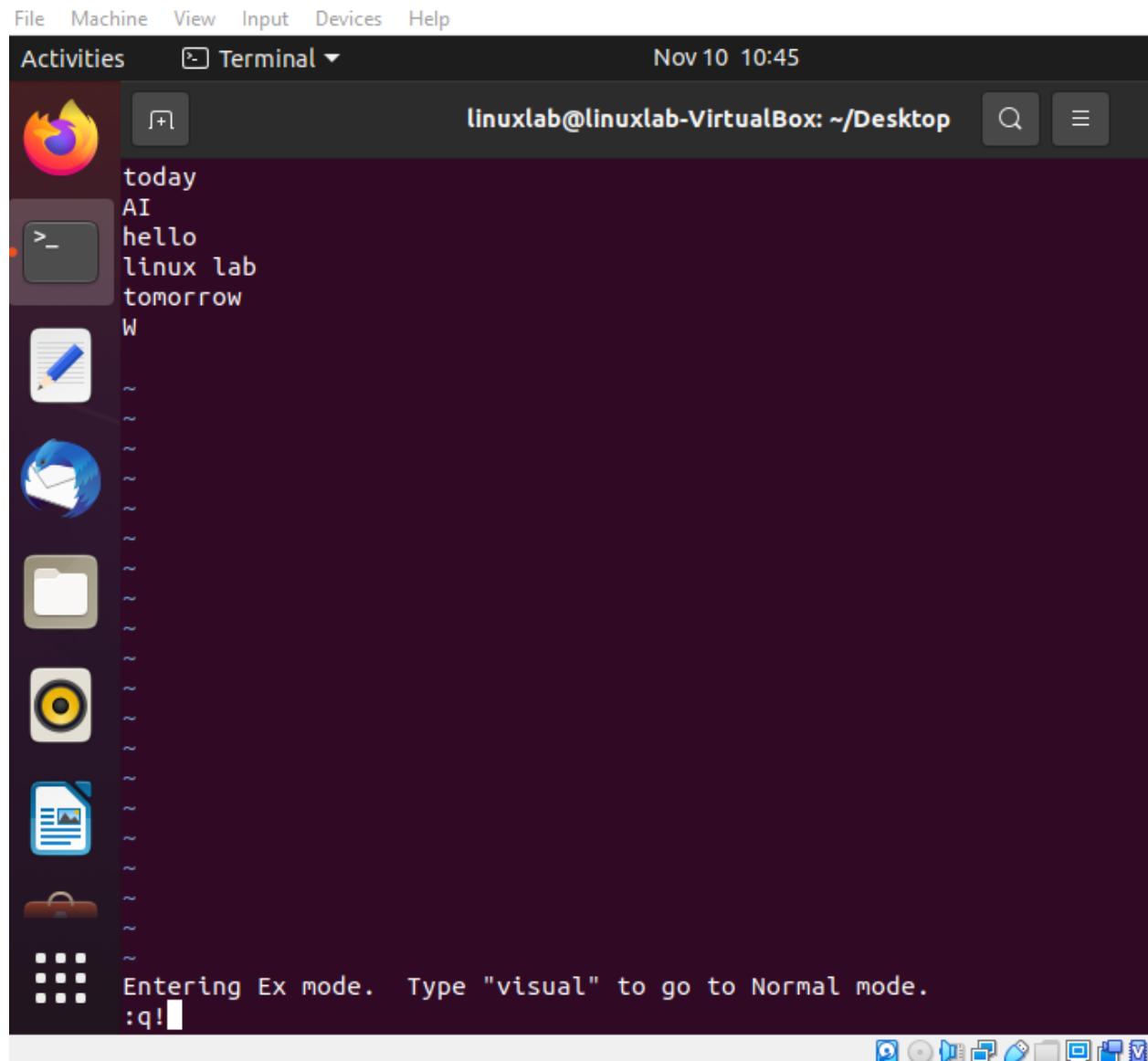
3. :wq- To exit vi editor

A screenshot of a Linux desktop environment. On the left is a dock with icons for various applications like a browser, file manager, and terminal. A terminal window is open in the top right corner, titled "Terminal". The terminal shows the command "linux lab" followed by "tomorrow" on the next line. The user has typed "W" and then pressed Enter. The terminal then displays "Entering Ex mode. Type "visual" to go to Normal mode." followed by the command ":wq". The status bar at the bottom shows the date and time as "Nov 10 10:40".

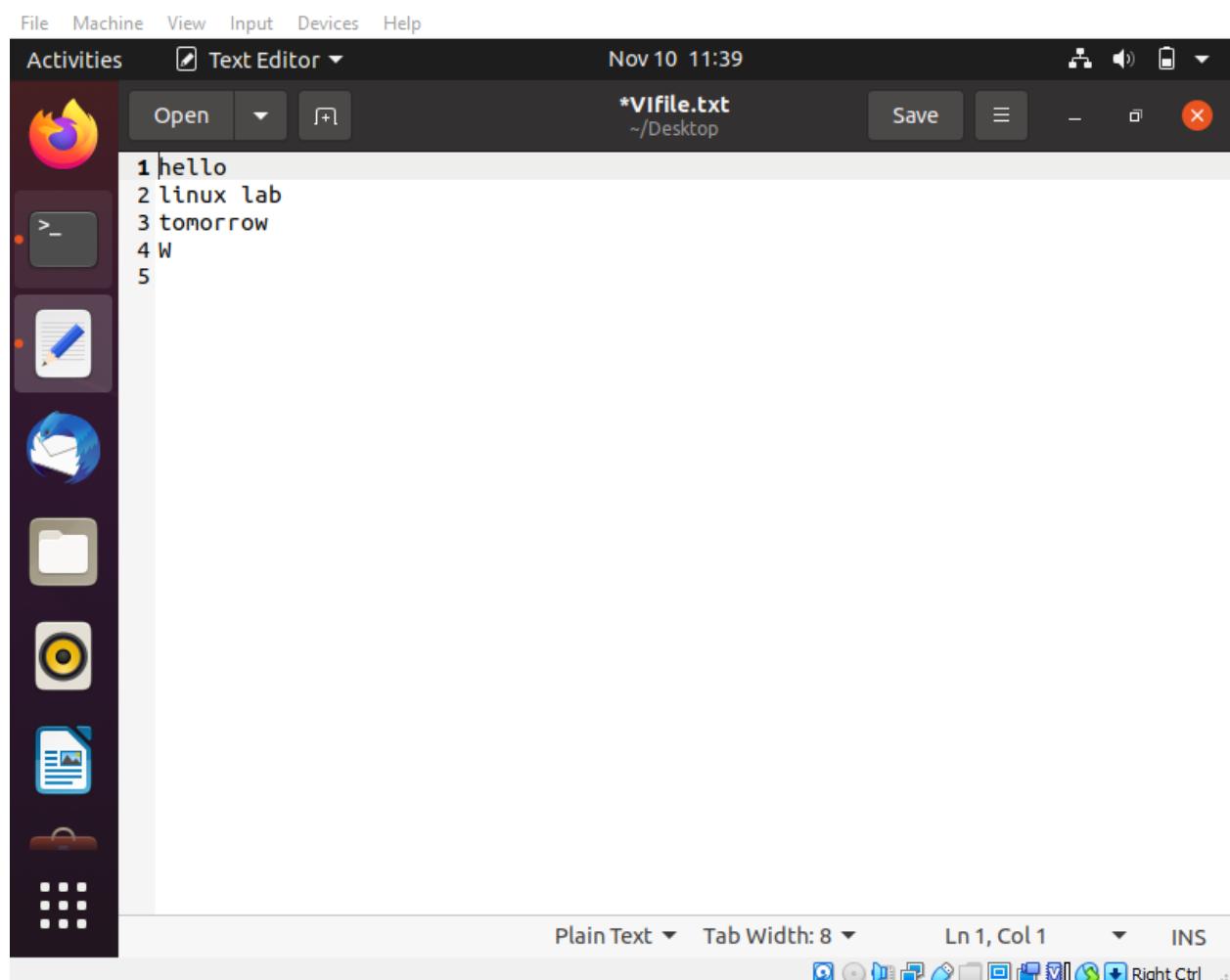
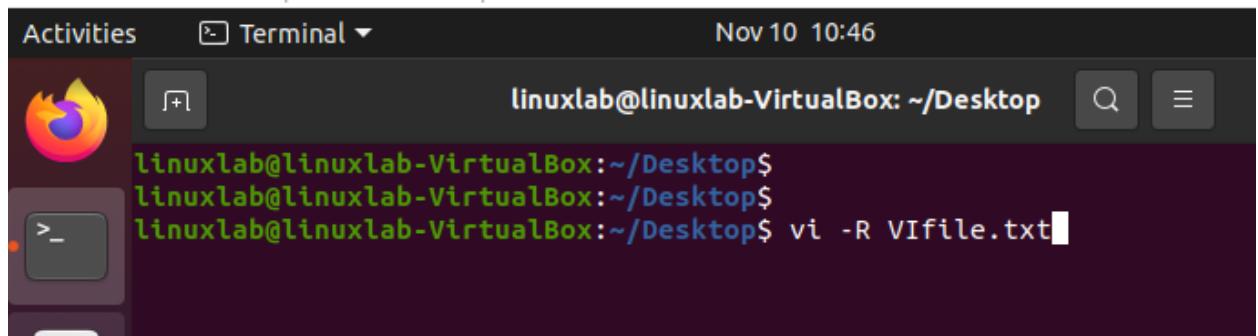
5:w- To save edited content in vi editor



6. **:q!-** To exit vi editor without saving



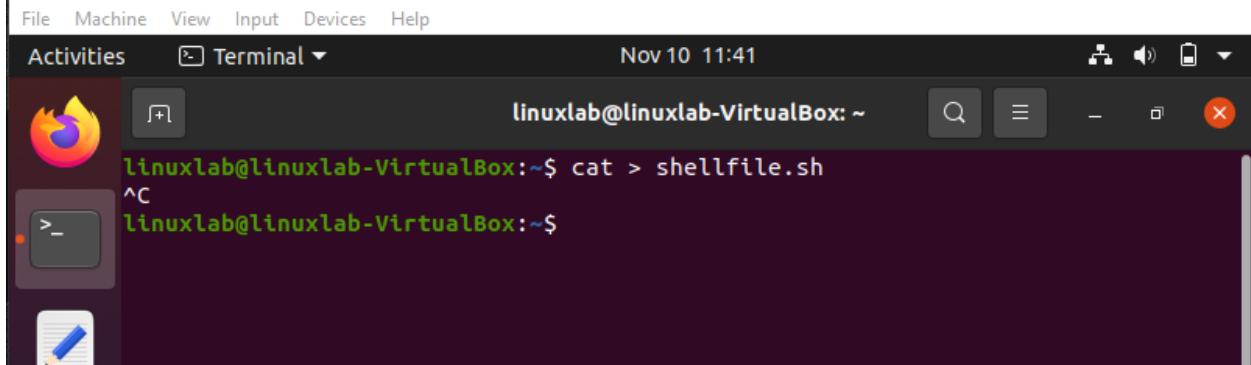
7. **vi -R filename:** Opens an existing file in the read-only mode.



Study of Bash shell, Bourne shell and C shell in Unix/Linux operating system.

PROCEDURE:

1.Creating <filename>.sh file:

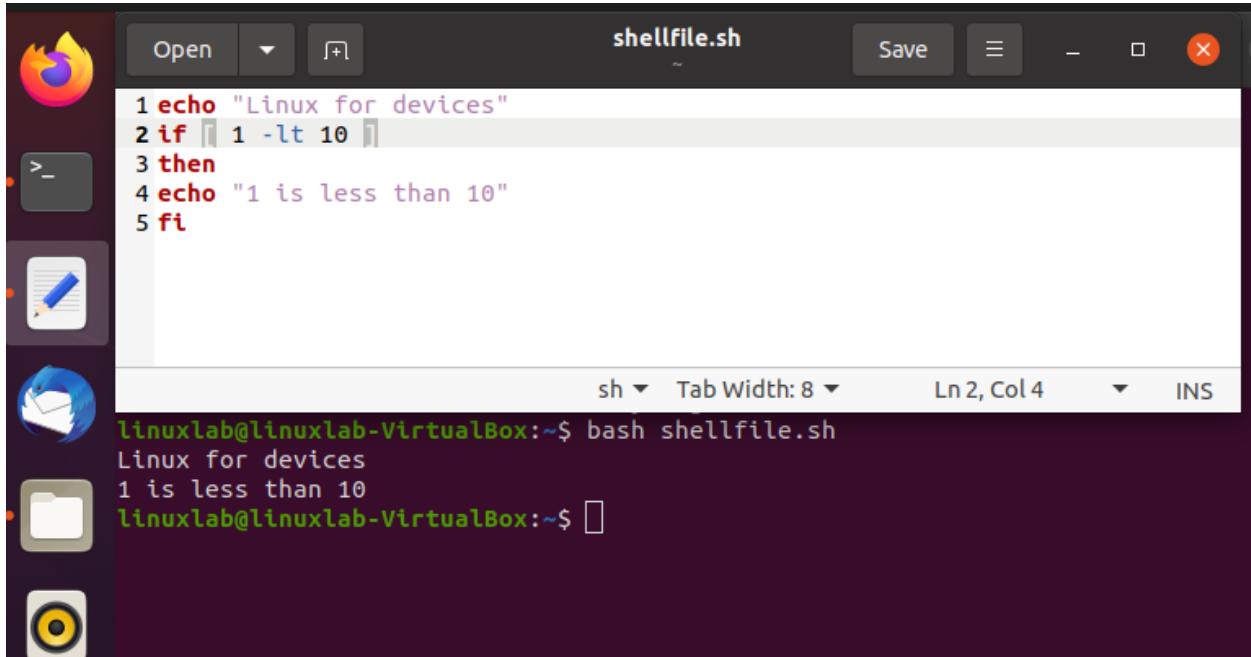


2.Inserting linux command in .sh file:



3.Saving and exiting .sh file:

4.running the .sh file:



The screenshot shows a terminal window titled "shellfile.sh" containing the following code:

```
1 echo "Linux for devices"
2 if [ 1 -lt 10 ]
3 then
4 echo "1 is less than 10"
5 fi
```

The terminal output below the code shows the script being run and its execution:

```
sh Tab Width: 8 Ln 2, Col 4 INS
linuxlab@linuxlab-VirtualBox:~$ bash shellfile.sh
Linux for devices
1 is less than 10
linuxlab@linuxlab-VirtualBox:~$
```

6.Creating Variable in .sh file and executing it:

The screenshot shows a Linux desktop environment with a dark theme. At the top, there is a header bar with icons for File, Machine, View, Input, Devices, and Help. Below the header is an Activities overview panel showing various application icons like a browser, file manager, and terminal. A central window is a Text Editor titled "shellfile.sh" containing the following shell script:

```
1 num=2
2 echo "$num"
```

The terminal window below shows the command being run:

```
linuxlab@linuxlab-VirtualBox:~$ cat > shellfile.sh
^Z
1 num=2
2 echo "$num"
^Z
2
linuxlab@linuxlab-VirtualBox:~$
```

The terminal also displays status information: sh, Tab Width: 8, Ln 2, Col 12, and INS.

PRACTICAL 4 & 5

AIM- Write a shell script program to display “HELLO WORLD”. write a shell script program to develop a scientific calculator.

THEORY-

Bash is a command language interpreter. It is widely available on various operating systems and is a default command interpreter on most GNU/Linux systems. The name is an acronym for the ‘Bourne-Again Shell’. Bash is a shell program. Bash is a command processor that typically runs in a text window where the user types commands that cause actions. Bash can also read and execute commands from a file, called a shell script. A shell program is typically an executable binary that takes commands that you type and (once you hit return), translates those commands into (ultimately) system calls to the Operating System API. Bash is not the only kind of shell. Other shells include:

Sh, ash, dash, ksh, tcsh, zsh, tclsh.

Shell

Shell is a macro processor which allows for an interactive or non-interactive command execution.

Scripting

Scripting allows for an automatic commands execution that would otherwise be executed interactively one-by-one.

all our scripts will include shell interpreter definition `#!/bin/bash`.

Script

In Computer programming, a script is a set of commands for an appropriate run time environment which is used to automate the execution of tasks.

Bash Script:

A Bash Shell Script is a plain text file containing a set of various commands that we usually type in the command line. It is used to automate repetitive tasks on Linux filesystem. It might include a set of commands, or a single command, or it might contain the hallmarks of imperative programming like loops, functions, conditional constructs, etc. Effectively, a Bash script is a computer program written in the Bash programming language.

How to create and run a Bash Script?

- To create an empty bash script, first, change the directory in which you want to save your script using cd command. Try to use text editor like gedit in which you want to type the shell commands.
- Use touch command to create the zero bytes sized script.
 1. touch file_name
 - To open the script in the text editor (eg., gedit), type
 1. gedit file_name.sh

Here, .sh is suffixed as an extension that you have to provide for execution.

- Type the shell commands for your bash script in the newly opened text window or the text editor. Before typing bash shell commands, first, look at the base of any bash script.

Each Bash based Linux script starts by the line-

1. #! /bin/bash

Where #! is referred to as the shebang and rest of the line is the path to the interpreter specifying the location of bash shell in our operating system.

Bash use # to comment any line.

Bash use echo command to print the output.

At the end, execute the bash script prefixing with ./.

Have a look at the basic terms of a Bash Script, i.e., SheBang and echo command.

SheBang (#!)

The She Bang (#!) is a character sequence consisting of the characters number sign (#) and exclamation mark (!) at the beginning of a script.

Under the Unix-like operating systems, when a script with a shebang runs as a program, the program loader parses the rest of the lines with the first line as an interpreter directive. So, SheBang denotes an interpreter to execute the script lines, and it is known as the path directive for the execution of different kinds of Scripts like Bash, Python, etc.

Here is the correct SheBang format for the discussed Bash Script.

1. `#!/bin/bash`

The formatting for shebang is most important. Its incorrect format can cause improper working of commands. So, always remember these two points of SheBang formatting while creating a Script as follows:

1. It should always be on the very first line of the Script.
2. There should not be any space before the hash (#), between the hash exclamation marks (#!), and the path to the interpreter.

`echo`

`echo` is a built-in command in Bash, which is used to display the standard output by passing the arguments. It is the most widely used command for printing the lines of text/String to the screen. Its performance is the same on both the platforms: Bash Shell and Command Line Terminal.

OUTPUT:

HELLO WORLD:

The screenshot shows a Linux desktop environment running in Oracle VM VirtualBox. The desktop interface includes a dock with icons for various applications like a browser, file manager, and terminal. A terminal window is open with the command:

```
logical.sh does not exist
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash file.sh
```

A file editor window titled "helloworld.sh" is open, showing the following script content:

```
1 echo "Hello World "
2 echo "MIHIR DADWAL"
```

The terminal window then executes the script:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > helloworld.sh
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash helloworld.sh
Hello World
MIHIR DADWAL
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

Calculator:

A screenshot of a Linux desktop environment. On the left is a dock with icons for various applications: Firefox, Mail, Files, Dash, Terminal, and Help. The main window is a terminal-like application titled "calculator.sh" located at "/Desktop". The code in the editor is:

```
1 echo "Enter Two numbers : "
2 read a
3 read b
4
5 echo "Enter Choice :"
6 echo "1. Addition"
7 echo "2. Subtraction"
8 echo "3. Multiplication"
9 echo "4. Division"
10 read ch
11 case $ch in
12   1)res=`echo $a + $b | bc`;;
13   ;;
14   2)res=`echo $a - $b | bc`;;
15   ;;
16   3)res=`echo $a \* $b | bc`;;
17   ;;
18   4)res=`echo "scale=2; $a / $b" | bc`;;
19   ;;
20 esac
21 echo "Result : $res"
```

A screenshot of a terminal window titled "Terminal" located at "Nov 9 21:28". The terminal shows the execution of the script:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > calculator.sh
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash calculator.sh
Enter Two numbers :
4
7
Enter Choice :
1. Addition
2. Subtraction
3. Multiplication
4. Division
3
Result : 28
linuxlab@linuxlab-VirtualBox:~/Desktop$ S
```

PRACTICAL 6

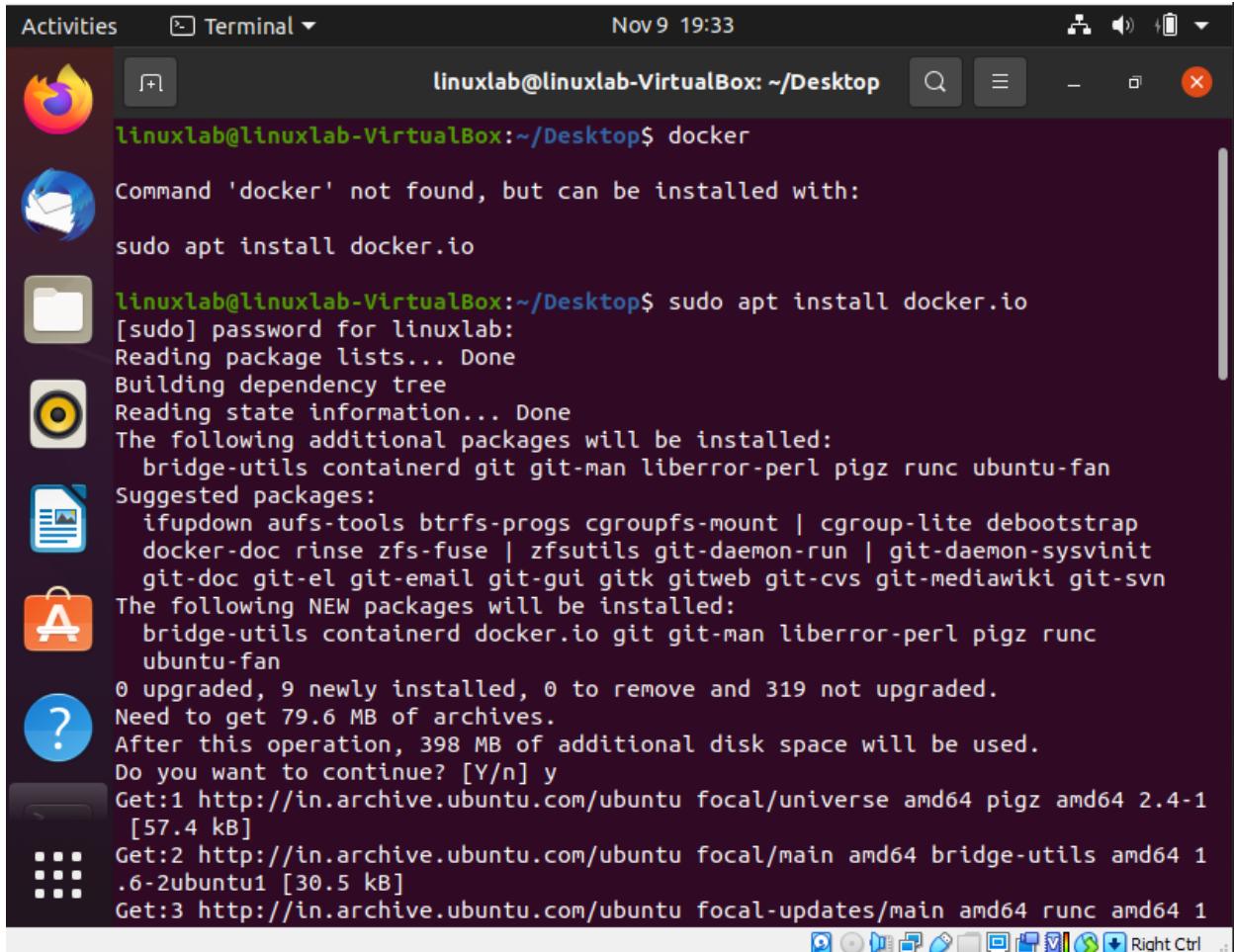
AIM- Install Docker and run :Start, stop, push, pull, log, docker ps, docker ps –a,create account, docker compose, docker build, docker run

THEORY-

- Docker is an open source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment. Containers simplify delivery of distributed applications, and have become increasingly popular as organizations shift to cloud-native development and hybrid multicloud environments.
- Developers can create containers without Docker, but the platform makes it easier, simpler, and safer to build, deploy and manage containers. Docker is essentially a toolkit that enables developers to build, deploy, run, update, and stop containers using simple commands and work-saving automation through a single API.
- Docker is so popular today that “Docker” and “containers” are used interchangeably. But the first container-related technologies were available for years — even decades (link resides outside IBM) — before Docker was released to the public in 2013.

PROCEDURE-

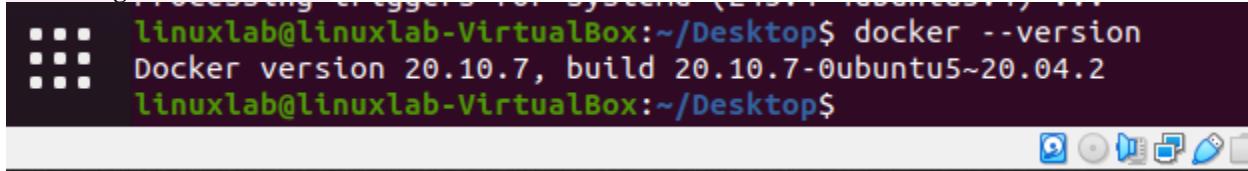
1.installing docker



```
Activities Terminal Nov 9 19:33
linuxlab@linuxlab-VirtualBox: ~/Desktop$ docker
Command 'docker' not found, but can be installed with:
sudo apt install docker.io

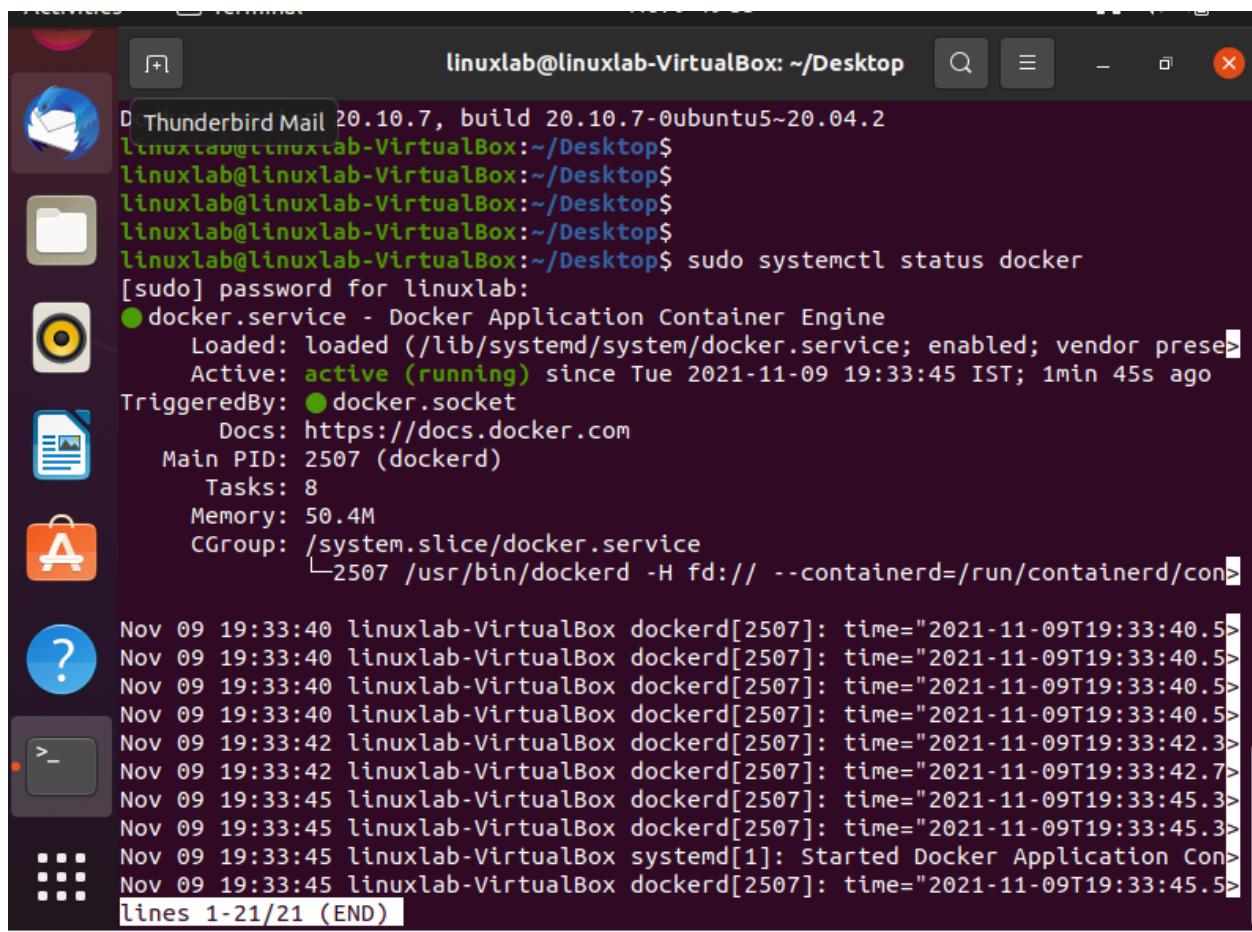
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo apt install docker.io
[sudo] password for linuxlab:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd git git-man liberror-perl pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools btrfs-progs cgroupfs-mount | cgroup-lite debootstrap
  docker-doc rinse zfs-fuse | zfsutils git-daemon-run | git-daemon-sysvinit
  git-doc git-el git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  bridge-utils containerd docker.io git git-man liberror-perl pigz runc
  ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 319 not upgraded.
Need to get 79.6 MB of archives.
After this operation, 398 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1
[57.4 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/main amd64 bridge-utils amd64 1
.6-2ubuntu1 [30.5 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 runc amd64 1
```

2.checking version of docker



```
linuxlab@linuxlab-VirtualBox:~/Desktop$ docker --version
Docker version 20.10.7, build 20.10.7-0ubuntu5~20.04.2
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

3.Checking enable status of docker(active/inactive)

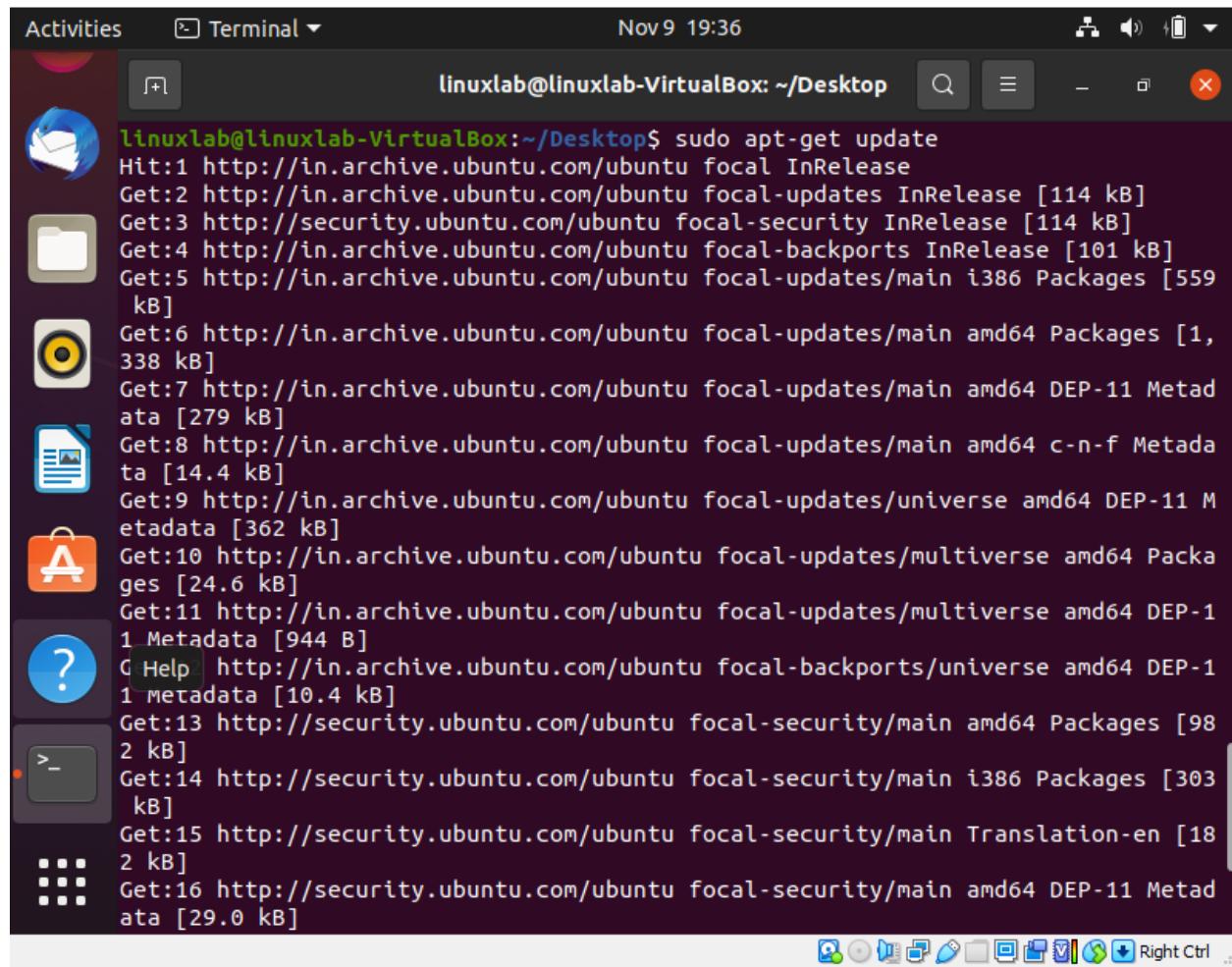


A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window titled "Terminal". The terminal window has a dark theme and displays a command-line session. The session starts with a message from Thunderbird Mail, followed by several blank lines, then a command to check the status of the Docker service. The user enters a password for the "linuxlab" user. The Docker service is shown as active and running. The terminal then displays log entries from the docker daemon, starting at November 9, 2021, at 19:33:40. The logs show the daemon starting and the system manager (systemd) starting the Docker application container engine.

```
D Thunderbird Mail 20.10.7, build 20.10.7-0ubuntu5~20.04.2
linuxlab@linuxlab-VirtualBox:~/Desktop$ 
linuxlab@linuxlab-VirtualBox:~/Desktop$ 
linuxlab@linuxlab-VirtualBox:~/Desktop$ 
linuxlab@linuxlab-VirtualBox:~/Desktop$ 
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo systemctl status docker
[sudo] password for linuxlab:
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor prese>
   Active: active (running) since Tue 2021-11-09 19:33:45 IST; 1min 45s ago
     TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
   Main PID: 2507 (dockerd)
     Tasks: 8
    Memory: 50.4M
      CGroup: /system.slice/docker.service
              └─2507 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con>

Nov 09 19:33:40 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:40.5>
Nov 09 19:33:42 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:42.3>
Nov 09 19:33:42 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:42.7>
Nov 09 19:33:45 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:45.3>
Nov 09 19:33:45 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:45.3>
Nov 09 19:33:45 linuxlab-VirtualBox systemd[1]: Started Docker Application Con>
Nov 09 19:33:45 linuxlab-VirtualBox dockerd[2507]: time="2021-11-09T19:33:45.5>
```

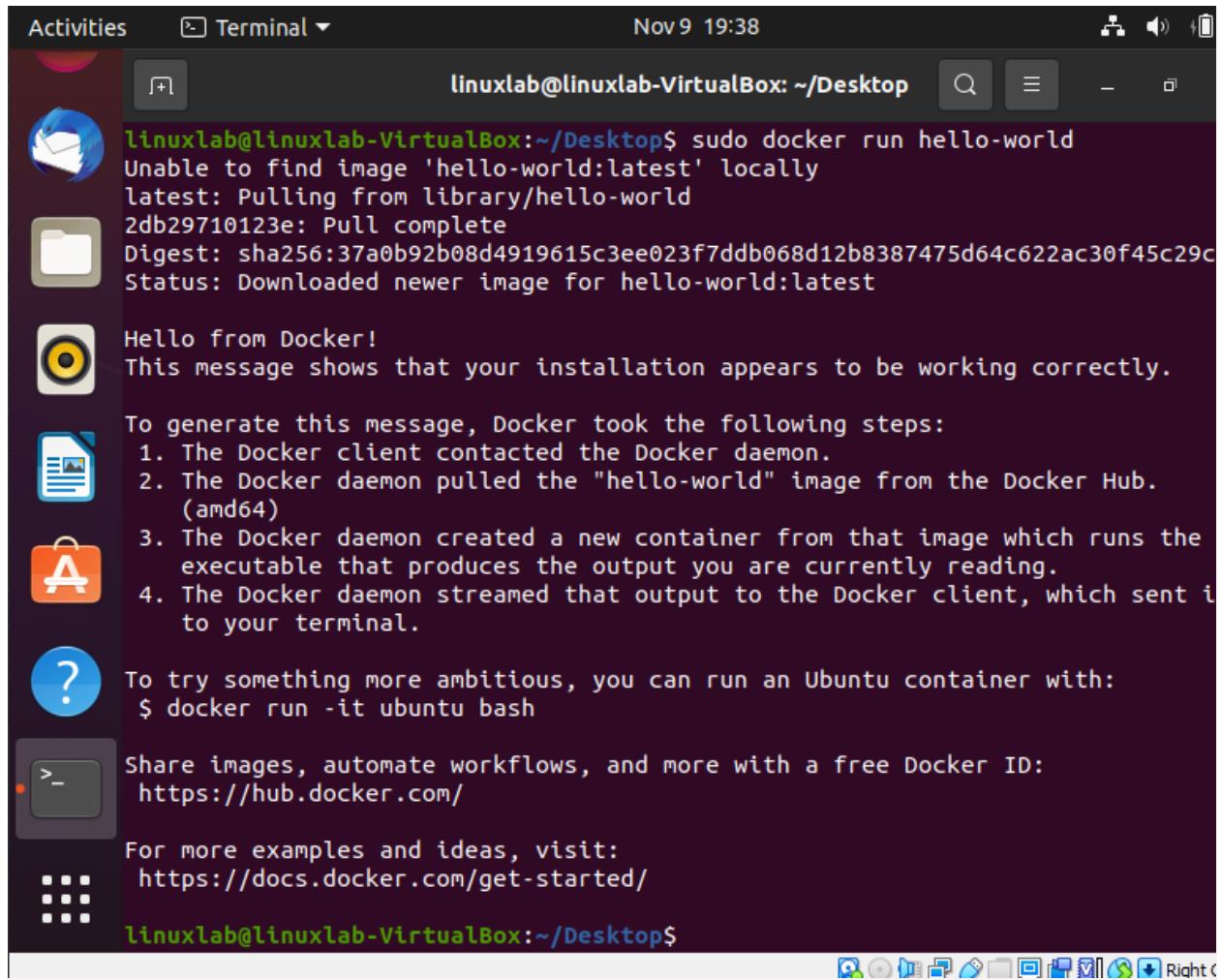
lines 1-21/21 (END)



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window title is "linuxlab@linuxlab-VirtualBox: ~/Desktop". The terminal content shows the output of the command "sudo apt-get update", which lists various package downloads from the Ubuntu repositories. The desktop background features a dark theme with icons for various applications like file manager, terminal, and system settings.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo apt-get update
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:4 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease [101 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [559 kB]
Get:6 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1,338 kB]
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 DEP-11 Metadata [279 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 c-n-f Metadata [14.4 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 DEP-11 Metadata [362 kB]
Get:10 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [24.6 kB]
Get:11 http://in.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 DEP-11 Metadata [944 B]
C Help http://in.archive.ubuntu.com/ubuntu focal-backports/universe amd64 DEP-11 metadata [10.4 kB]
Get:13 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [982 kB]
Get:14 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [303 kB]
Get:15 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [182 kB]
Get:16 http://security.ubuntu.com/ubuntu focal-security/main amd64 DEP-11 Metadata [29.0 kB]
```

4. Run hello world



A screenshot of an Ubuntu desktop environment. At the top, there's a dock with icons for Dash, Home, Activities, Terminal, and others. The terminal window is open at the command line: `linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo docker run hello-world`. The output shows the Docker daemon pulling the image from the Docker Hub and running the container, which prints "Hello from Docker!" and its status. Below the terminal, several informational cards provide tips about Docker usage, such as running an Ubuntu container and sharing images. The bottom of the screen shows the standard Unity interface with a dock of application icons.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

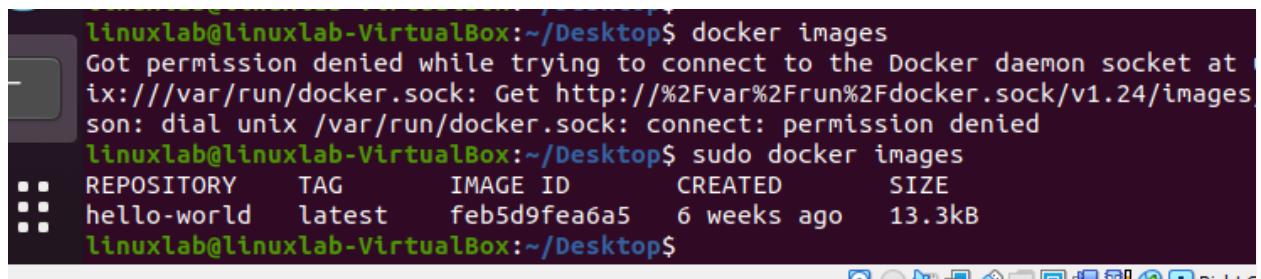
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

linuxlab@linuxlab-VirtualBox:~/Desktop\$

5. Run image pulled from repository



A screenshot of an Ubuntu desktop environment. The terminal window shows the command `docker images` being run, but it fails with a permission denied error because it's not run as root. The user then runs `sudo docker images`, which successfully lists the available Docker images. The output shows one image: "hello-world" with tag "latest", created 6 weeks ago, and a size of 13.3kB.

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ docker images
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://<2>/var%2Frun%2Fdocker.sock/v1.24/images?filter%40all: dial unix /var/run/docker.sock: connect: permission denied
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo docker images
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
hello-world    latest    feb5d9fea6a5   6 weeks ago   13.3kB
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

6.

Checking if containers are still running in machine

docker ps: Checks if container is running

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ docker ps
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://<2Fvar%2Frun%2Fdocker.sock/v1.24/containers/json: dial unix /var/run/docker.sock: connect: permission denied
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
                  PORTS           NAMES
4fc3eb5d21b3        hello-world        "/hello"          About a minute ago   Exited (0) About a
minute ago          clever_galois
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

No container is running

PRACTICAL 7

AIM Write a shell script program to illustrate the implementation of following:

- a) Integer Comparison
- b) String comparison
- c) Logical operators
- d) File tests
- e) Conditional control structure
- f) Loop control structures

THEORY-

Bash is a command language interpreter. It is widely available on various operating systems and is a default command interpreter on most GNU/Linux systems. The name is an acronym for the ‘Bourne-Again Shell’. Bash is a shell program. Bash is a command processor that typically runs in a text window where the user types commands that cause actions. Bash can also read and execute commands from a file, called a shell script. A shell program is typically an executable binary that takes commands that you type and (once you hit return), translates those commands into (ultimately) system calls to the Operating System API. Bash is not the only kind of shell.

PROCEDURE:

1. Integer comparison

Different kinds of integer operator in bash are-

The general form of integer comparisons is int1 -operator int2.

Operator	Purpose
-eq	Integer equality
-ne	Integer inequality
-lt	Integer less than

-le Integer less than or equal to

-gt Integer greater than

-ge Integer greater than or equal to

The screenshot shows a Linux desktop environment with a terminal window and a text editor window.

The terminal window (bottom) shows the following session:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > compare.sh
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash compare.sh
enter number for comparison
4
8
>-
4 is less than 8
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash compare.sh
enter number for comparison
6
2
6 is greater than 2
```

The text editor window (top) shows a script named "compare.sh" with the following content:

```
1 echo "enter number for comparison"
2 read num1
3 read num2
4 if test $num1 -eq $num2
5 then
6 echo "$num1 is equal than $num2"
7 elif test $num1 -gt $num2
8 then
9 echo "$num1 is greater than $num2"
10 elif test $num1 -lt $num2
11 then
12 echo "$num1 is less than $num2"
13 fi
```

2.String comparison

String Comparison Operators

Comparison operators are operators that compare values and return true or false. When comparing strings in Bash you can use the following operators:

string1 = string2 and string1 == string2 - The equality operator returns true if the operands are equal.

- Use the `=` operator with the `test [` command.
- Use the `==` operator with the `[[` command for pattern matching.

string1 != string2 - The inequality operator returns true if the operands are not equal.

string1 =~ regex - The regex operator returns true if the left operand matches the extended regular expression on the right.

string1 > string2 - The greater than operator returns true if the left operand is greater than the right sorted by lexicographical (alphabetical) order.

string1 < string2 - The less than operator returns true if the right operand is greater than the right sorted by lexicographical (alphabetical) order.

-z string - True if the string length is zero.

-n string - True if the string length is non-zero.

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Text Editor" and the path is "linuxlab@linuxlab-VirtualBox: ~/Desktop". The file being edited is "string.sh" located at "~/Desktop". The script content is:

```
1 echo "enter string"
2 read a1
3 read a2
4 if [ "$a1" == "$a2" ]
5 then
6 echo "strings match"
7 else
8 echo "strings do not match"
9 fi
```

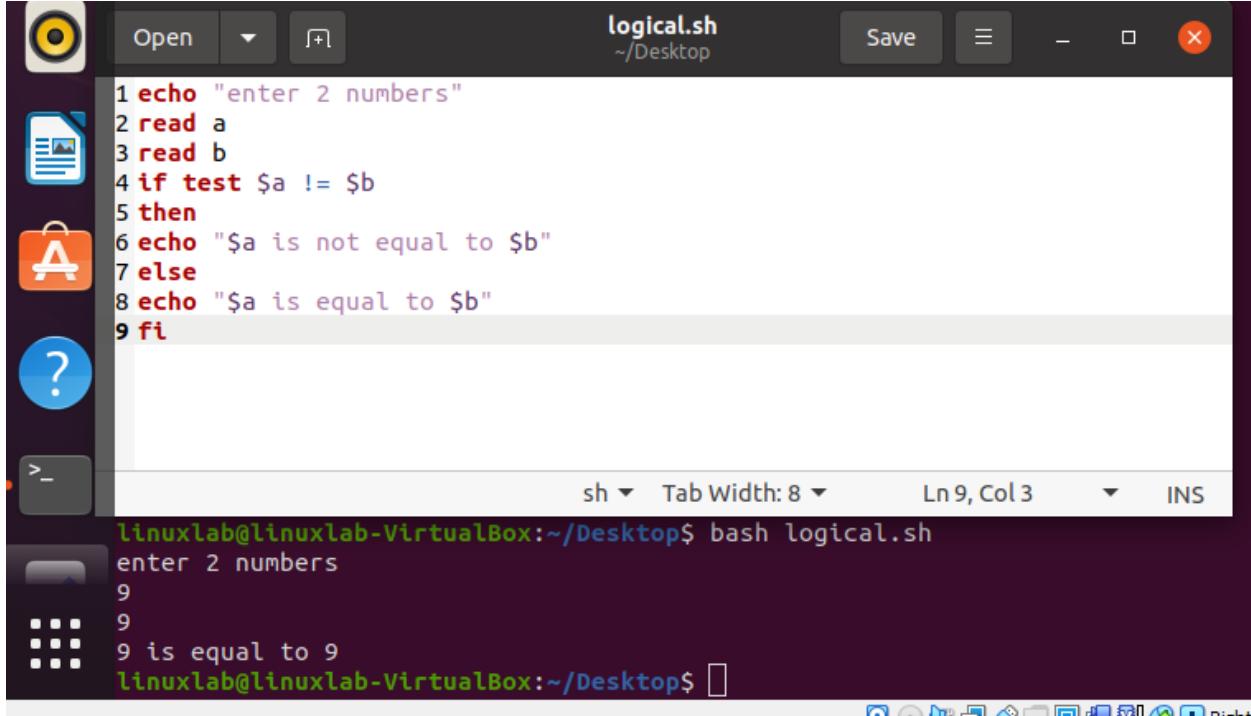
The terminal window shows the output of running the script:

```
bash: string.sh: No such file or directory
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash string.sh
enter string
d
d
strings match
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash string.sh
enter string
x
c
strings do not match
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

3. Logical operators

Different Logical Operator in Linux are:

- !/: This is logical negation. This inverts a true condition into false and vice versa.
- o: This is logical OR. If one of the operands is true, then the condition becomes true.
- a: This is logical AND. If both the operands are true, then the condition becomes true otherwise false.



```
logical.sh
~/Desktop
Save
-
X
Open ▾
logical.sh
~/Desktop
Save
-
X
1 echo "enter 2 numbers"
2 read a
3 read b
4 if test $a != $b
5 then
6 echo "$a is not equal to $b"
7 else
8 echo "$a is equal to $b"
9 fi
sh ▾ Tab Width: 8 ▾ Ln 9, Col 3 ▾ INS
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash logical.sh
enter 2 numbers
9
9
9 is equal to 9
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

4.File tests

To check whether a file exists,most commonly used file operators are -e and -f.
-e checks whether a file exists regardless of its type,
-f return true only if file is a regular file,not a directory or device.

To check whether a file exists or not, test command is used with if statement.

The screenshot shows a terminal window with a dark background. At the top, there's a menu bar with icons for file operations like Open, Save, and Close. The title bar says "file.sh ~/Desktop". Below the title bar, there's a status bar showing "sh Tab Width: 8 Ln 3, Col 15 INS". The main area of the terminal contains the following text:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash file.sh
enter filename
compare.sh
compare.sh exists
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash file.sh
enter filename
xyzabcd
xyzabcd does not exist
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

The terminal window has a toolbar at the bottom with various icons for file operations like copy, paste, and save.

5. Conditional control structure

Conditional Statements: There are a total of five conditional statements which can be used in bash programming.

1. if statement
2. if-else statement
3. if..elif..else..fi statement (Else If ladder)
4. if..then..else..if..then..fi..fi..(Nested if)

5. switch statement

simple program for integer comparison using if statement

The screenshot shows a Linux desktop environment with a terminal window and a text editor window.

The terminal window (bottom) shows the following session:

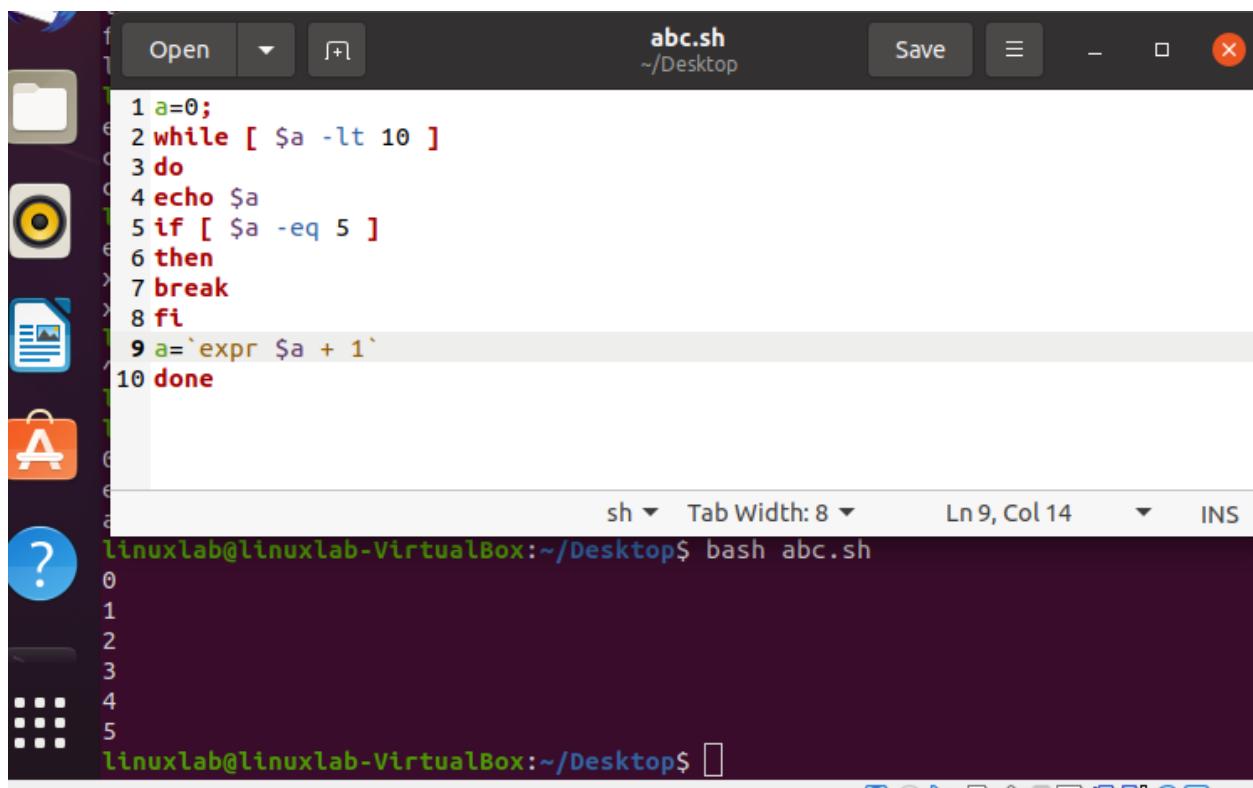
```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > compare.sh
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash compare.sh
enter number for comparison
4
8
4 is less than 8
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash compare.sh
enter number for comparison
6
2
6 is greater than 2
```

The text editor window (top) shows a script named "compare.sh" with the following content:

```
1 echo "enter number for comparison"
2 read num1
3 read num2
4 if test $num1 -eq $num2
5 then
6 echo "$num1 is equal than $num2"
7 elif test $num1 -gt $num2
8 then
9 echo "$num1 is greater than $num2"
10 elif test $num1 -lt $num2
11 then
12 echo "$num1 is less than $num2"
13 fi
```

6.Loop control structures

This bash shows that loop terminates as soon as **a** becomes 5 –



The screenshot shows a terminal window with a dark theme. At the top, there's a toolbar with icons for Open, Save, and other file operations. The title bar says "abc.sh ~/Desktop". Below the toolbar, the script content is displayed:

```
1 a=0;
2 while [ $a -lt 10 ]
3 do
4 echo $a
5 if [ $a -eq 5 ]
6 then
7 break
8 fi
9 a=`expr $a + 1`
10 done
```

Below the script, the terminal prompt is "sh ▾ Tab Width: 8 ▾ Ln 9, Col 14 ▾ INS". The command "bash abc.sh" is run, and the output shows the values 0 through 5 being printed sequentially:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash abc.sh
0
1
2
3
4
5
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

PRACTICAL 8

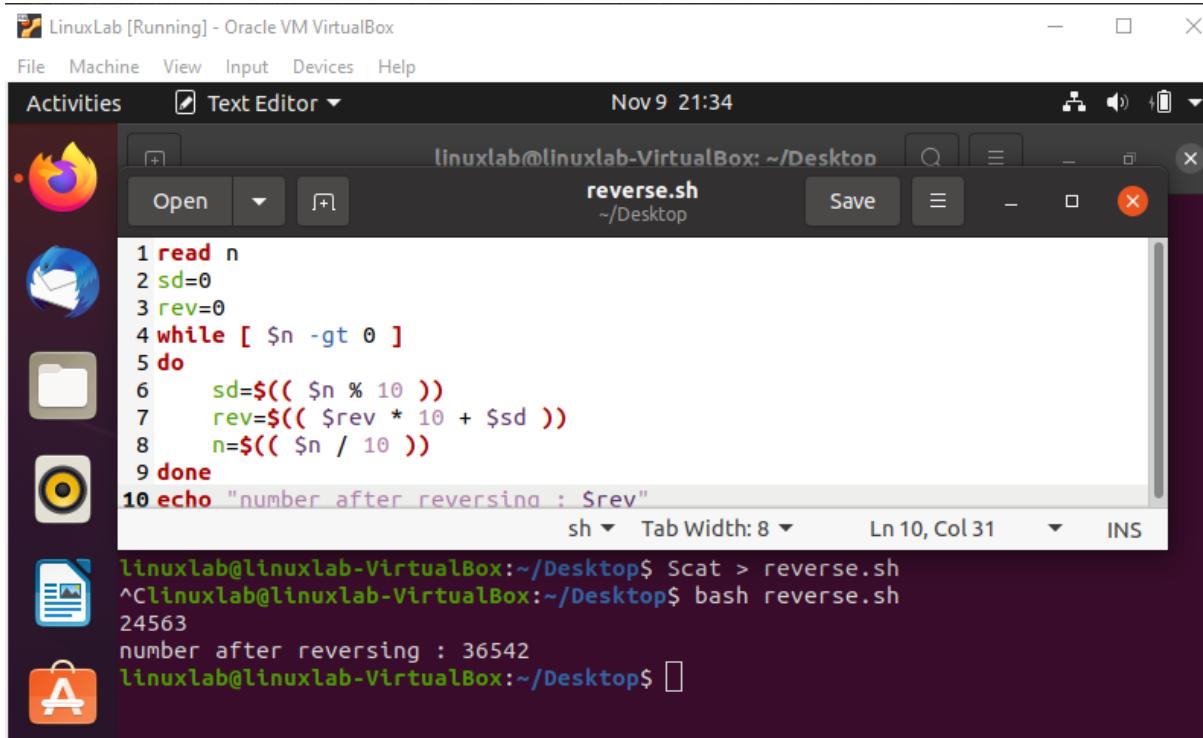
AIM-

Write a shell script to

- a) print a number in reverse order.
- b) to reverse the string and reverse each string further in the list
- c) find the sum of all numbers in a file in Linux
- d) validate password strength. Here are a few assumptions for the password string. Length – minimum of 8 characters, Contain both alphabet and number, Include both the small and capital case letters.

PROCEDURE-

1. Reverse a number



The screenshot shows a Linux desktop environment with a terminal window and a text editor window.

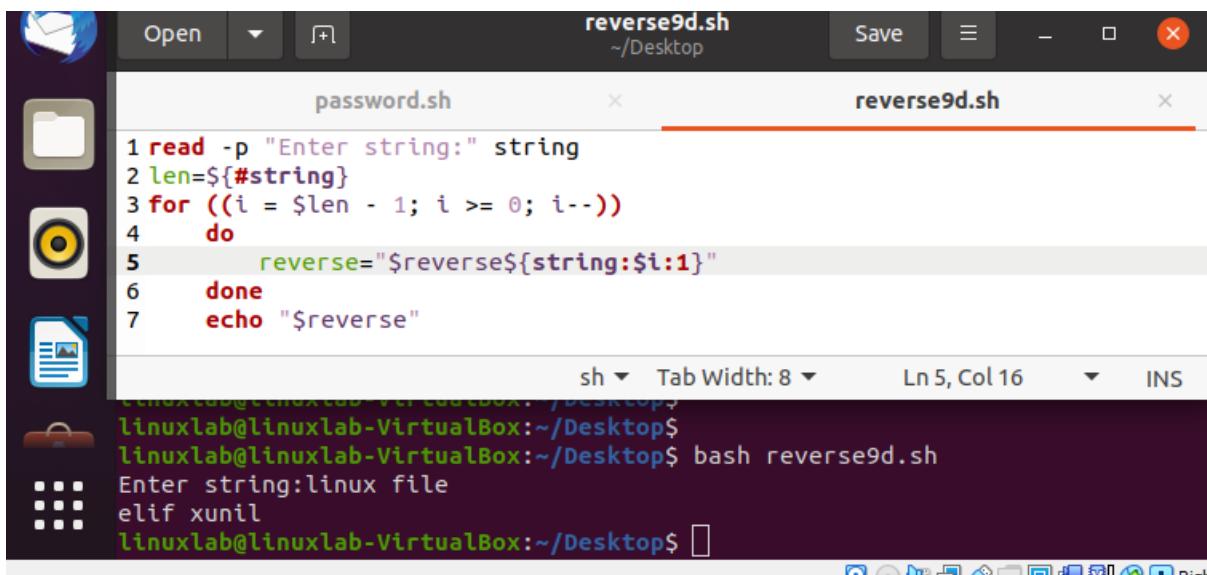
Text Editor Window: The title bar says "reverse.sh ~/Desktop". The code in the editor is:

```
1 read n
2 sd=0
3 rev=0
4 while [ $n -gt 0 ]
5 do
6     sd=$(( $n % 10 ))
7     rev=$(( $rev * 10 + $sd ))
8     n=$(( $n / 10 ))
9 done
10 echo "number after reversing : $rev"
```

Terminal Window: The terminal shows the following session:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ Scat > reverse.sh
^Clinuxlab@linuxlab-VirtualBox:~/Desktop$ bash reverse.sh
24563
number after reversing : 36542
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

2.reverse a string and reverse each string further in the list



The screenshot shows a terminal window with two tabs. The left tab is titled 'password.sh' and contains the following code:

```
1 read -p "Enter string:" string
2 len=${#string}
3 for ((i = $len - 1; i >= 0; i--))
4 do
5     reverse="$reverse${string:$i:1}"
6 done
7 echo "$reverse"
```

The right tab is titled 'reverse9d.sh'. Below the tabs, the terminal prompt is shown:

```
sh ▾ Tab Width: 8 ▾ Ln 5, Col 16 ▾ INS
```

The terminal history shows:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash reverse9d.sh
Enter string:linux file
elif xunil
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

3.Print the sum of all numbers in file

The screenshot shows a terminal window with a dark background. At the top, there's a menu bar with icons for file operations like Open, Save, and Exit. The title bar says "sumInFile.sh ~/Desktop". Below the title bar, there's a status bar showing "sh Tab Width: 8 Ln 6, Col 10 INS". The main area of the terminal contains the following text:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > num.txt
1
2
6
7
4
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > sumInFile.sh
^C
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash sumInFile.sh
20
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

The terminal window has a sidebar on the left with various icons for file operations.

4.validate password strength. Here are a few assumptions for the password string.Length – minimum of 8 characters,Contain both alphabet and number,Include both the small and capital case letters.

The screenshot shows a terminal window with a dark background. At the top, there's a menu bar with icons for file operations like Open, Save, and Exit. The title bar says "password.sh ~/Desktop". Below the title bar, there's a status bar showing "sh Tab Width: 8 Ln 4, Col 45 INS". The main area of the terminal contains the following text:

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ cat > password.sh
1 read P
2 count=`echo ${#pass}`
3 if [[ $count -ne 8 ]];then
4 echo "Password length should be 8 charactore"
5 exit 1;
6 fi
7 echo $count | grep "[A-Z]" | grep "[a-z]" | grep "[0-9]" | grep "[@#$%^&*]"
8
9 if [[ $? -ne 0 ]];then
10
```

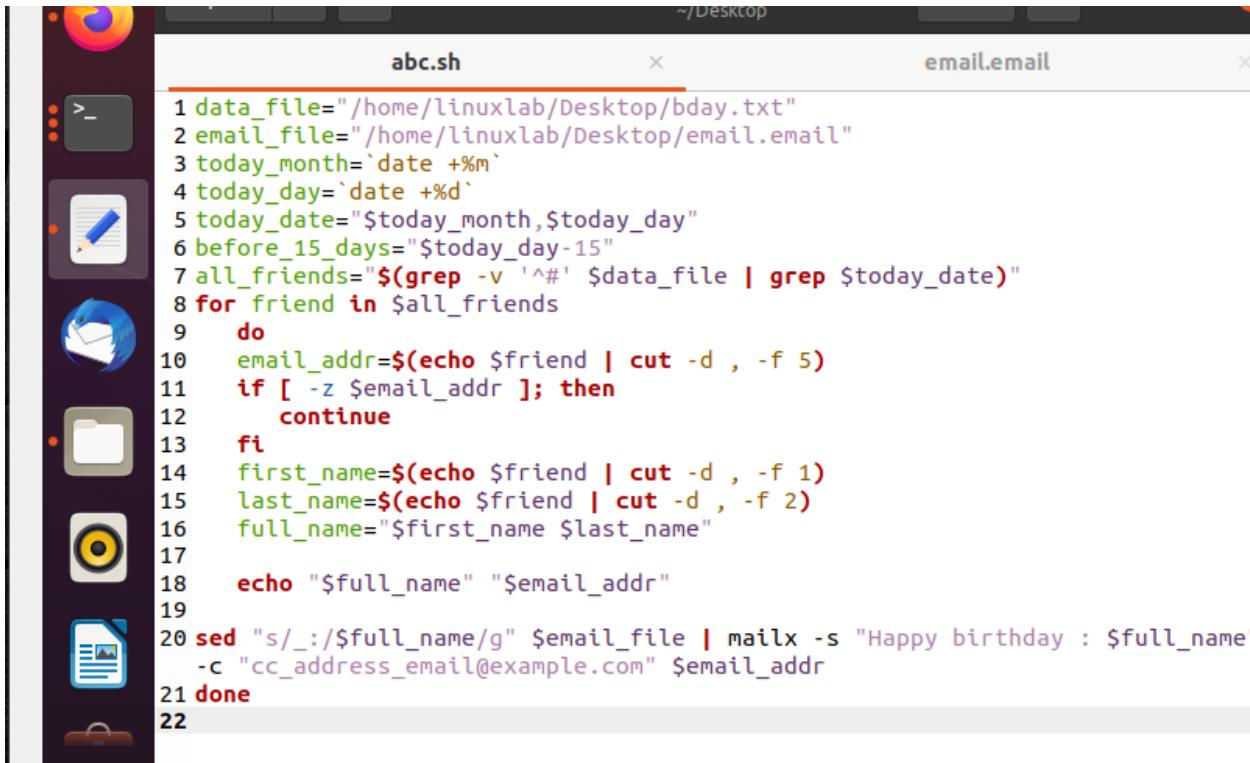
The terminal window has a sidebar on the left with various icons for file operations.

PRACTICAL 9

AIM: Design and develop a “Birthday Reminder” that can automatically send birthday wishes with a personalized message via email.

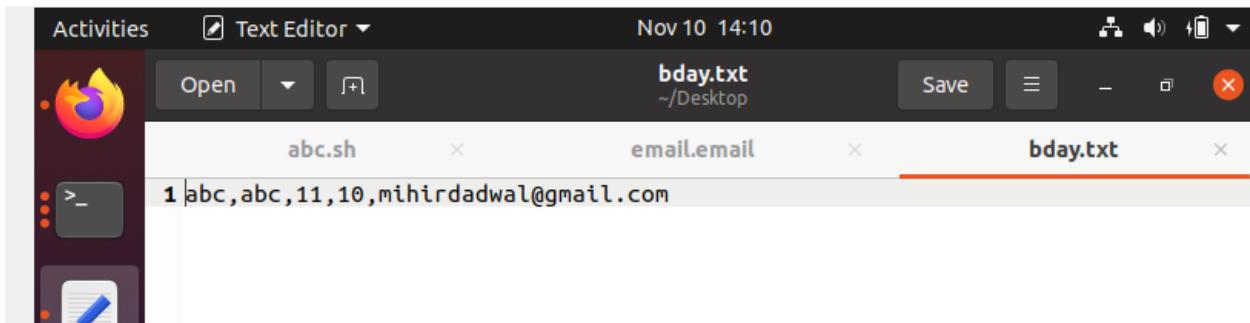
PROCEDURE-

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ sudo apt install ssmtp
[sudo] password for linuxlab:
Reading package lists... Done
Building dependency tree...
Reading state information... Done
The following additional packages will be installed:
  libgnutls-openssl27 libgnutls30
Suggested packages:
  gnutls-bin
The following NEW packages will be installed:
  libgnutls-openssl27 ssmtp
The following packages will be upgraded:
  libgnutls30
1 upgraded, 2 newly installed, 0 to remove and 318 not upgraded.
Need to get 71.0 kB/899 kB of archives.
After this operation, 156 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 libgnutls-openssl27 amd64 3.6.13-2ubuntu1.6 [29.8 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 ssmtp amd64 2.64-8.1ubuntu1 [41.2 kB]
Fetched 71.0 kB in 0s (167 kB/s)
Preconfiguring packages ...
(Reading database ... 184760 files and directories currently installed.)
Preparing to unpack .../libgnutls30_3.6.13-2ubuntu1.6_amd64.deb ...
Unpacking libgnutls30:amd64 (3.6.13-2ubuntu1.6) over (3.6.13-2ubuntu1.3) ...
Setting up libgnutls30:amd64 (3.6.13-2ubuntu1.6) ...
Selecting previously unselected package libgnutls-openssl27:amd64.
(Reading database ... 184760 files and directories currently installed.)
```



The screenshot shows a terminal window with three tabs: abc.sh, email.email, and bday.txt. The abc.sh tab contains a shell script with the following code:

```
1 data_file="/home/linuxlab/Desktop/bday.txt"
2 email_file="/home/linuxlab/Desktop/email.email"
3 today_month=`date +%m`
4 today_day=`date +%d`
5 today_date="$today_month,$today_day"
6 before_15_days="$today_day-15"
7 all_friends=$(grep -v '^#' $data_file | grep $today_date)"
8 for friend in $all_friends
9 do
10   email_addr=$(echo $friend | cut -d , -f 5)
11   if [ -z $email_addr ]; then
12     continue
13   fi
14   first_name=$(echo $friend | cut -d , -f 1)
15   last_name=$(echo $friend | cut -d , -f 2)
16   full_name="$first_name $last_name"
17
18   echo "$full_name" "$email_addr"
19
20 sed "s/_:$full_name/g" $email_file | mailx -s "Happy birthday : $full_name"
-c "cc_address_email@example.com" $email_addr
21 done
22
```



The screenshot shows a terminal window with three tabs: abc.sh, email.email, and bday.txt. The bday.txt tab displays the following content:

```
1|abc,abc,11,10,mihirdadwal@gmail.com
```

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash abc.sh
abc abc mihirdadwal@gmail.com
```

OUTPUT-

Brtday mail sent successfully.

Happy birthday : Mihir Dadwal ➤



Mihir Dadwal <mihirdadwal@gmail.com>

to me ▾

happy birthday
have a good day

The image shows a desktop interface with two windows side-by-side. Both windows have dark grey headers and toolbars.

Top Window: The title bar says ***bday.txt** and **~/Desktop**. The toolbar includes **Open**, **Save**, and standard window controls. The main area contains the following text:

```
1 Mihir,Dadwal,11,10,mihirdadwal@gmail.com
```

Bottom Window: The title bar says **email.email** and **~/Desktop**. The toolbar includes **Open**, **Save**, and standard window controls. The main area contains the following text:

```
1 happy birthday .  
2 have a good day  
3
```

Both windows have a status bar at the bottom with the text **Plain Text**, **Tab Width: 8**, **Ln 1, Col 13** (or 1), and **INS**.

PRACTICAL 10

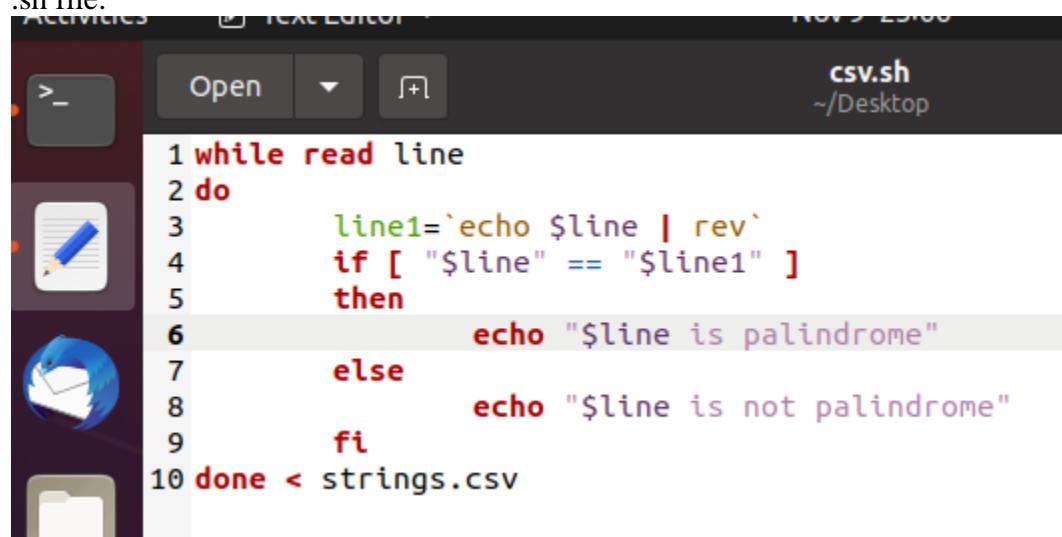
AIM: Check whether strings in csv files are palindrome or not.

Procedure:

Strings present in the csv file:

Fields	
	Column type:
1	Standard
2	abba
3	asdfa
4	linux
	abcdcba

.sh file:



```
1 while read line
2 do
3     line1=`echo $line | rev`
4     if [ "$line" == "$line1" ]
5     then
6         echo "$line is palindrome"
7     else
8         echo "$line is not palindrome"
9     fi
10 done < strings.csv
```

```
linuxlab@linuxlab-VirtualBox:~/Desktop$ bash csv.sh
abba is palindrome
asdfa is not palindrome
linux is not palindrome
abcdcba is palindrome
linuxlab@linuxlab-VirtualBox:~/Desktop$
```

PRACTICAL 11

AIM: Implement NIC Bonding and Teaming.

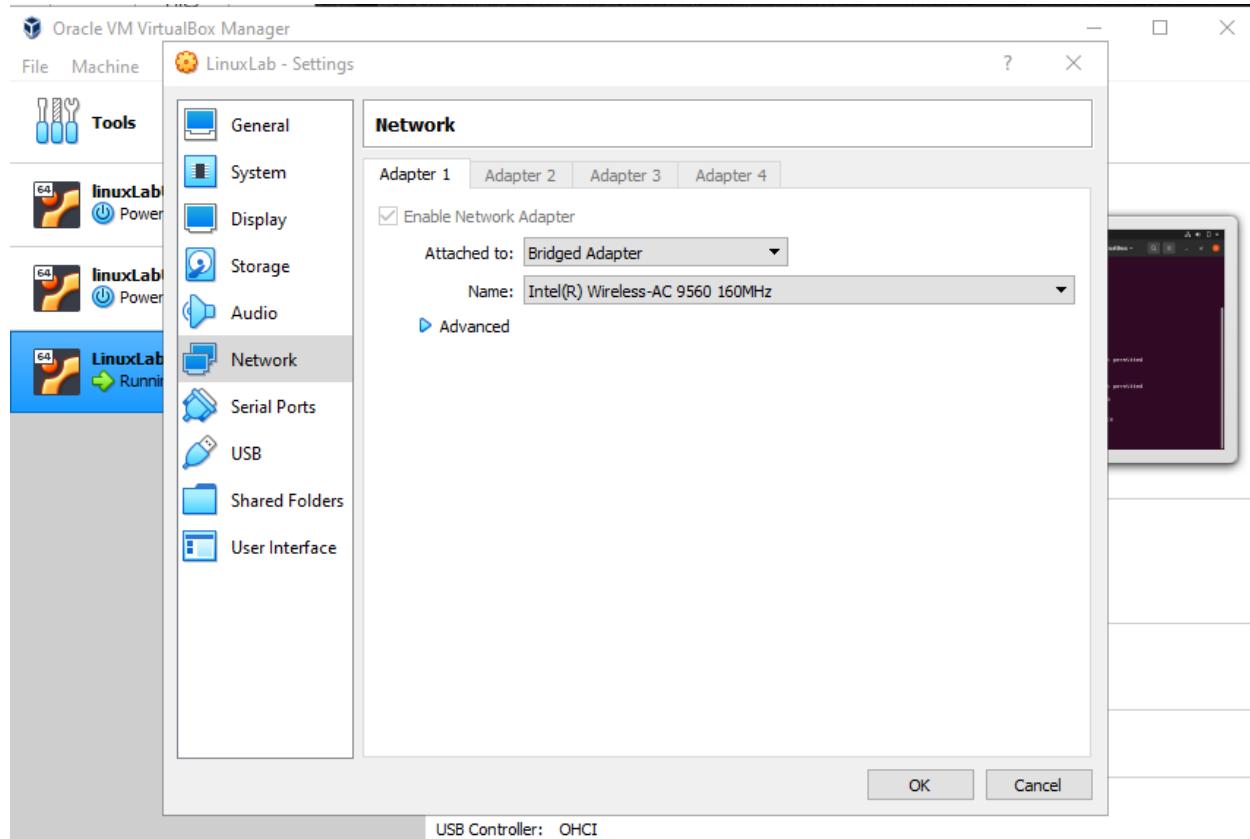
THEORY:

Network Interface Bonding is a mechanism used in Linux servers which consists of binding more physical network interfaces in order to provide more bandwidth than a single interface can provide or provide link redundancy in case of a cable failure. This type of link redundancy has multiple names in Linux, such as **Bonding**, **Teaming** or **Link Aggregation Groups (LAG)**.

Bonding is nothing but Linux kernel feature that allows to aggregate multiple link interfaces (such as eth0, eth1) into a single virtual link such as bond0. The idea is pretty simple get higher data rates and as well as link failover. Linux allows administrators to bind multiple network interfaces together into a single channel using the bonding kernel module and a special network interface called a channel bonding interface. Channel bonding enables two or more network interfaces to act as one, simultaneously increasing the bandwidth and providing redundancy.

```
linuxlab@linuxlab-VirtualBox:~$ ethtool docker0
Settings for docker0:
    Supported ports: [ ]
    Supported link modes:  Not reported
    Supported pause frame use: No
    Supports auto-negotiation: No
    Supported FEC modes: Not reported
    Advertised link modes:  Not reported
    Advertised pause frame use: No
    Advertised auto-negotiation: No
    Advertised FEC modes: Not reported
    Speed: Unknown!
    Duplex: Unknown! (255)
    Port: Other
    PHYAD: 0
    Transceiver: internal
    Auto-negotiation: off
Cannot get wake-on-lan settings: Operation not permitted
    Link detected: no
linuxlab@linuxlab-VirtualBox:~$
```

```
linuxlab@linuxlab-VirtualBox:~$ ethtool lo
Settings for lo:
Cannot get wake-on-lan settings: Operation not permitted
    Link detected: yes
linuxlab@linuxlab-VirtualBox:~$ ethtool enp1s0
Settings for enp1s0:
Cannot get device settings: No such device
Cannot get wake-on-lan settings: No such device
Cannot get message level: No such device
Cannot get link status: No such device
No data available
linuxlab@linuxlab-VirtualBox:~$
```



linuxlab@linuxlab-VirtualBox:~\$ modinfo bonding

```
filename:      /lib/modules/5.11.0-40-generic/kernel/drivers/net/bonding/bonding.ko
author:       Thomas Davis, tadavis@lbl.gov and many others
description:  Ethernet Channel Bonding Driver
license:      GPL
alias:        rtnl-link-bond
srcversion:   6504B4DC3842F2E84C3DFFB
depends:      tls
retpoline:    Y
intree:       Y
name:        bonding
vermagic:    5.11.0-40-generic SMP mod_unload modversions
sig_id:      PKCS#7
signer:      Build time autogenerated kernel key
sig_key:     3D:D4:46:D1:D7:53:DC:D7:67:4C:63:C3:82:F9:49:25:FA:67:F0:15
sig_hashalgo: sha512
signature:   5D:E9:69:91:57:21:DE:B5:24:94:69:77:30:14:D7:FF:57:86:E1:D2:
             11:CE:BE:15:4C:45:0E:81:0A:71:5A:49:05:44:F0:64:BD:D2:D5:03:
             21:B6:1C:DF:8A:05:66:4D:92:64:49:97:E0:2F:6C:FB:F7:A4:70:75:
             9A:9F:5E:32:6E:B3:D0:C0:D5:CC:E2:86:1E:4A:B5:26:E4:80:44:78:
             B8:7A:A3:4C:E0:E9:54:83:F2:6D:96:D2:39:6E:D0:B0:D0:CD:5E:28:
             03:88:FC:35:7D:40:A1:2A:4F:98:A4:F2:02:C1:74:6A:8B:47:43:A6:
             94:28:01:3E:8C:5F:90:5A:89:01:3D:F2:59:98:4D:34:75:AD:2E:32:
             F5:C4:3E:5D:08:E8:3C:04:C6:81:4F:B4:A0:E0:A3:65:99:AF:11:09:
             9F:75:DF:EC:6A:41:E5:BB:0C:8B:45:DC:8A:CC:74:61:35:C1:25:8F:
             03:AE:7B:36:DB:18:42:6A:4A:AC:16:7A:84:50:7C:21:3A:65:26:E5:
             26:DD:FC:99:7D:87:C5:F2:3E:F6:4E:96:8D:1A:23:42:A4:3D:F1:AE:
             E4:C1:7E:79:D5:8E:FC:67:8C:F8:C7:68:C2:13:3F:24:95:69:AF:AB:
```

parm: tx_queues:Max number of transmit queues (default = 16) (int)

parm: num_grat_arp:Number of peer notifications to send on failover event (alias of num_unsol_na) (int)

parm: num_unsol_na:Number of peer notifications to send on failover event (alias of num_grat_arp) (int)

parm: miimon:Link check interval in milliseconds (int)

parm: updelay:Delay before considering link up, in milliseconds (int)

parm: downdelay:Delay before considering link down, in milliseconds (int)

parm: use_carrier:Use netif_carrier_ok (vs MII iocntls) in miimon; 0 for off, 1 for on (default) (int)

parm: mode:Mode of operation; 0 for balance-rr, 1 for active-backup, 2 for balance-xor, 3 for broadcast, 4 for 802.3x load balancing, 5 for 802.3ad link aggregation, 6 for balance-alb (charp)

parm: primary:Primary network device to use (charp)

parm: primary_reselect:Reselect primary slave once it comes up; 0 for always (default), 1 for only if speed of primary is better, 2 for only on active slave failure (charp)

parm: lacp_rate:LACPDU tx rate to request from 802.3ad partner; 0 for slow, 1 for fast (charp)

parm: ad_select:802.3ad aggregation selection logic; 0 for stable (default), 1 for bandwidth, 2 for count (charp)

parm: min_links:Minimum number of available links before turning on carrier (int)

parm: xmit_hash_policy:balance-alb, balance-tlb, balance-xor, 802.3ad hashing method; 0 for layer 2 (default), 1 for layer 3+4, 2 for layer 2+3, 3 for encapsulation 2+3, 4 for encapsulation 3+4 (charp)

parm: arp_interval:arp interval in milliseconds (int)

parm: arp_ip_target:arp targets in n.n.n.n form (array of charp)

parm: arp_validate:validate src/dst of ARP probes; 0 for none (default), 1 for active, 2 for backup, 3 for all (charp)

parm: arp_all_targets:fail on any/all arp targets timeout; 0 for any (default), 1 for all (charp)

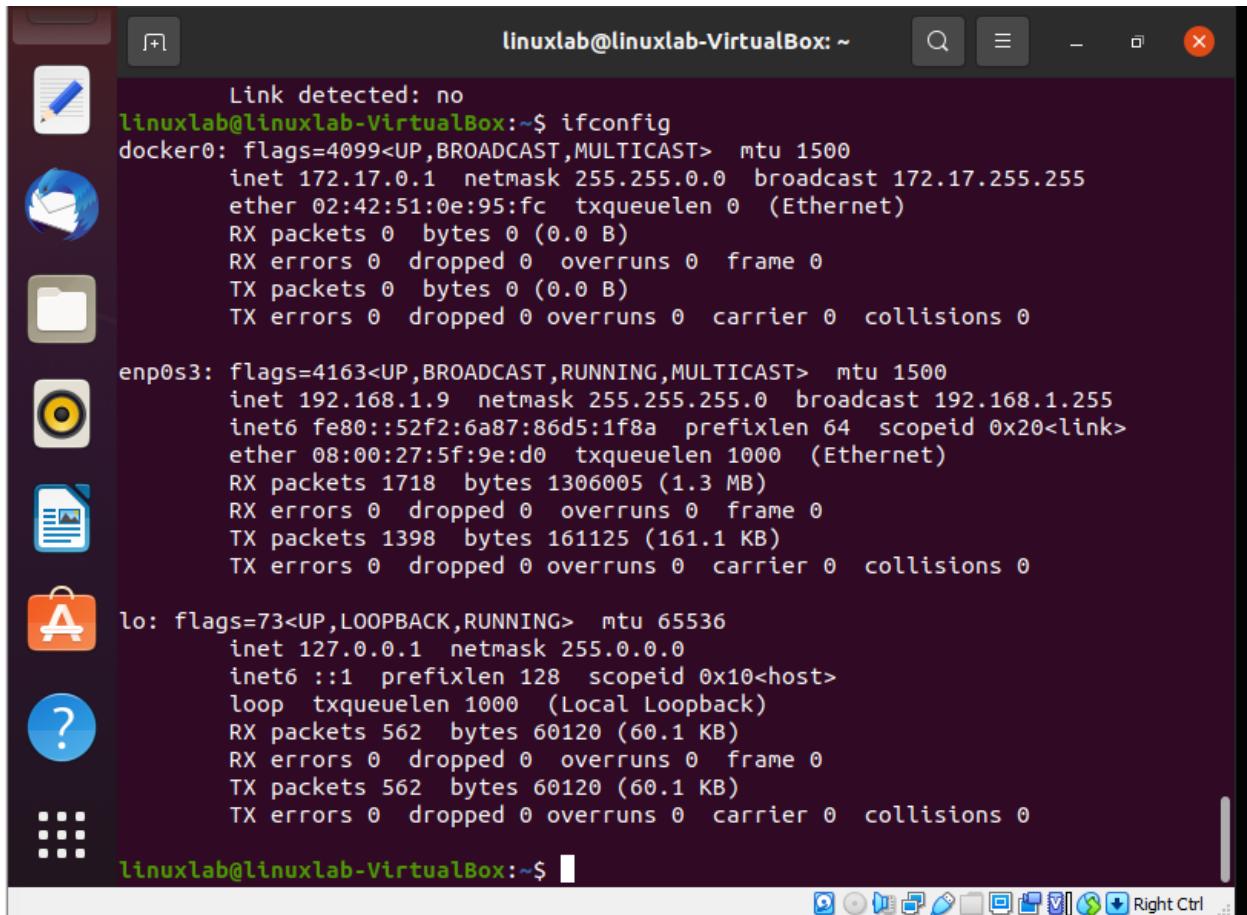
parm: fail_over_mac:For active-backup, do not set all slaves to the same MAC; 0 for none (default), 1 for active, 2 for backup (charp)

parm: all_slaves_active:Keep all frames received on an interface by setting active flag for all slaves; 0 for never (int)

parm: resend_igmp:Number of IGMP membership reports to send on link failure (int)

parm: packets_per_slave:Packets to send per slave in balance-rr mode; 0 for a random slave, 1 packet per slave, >1 packets per slave. (int)

parm: lp_interval:The number of seconds between instances where the bonding driver sends learning packets to the peer switch. The default is 1. (uint)



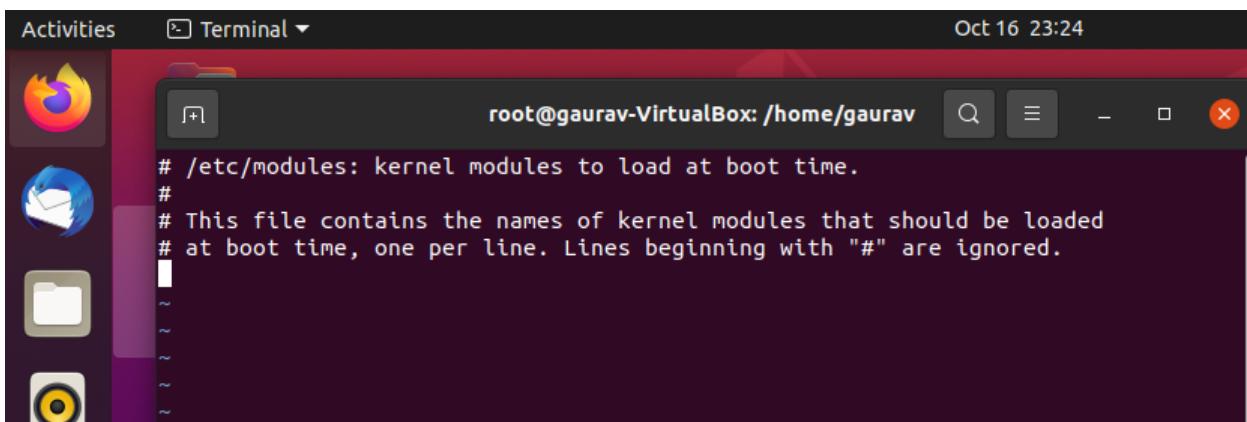
Link detected: no

```
linuxlab@linuxlab-VirtualBox:~$ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
        inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
                ether 02:42:51:0e:95:fc txqueuelen 0 (Ethernet)
                RX packets 0 bytes 0 (0.0 B)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 0 bytes 0 (0.0 B)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.1.9 netmask 255.255.255.0 broadcast 192.168.1.255
        inet6 fe80::52f2:6a87:86d5:1f8a prefixlen 64 scopeid 0x20<link>
                ether 08:00:27:5f:9e:d0 txqueuelen 1000 (Ethernet)
                RX packets 1718 bytes 1306005 (1.3 MB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 1398 bytes 161125 (161.1 KB)
                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
        inet6 ::1 prefixlen 128 scopeid 0x10<host>
                loop txqueuelen 1000 (Local Loopback)
                RX packets 562 bytes 60120 (60.1 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 562 bytes 60120 (60.1 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

linuxlab@linuxlab-VirtualBox:~\$



Activities Terminal ▾ Oct 16 23:24

```
root@gaurav-VirtualBox: /home/gaurav
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
#
#
```

```
root@linuxlab-VirtualBox: /home/linuxlab
linuxlab@linuxlab-VirtualBox:~$ sudo -s
root@linuxlab-VirtualBox:/home/linuxlab# vi /etc/module
root@linuxlab-VirtualBox:/home/linuxlab# vi /etc/modules
root@linuxlab-VirtualBox:/home/linuxlab# vi /etc/modules
root@linuxlab-VirtualBox:/home/linuxlab# vi /etc/network/interfaces
root@linuxlab-VirtualBox:/home/linuxlab# apt-get install ifenslave-2.6
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  vim-runtime
Use 'apt autoremove' to remove it.
The following additional packages will be installed:
  ifenslave ifupdown
Suggested packages:
  rdnssd
The following NEW packages will be installed:
  ifenslave ifenslave-2.6 ifupdown
0 upgraded, 3 newly installed, 0 to remove and 316 not upgraded.
Need to get 75.5 kB of archives.
After this operation, 297 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 ifupdown amd64 0
.8.35ubuntu1 [60.5 kB]
Get:2 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 ifenslave all 2.
9ubuntu1 [13.3 kB]
Get:3 http://in.archive.ubuntu.com/ubuntu focal/universe amd64 ifenslave-2.6 al
l 2.9ubuntu1 [1,740 B]
```

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

#enp0s3 configuration
auto enp0s3
iface enp0s3 inet manual
bond-master bond0
bond-primary enp0s3

#enp0s8 configuration
auto enp0s8
iface enp0s8 inet manual
bond-master bond0

# Bonding enp0s3 & enp0s8 create bond0 NIC
auto bond0
iface bond0 inet static
address 192.168.100.200
gateway 192.168.100.255
netmask 255.255.255.0
bond-mode active-backup
bond-mimon 100
bond-slaves none
```

~ Show Applications
-- INSERT --

```
Bonding Mode: fault-tolerance (active-backup)
Primary Slave: enp0s3 (primary_reselect always)
Currently Active Slave: enp0s3
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0
Peer Notification Delay (ms): 0

Slave Interface: enp0s3
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:15:8b:df
Slave queue ID: 0

Slave Interface: enp0s8
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:e2:f6:1f
Slave queue ID: 0
```

```
root@gaurav-VirtualBox:/home/gaurav# ifconfig
bond0: flags=5187<UP,BROADCAST,RUNNING,MASTER,MULTICAST> mtu 1500
        inet 192.168.100.200 netmask 255.255.255.0 broadcast 192.168.100.255
        inet6 fe80::a00:27ff:fe15:8bdf prefixlen 64 scopeid 0x20<link>
          ether 08:00:27:15:8b:df txqueuelen 1000 (Ethernet)
            RX packets 68 bytes 9431 (9.4 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 55 bytes 7729 (7.7 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
        inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
        ether 02:42:04:4d:0f:ac txqueuelen 0 (Ethernet)
          RX packets 0 bytes 0 (0.0 B)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 0 bytes 0 (0.0 B)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s3: flags=6211<UP,BROADCAST,RUNNING,SLAVE,MULTICAST> mtu 1500
        inet 192.168.100.3 netmask 255.255.255.0 broadcast 192.168.100.255
        inet6 fe80::de61:8b4d:29ce:5bf6 prefixlen 64 scopeid 0x20<link>
          ether 08:00:27:15:8b:df txqueuelen 1000 (Ethernet)
            RX packets 461 bytes 129174 (129.1 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 227 bytes 30431 (30.4 KB)
```