OPERATION SYSTEMS LAB
(Code 18CSC205J)
B.Tech (CSE) – 2nd year/4th Semester

Name: Sahil Yadav
Registration No: RA2011003030176

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING FACULTY OF ENGINEERING & TECHNOLOGY

SRM INSTITUTE OF SCIENCE & TECHNOLOGY, DELHI NCR CAMPUS, MODINAGAR

SIKRI KALAN, DELHI MEERUT ROAD, DIST. – GHAZIABAD - 201204

https://www.srmup.in/

Even Semester (2021-2022)

`

BONAFIDE CERTIFICATE

Registration no. RA2011003030176

Certified to be the bonafide record of work done by Sahil Yadav of 4th semester 2nd year B.TECH degree course in SRM INSTITUTE OF SCIENCE AND TECHNOLOGY, NCR Campus of Department of Computer Science & Engineering, in OPERATING SYSTEM LAB, during the academic year 2021-2022.

Lab In charge

Head of the Department (CSE)

Submitted for university examination held on__/____/___ at SRM IST, NCR Campus.

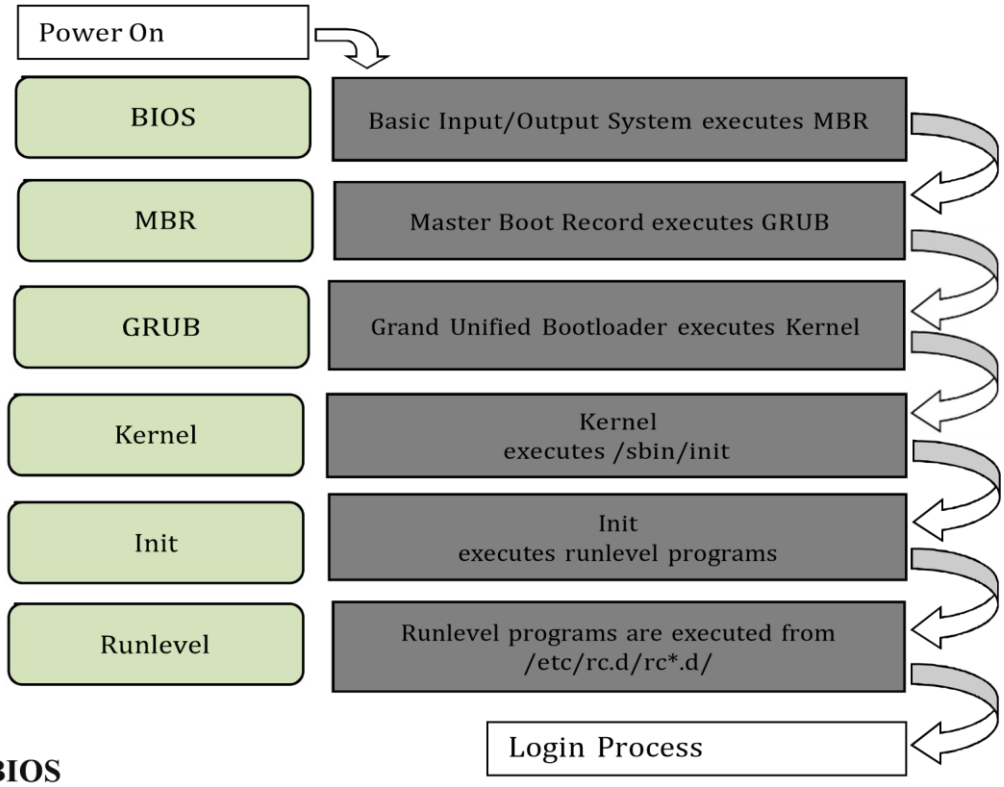
Internal Examiner-I

Internal Examiner-II

INDEX

Ex. No. 1	BOOTING PROCESS OF LINUX	Date: 25/03/22

Press the power button on your system, and after few moments you see the Linux login prompt. From the time you press the power button until the Linux login prompt appears, the following sequence occurs. The following are the 6 high level stages of a typical Linux boot process.



Step 1.BIOS

- BIOS stands for Basic Input/Output System
- Performs some system integrity checks

- Searches, loads, and executes the boot loader program.
- It looks for boot loader in floppy, CD-ROMs, or hard drive. You can press a key (typically F12 or F2, but it depends on your system) during the BIOS startup to change the boot sequence.
- Once the boot loader program is detected and loaded into the memory, BIOS gives the control to it.
- So, in simple terms BIOS loads and executes the MBR boot loader.

Step 2. MBR

- MBR stands for Master Boot Record.
- It is located in the 1st sector of the bootable disk. Typically /dev/hda, or /dev/sda
- MBR is less than 512 bytes in size. This has three components 1) primary boot loader info in 1st 446 bytes 2) partition table info in next 64 bytes 3) mbr validation check in last 2 bytes.

- It contains information about GRUB (or LILO in old systems).
- So, in simple terms MBR loads and executes the GRUB boot loader.

Step 3. GRUB

- GRUB stands for Grand Unified Bootloader.
- If you have multiple kernel images installed on your system, you can choose which one to be executed.
- GRUB displays a splash screen, waits for few seconds, if you don't enter anything, it loads the default kernel image as specified in the grub configuration file.
- GRUB has the knowledge of the filesystem (the older Linux loader LILO didn't understand filesystem).
- Grub configuration file is /boot/grub/grub.conf (/etc/grub.conf is a link to this). The following is sample grub.conf of CentOS.

```
#boot=/dev/sda default=0 timeout=5
splashimage=(hd0,0)/boot/grub/splash.xpm.gz hiddenmenu
title CentOS(2.6.18-194.el5PAE) root(hd0,0)
kernel/boot/vmlinuz-2.6.18-194.el5PAE ro root=LABEL=/ initrd /boot/initrd-2.6.18-194.el5PAE.img
```

- As you notice from the above info, it contains kernel and initrd image.
- So, in simple terms GRUB just loads and executes Kernel and initrd images.

Step 4. Kernel

- Mounts the root file system as specified in the "root=" in grub.conf
- Kernel executes the /sbin/init program
- Since init was the 1st program to be executed by Linux Kernel, it has the process id (PID) of 1. Do a 'ps -ef | grep init' and check the pid.
- initrd stands for Initial RAM Disk.
- initrd is used by kernel as temporary root file system until kernel is booted and the real root file system is mounted. It also contains necessary drivers compiled inside, which helps it to access the hard drive partitions, and other hardware.

Step 5. Init

- Looks at the /etc/inittab file to decide the Linux run level.
- Following are the available run levels
    - 0 – halt
    - 1 – Single user mode
    - 2 – Multiuser, without NFS
    - 3 – Full multiuser mode
    - 4 – unused
    - 5 – X11
    - 6 – reboot
- Init identifies the default initlevel from /etc/inittab and uses that to load all appropriate program.
- Execute 'grep initdefault /etc/inittab' on your system to identify the default run level

- If you want to get into trouble, you can set the default run level to 0 or 6. Since you know what 0 and 6 means, probably you might not do that.
- Typically you would set the default run level to either 3 or 5.

Step 6. Runlevel programs

- When the Linux system is booting up, you might see various services getting started. For example, it might say “starting sendmail …. OK”. Those are the runlevel programs, executed from the run level directory as defined by your run level.
- Depending on your default init level setting, the system will execute the programs from one of the following directories. o Run level 0 – /etc/rc.d/rc0.d/ o Run level 1 – /etc/rc.d/rc1.d/ o Run level 2 – /etc/rc.d/rc2.d/ o Run level 3 – /etc/rc.d/rc3.d/ o Run level 4 – /etc/rc.d/rc4.d/ o Run level 5 – /etc/rc.d/rc5.d/ o Run level 6 – /etc/rc.d/rc6.d/
- Please note that there are also symbolic links available for these directory under /etc directly. So, /etc/rc0.d is linked to /etc/rc.d/rc0.d.
- Under the /etc/rc.d/rc*.d/ directories, you would see programs that start with S and K.
- Programs starts with S are used during startup. S for startup.
- Programs starts with K are used during shutdown. K for kill.
- There are numbers right next to S and K in the program names. Those are the sequence number in which the programs should be started or killed.
- For example, S12syslog is to start the syslog deamon, which has the sequence number of 12. S80sendmail is to start the sendmail daemon, which has the sequence number of 80. So, syslog program will be started before sendmail.

Login Process

1. Users enter their username and password
2. The operating system confirms your name and password.
3. A "shell" is created for you based on your entry in the "/etc/passwd" file
4. You are "placed" in your "home"directory.
5. Start-up information is read from the file named "/etc/profile". This file is known as the system login file. When every user logs in, they read the information in this file.
6. Additional information is read from the file named ".profile" that is located in your "home" directory. This file is known as your personal login file.

Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	**Date :**

Ex. No. 2	BASIC LINUX COMMANDS	Date : 01/04/22

a) Basics

1. echo SRM to display the string SRM
2. clear to clear the screen
3. date to display the current date and time
4. cal 2003 to display the calendar for the year 2003 cal 6 2003 to display the calendar for the June-2003
5. passwd to change password

b) Working with Files

1. ls list files in the present working directory ls –l list files with detailed information (long list) ls –a list all files including the hidden files
2. cat > f1 to create a file (Press ^d to finish typing)
3. cat f1 display the content of the file f1
4. wc f1 list no. of characters, words & lines of a file f1 wc –c f1 list only no. of characters of file f1 wc –w f1 list only no. of words of file f1 wc –l f1 list only no. of lines of file f1
5. cp f1 f2 copy file f1 into f2
6. mv f1 f2 rename file f1 as f2
7. rm f1 remove the file f1
8. head –5 f1 list first 5 lines of the file f1 tail –5 f1 list last 5 lines of the file f1

c) Working with Directories

1. mkdir elias to create the directory elias
2. cd elias to change the directory as elias
3. rmdir elias to remove the directory elias
4. pwd to display the path of the present working directory

5. cd to go to the home directory cd .. to go to the parent directory cd - to go to the previous working directory cd / to go to the root directory

d) File name substitution

1. ls f? list files start with 'f' and followed by any one character

2. ls *.c list files with extension 'c'

3. ls [gpy]et list files whose first letter is any one of the character g, p or y and followed by the word et

4. ls [a-d,l-m]ring list files whose first letter is any one of the character from a to d and l to m and followed by the word ring.

e) I/O Redirection

1. Input redirection

   wc –l < ex1 To find the number of lines of the file 'ex1'

2. Output redirection

   who > f2 the output of 'who' will be redirected to file f2

3. cat >> f1 to append more into the file f1

f) Piping

Syntax : Command1 | command2

Output of the command1 is transferred to the command2 as input. Finally output of the command2 will be displayed on the monitor.

ex. cat f1 | more list the contents of file f1 screen by screen

head –6 f1 |tail –2 prints the 5th & 6th lines of the file f1.

g) Environment variables

1. echo $HOME display the path of the home directory

2. echo $PS1 display the prompt string $

3. echo $PS2 display the second prompt string (> symbol by default)

4. echo $LOGNAME login name

5. echo $PATH list of pathname where the OS searches for an executable file

h) File Permission

-- chmod command is used to change the access permission of a file.

Method-1

Syntax : chmod [ugo] [+/-] [rwxa] filename

u : user, g : group, o : others
+ : Add permission - : Remove the permission r
: read, w : write, x : execute, a : all permissions

ex. chmod ug+rw f1
adding 'read & write' permissions of file f1 to both user and group members.

Method-2

Syntax : chmod octnum file1

The 3 digit octal number represents as follows

- first digit -- file permissions for the user
- second digit -- file permissions for the group
- third digit -- file permissions for others

Each digit is specified as the sum of following
4 – read permission, 2 – write permission, 1 – execute permission

ex. chmod 754 f1

it change the file permission for the file as follows

- read, write & execute permissions for the user ie; 4+2+1 = 7
- read, & execute permissions for the group members ie; 4+0+1 = 5
- only read permission for others ie; 4+0+0 = 4

Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	Date :

Ex. No. 3	LINUX FILE SYSTEM	Date : 22/04/22

Linux File System

Linux File System or any file system generally is a layer which is under the operating system that handles the positioning of your data on the storage, without it; the system cannot knows which file starts from where and ends where.

Linux offers many file systems types like:

- Ext: an old one and no longer used due to limitations.
- Ext2: first Linux file system that allows 2 terabytes of data allowed.
- Ext3: came from Ext2, but with upgrades and backward compatibility.
- Ext4: faster and allow large files with significant speed.(Best Linux File System) It is a very good option for SSD disks and you notice when you try to install any Linux distro that this one is the default file system that Linux suggests.
- JFS: old file system made by IBM. It works very well with small and big files, but it failed and files corrupted after long time use, reports say.
- XFS: old file system and works slowly with small files.
- Btrfs: made by Oracle. It is not stable as Ext in some distros, but you can say that it is a replacement for it if you have to. It has a good performance.

File System Structure

The following table provides a short overview of the most important higher-level directories you find on a Linux system

Directory	Contents
/	Root directory—the starting point of the directory tree.
/bin	Essential binary files. Binary Executable files
/boot	Static files of the boot loader.
/dev	Files needed to access host-specific devices.
/etc	Host-specific system configuration files.
/lib	Essential shared libraries and kernel modules.
/media	Mount points for removable media.

/mnt	Mount point for temporarily mounting a file system.
/opt	Add-on application software packages.
/root	Home directory for the superuser root.
/sbin	Essential system binaries.
/srv	Data for services provided by the system.
/tmp	Temporary files.
/usr	Secondary hierarchy with read-only data.
/var	Variable data such as log files

Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	**Date :**

Ex. No. 4	EDITORS AND FILTERS	Date : 29/04/22

VI EDITOR

- vi fname to open the file fname

- There are two types of mode in vi editor
  Escape mode – used to give commands – to switch to escape mode, press <Esc> key
  Command mode – used to edit the text – to switch to command mode, press any one the following inserting text command

a) Inserting Text i insert text before the cursor
a append text after the cursor I insert text at the beginning of the line A append text to the end of the line
r replace character under the cursor with the next character typed R Overwrite characters until the end of the line
o (small o) open new line after the current line to type text
O (capital O) open new line before the current line to type text

b) Cursor movements h left
j down
k up
l right
(The arrow keys usually work also)

^F forward one screen
^B back one screen
^D down half screen
^U up half screen
(^ indicates control key; case does not matter)

0 (zero) beginning of line
$ end of line

c) Deleting text
Note : (n) indicates a number, and is optional

dd deletes current line
(n) dd deletes (n) line(s) ex. 5dd deletes 5 lines
(n)dw deletes (n) word(s)
D deletes from cursor to end of line x deletes current character (n)x deletes (n) character(s)

X deletes previous character

d) Saving files

:w to save & resume editing (write & resume)

:wq to save & exit (write & quit)

:q! quit without save

e) Cut, Copy and Paste yy copies current line

(n) yy copies (n) lines from the current line. ex. 4yy copies 4 lines. p

 paste deleted or yanked (copied) lines after the cursor

FILTERS

1. cut

- Used to cut characters or fileds from a file/input

Syntax : cut -cchars filename
-ffieldnos filename

- By default, tab is the filed separator(delimiter). If the fileds of the files are separated by any other character, we need to specify explicitly by –d option

cut -ddelimitchar -ffileds filname

2. paste

- Paste files vertically. That is n^{th} line of first file and n^{th} line of second file are pasted as the nth line of result

Syntax : paste file1 file2

-ddchar option is used to paste the lines using the delimiting character dchar

-s option is used paste the lines of the file in a single line

3. tr

- Used to translate characters from standard input

Syntax : tr char1 char2 < filename

It translates char1 into char2 in file filename

- Octal representation characters can also be used

Octal value	Character
'\7'	Bell
'\10'	Backspace
'\11'	Tab
'\12'	Newline

'\33' Escape
Ex. tr : '\11' < f1 translates all : into tab of f ile f1

-s Option translate multiple occurrences of a character by single character.
-d Option is to delete a character

4. grep
 Used to search one or more files for a particular pattern.

Syntax : grep pattern filename(s)

- Lines that contain the pattern in the file(s)
- get displayed pattern can be any regular expressions
- More than one files can be searched for a pattern

-v option displays the lines that do not contain the pattern
-l list only name of the files that contain the pattern
-n displays also the line number along with the lines that matches the pattern

5. sort
 Used to sort the file in order

-
-

Syntax : sort filename
Sorts the data as text by default
Sorts by the first filed by default

-r option sorts the file in descending order
-u eliminates duplicate lines
-o filename writes sorted data into the file fname
-tdchar sorts the file in which fileds are separated by dchar
-n sorts the data as number
+1n skip first filed and sort the file by second filed numerically

6. Uniq
 Displays unique lines of a sorted file
Syntax : uniq filename

-d option displays only the duplicate lines -c
displays unique lines with no. of occurrences.

7. cmp
 Used to compare two files

Syntax : cmp f1 f2 compare two files f1 & f2 and prints the line of first difference .

8. diff

 Used to differentiate two files

Syntax : diff f1 f2 compare two files f1 & f2 and prints all the lines that are differed between f1 & f2.

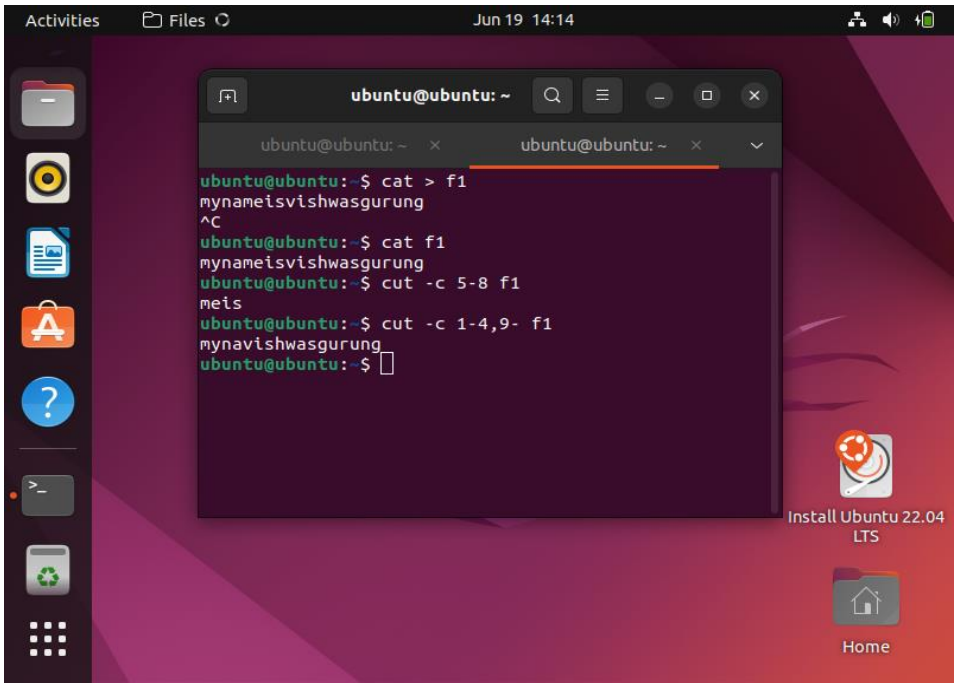
9. comm

 Used to compare two sorted files Syntax : comm file1 file2

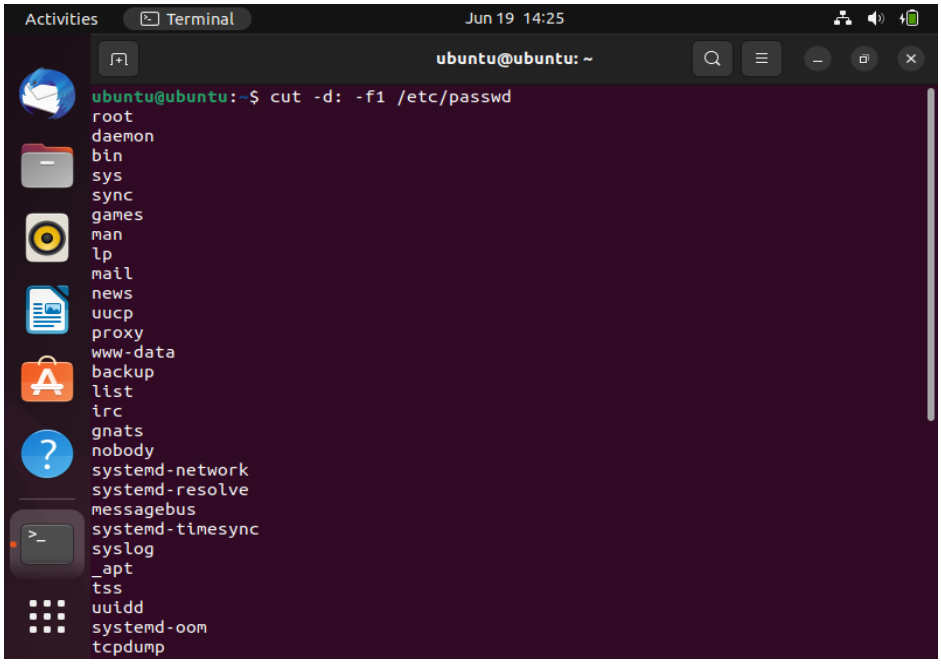
Three columns of output will be displayed.
First column displays the lines that are unique to file1
Second column displays the lines that are unique to file2
Third column displays the lines that are appears in both the files

-1	option suppress first column
-2	option suppress second column
-3	option suppress third column
-12	option display only third column
-13	option display only second column
-23	option display only first column

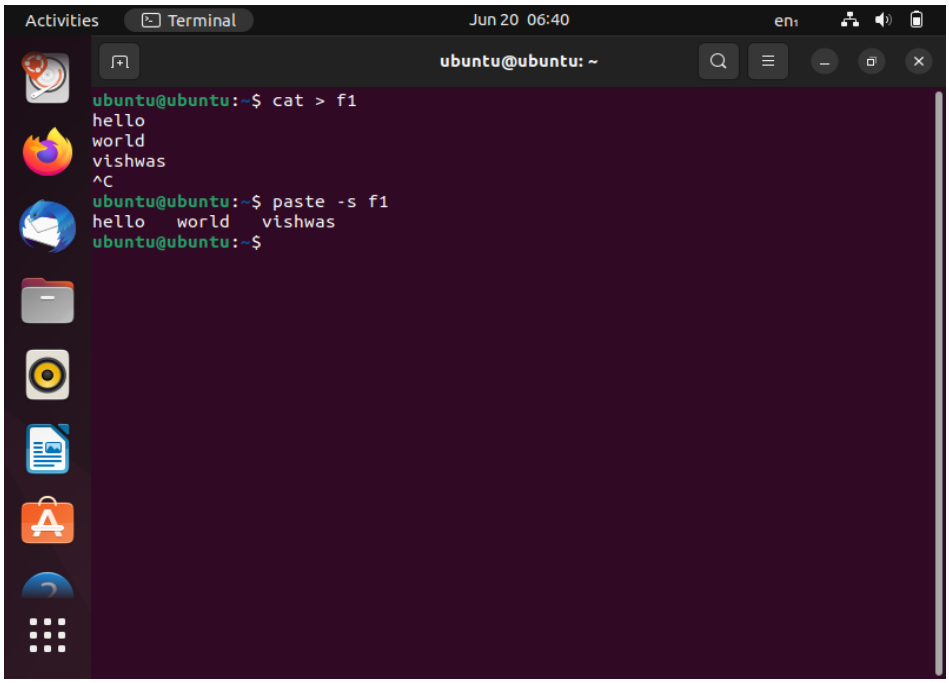
Q1. Write a command to cut 5 to 8 characters of the file f1.



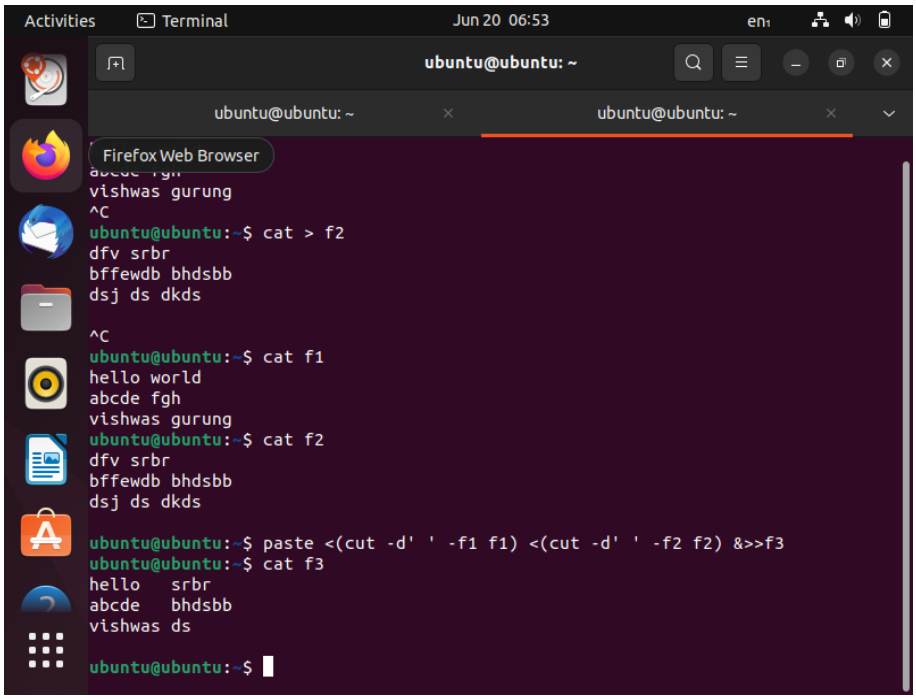
Q2. Write a command to display user-id of all the users in your system.



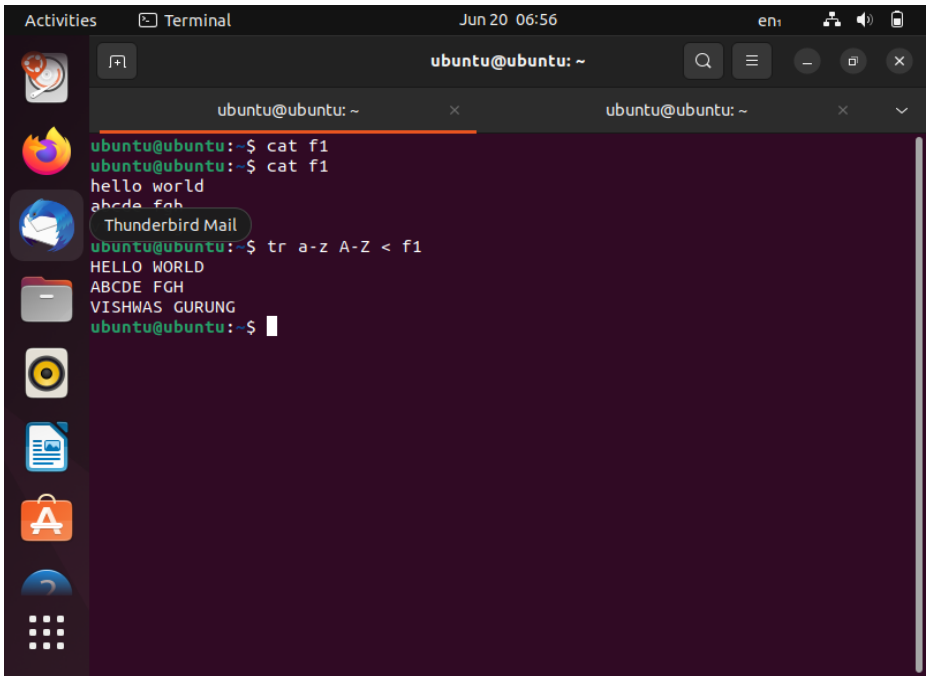
Q3. Write a command to paste all the lines of the file f1 into single line



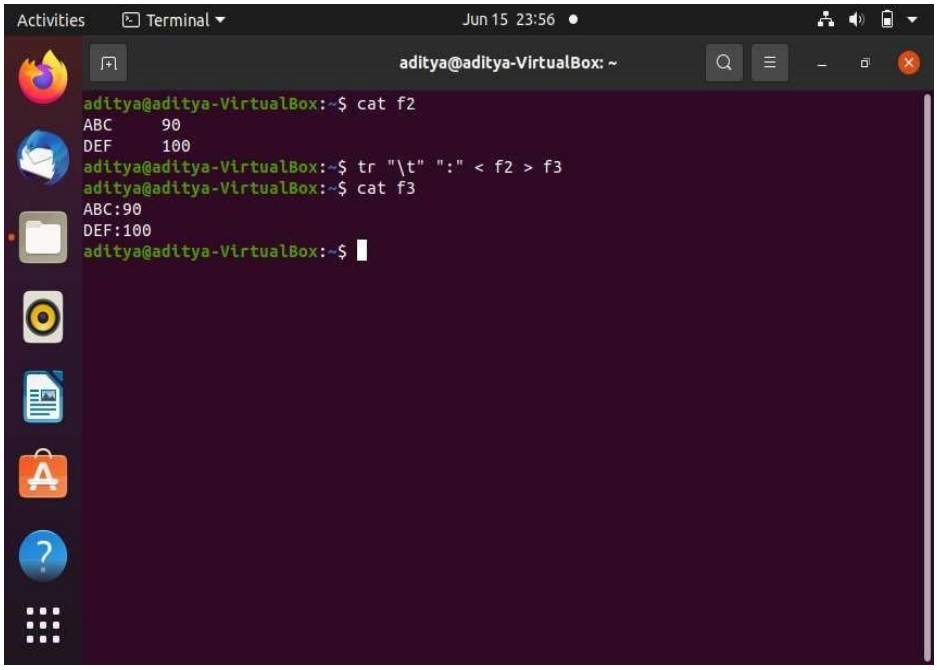
Q4. Write a command to cut the first field of file f1 and second field of file f2 and paste into the file f3.



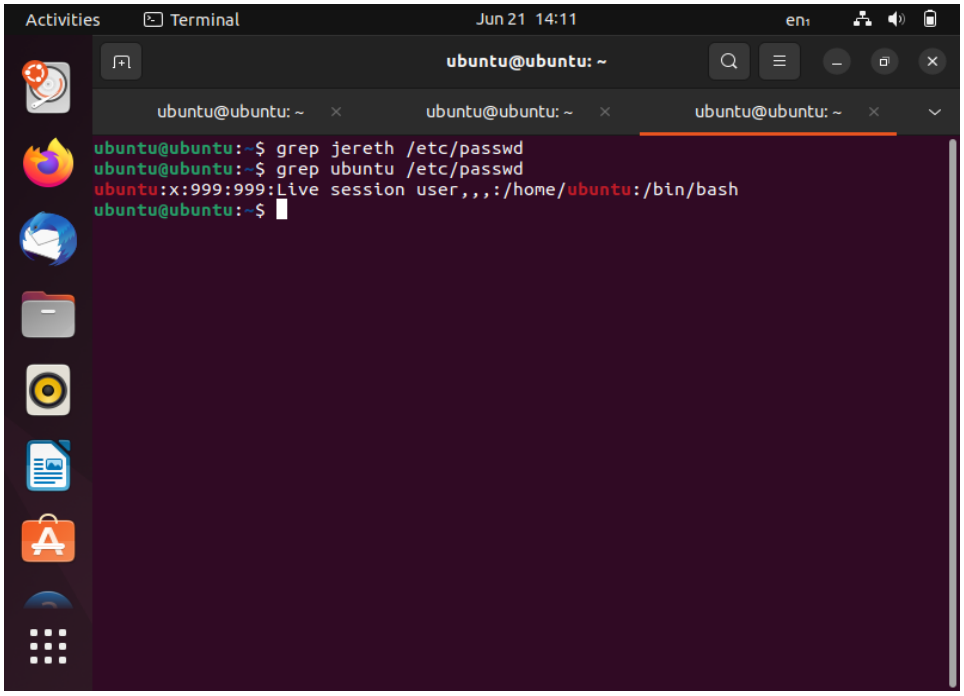
Q5. Write a command to change all small case letters to capitals of file f2.



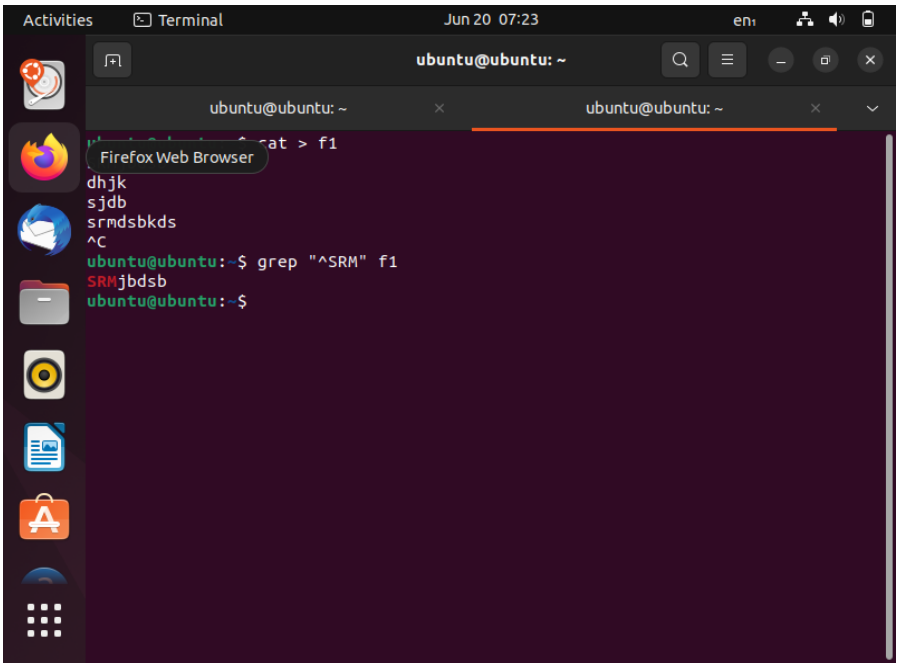
Q6. Write a command to replace all tab character in the file f2 by :



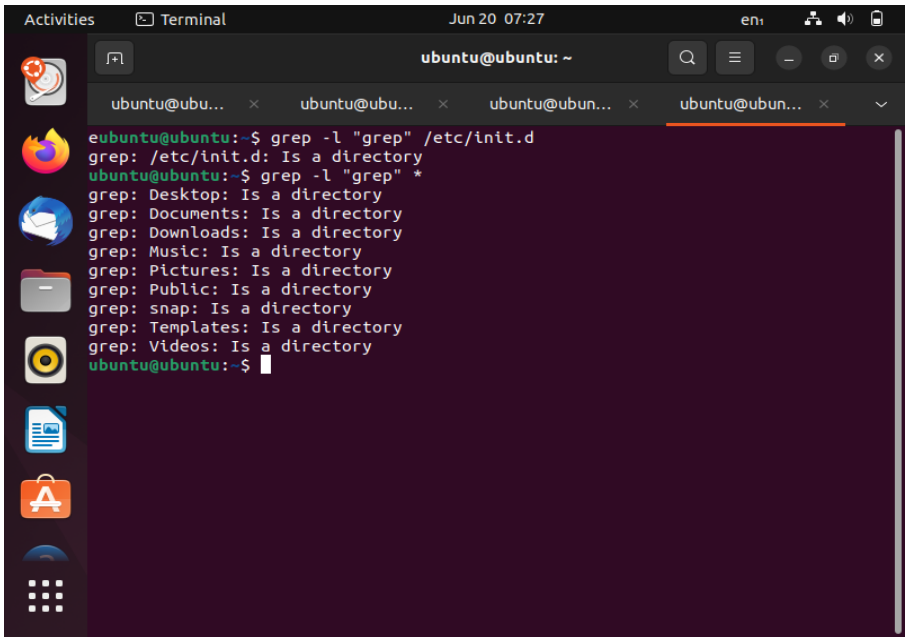
Q7. Write a command to check whether the user judith is available in your system or not. (use grep)



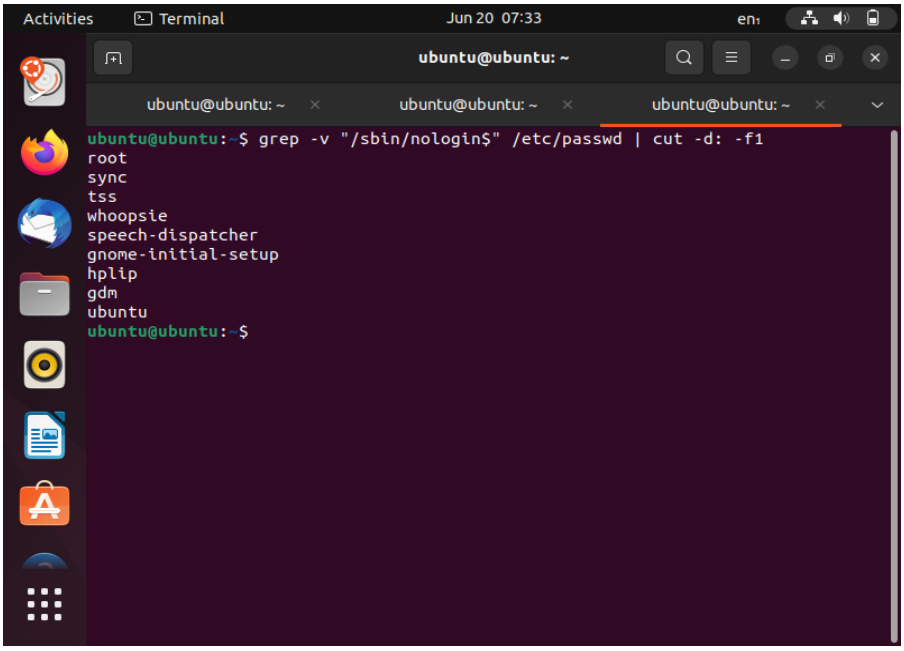
Q8. Write a command to display the lines of the file f1 starts with SRM.



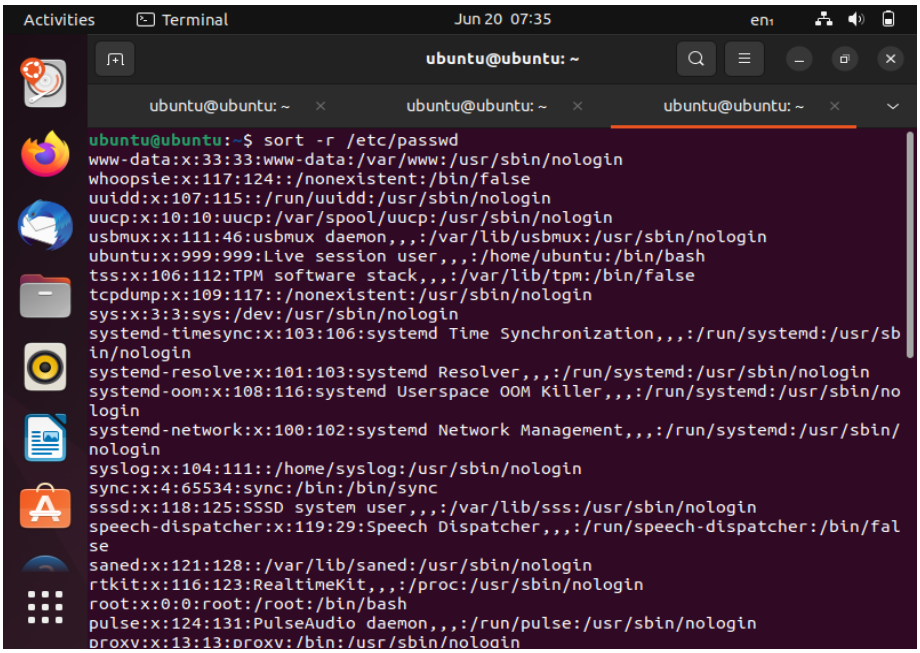
Q9. Write a command to display the name of the files in the directory /etc/init.d that contains the pattern grep.



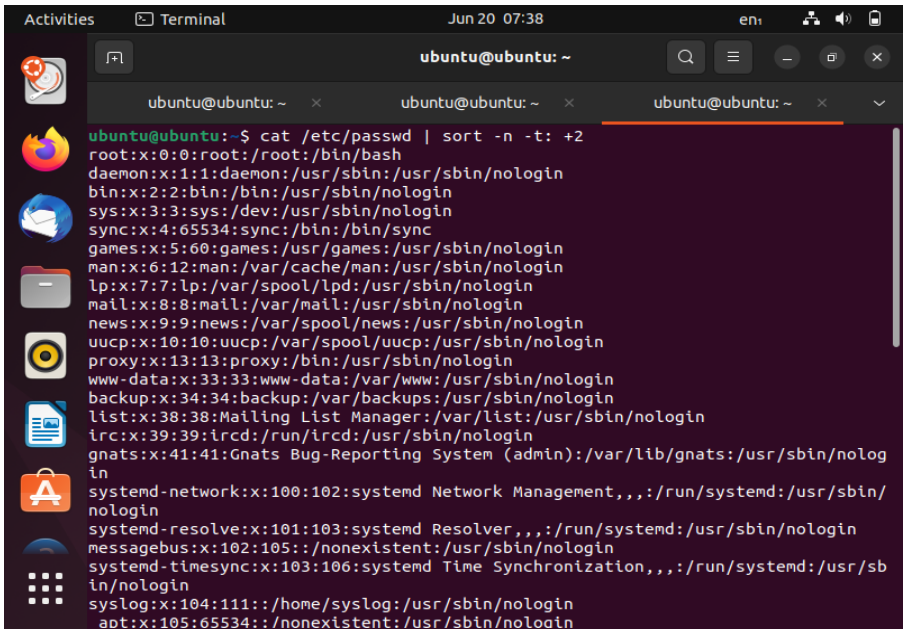
Q10. Write a command to display the names of nologin users. (Hint: the command nologin is specified in the last filed of the file /etc/passwd for nologin users)



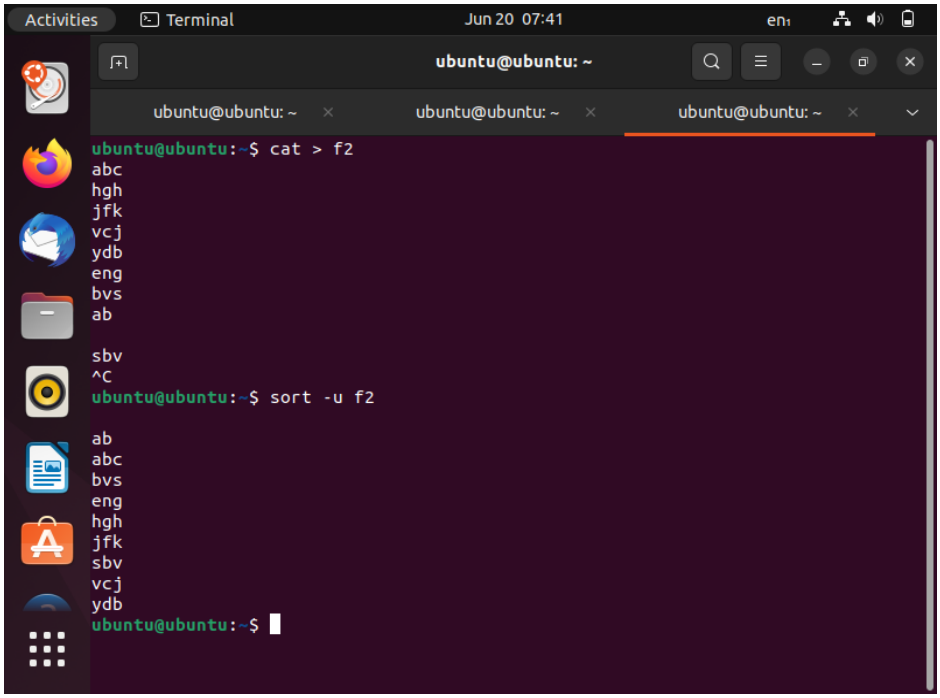
Q11. Write a command to sort the file /etc/passwd in descending order



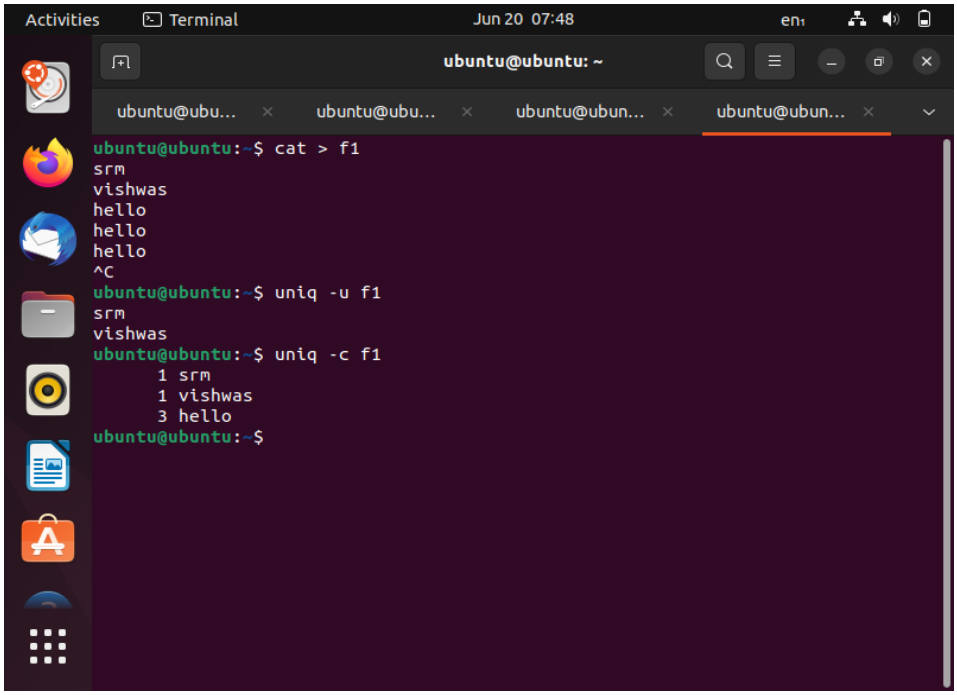
Q12. Write a command to sort the file /etc/passwd by user-id numerically. (Hint : user-id is in 3rd field)



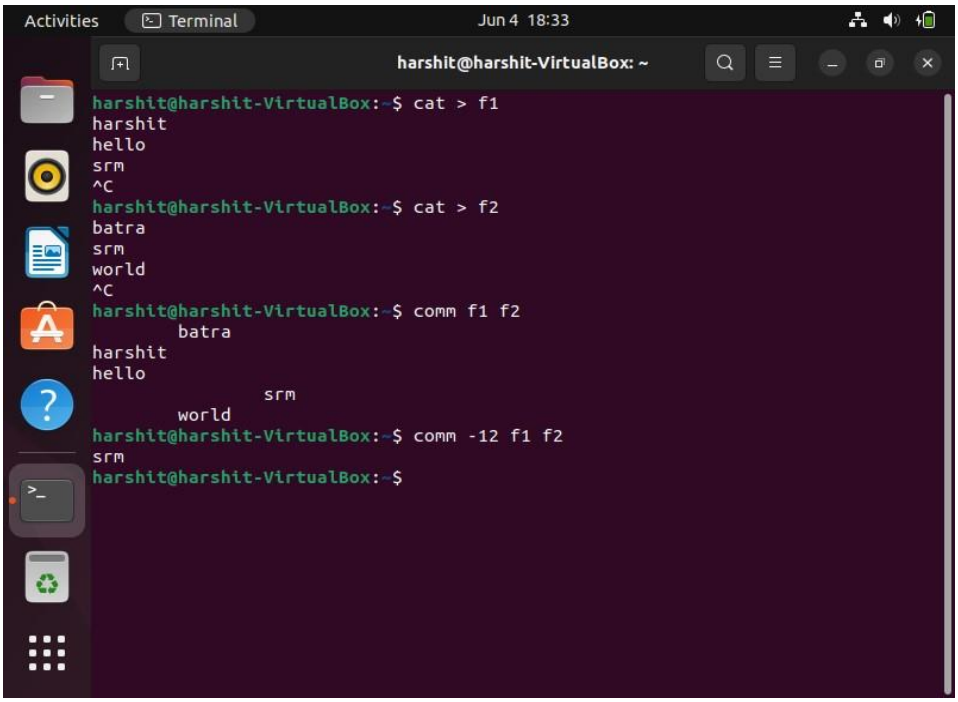
Q13. Write a command to sort the file f2 and write the output into the file f22. Also eliminate duplicate lines.



Q14. Write a command to display the unique lines of the sorted file f21. Also display the number of occurrences of each line.



Q15. Write a command to display the lines that are common to the files f1 and f2.



Result-The given program has been performed successfully.

Staff In-charge Sign :	**Date :**

Ex. No. 5	COMPILATION OF C PROGRAM	Date : 06/05/22

Compilation of C Program

Step 1 : Open the terminal and edit your program using vi editor/gedit editor and save with extension ".c"
Ex. vi test.c
(or) gedit text.c

Step 2 : Compile your program using gcc compiler
Ex. gcc test.c Output file will be "a.out"
(or) gcc –o test text.c Output file will be "test"

Step 3 : Correct the errors if any and run the program
Ex. ./a.out
or ./test

Optional Step : In order to avoid ./ prefix each time a program is to be executed, insert the following as the last line in the file .profile export PATH=.:$PATH
This Step needs only to be done once.

Debug C Programs using gdb debugger

Step 1 : Compile C program with debugging option –g Ex. gcc –g test.c

Step 2 : Launch gdb. You will get gdb prompt
Ex. gdb a.out

Step 3 : Step break points inside C program
Ex. (gdb) b 10
Break points set up at line number 10. We can have any number of break points

Step 4 : Run the program inside gdb
Ex. (gdb) r

Step 5 : Print variable to get the intermediate values of the variables at break point Ex. (gdb) p i Prints the value of the variable 'i'

Step 6 : Continue or stepping over the program using the following gdb commands c continue till the next break
n Execute the next line. Treats function as single statement s Similar to 'n' but executes function statements line by line l List the program statements

Step 7 : Quit the debugger
(gdb) q

Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	**Date :**

Ex. No. 6	PROCESS CREATION	Date : 13/05/22

Syntax for process creation

int fork();

Returns 0 in child process and child process ID in parent process.

Other Related Functions int getpid() returns the current process ID int getppid() returns the parent process ID

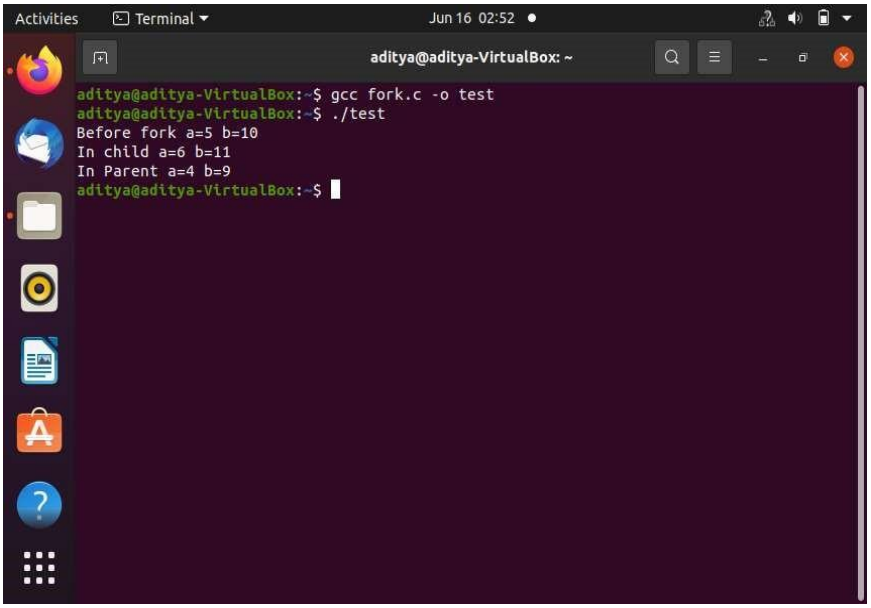
wait() makes a process wait for other process to complete

Virtual fork vfork() function is similar to fork but both processes shares the same address space.

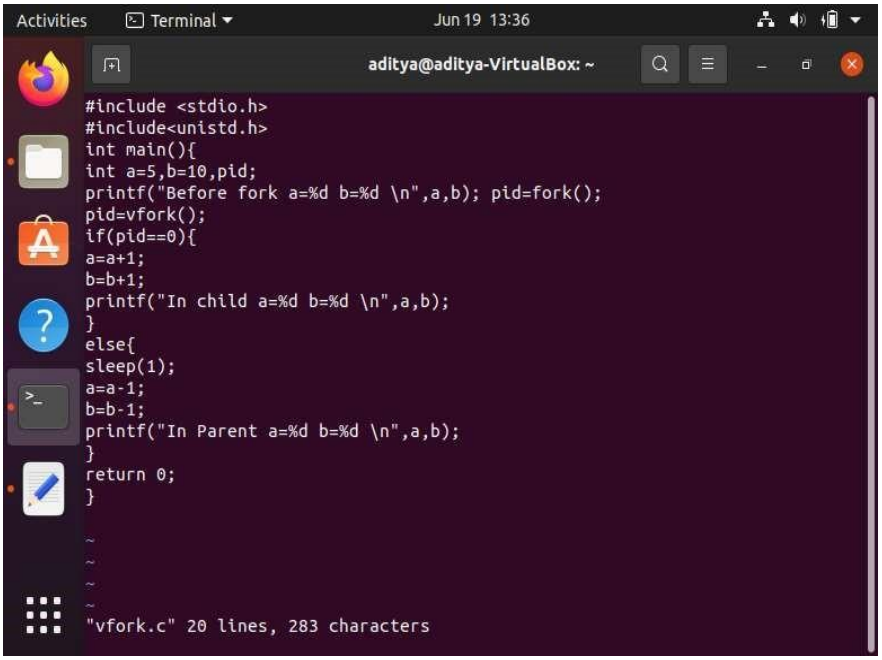
Q1. Find the output of the following program

```
#include <stdio.h>
#include<unistd.h>

int main()
{
   int a=5,b=10,pid;
printf("Before fork a=%d b=%d \n",a,b); pid=fork();

   if(pid==0)
   {
      a=a+1; b=b+1;
      printf("In child a=%d b=%d \n",a,b);
   }
   else
   {
      sleep(1); a=a-
      1; b=b-1;
      printf("In Parent a=%d b=%d \n",a,b);
   }
   return 0;
}
```

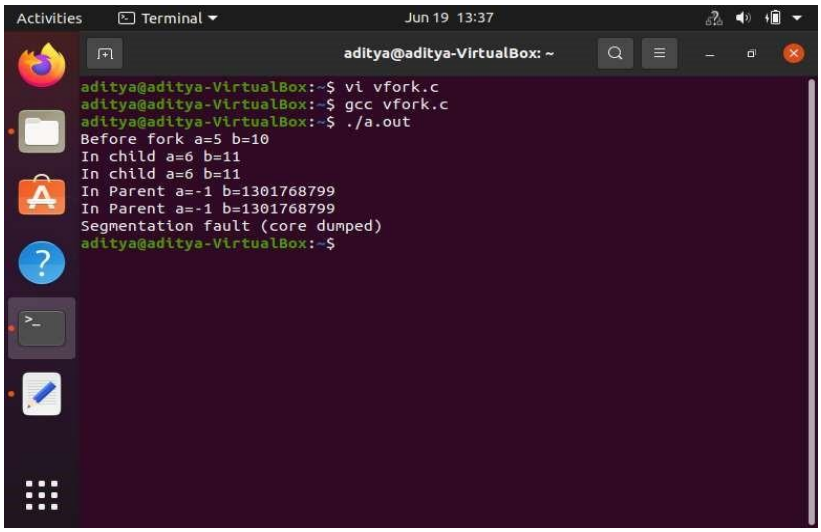
Output :-



Q2. Rewrite the program in Q1 using vfork() and write the output.



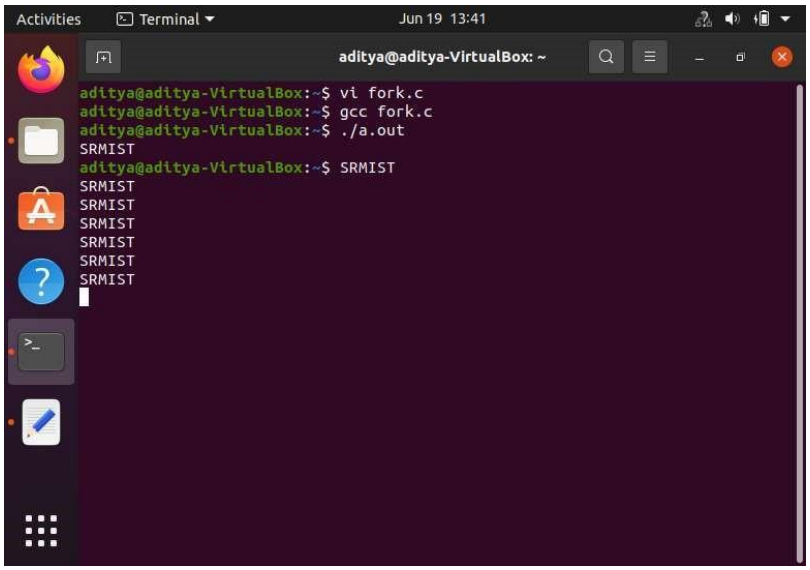
Output :-



Q3. Calculate the number of times the text “SRMIST” is printed.

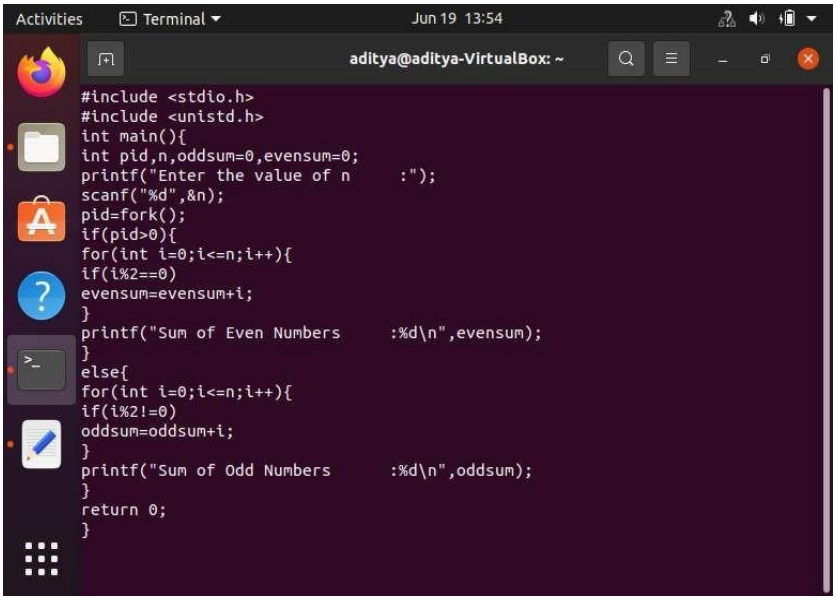
```
#include
<stdio.h> #include<unistd.
h>

int main()
{
fork();    fork();
                fork();
printf(“SRMIST\n”);        return 0;
}
```

Output :

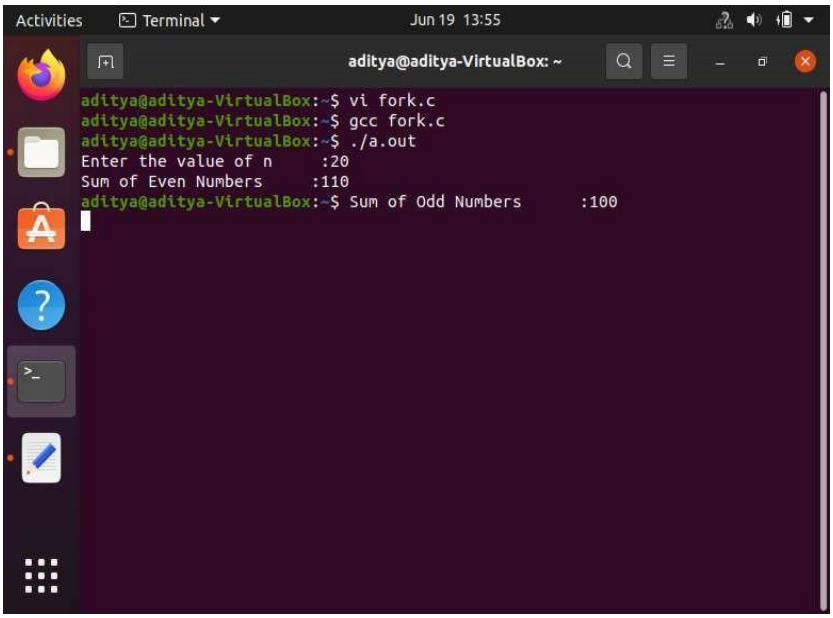


Q4. Complete the following program as described below :

The child process calculates the sum of odd numbers and the parent process calculate the sum of even numbers up to the number ‘n’. Ensure the Parent process waits for the child process to finish.



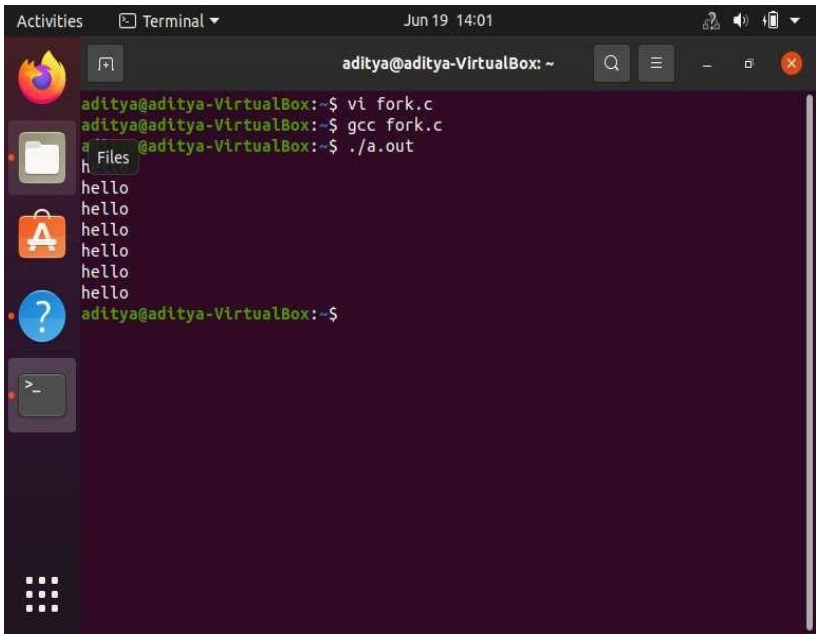
Output :



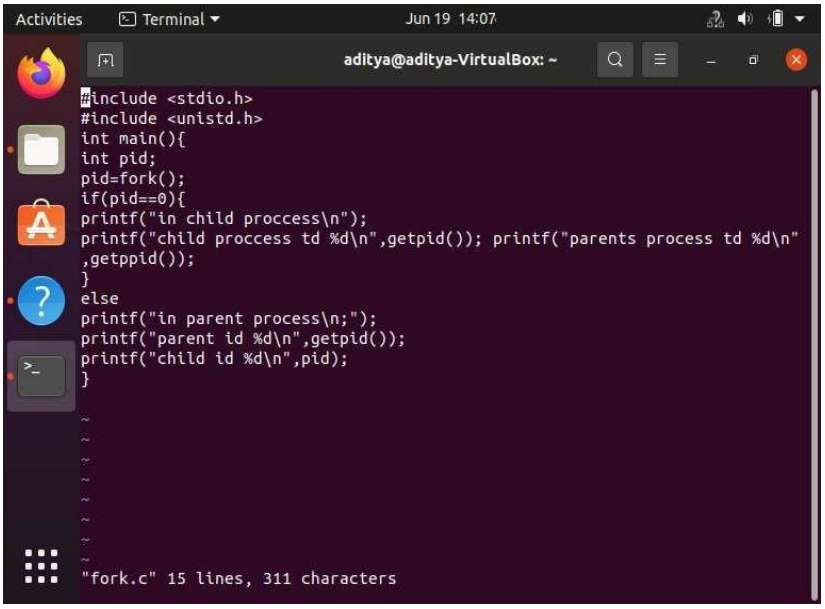
Q5. How many child processes are created for the following code? Hint : Check with small values of ‘n’.

```
for (i=0; i<n; i++) fork();
```

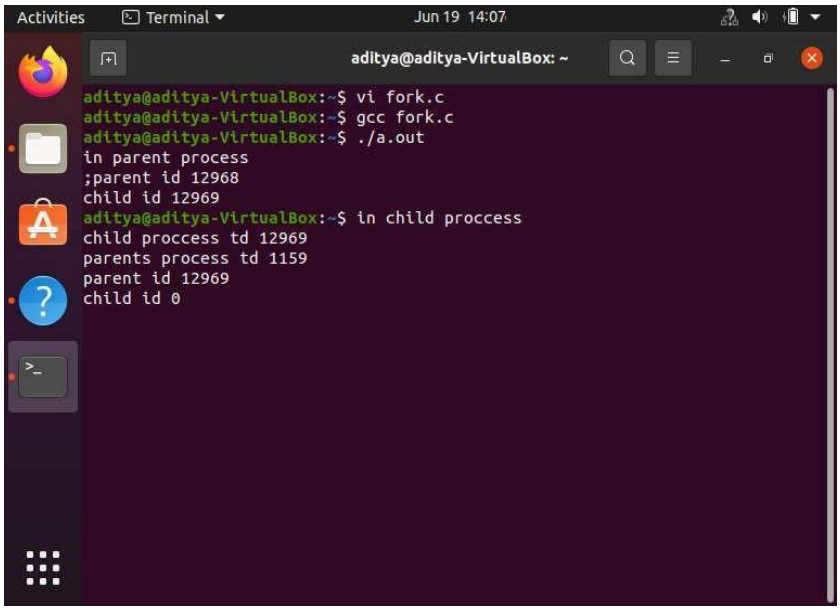
Output :



Q6. Write a program to print the Child process ID and Parent process ID in both Child and Parent processes



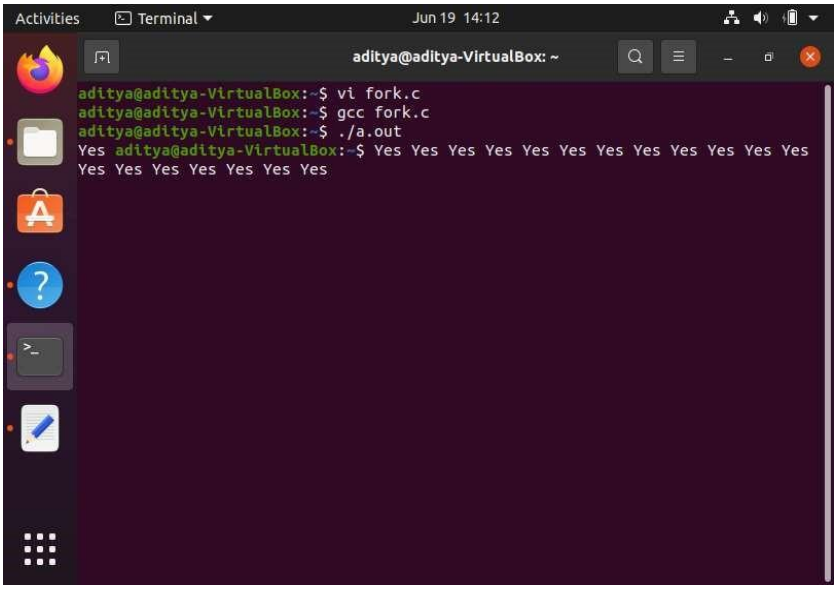
Output :



Q7. How many child processes are created for the following code?

```
#include <stdio.h>
#include<unistd.h> int
main() { fork();
fork()&&fork()||fork();  fork();
printf("Yes ");  return 0;
}
```

Output :



Result- The given program has been performed successfully.

Verified by

Staff In-charge Sign :	**Date :**

Ex. No. 7	SYSTEM ADMIN COMMANDS (For Ubuntu Linux)	Date : 20/05/22

INSTALLING SOFTWARE

To Update the package repositories

sudo apt-get update

To update installed software

sudo apt-get upgrade

To install a package/software

sudo apt-get install <package-name>

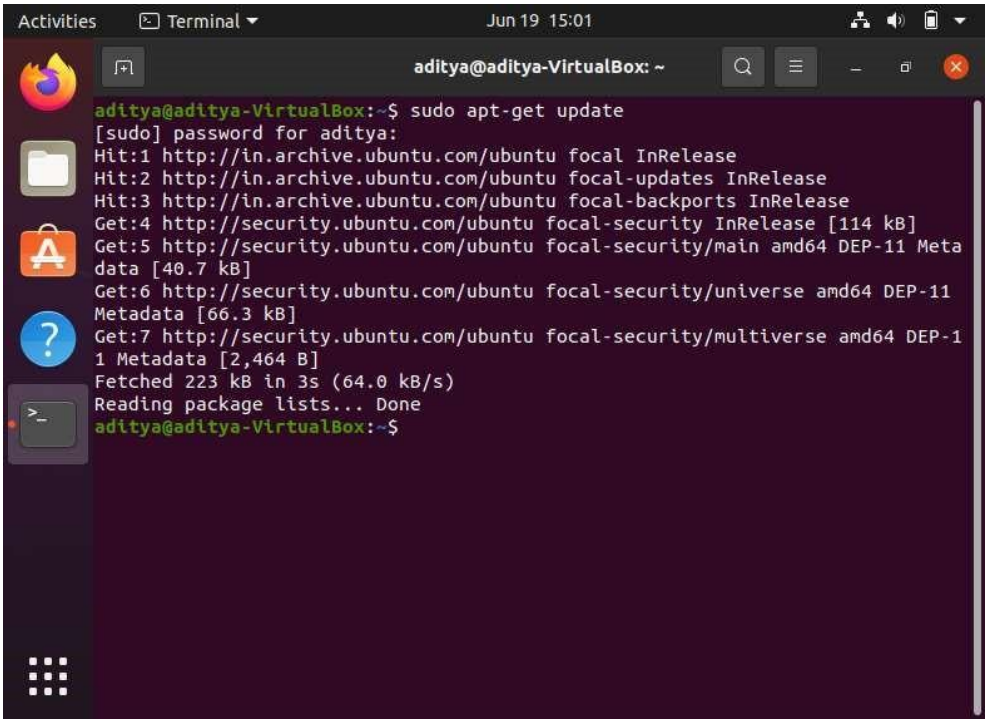
To remove a package from the system

sudo apt-get remove <package-name>

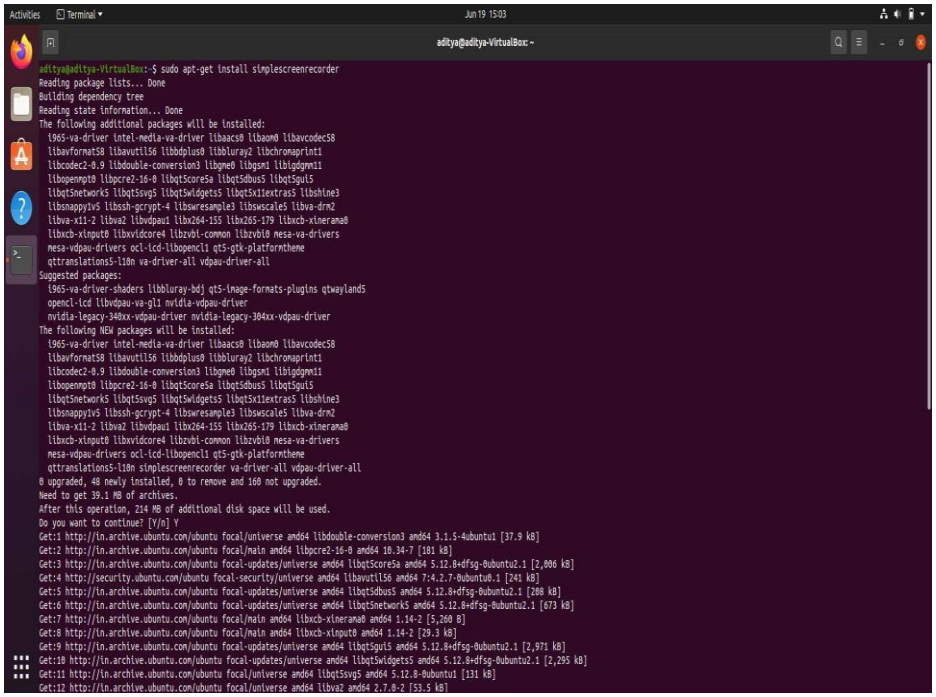
To reinstall a package

sudo apt-get install <package-name> --reinstall

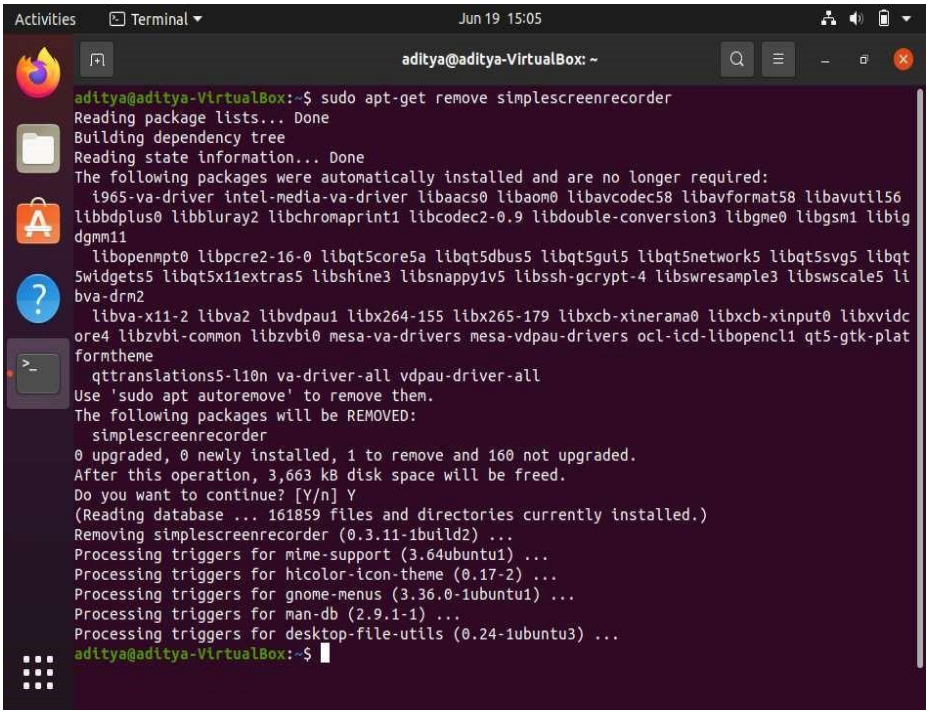
Q1. Update the package repositories



Q2. Install the package “simplescreenrecorder”



Q3. Remove the package “simplescreenrecorder”



MANAGING USERS

- Managing users is a critical aspect of server management.
- In Ubuntu, the root user is disabled for safety.

- Root access can be completed by using the sudo command by a user who is in the "admin" group.
- When you create a user during installation, that user is added automatically to the admin group.

To add a user:

sudo adduser username

To disable a user:

sudo passwd -l username

To enable a user:

sudo passwd -u username

To delete a user:

sudo userdel –r username

To create a group:

sudo addgroup groupname

To delete a group:

sudo delgroup groupname

To create a user with group:

sudo adduser username groupname

To see the password expiry value for a user,

sudo chage -l username

To make changes:

sudo chage username

GUI Tool for user management

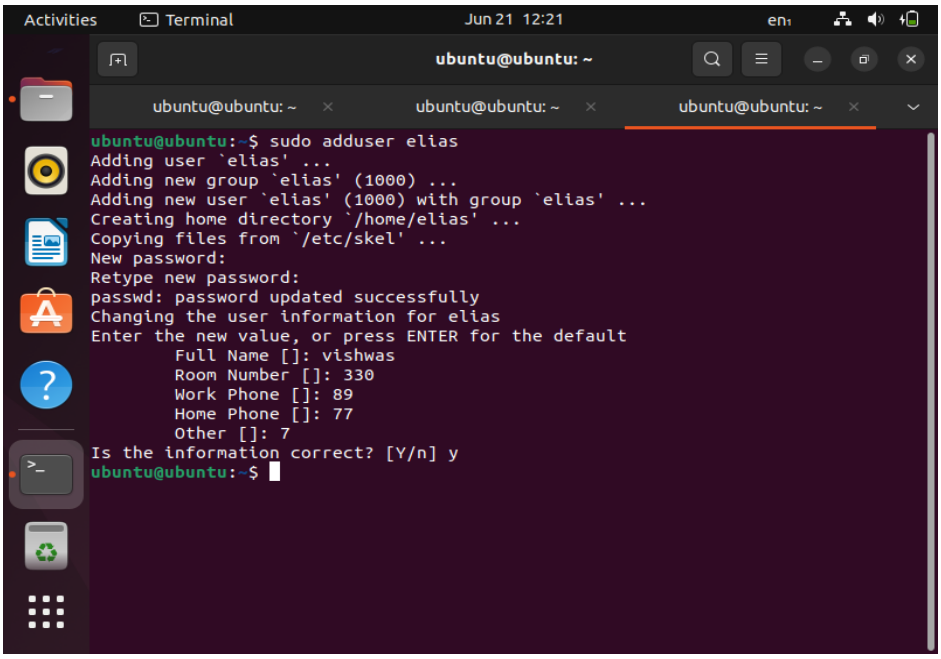
If you do not want to run the commands in terminal to manage users and groups, then you can install a GUI add-on .

sudo apt install gnome-system-tools

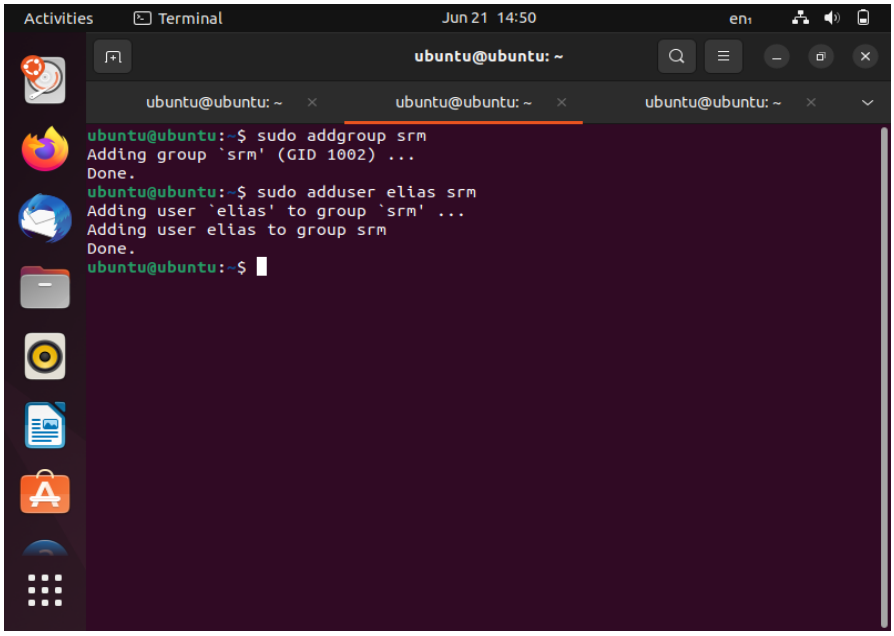
Once done, type

users-admin

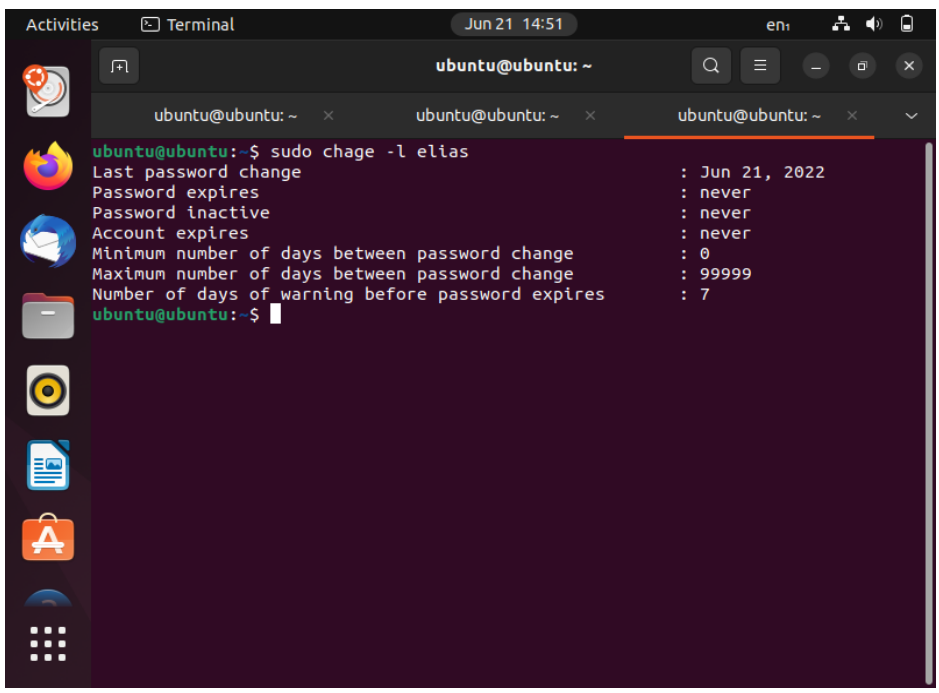
Q4. Create a user 'elias'. Login to the newly created user and exit.



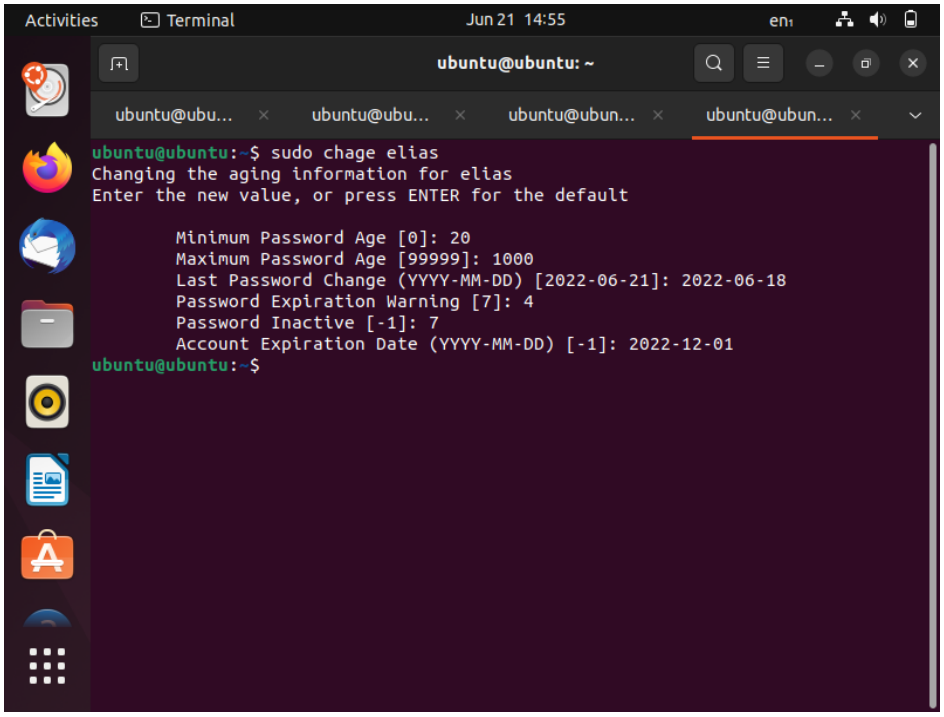
Q5. Create a group ‘cse’ and add the user ‘elias’ in that group .



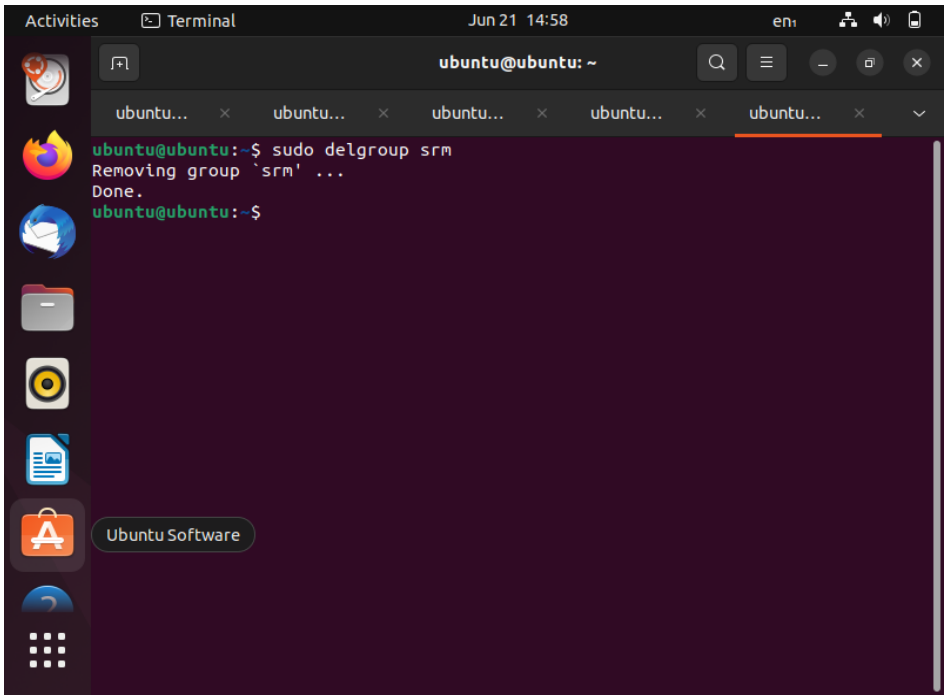
Q6. List the account expiry information of the user ‘elias’



Q7. Change the ‘Number of days warning before password expires’ as 5 for the user ‘elias’.



Q8. Delete the user ‘elias’ and then delete the group ‘cse’.



FILE SYSTEM

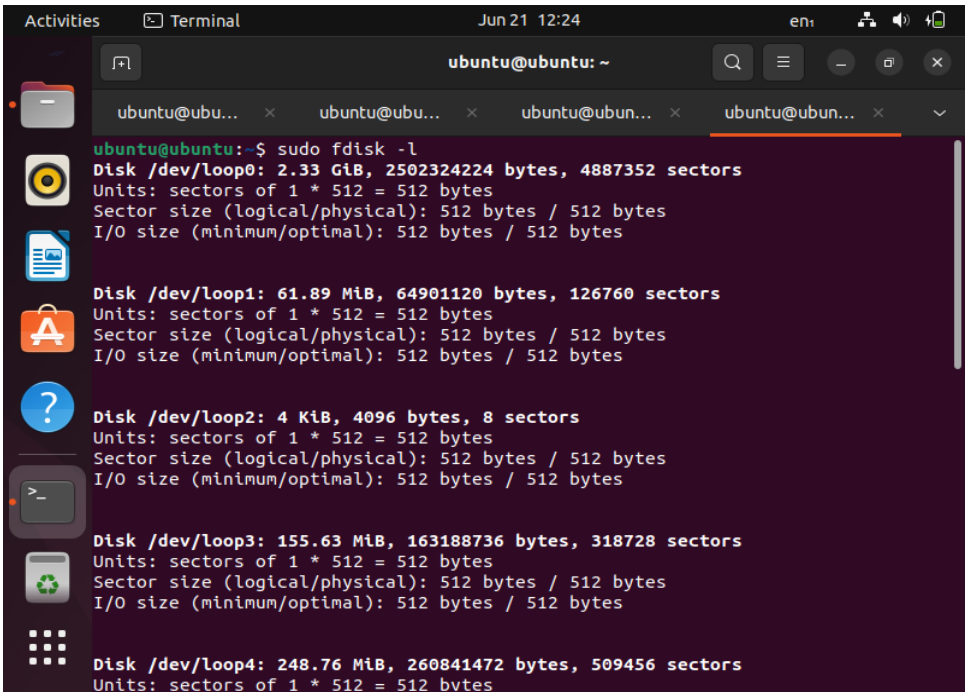
A filesystem is a permanent storage for containing data. Any non-volatile storage device like hard disk, usb etc has a filesystem in place, on top of which data is stored. While installing Linux, you may opt for either EXT4 or EXT3 file system.

EXT3: A journaling filesystem: logs changes in a journal to increase reliability in case of power failure or system crash.

EXT4: It is an advanced file syste. This file system supports 64-bit storage limits, columns up to 1 exabytes and you may store files up to 16 terabytes

Disk Partitions can be viewed by the command sudo fdisk -l File system information are available in the file /etc/fstab

Q.9. List the partitions available in your system



NETWORKING

Most networking is configured by editing two files:

- /etc/network/interfaces
  - Ethernet, TCP/IP, bridging
- /etc/resolv.conf o DNS

Other networking files:

- /etc/hosts
- /etc/dhcp3/dhcpd.conf

To test any host's connectivity

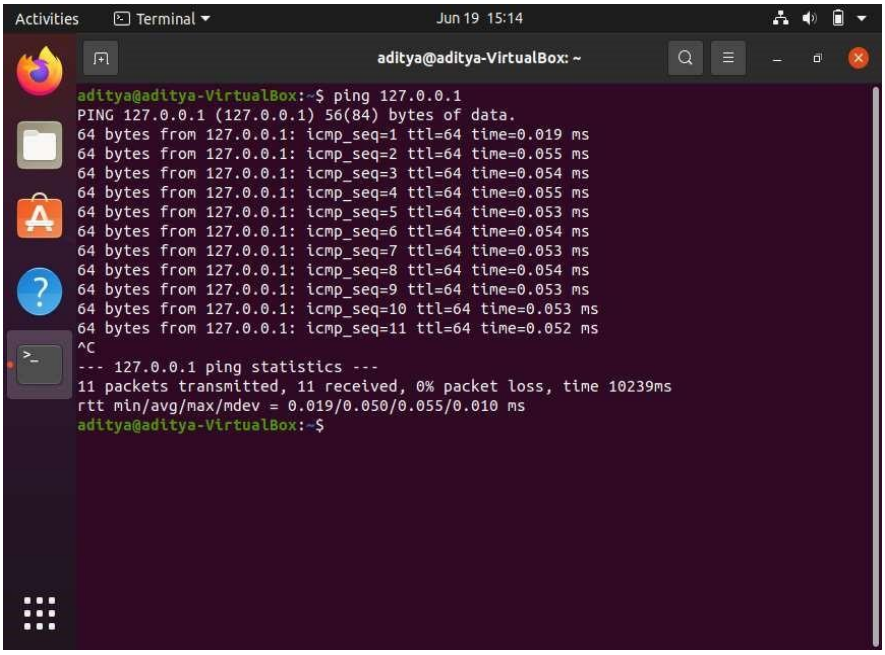
ping <ip-address>

To start/stop/restart/reload networking services

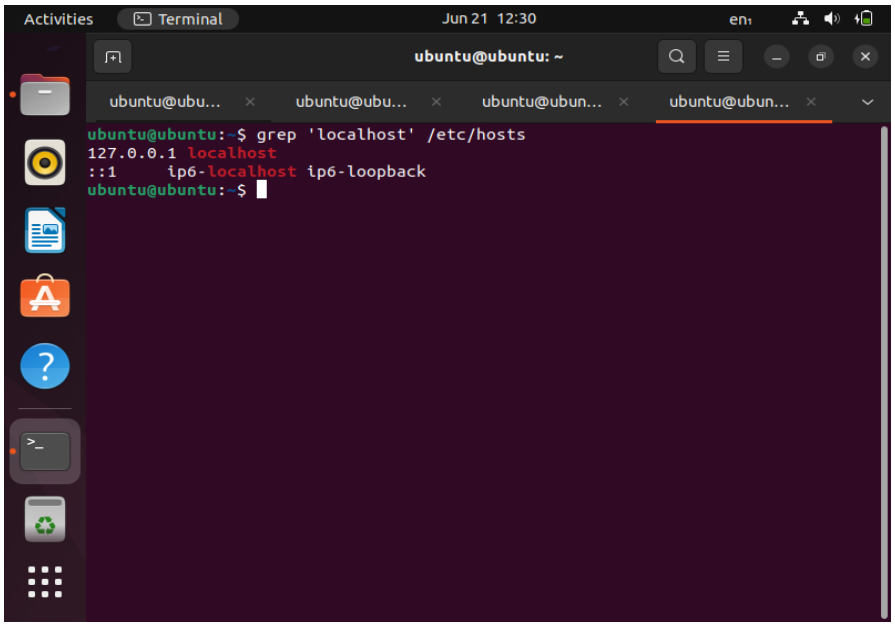
sudo /etc/init.d/mnetworking <function>

Note : <function> can be any one of stop or start or reload or restart

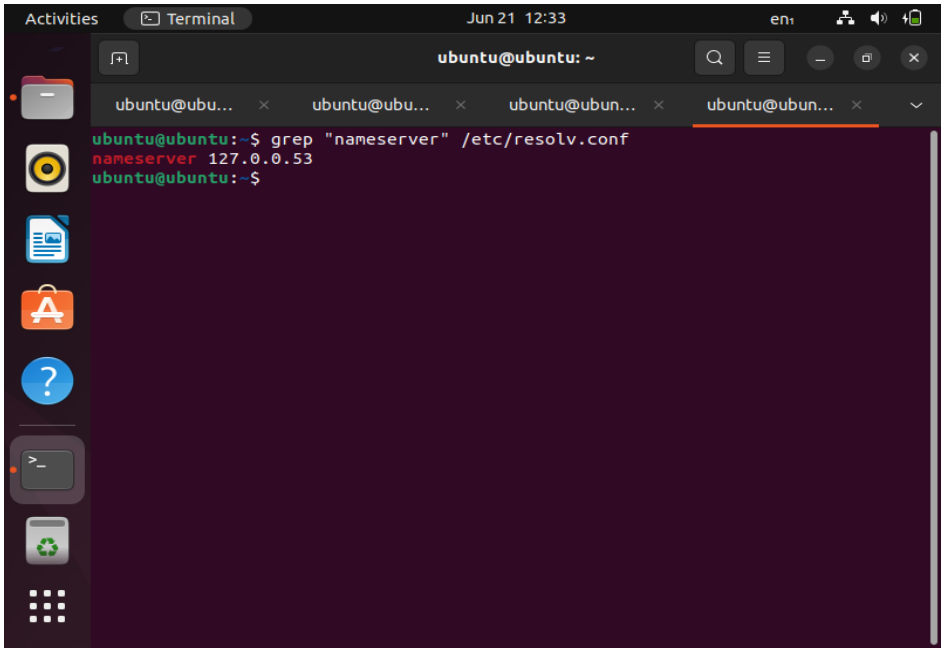
Q10. Check the connectivity of the host with IP address 127.0.0.1 .



Q11. Find the IP address of the localhost.



Q12. Find the IP address of the DNS Server (name server)



INSTALLING INTERNET SERVICES

Installing Apache server

sudo apt-get install apache2

Configuration file for Apache server

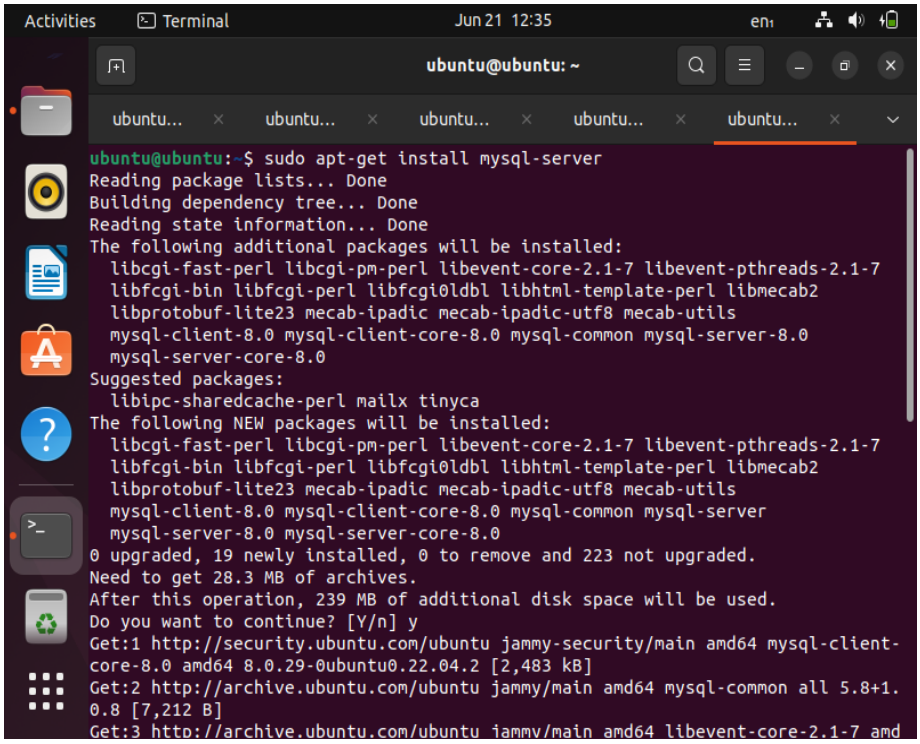
apache2.conf

Restart apache services after any configuration changes made

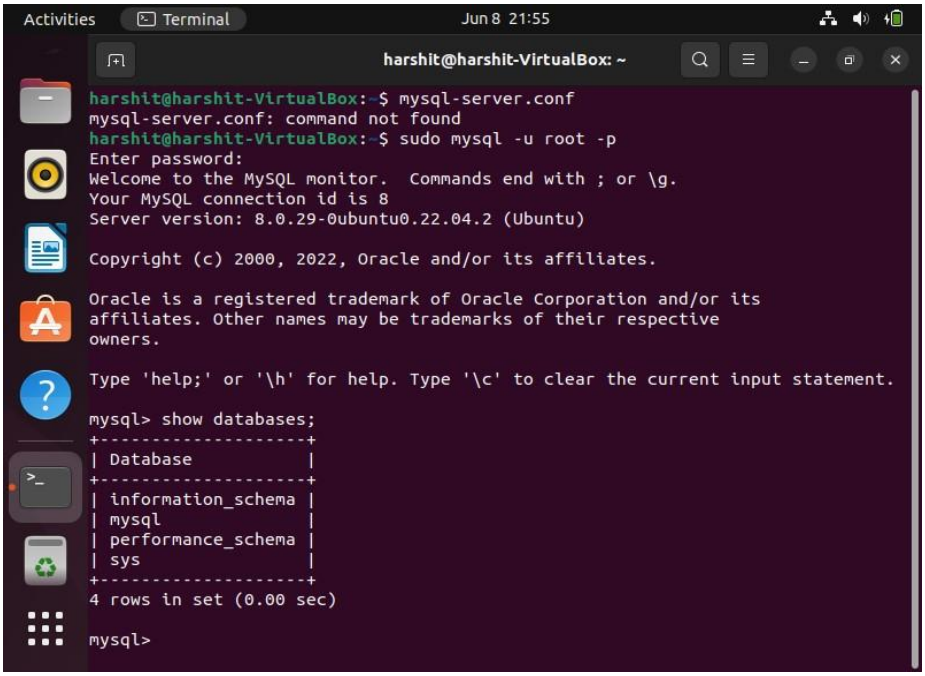
sudo /etc/init.d/mnetworking restart

Similarly all services can be installed, configured and restarted

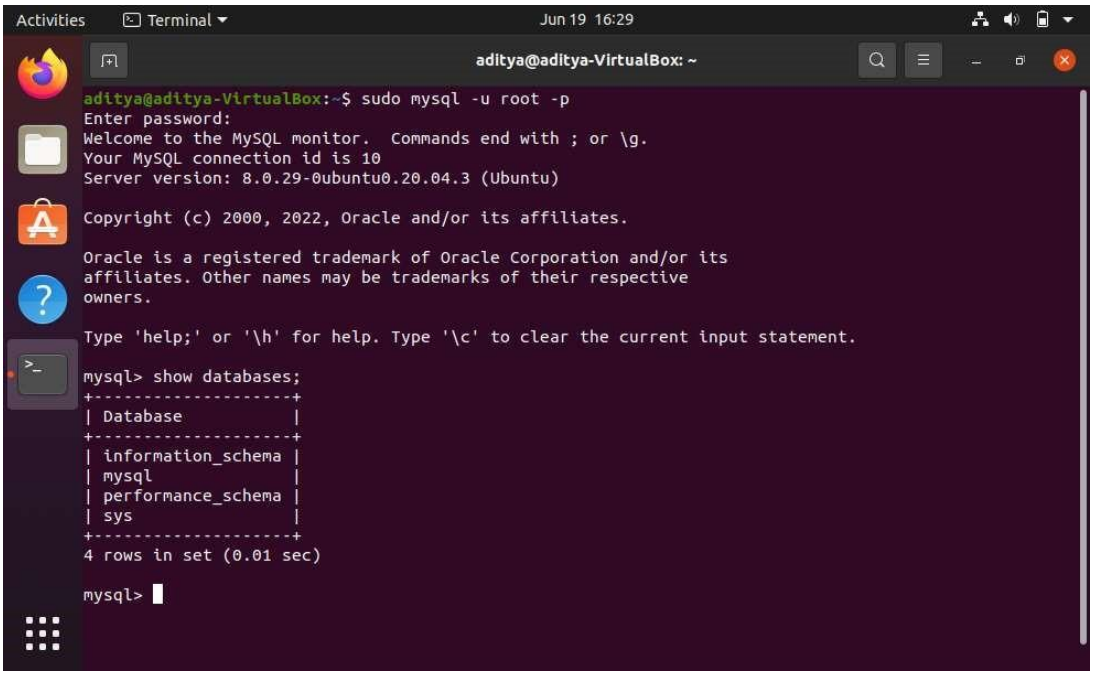
Q13. Install mysql server



Q14. Log on as root into mysql server



Q15. Create a new database for mysql server



Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	Date :

Ex. No. 8	SIMPLE TASK AUTOMATION	Date : 03/06/22

Linux Cron utility is an effective way to schedule a routine background job at a specific time and/or day on an on-going basis. You can use this to schedule activities, either as one-time events or as recurring tasks.

Crontab Syntax m h dom mon dow command

m – The minute when the cron job will run (0-59) h - a numeric value determining the hour when the tasks will run (0-23) dom – Day of the Month when the cron job will run (1-31) mon - The month when the cron job will run (1-12) dow – Day Of the Week from 0-6 with Sunday being 0 command- The linux command you wish to execute

Scheduling of Tasks (For Ubuntu)

Step 1 : Open terminal and type the command

crontab -e

Step 2 : Choose the editor. Better to select nano editor

Step 3 : Edit the file based on the syntax given above

Step 4 : Save and Exit the file

Step 5 : Start cron daemon using the following command

systemctl start cron

Example of crontab entry

0 8 * * 1 echo Have a Good Week > >tmpfile

Every Monday 8:00 am the message “Have a Good Week” transferred to the file ‘tmpfile’

Special Crontab Characters

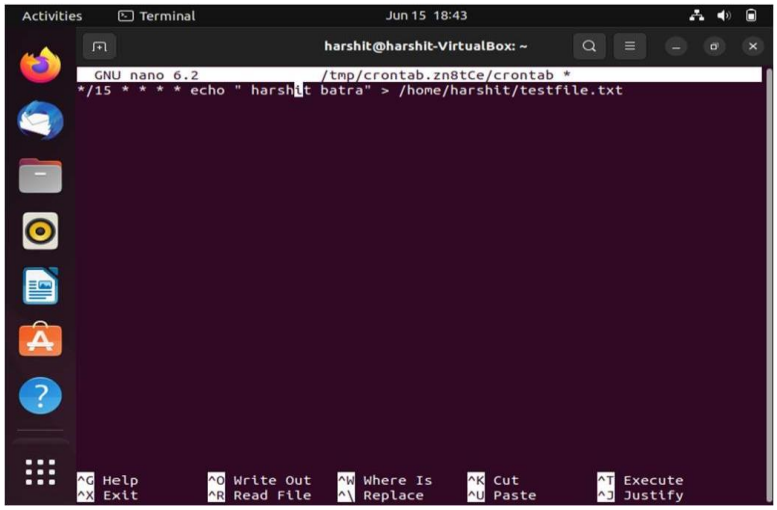
* represents all possible value

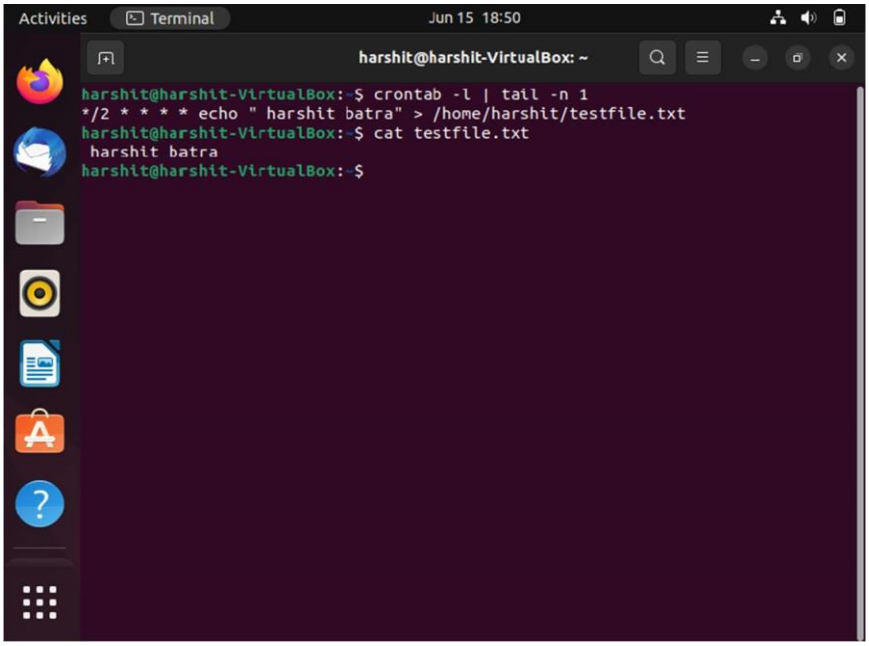
/ represents partial value. Ex. */10 in minute column specifies every 10 minutes

- represent range of values. Ex. 6-9 in hour column specifies 6am to 9 am

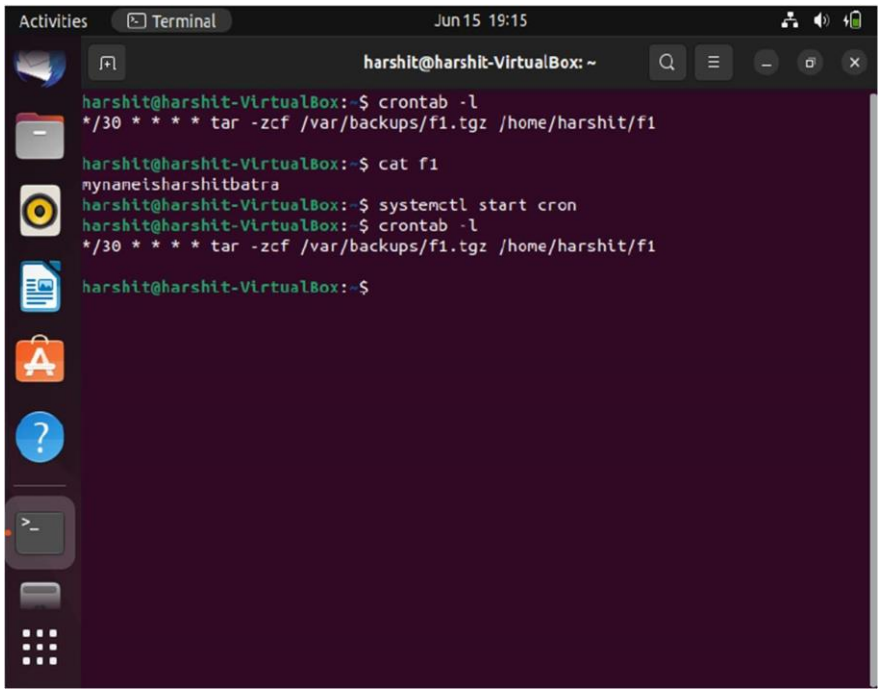
, (Comma) represent different set of values. Ex. 1,4 in month specifies Jan and Apr month

Q1. Schedule a task to display the following message on the monitor for every 2 minutes.

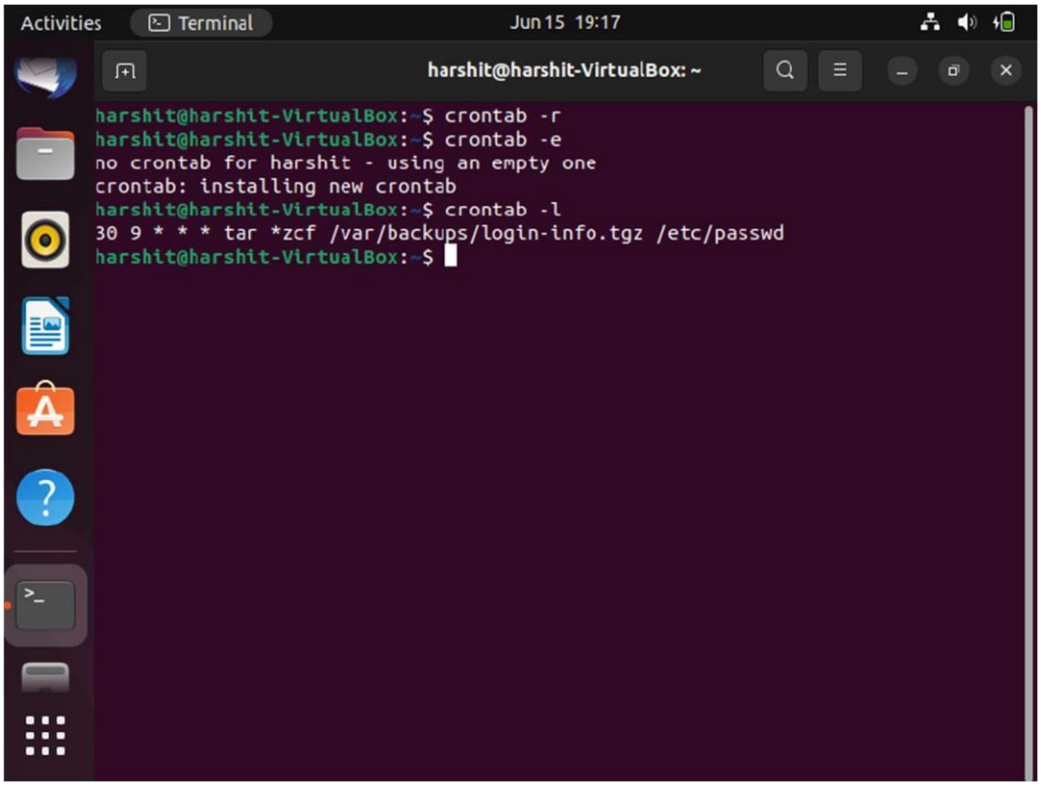




Q2. Schedule a task to take backup of your important file (say file f1) for every 30 minutes



Q3. Schedule a task to take backup of login information everyday 9:30am.



Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	Date :

Ex. No. 9	SHELL PROGRAMS	Date : 10/06/22

How to run a Shell Script

- Edit and save your program using editor Add execute permission by chmod command
- Run your program using the name of your program

  ./program-name

Important Hints

- No space before and after the assignment operator Ex. sum=0
- Single quote ignores all special characters. Dollar sign, Back quote and Back slash are not ignored inside Double quote. Back quote is used as command substitution. Back slash is used to remove the special meaning of a character.
- Arithmetic expression can be written as follows : i=$((i+1) or i=$(expr $i + 1)
- Command line arguments are referred inside the programme as $1, $2, ..and so on
- $* represents all arguments, $# specifies the number of arguments
- read statement is used to get input from input device. Ex. read a b

Syntax for if statement

```
if [ condition ] then
        ...
elif [ condition ] then
        ... else
        ...
fi
```

Syntax for case structure

```
case value in pat1) ...
statement;; pat2) ...
                        Statement;;
 *)          ...
                        Statement;;
esac
```

Syntax for for-loop

```
for var in list-of-values do
                                                                    ...
                ...
done
```

Syntax for While loop

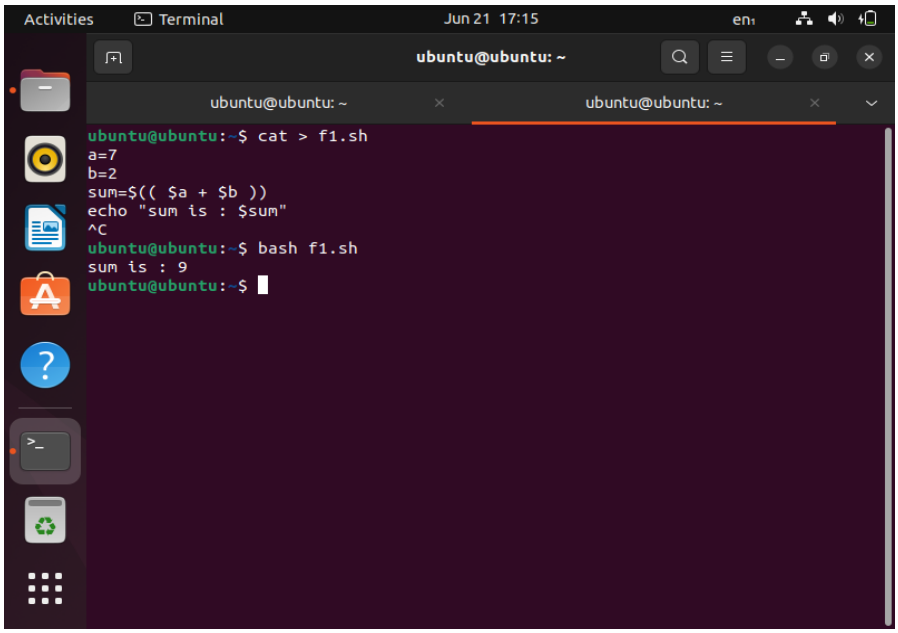
```
while command_t do
                ...
                ...
done
```

Syntax for printf statement

printf “string and format” arg1 arg2 … …

- Break and continue statements functions similar to C programming
- Relational operators are –lt, -le, -gt, -ge, -eq,-ne
- Ex. (i>= 10) is written as [$i -ge 10]
- Logical operators (and, or, not) are -o, -a, !
- Ex. (a>b) && (a>c) is written as [$a –gt $b –a $a –gt $c] Two strings can be compared using = operator

Q1. Write a program to do sum using shell.



Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	Date :

Ex. No. 10	PIPES	Date : 17/06/22

Pipe is a communication medium between two or more processes. The system call for creating pipe is

int pipe(int p[2]);

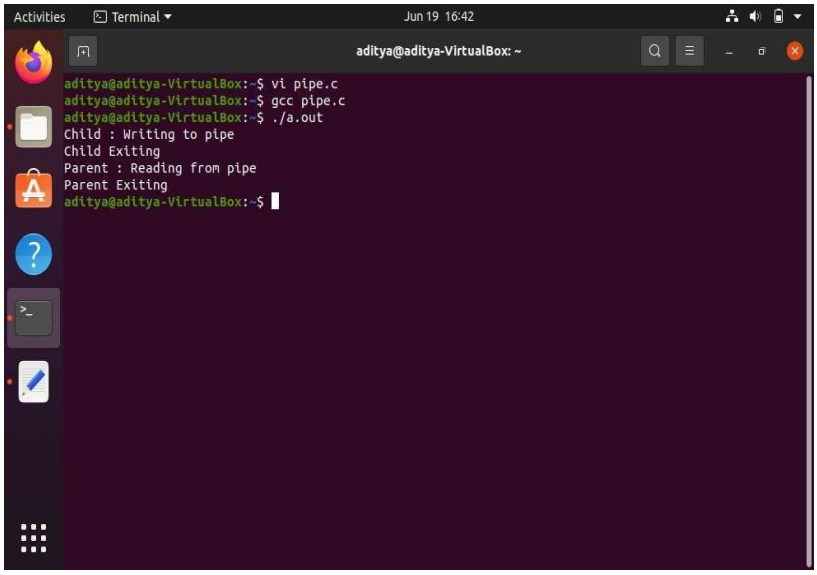
This system call would create a pipe for one-way communication i.e., it creates two descriptors, first one is connected to read from the pipe and other one is connected to write into the pipe.

Descriptor p[0] is for reading and p[1] is for writing. Whatever is written into p[1] can be read from p[0].

Q1. Write the output of the following program

```
#include <stdio.h>
#include<unistd.h>
#include<sys/wait.h> int main()
{   int   p[2];   char
    buff[25];
    if(fork()==0)
    {   printf("Child   :   Writing   to   pipe   \n");
        write(p[1],"Welcome",8); printf("Child Exiting\n");
    } else
    { wait(NULL);
        printf("Parent : Reading from pipe \n"); read(p[1],buff,8); printf("Parent Exiting\n");
    } return 0;
}
```

Output :



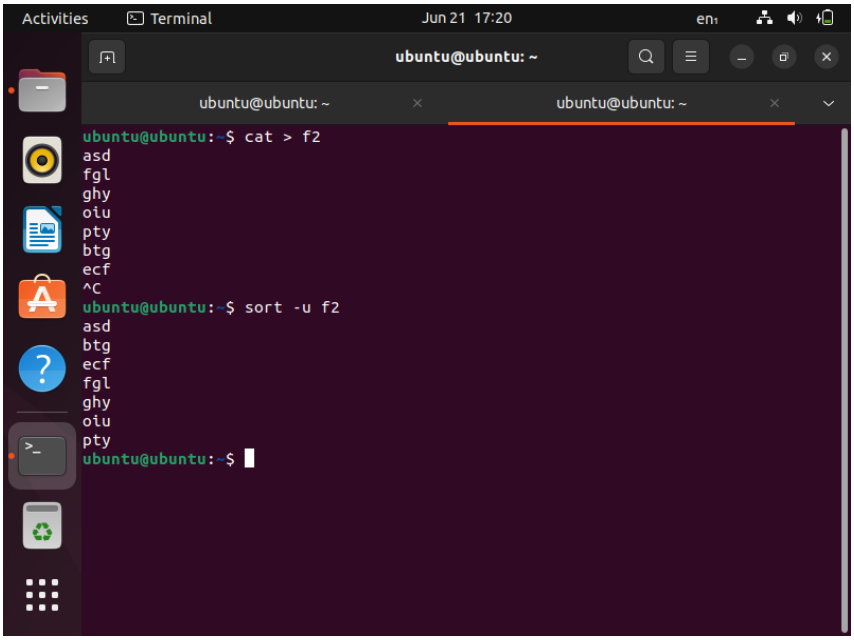
Implementing command line pipe using exec() family of functions00 Follow the steps to transfer the output of a process to pipe:

(i) Close the standard output descriptor
(ii) Use the following system calls, to take duplicate of output file descriptor of the pipe

```
int dup(int fd);
int dup2(int oldfd, int newfd);
```

(iii) Close the input file descriptor of the pipe
(iv) Now execute the process

Q.2. Write a program to sort a file using pipes.



Result-The given program has been performed successfully.

Verified by

Staff In-charge Sign :	Date :