HARSH KASHYAP
CSE 4
101917088
hkashyap_be19@thapar.edu

A Practical activity Report submitted
for Practical Computing (UCS311)

PRACTICAL COMPUTING

Evaluation Assignment 1

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering
Patiala Campus
2020

Submitted to
Aashima Sharma

———————————————————————————————————

Question 1

Create a file named Rolllist in your home directory and Insert your roll number and one another roll number (matching pattern of your roll number) into it (Rolllistfile).

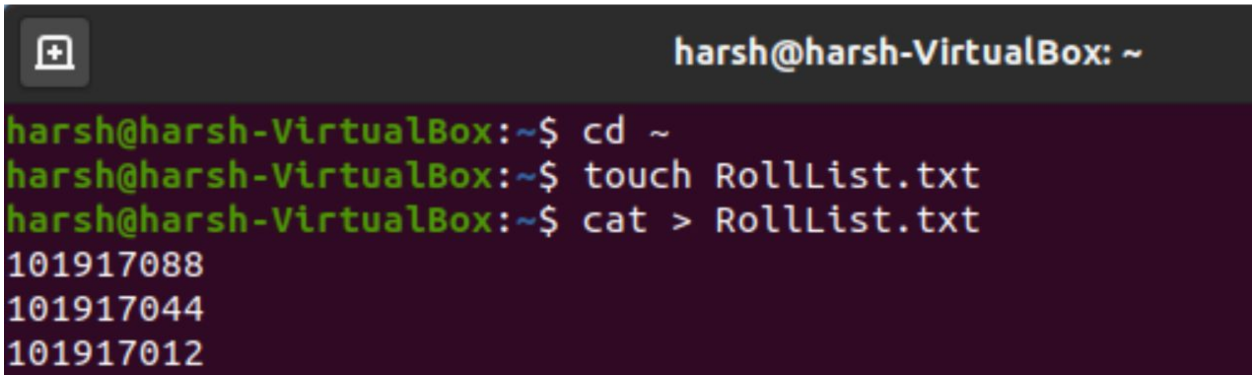
- Write a shell script that:
- accepts your roll no as command-line argument.
- matches the command line roll no with the Rolllist File.
- If script find roll no in Roll list file then Print the reverse of the roll number otherwise display the message "Roll number is not found in the file"

Write commands for every activity (1, 2 (a, b, c)) and attach screen shot.

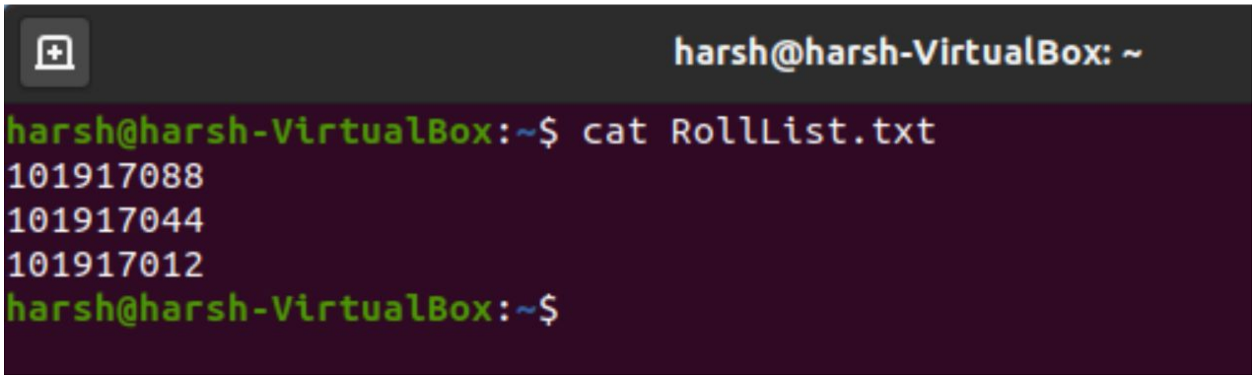
Solution -

Creating RollList.txt

1. First of all, go to the home directory to create the file Rollist.
2. Then, create a file named Rollist.txt (namely, creating a text file), using the touch command.
3. Following this, use the cat command to add your and other few roll number in the file.



```
harsh@harsh-VirtualBox:~$ cd ~
harsh@harsh-VirtualBox:~$ touch RollList.txt
harsh@harsh-VirtualBox:~$ cat > RollList.txt
101917088
101917044
101917012
```



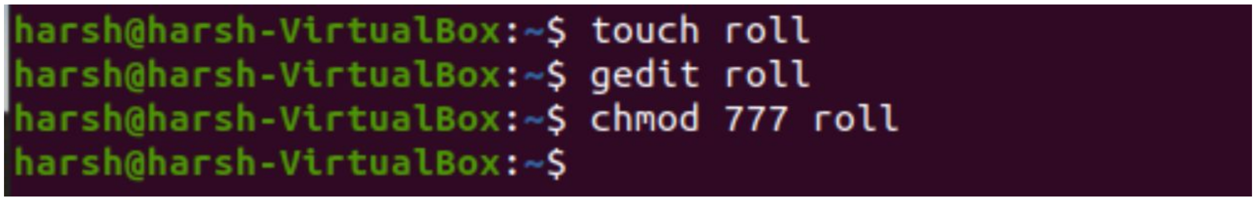
```
harsh@harsh-VirtualBox:~$ cat RollList.txt
101917088
101917044
101917012
harsh@harsh-VirtualBox:~$
```

Now, RollList file is created.
Now we will create a shell file, that will compare and match the Roll number witty file line argument.

<u>**Creating roll.sh**</u>

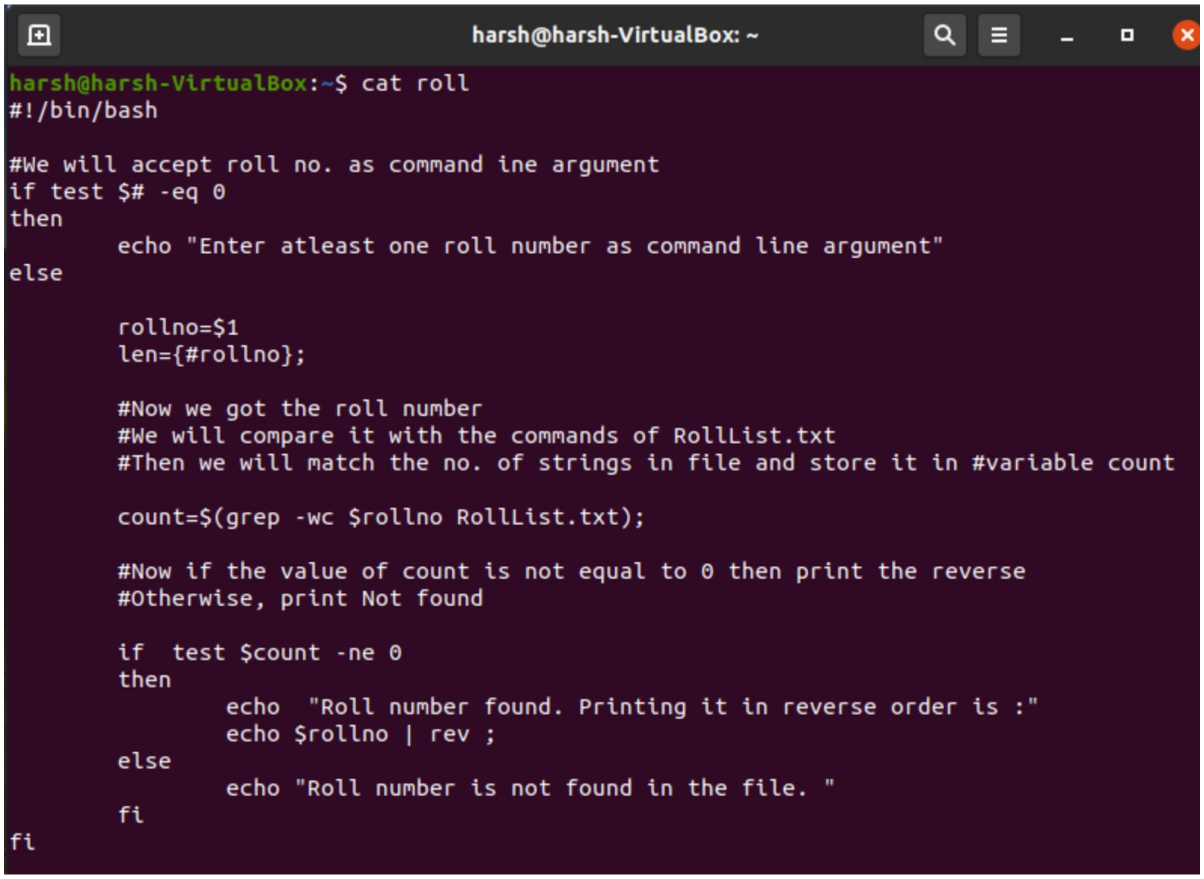
1. **Create a shell script file.**
2. **Give command lines for better understanding**
3. **Take the input from the user as command-line argument and store it in a variable**
4. **There exist three conditions which needs to be checked:-**
    a. ***When no input is given as command line***
    Check whether there is a command-line argument using $#. If there is no input, display an appropriate message and return.
    b. ***When the input passed as an argument does not match any roll number in the file.***
    Display the message “Roll number is not found in the file”
    c. ***When the input passed as an argument matches any roll number in the file.***
    Print the roll number in reverse order.
5. **Exit**

Here are some snapshots of the following:-



```
harsh@harsh-VirtualBox:~$ touch roll
harsh@harsh-VirtualBox:~$ gedit roll
harsh@harsh-VirtualBox:~$ chmod 777 roll
harsh@harsh-VirtualBox:~$
```

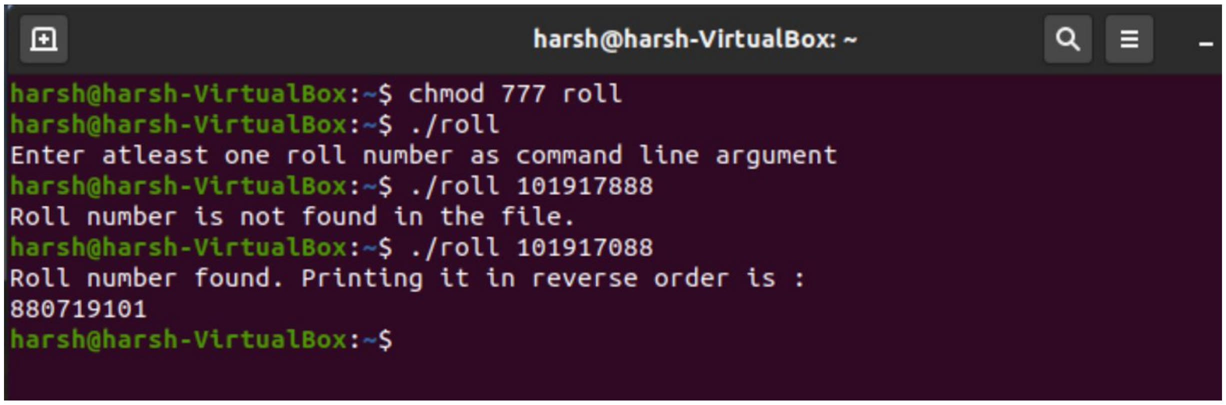
The Shell Script of the program



```
harsh@harsh-VirtualBox:~$ cat roll
#!/bin/bash

#We will accept roll no. as command ine argument
if test $# -eq 0
then
        echo "Enter atleast one roll number as command line argument"
else

        rollno=$1
        len={#rollno};

        #Now we got the roll number
        #We will compare it with the commands of RollList.txt
        #Then we will match the no. of strings in file and store it in #variable count

        count=$(grep -wc $rollno RollList.txt);

        #Now if the value of count is not equal to 0 then print the reverse
        #Otherwise, print Not found

        if  test $count -ne 0
        then
                echo  "Roll number found. Printing it in reverse order is :"
                echo $rollno | rev ;
        else
                echo "Roll number is not found in the file. "
        fi
fi
```

<u>OUTPUT</u> (Various Command as parameters and their output)

Fully Functional and Fault Tolerant.



```
harsh@harsh-VirtualBox:~$ chmod 777 roll
harsh@harsh-VirtualBox:~$ ./roll
Enter atleast one roll number as command line argument
harsh@harsh-VirtualBox:~$ ./roll 101917888
Roll number is not found in the file.
harsh@harsh-VirtualBox:~$ ./roll 101917088
Roll number found. Printing it in reverse order is :
880719101
harsh@harsh-VirtualBox:~$
```

——————————————————————————————

Question 2

Write a program to create the weekly schedule of your Practical computing class with menu driven option in shell script and save it as "PCSchedule".

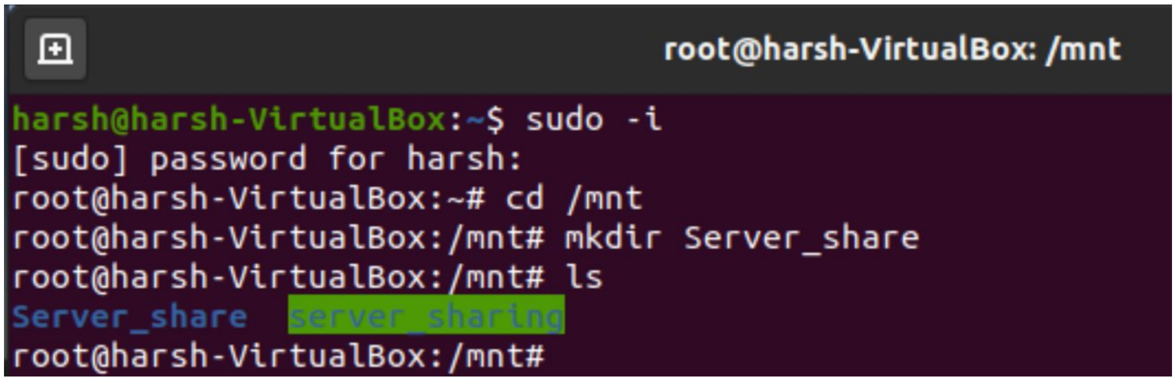
Hint: Menu –1. Monday, 2. Tuesday…………3. Saturday If option 1 (Monday) is selected, Display: Practical computing is scheduled at 8:00 am.

Save this script in a folder named "Server_share" on the server.
Further, access this file (script) from client folder named "Client_share" and execute it on the client's terminal.

Solution -

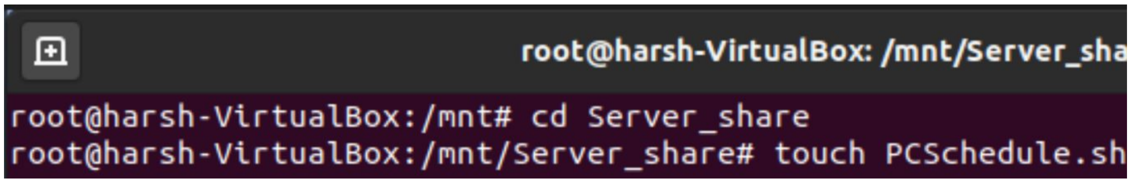
In the Server System

1. Creating a folder name Server_share
    a. We will create this folder in /mnt
    b. We will log in as administrator using sudo -i
    c. First, we will visit mnt directory using command cd.
    d. We will create this folder using mkdir command



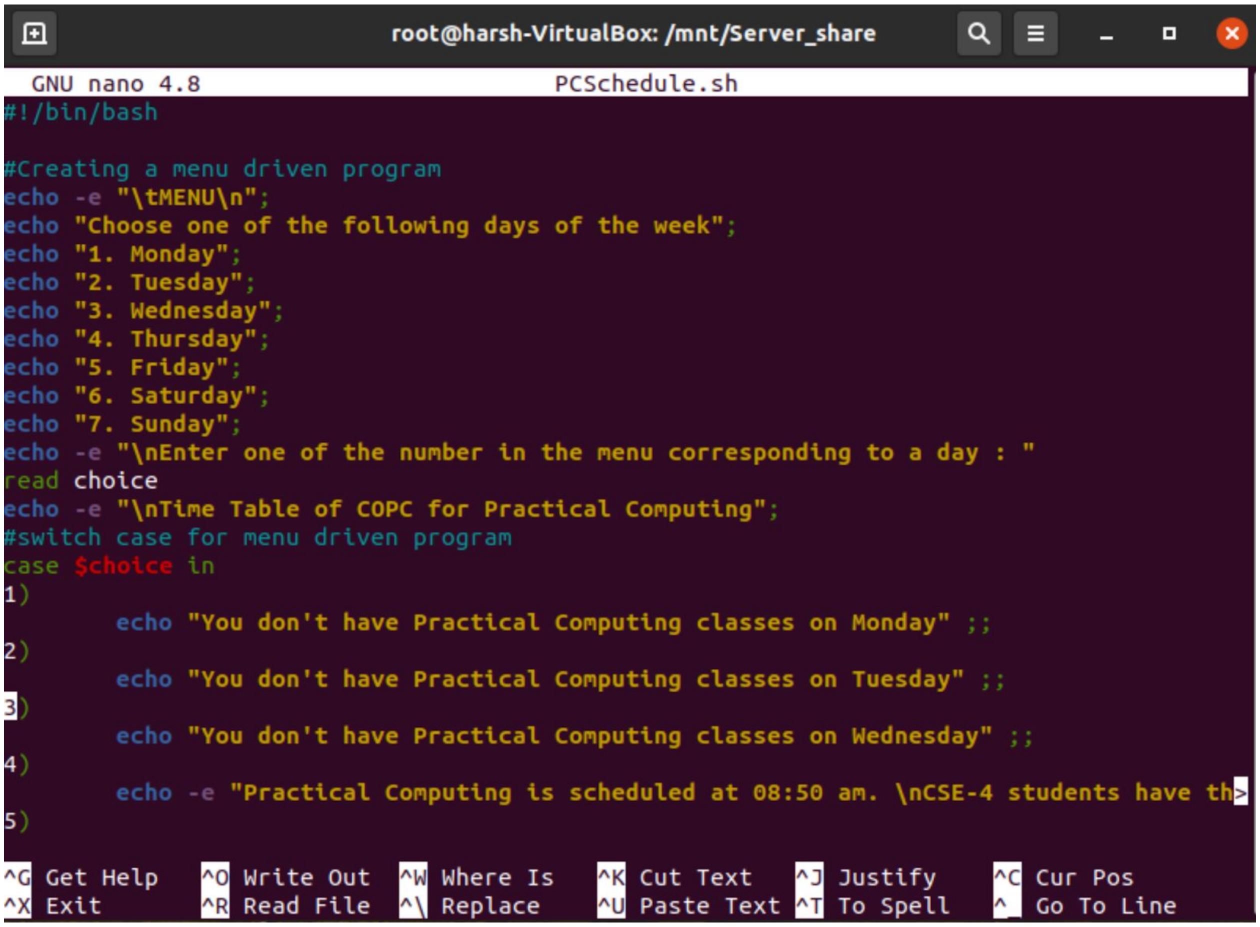
```
root@harsh-VirtualBox: /mnt
harsh@harsh-VirtualBox:~$ sudo -i
[sudo] password for harsh:
root@harsh-VirtualBox:~# cd /mnt
root@harsh-VirtualBox:/mnt# mkdir Server_share
root@harsh-VirtualBox:/mnt# ls
Server_share  server_sharing
root@harsh-VirtualBox:/mnt#
```

    e. Now we will create the shell script file named PCSchedule inside it
2. Working on the Shell Script PCSchedule.
    a. Moving inside Server_share and creating shell file.



```
root@harsh-VirtualBox: /mnt/Server_sha
root@harsh-VirtualBox:/mnt# cd Server_share
root@harsh-VirtualBox:/mnt/Server_share# touch PCSchedule.sh
```

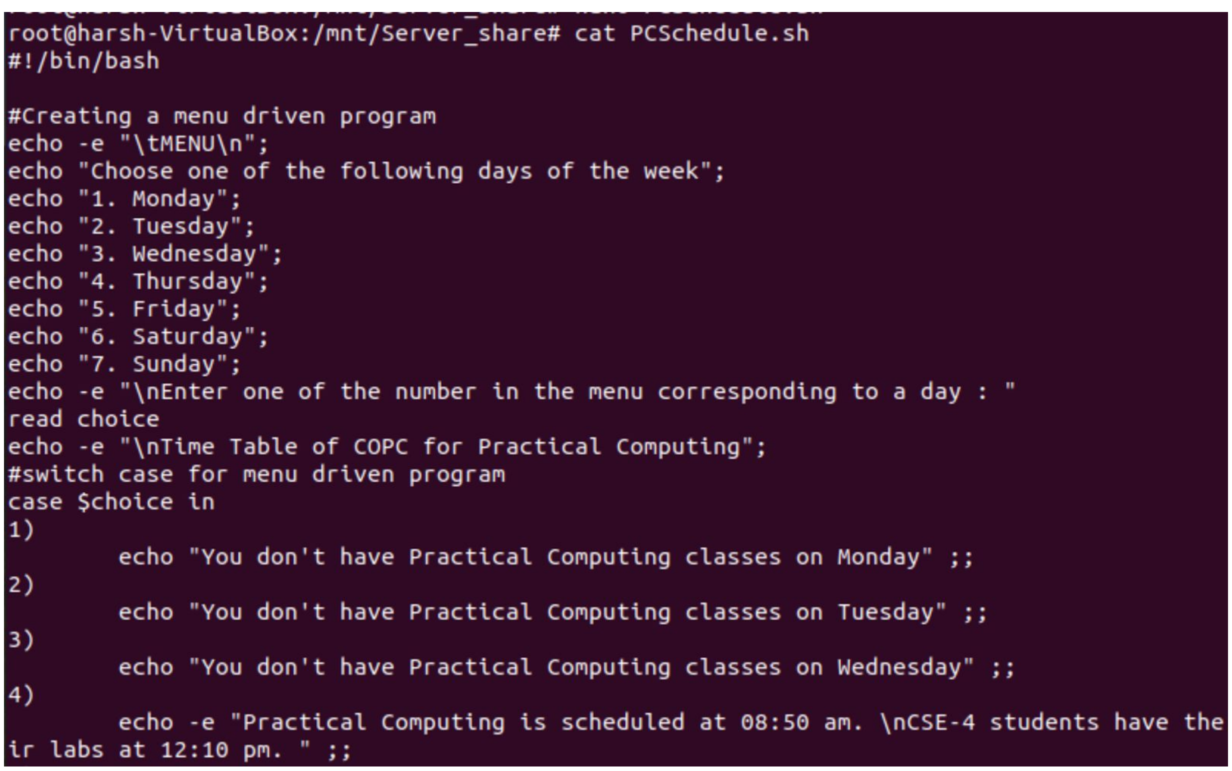
    b. Write a Shell script in nano editor.



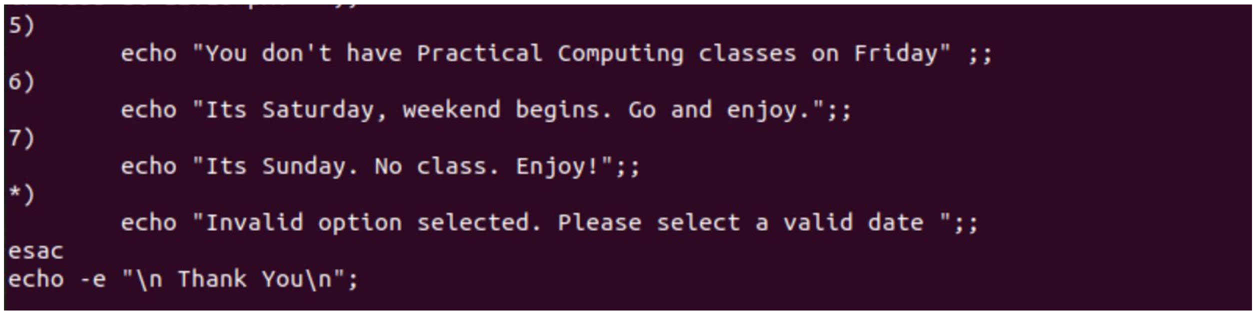
```
root@harsh-VirtualBox: /mnt/Server_share
  GNU nano 4.8                         PCSchedule.sh
#!/bin/bash

#Creating a menu driven program
echo -e "\tMENU\n";
echo "Choose one of the following days of the week";
echo "1. Monday";
echo "2. Tuesday";
echo "3. Wednesday";
echo "4. Thursday";
echo "5. Friday";
echo "6. Saturday";
echo "7. Sunday";
echo -e "\nEnter one of the number in the menu corresponding to a day : "
read choice
echo -e "\nTime Table of COPC for Practical Computing";
#switch case for menu driven program
case $choice in
1)
        echo "You don't have Practical Computing classes on Monday" ;;
2)
        echo "You don't have Practical Computing classes on Tuesday" ;;
3)
        echo "You don't have Practical Computing classes on Wednesday" ;;
4)
        echo -e "Practical Computing is scheduled at 08:50 am. \nCSE-4 students have th>
5)

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line
```

(For demo purpose, showing how command was written using the editor)

The shell script using cat is displayed below.

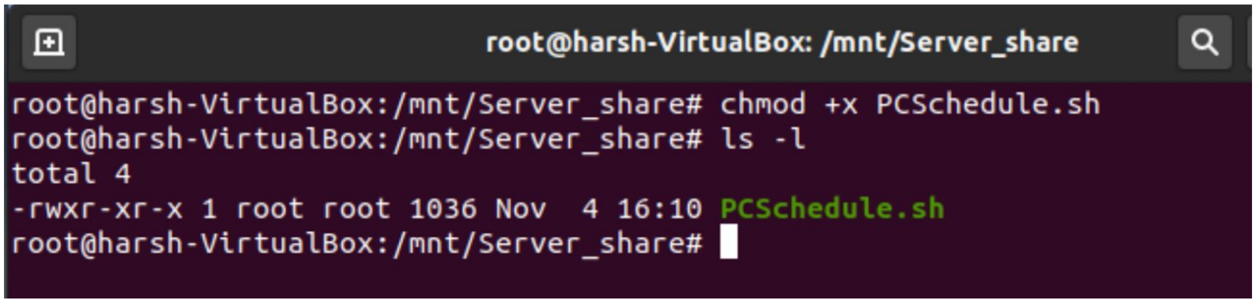


```
root@harsh-VirtualBox:/mnt/Server_share# cat PCSchedule.sh
#!/bin/bash

#Creating a menu driven program
echo -e "\tMENU\n";
echo "Choose one of the following days of the week";
echo "1. Monday";
echo "2. Tuesday";
echo "3. Wednesday";
echo "4. Thursday";
echo "5. Friday";
echo "6. Saturday";
echo "7. Sunday";
echo -e "\nEnter one of the number in the menu corresponding to a day : "
read choice
echo -e "\nTime Table of COPC for Practical Computing";
#switch case for menu driven program
case $choice in
1)
        echo "You don't have Practical Computing classes on Monday" ;;
2)
        echo "You don't have Practical Computing classes on Tuesday" ;;
3)
        echo "You don't have Practical Computing classes on Wednesday" ;;
4)
        echo -e "Practical Computing is scheduled at 08:50 am. \nCSE-4 students have the
ir labs at 12:10 pm. " ;;
```



```
5)
        echo "You don't have Practical Computing classes on Friday" ;;
6)
        echo "Its Saturday, weekend begins. Go and enjoy.";;
7)
        echo "Its Sunday. No class. Enjoy!";;
*)
        echo "Invalid option selected. Please select a valid date ";;
esac
echo -e "\n Thank You\n";
```

c. Finally, give permission to execute using chmod.



root@harsh-VirtualBox: /mnt/Server_share

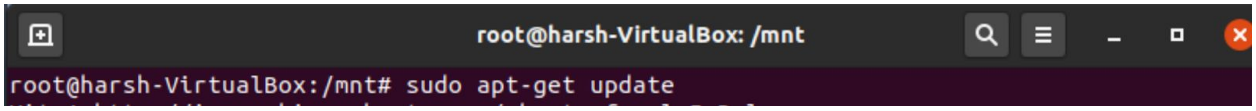
```
root@harsh-VirtualBox:/mnt/Server_share# chmod +x PCSchedule.sh
root@harsh-VirtualBox:/mnt/Server_share# ls -l
total 4
-rwxr-xr-x 1 root root 1036 Nov  4 16:10 PCSchedule.sh
root@harsh-VirtualBox:/mnt/Server_share# 
```

3. Now create a NFS server to act as a bridge between Server and Client.
   We have already installed NFS on both server and client systems

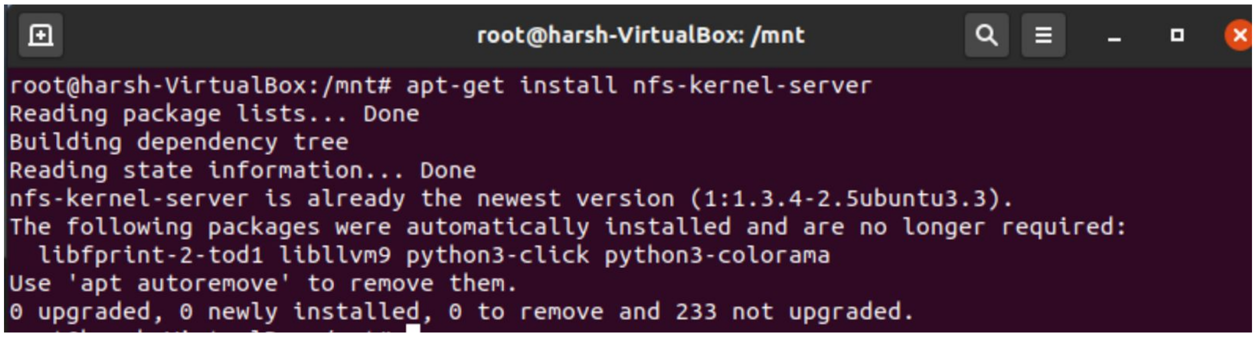
Doing further operation and attaching screenshots of the same.

SERVER SYSTEM

- ❏ Installing updates, and NFS server and doing the basics.
- ❏ Start the NFS service (if not already started)

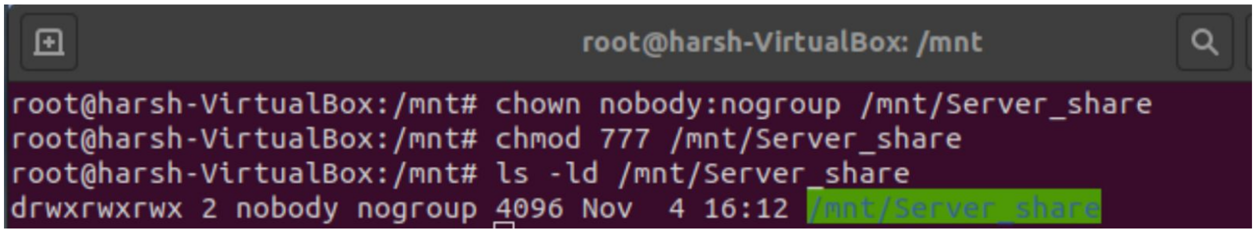


```
root@harsh-VirtualBox:/mnt# sudo apt-get update
```



```
root@harsh-VirtualBox:/mnt# apt-get install nfs-kernel-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
nfs-kernel-server is already the newest version (1:1.3.4-2.5ubuntu3.3).
The following packages were automatically installed and are no longer required:
  libfprint-2-tod1 libllvm9 python3-click python3-colorama
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 233 not upgraded.
```

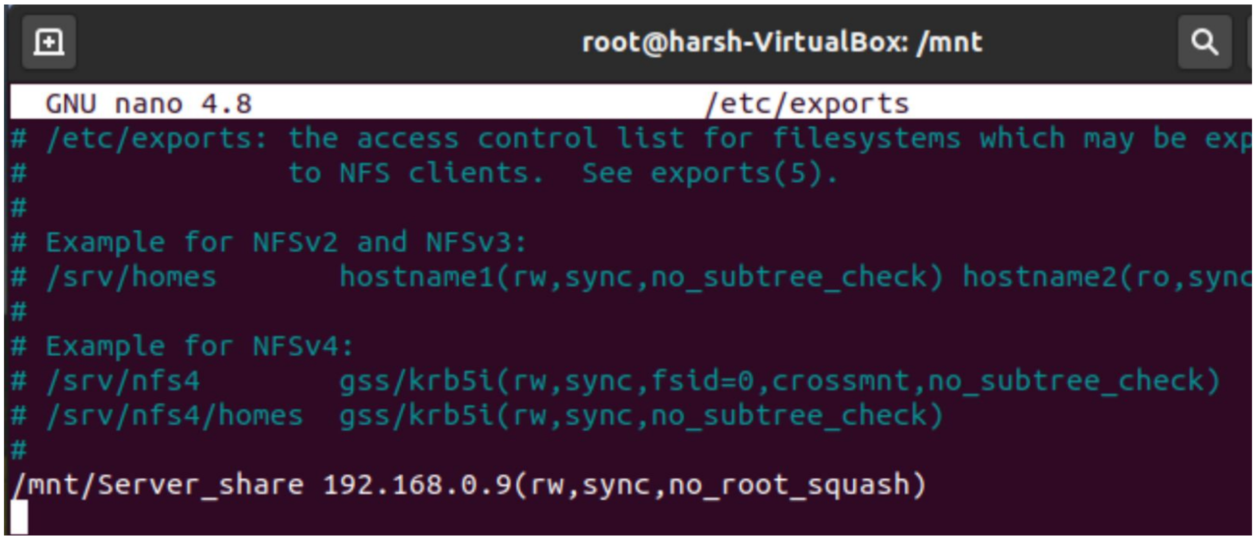
- ❏ Changing the ownership and modifying the directory in which folder Server_share is stored. Change the owner/group of the directory to nobody/nogroup and give rwx to all.



```
root@harsh-VirtualBox:/mnt# chown nobody:nogroup /mnt/Server_share
root@harsh-VirtualBox:/mnt# chmod 777 /mnt/Server_share
root@harsh-VirtualBox:/mnt# ls -ld /mnt/Server_share
drwxrwxrwx 2 nobody nogroup 4096 Nov  4 16:12 /mnt/Server_share
```

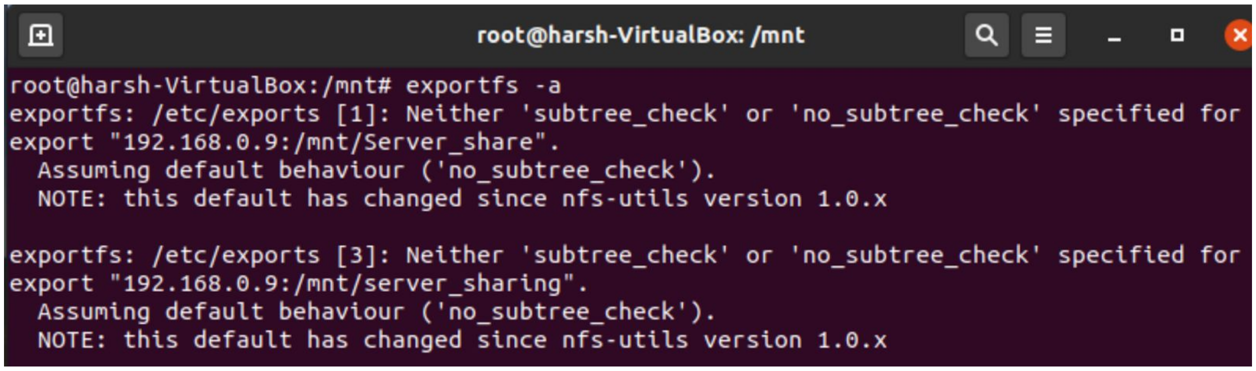
- ❏ Using the nano editor on /etc/exports and making a bridge of the client on server.
- ❏ Go to the /etc/exports file to add the new entry to the file

  /server_share (rw,sync,no_root_squash)



root@harsh-VirtualBox: /mnt

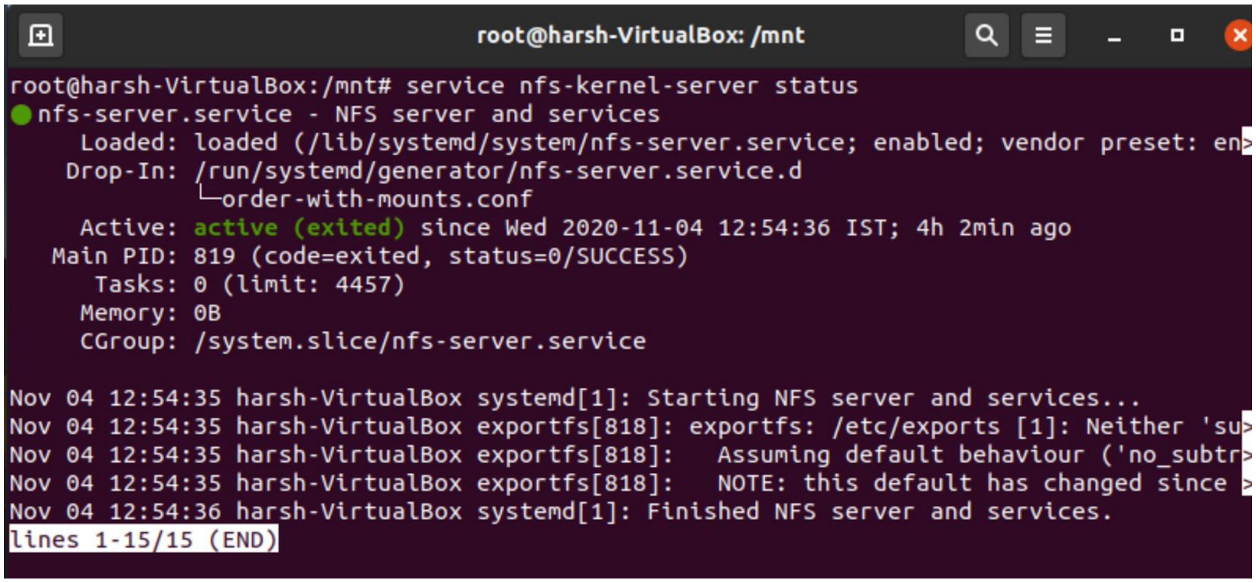
```
 GNU nano 4.8                              /etc/exports
# /etc/exports: the access control list for filesystems which may be exp
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/mnt/Server_share 192.168.0.9(rw,sync,no_root_squash)
```



root@harsh-VirtualBox: /mnt

```
root@harsh-VirtualBox:/mnt# exportfs -a
exportfs: /etc/exports [1]: Neither 'subtree_check' or 'no_subtree_check' specified for
export "192.168.0.9:/mnt/Server_share".
  Assuming default behaviour ('no_subtree_check').
  NOTE: this default has changed since nfs-utils version 1.0.x

exportfs: /etc/exports [3]: Neither 'subtree_check' or 'no_subtree_check' specified for
export "192.168.0.9:/mnt/server_sharing".
  Assuming default behaviour ('no_subtree_check').
  NOTE: this default has changed since nfs-utils version 1.0.x
```

- Checking if NFS server is active or not.

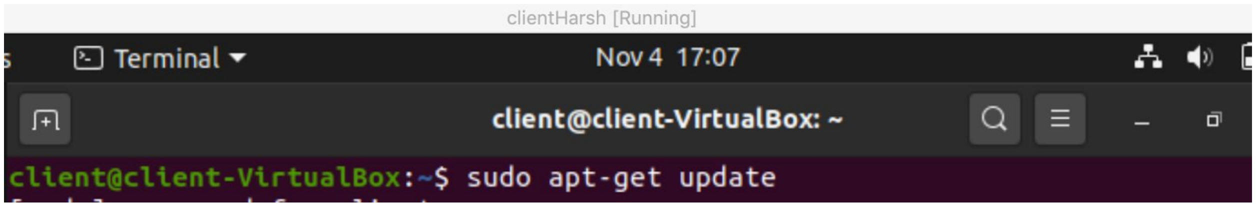


root@harsh-VirtualBox: /mnt

```
root@harsh-VirtualBox:/mnt# service nfs-kernel-server status
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor preset: en>
    Drop-In: /run/systemd/generator/nfs-server.service.d
             └─order-with-mounts.conf
     Active: active (exited) since Wed 2020-11-04 12:54:36 IST; 4h 2min ago
   Main PID: 819 (code=exited, status=0/SUCCESS)
      Tasks: 0 (limit: 4457)
     Memory: 0B
     CGroup: /system.slice/nfs-server.service

Nov 04 12:54:35 harsh-VirtualBox systemd[1]: Starting NFS server and services...
Nov 04 12:54:35 harsh-VirtualBox exportfs[818]: exportfs: /etc/exports [1]: Neither 'su>
Nov 04 12:54:35 harsh-VirtualBox exportfs[818]:   Assuming default behaviour ('no_subtr>
Nov 04 12:54:35 harsh-VirtualBox exportfs[818]:   NOTE: this default has changed since >
Nov 04 12:54:36 harsh-VirtualBox systemd[1]: Finished NFS server and services.
lines 1-15/15 (END)
```

CLIENT SYSTEM

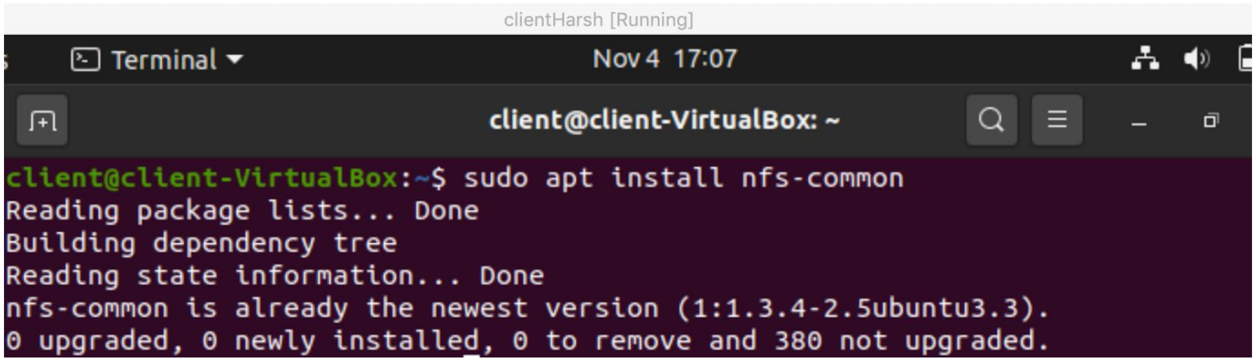
- Installing the update, and NFS common and doing the basics.


clientHarsh [Running]

Terminal ▾ Nov 4 17:07

client@client-VirtualBox: ~

```
client@client-VirtualBox:~$ sudo apt-get update
```

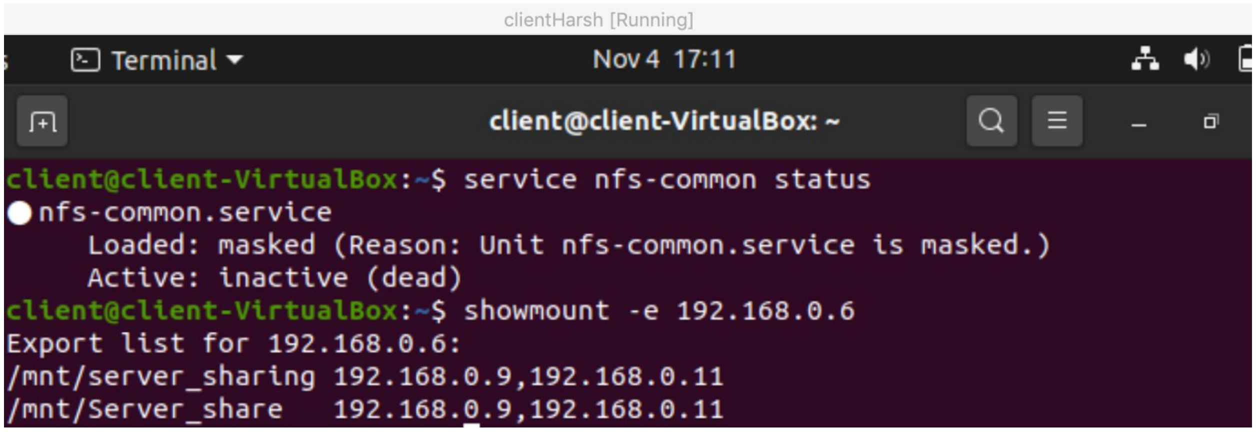

clientHarsh [Running]

Terminal ▾ Nov 4 17:07

client@client-VirtualBox: ~

```
client@client-VirtualBox:~$ sudo apt install nfs-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
nfs-common is already the newest version (1:1.3.4-2.5ubuntu3.3).
0 upgraded, 0 newly installed, 0 to remove and 380 not upgraded.
```

- ❑ Checking status of the server and using showmount with IP address to view the server file on Client system.

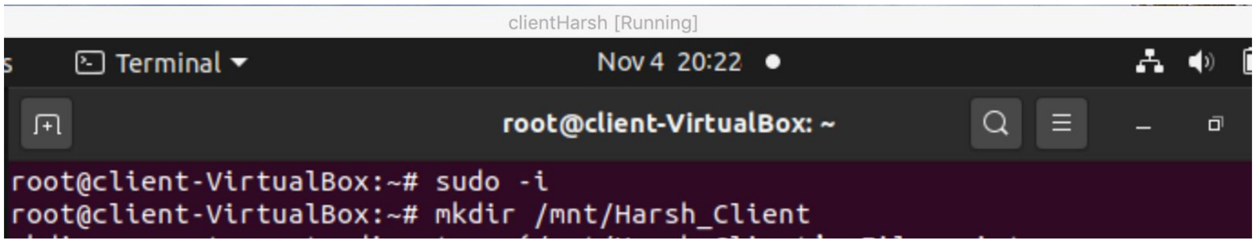

clientHarsh [Running]

Terminal ▾ Nov 4 17:11

client@client-VirtualBox: ~

```
client@client-VirtualBox:~$ service nfs-common status
● nfs-common.service
     Loaded: masked (Reason: Unit nfs-common.service is masked.)
     Active: inactive (dead)
client@client-VirtualBox:~$ showmount -e 192.168.0.6
Export list for 192.168.0.6:
/mnt/server_sharing 192.168.0.9,192.168.0.11
/mnt/Server_share   192.168.0.9,192.168.0.11
```

- ❑ Login as root and make a directory to connect to the server "Server_share"


clientHarsh [Running]

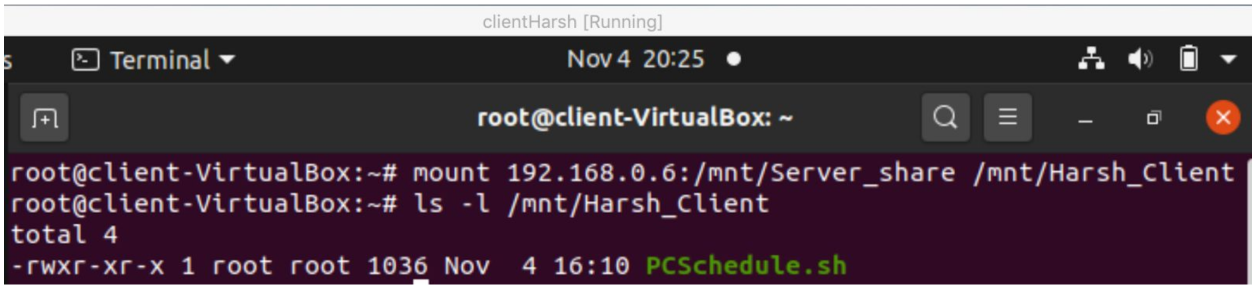
Terminal ▾ Nov 4 20:22 ●

root@client-VirtualBox: ~

```
root@client-VirtualBox:~# sudo -i
root@client-VirtualBox:~# mkdir /mnt/Harsh_Client
```

- ❑ Now use mount with IP address of server to attach the Client directory with server directory. We can see the file PCSchedule when we use ls command in Client System.

- ❑ Now you can access the file on the client machine by going to the directory and executing the file



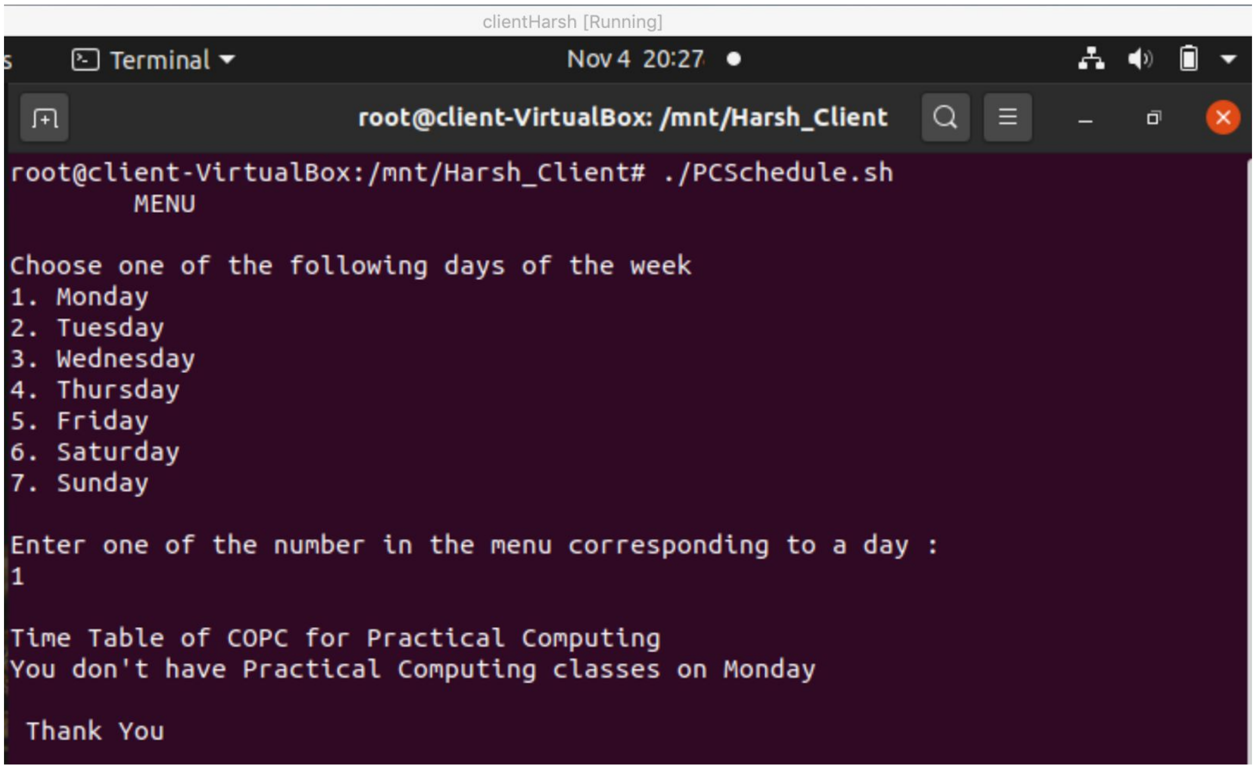
```
root@client-VirtualBox:~# mount 192.168.0.6:/mnt/Server_share /mnt/Harsh_Client
root@client-VirtualBox:~# ls -l /mnt/Harsh_Client
total 4
-rwxr-xr-x 1 root root 1036 Nov  4 16:10 PCSchedule.sh
```

4. Executing PCSchedule on client System

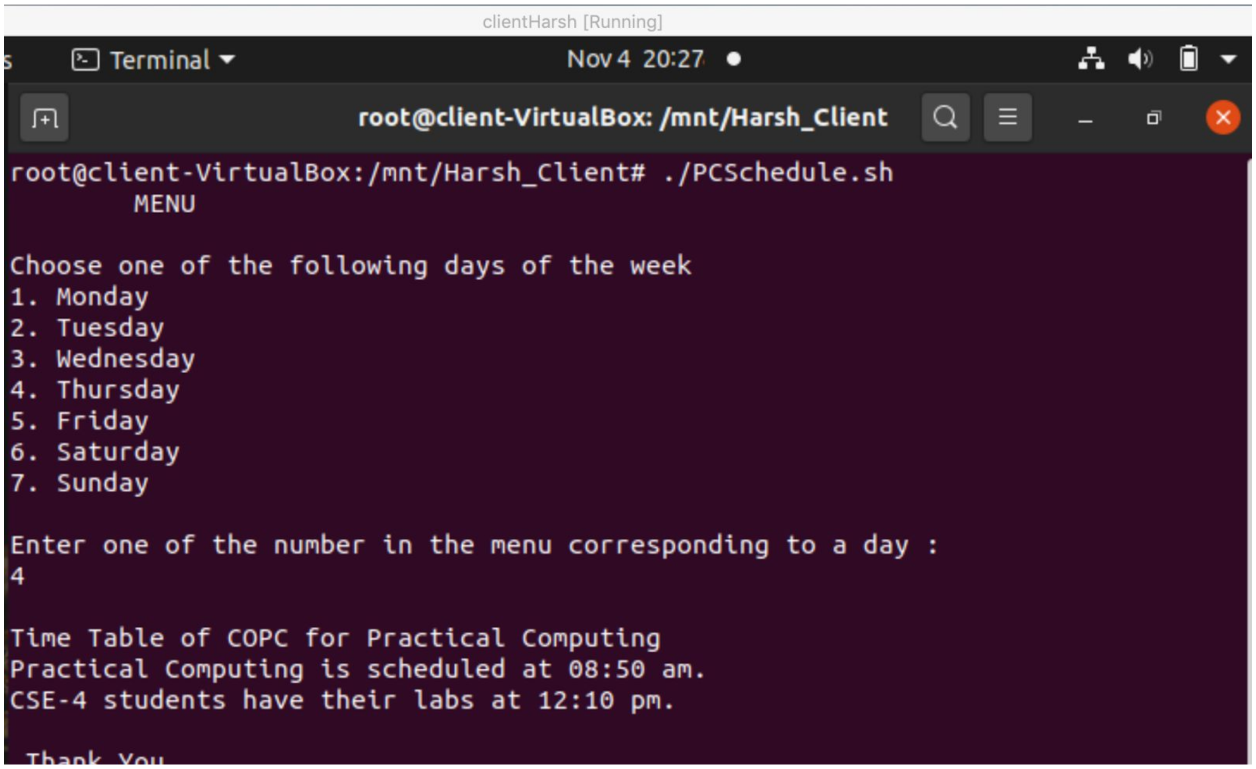
After this, you will see the same output window as shown above for the shell script PCSchedule.sh

OUTPUT

1. Fully Functional

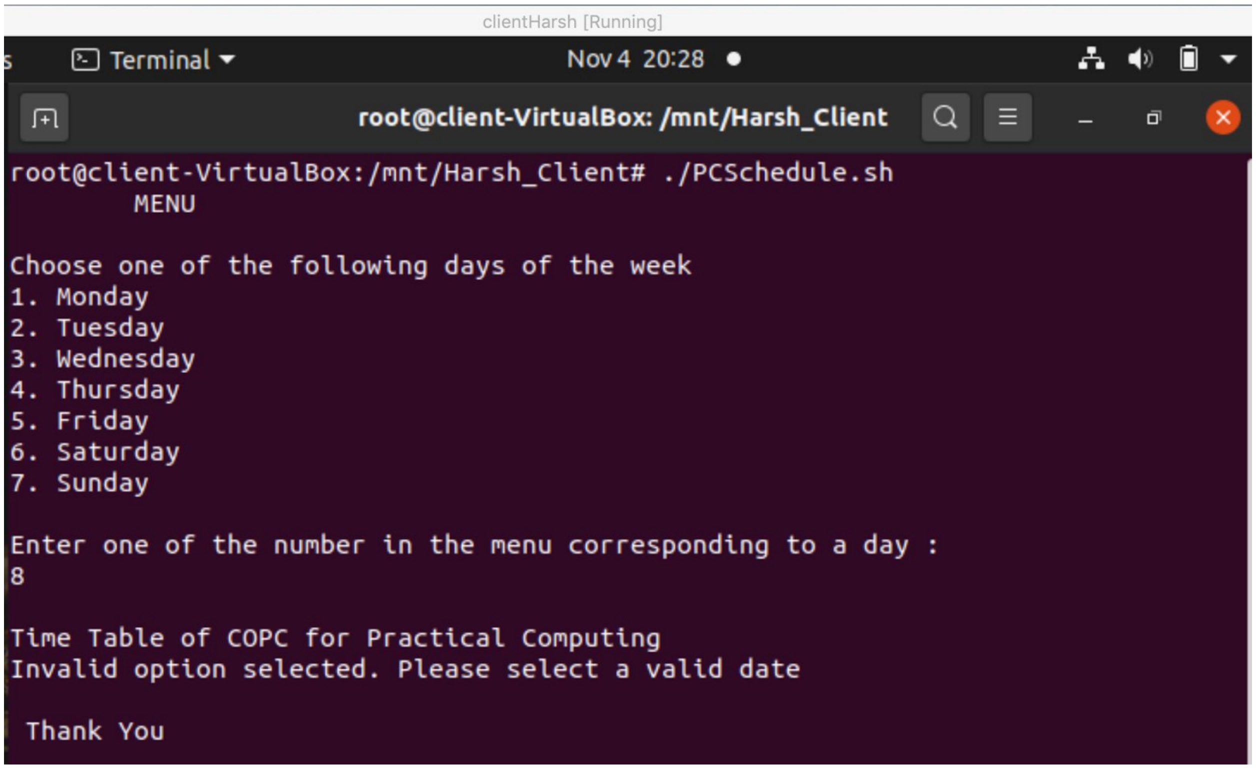


```
root@client-VirtualBox:/mnt/Harsh_Client# ./PCSchedule.sh
        MENU

Choose one of the following days of the week
1. Monday
2. Tuesday
3. Wednesday
4. Thursday
5. Friday
6. Saturday
7. Sunday

Enter one of the number in the menu corresponding to a day :
1

Time Table of COPC for Practical Computing
You don't have Practical Computing classes on Monday

 Thank You
```



```
root@client-VirtualBox:/mnt/Harsh_Client# ./PCSchedule.sh
        MENU

Choose one of the following days of the week
1. Monday
2. Tuesday
3. Wednesday
4. Thursday
5. Friday
6. Saturday
7. Sunday

Enter one of the number in the menu corresponding to a day :
4

Time Table of COPC for Practical Computing
Practical Computing is scheduled at 08:50 am.
CSE-4 students have their labs at 12:10 pm.

 Thank You
```

2. Fault Tolerant



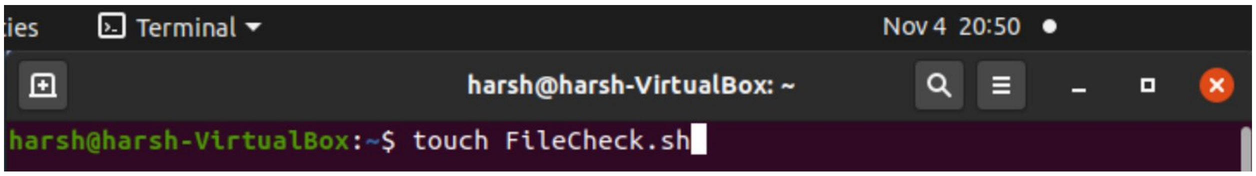
```
root@client-VirtualBox:/mnt/Harsh_Client# ./PCSchedule.sh
        MENU

Choose one of the following days of the week
1. Monday
2. Tuesday
3. Wednesday
4. Thursday
5. Friday
6. Saturday
7. Sunday

Enter one of the number in the menu corresponding to a day :
8

Time Table of COPC for Practical Computing
Invalid option selected. Please select a valid date

 Thank You
```

Question 3

Take a name as an input from the user.
Write a shell script to check whether there is a directory or file with that name. elf any of this exists then print "This is a directory" or "This is a file" respectively, otherwise give three options to the user for creating a new directory or file with that input name or do nothing. Create a directory or file or do nothing as per the selected option by user.

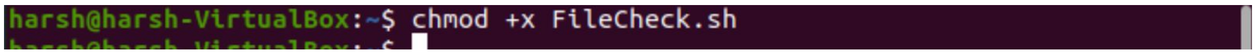
Solution -

Creating a Shell File for the Same and sharing the source code.



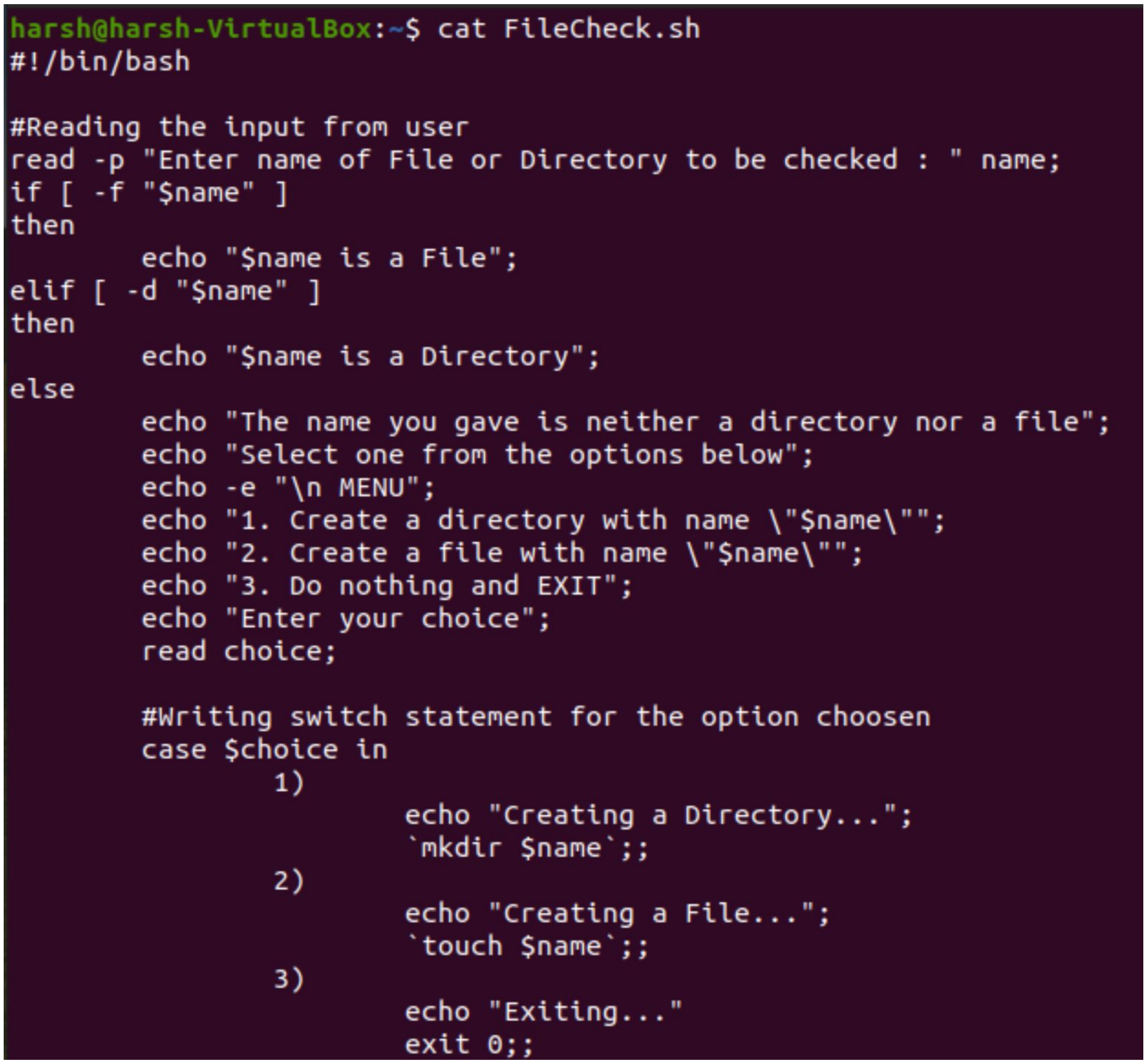
```
harsh@harsh-VirtualBox:~$ touch FileCheck.sh
```

Making the file executable.

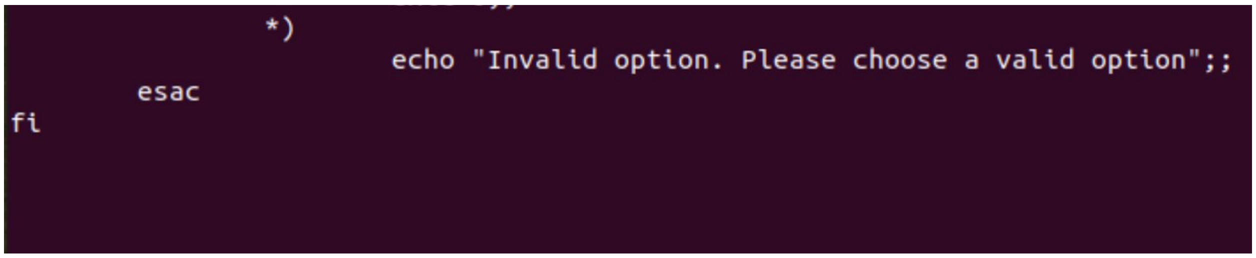


```
harsh@harsh-VirtualBox:~$ chmod +x FileCheck.sh
```

Source Code



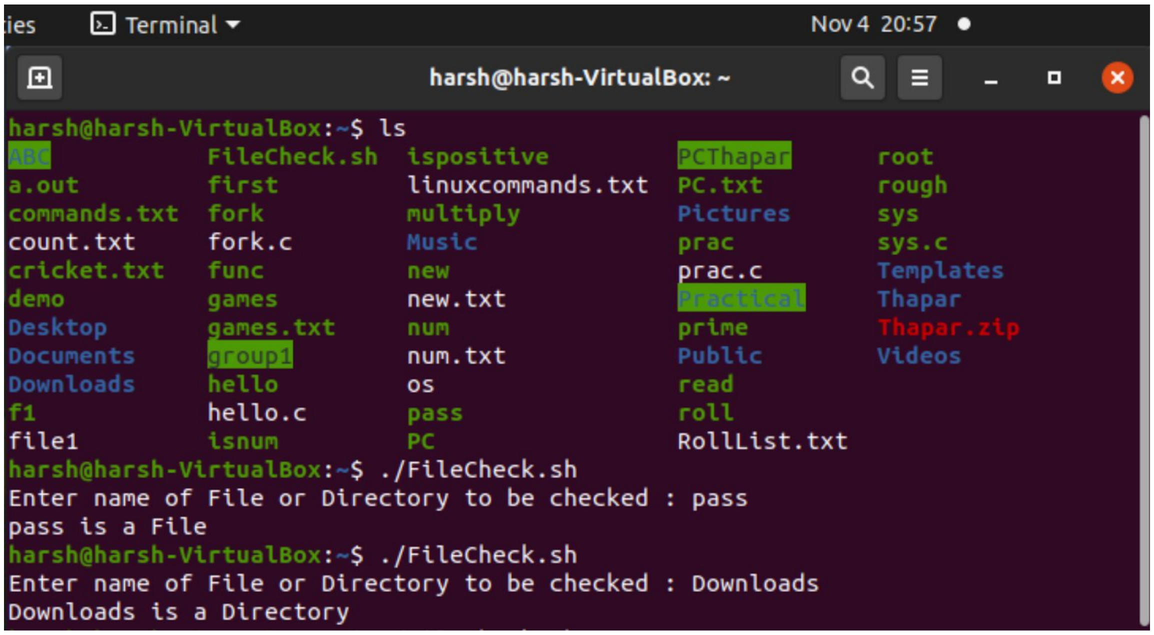
```
harsh@harsh-VirtualBox:~$ cat FileCheck.sh
#!/bin/bash

#Reading the input from user
read -p "Enter name of File or Directory to be checked : " name;
if [ -f "$name" ]
then
        echo "$name is a File";
elif [ -d "$name" ]
then
        echo "$name is a Directory";
else
        echo "The name you gave is neither a directory nor a file";
        echo "Select one from the options below";
        echo -e "\n MENU";
        echo "1. Create a directory with name \"$name\"";
        echo "2. Create a file with name \"$name\"";
        echo "3. Do nothing and EXIT";
        echo "Enter your choice";
        read choice;

        #Writing switch statement for the option choosen
        case $choice in
                1)
                        echo "Creating a Directory...";
                        `mkdir $name`;;
                2)
                        echo "Creating a File...";
                        `touch $name`;;
                3)
                        echo "Exiting..."
                        exit 0;;
```



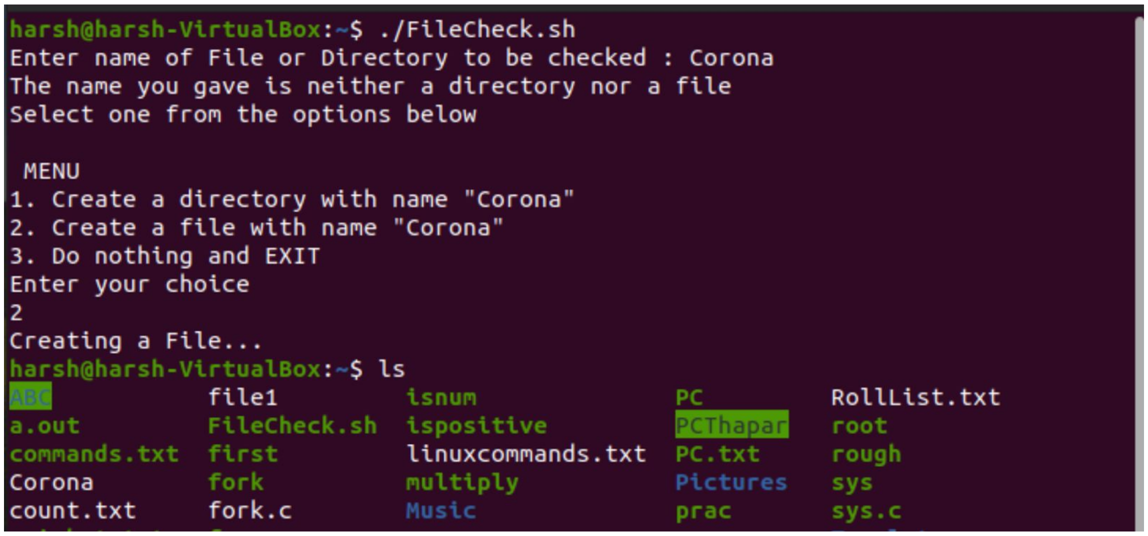
```
               *)
                        echo "Invalid option. Please choose a valid option";;
        esac
fi
```

OUTPUT

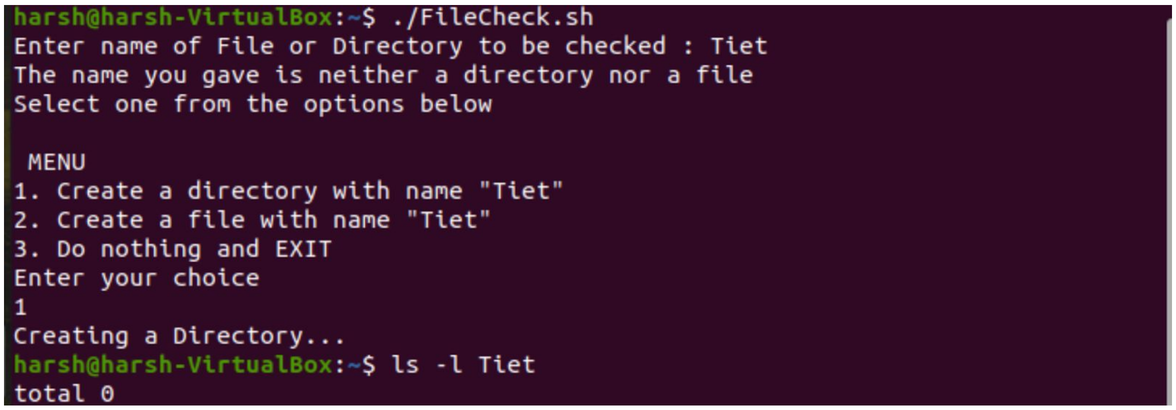
1. **Fully Functional**



```
harsh@harsh-VirtualBox:~$ ls
ABC            FileCheck.sh  ispositive         PCThapar   root
a.out          first         linuxcommands.txt  PC.txt     rough
commands.txt   fork          multiply           Pictures   sys
count.txt      fork.c        Music              prac       sys.c
cricket.txt    func          new                prac.c     Templates
demo           games         new.txt            Practical  Thapar
Desktop        games.txt     num                prime      Thapar.zip
Documents      group1        num.txt            Public     Videos
Downloads      hello         os                 read
f1             hello.c       pass               roll
file1          isnum         PC                 RollList.txt
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : pass
pass is a File
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : Downloads
Downloads is a Directory
```

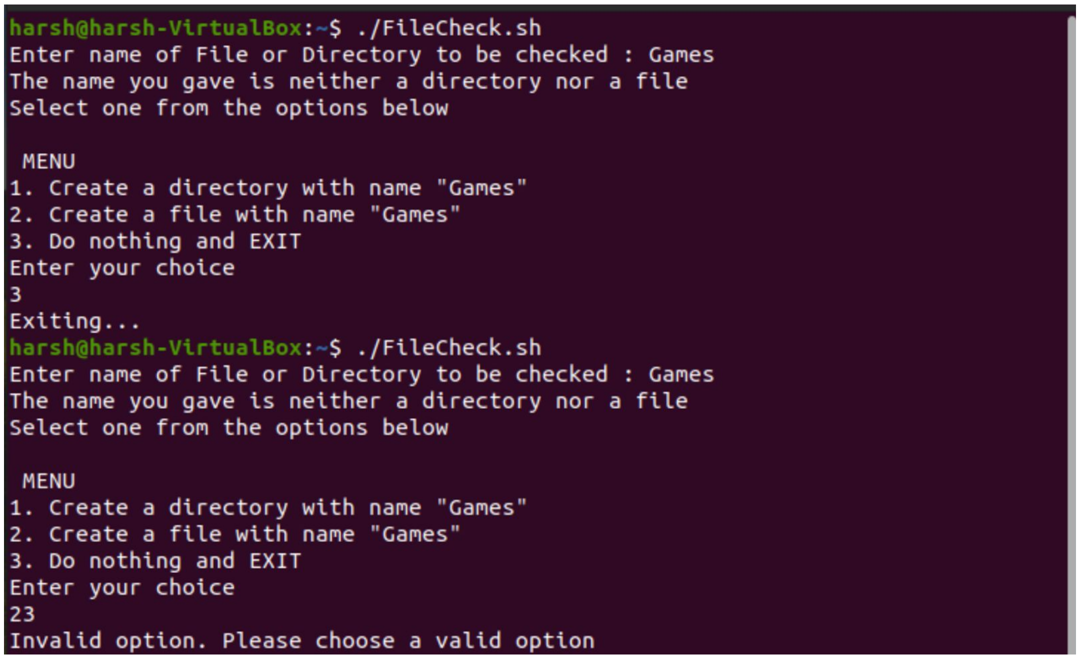


```
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : Corona
The name you gave is neither a directory nor a file
Select one from the options below

 MENU
1. Create a directory with name "Corona"
2. Create a file with name "Corona"
3. Do nothing and EXIT
Enter your choice
2
Creating a File...
harsh@harsh-VirtualBox:~$ ls
ABC            file1         isnum              PC         RollList.txt
a.out          FileCheck.sh  ispositive         PCThapar   root
commands.txt   first         linuxcommands.txt  PC.txt     rough
Corona         fork          multiply           Pictures   sys
count.txt      fork.c        Music              prac       sys.c
```



```
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : Tiet
The name you gave is neither a directory nor a file
Select one from the options below

 MENU
1. Create a directory with name "Tiet"
2. Create a file with name "Tiet"
3. Do nothing and EXIT
Enter your choice
1
Creating a Directory...
harsh@harsh-VirtualBox:~$ ls -l Tiet
total 0
```

2. **Fault Tolerant**



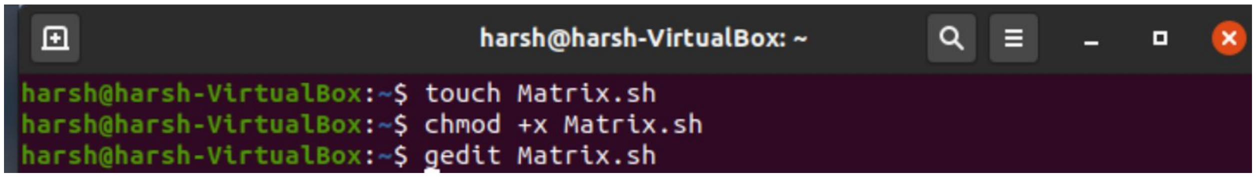
```
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : Games
The name you gave is neither a directory nor a file
Select one from the options below

 MENU
1. Create a directory with name "Games"
2. Create a file with name "Games"
3. Do nothing and EXIT
Enter your choice
3
Exiting...
harsh@harsh-VirtualBox:~$ ./FileCheck.sh
Enter name of File or Directory to be checked : Games
The name you gave is neither a directory nor a file
Select one from the options below

 MENU
1. Create a directory with name "Games"
2. Create a file with name "Games"
3. Do nothing and EXIT
Enter your choice
23
Invalid option. Please choose a valid option
```

Question 4

Write a shell script for multiplying 2 matrices and printing the resultant matrix. Here you need to represent a 2-dimensional matrix using a 1-dimensional array

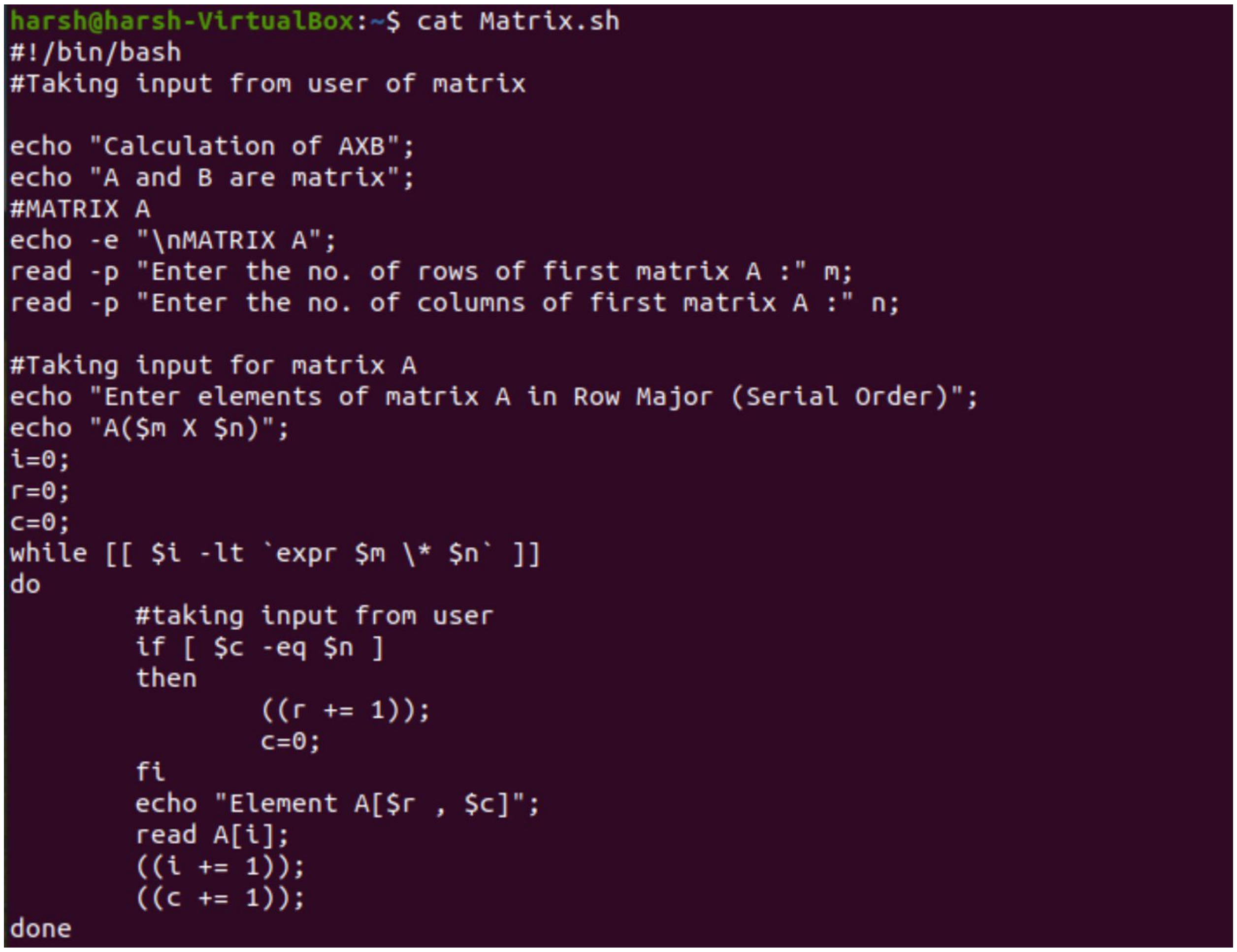
Solution -

Creating a Shell File for the Same and sharing the source code.

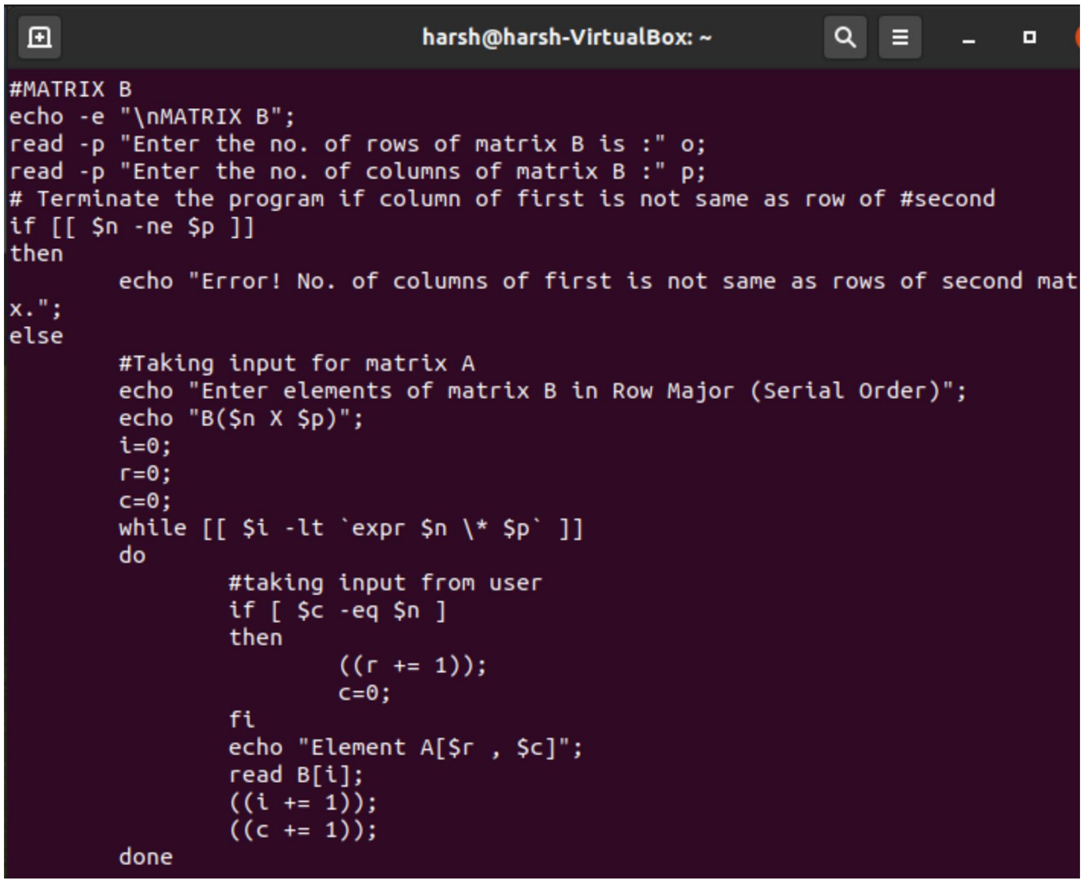


```
harsh@harsh-VirtualBox:~$ touch Matrix.sh
harsh@harsh-VirtualBox:~$ chmod +x Matrix.sh
harsh@harsh-VirtualBox:~$ gedit Matrix.sh
```

SOURCE CODE

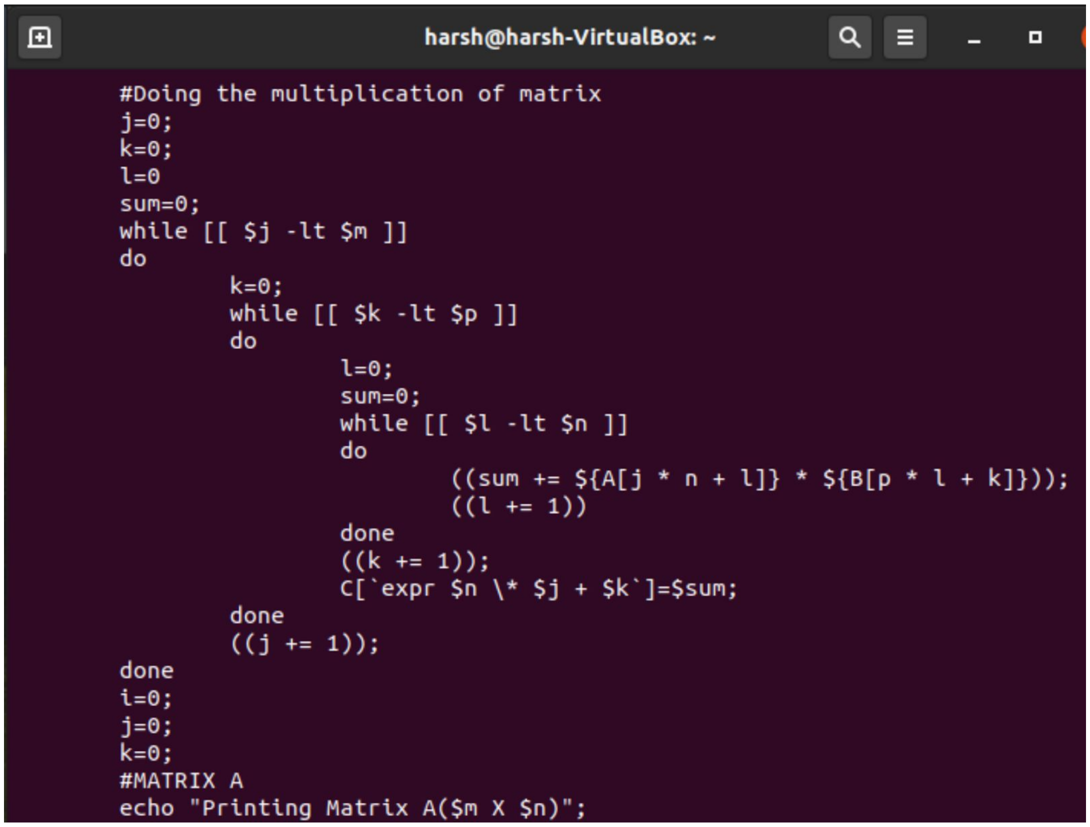


```
harsh@harsh-VirtualBox:~$ cat Matrix.sh
#!/bin/bash
#Taking input from user of matrix

echo "Calculation of AXB";
echo "A and B are matrix";
#MATRIX A
echo -e "\nMATRIX A";
read -p "Enter the no. of rows of first matrix A :" m;
read -p "Enter the no. of columns of first matrix A :" n;

#Taking input for matrix A
echo "Enter elements of matrix A in Row Major (Serial Order)";
echo "A($m X $n)";
i=0;
r=0;
c=0;
while [[ $i -lt `expr $m \* $n` ]]
do
        #taking input from user
        if [ $c -eq $n ]
        then
                ((r += 1));
                c=0;
        fi
        echo "Element A[$r , $c]";
        read A[i];
        ((i += 1));
        ((c += 1));
done
```



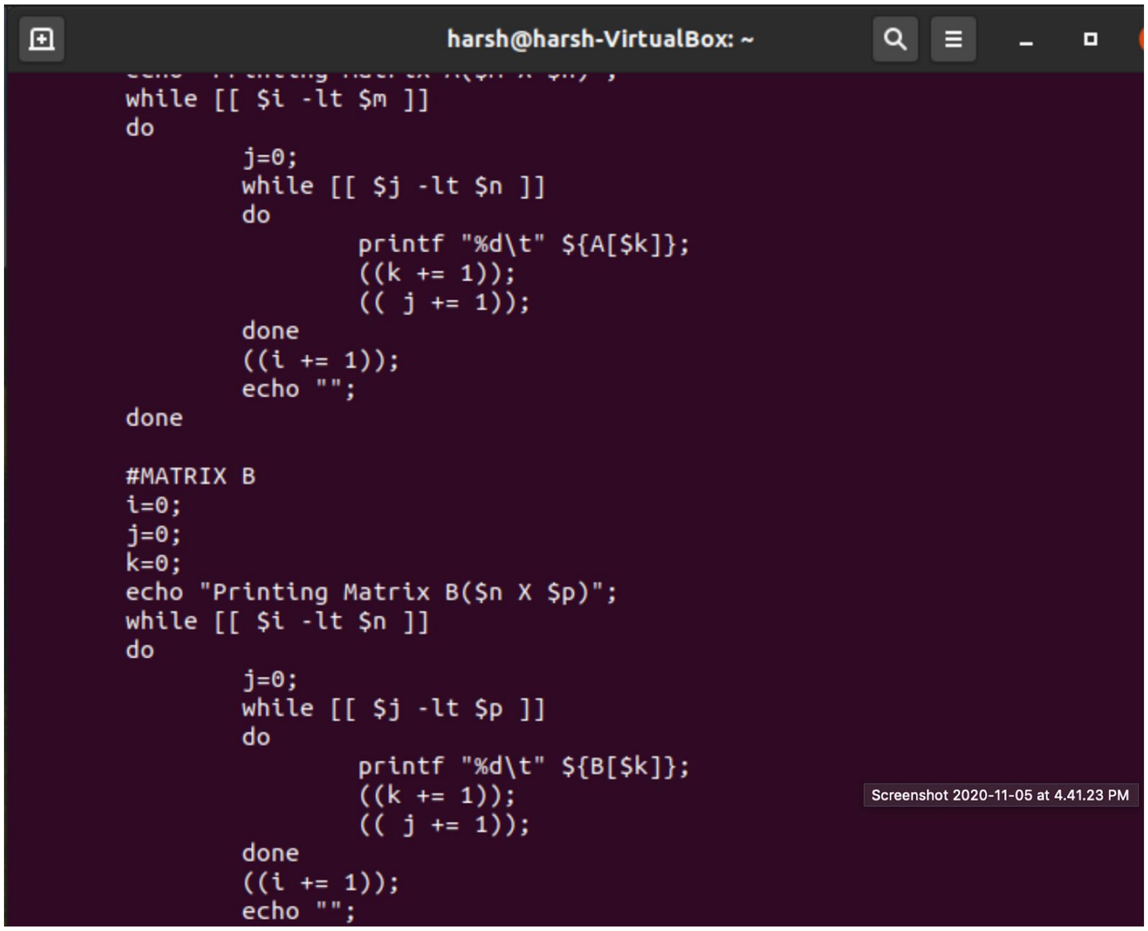
harsh@harsh-VirtualBox: ~

```
#MATRIX B
echo -e "\nMATRIX B";
read -p "Enter the no. of rows of matrix B is :" o;
read -p "Enter the no. of columns of matrix B :" p;
# Terminate the program if column of first is not same as row of #second
if [[ $n -ne $p ]]
then
        echo "Error! No. of columns of first is not same as rows of second mat
x.";
else
        #Taking input for matrix A
        echo "Enter elements of matrix B in Row Major (Serial Order)";
        echo "B($n X $p)";
        i=0;
        r=0;
        c=0;
        while [[ $i -lt `expr $n \* $p` ]]
        do
                #taking input from user
                if [ $c -eq $n ]
                then
                        ((r += 1));
                        c=0;
                fi
                echo "Element A[$r , $c]";
                read B[i];
                ((i += 1));
                ((c += 1));
        done
```

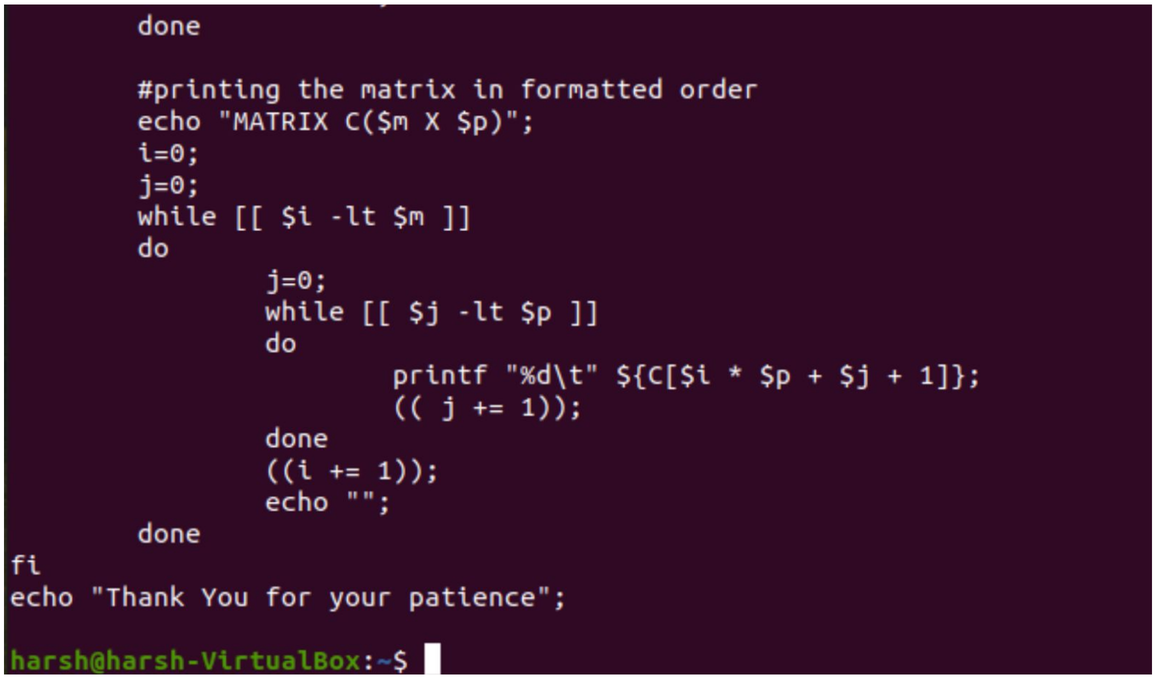


harsh@harsh-VirtualBox: ~

```
        #Doing the multiplication of matrix
        j=0;
        k=0;
        l=0
        sum=0;
        while [[ $j -lt $m ]]
        do
                k=0;
                while [[ $k -lt $p ]]
                do
                        l=0;
                        sum=0;
                        while [[ $l -lt $n ]]
                        do
                                ((sum += ${A[j * n + l]} * ${B[p * l + k]}));
                                ((l += 1))
                        done
                        ((k += 1));
                        C[`expr $n \* $j + $k`]=$sum;
                done
                ((j += 1));
        done
        i=0;
        j=0;
        k=0;
        #MATRIX A
        echo "Printing Matrix A($m X $n)";
```


harsh@harsh-VirtualBox: ~
Screenshot 2020-11-05 at 4.41.23 PM

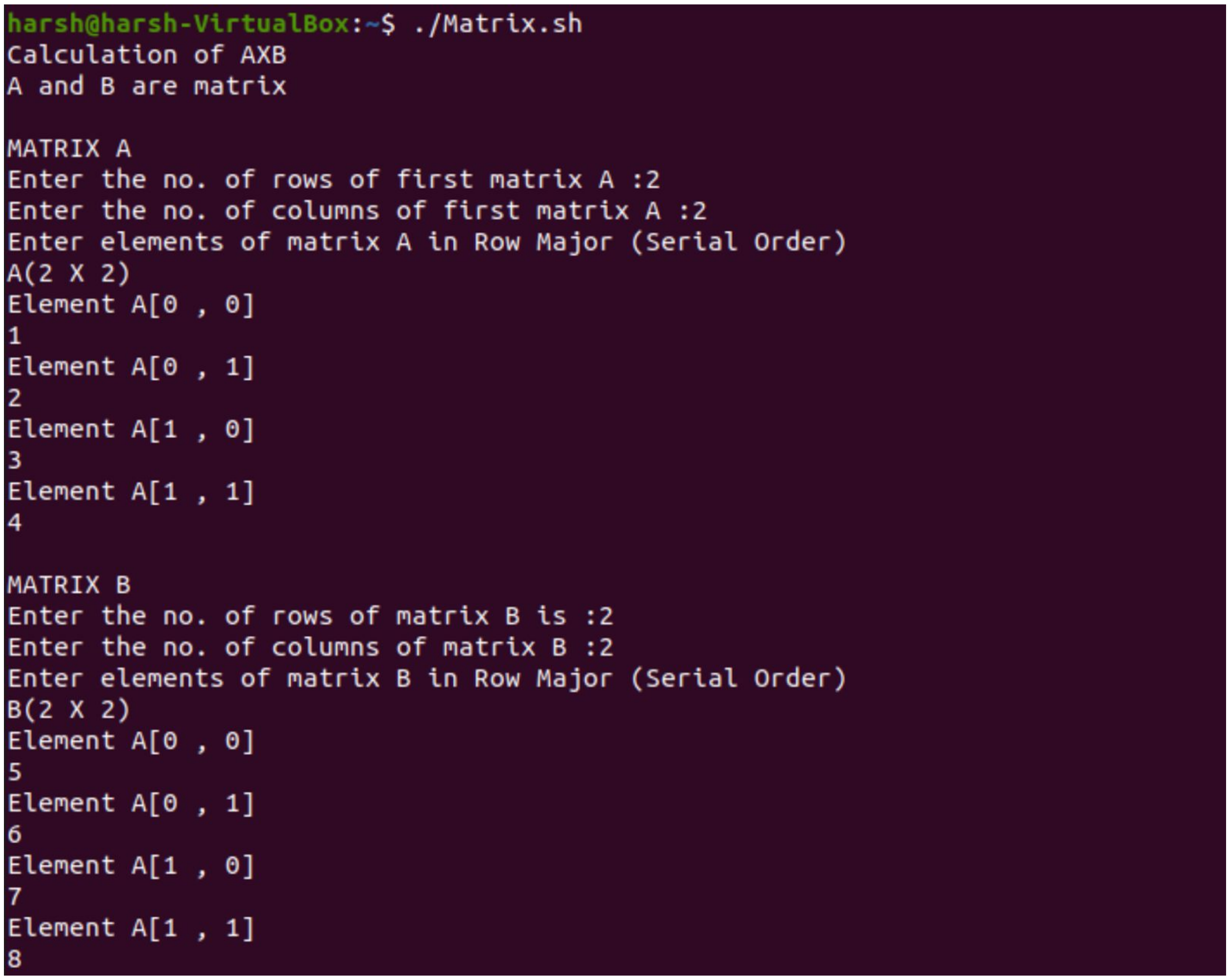
```
        while [[ $i -lt $m ]]
        do
                j=0;
                while [[ $j -lt $n ]]
                do
                        printf "%d\t" ${A[$k]};
                        ((k += 1));
                        (( j += 1));
                done
                ((i += 1));
                echo "";
        done

        #MATRIX B
        i=0;
        j=0;
        k=0;
        echo "Printing Matrix B($n X $p)";
        while [[ $i -lt $n ]]
        do
                j=0;
                while [[ $j -lt $p ]]
                do
                        printf "%d\t" ${B[$k]};
                        ((k += 1));
                        (( j += 1));
                done
                ((i += 1));
                echo "";
```



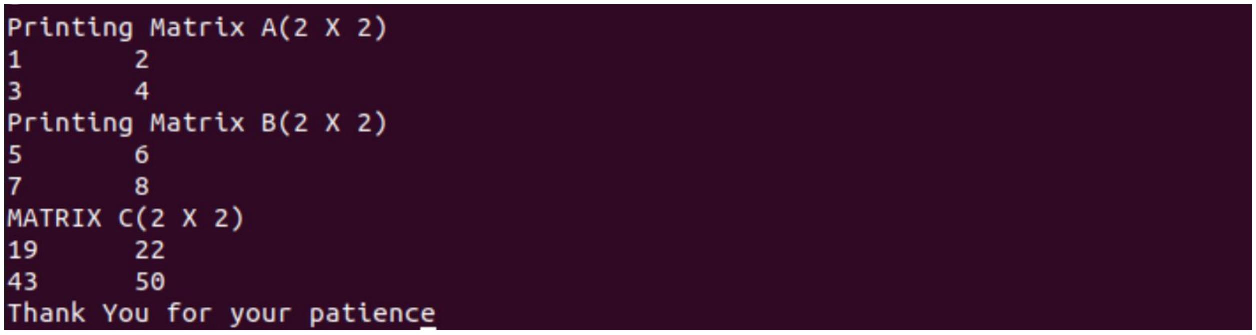
```
        done

        #printing the matrix in formatted order
        echo "MATRIX C($m X $p)";
        i=0;
        j=0;
        while [[ $i -lt $m ]]
        do
                j=0;
                while [[ $j -lt $p ]]
                do
                        printf "%d\t" ${C[$i * $p + $j + 1]};
                        (( j += 1));
                done
                ((i += 1));
                echo "";
        done
fi
echo "Thank You for your patience";

harsh@harsh-VirtualBox:~$ 
```

OUTPUT

1. **Fully Functional**

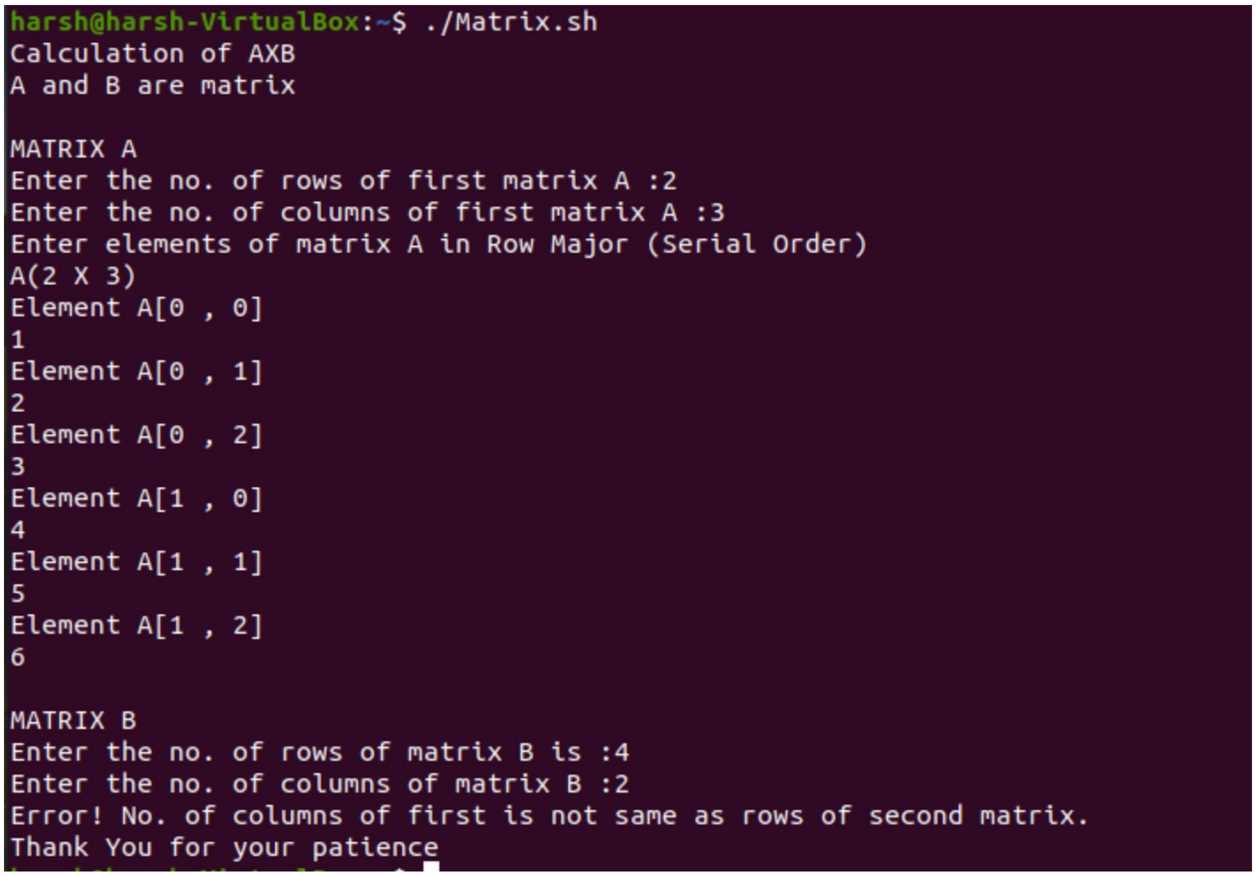


```
harsh@harsh-VirtualBox:~$ ./Matrix.sh
Calculation of AXB
A and B are matrix

MATRIX A
Enter the no. of rows of first matrix A :2
Enter the no. of columns of first matrix A :2
Enter elements of matrix A in Row Major (Serial Order)
A(2 X 2)
Element A[0 , 0]
1
Element A[0 , 1]
2
Element A[1 , 0]
3
Element A[1 , 1]
4

MATRIX B
Enter the no. of rows of matrix B is :2
Enter the no. of columns of matrix B :2
Enter elements of matrix B in Row Major (Serial Order)
B(2 X 2)
Element A[0 , 0]
5
Element A[0 , 1]
6
Element A[1 , 0]
7
Element A[1 , 1]
8
```



```
Printing Matrix A(2 X 2)
1       2
3       4
Printing Matrix B(2 X 2)
5       6
7       8
MATRIX C(2 X 2)
19      22
43      50
Thank You for your patience
```

2. **Fault Tolerant**



```
harsh@harsh-VirtualBox:~$ ./Matrix.sh
Calculation of AXB
A and B are matrix

MATRIX A
Enter the no. of rows of first matrix A :2
Enter the no. of columns of first matrix A :3
Enter elements of matrix A in Row Major (Serial Order)
A(2 X 3)
Element A[0 , 0]
1
Element A[0 , 1]
2
Element A[0 , 2]
3
Element A[1 , 0]
4
Element A[1 , 1]
5
Element A[1 , 2]
6

MATRIX B
Enter the no. of rows of matrix B is :4
Enter the no. of columns of matrix B :2
Error! No. of columns of first is not same as rows of second matrix.
Thank You for your patience
```

———————————————————————————————————

<u>Question 5</u>

Conversion:

- IPV4 to IPV6
- IPV6 to IPV4

<u>Solution</u> -

IPv4

IPv4 addresses are 32-bit numbers that are typically displayed in dotted decimal notation. A 32-bit address contains two primary parts: the network prefix and the host number. IPv4 is the most widely used version of the protocol despite the limitations of its 32-bit address space.

All hosts within a single network share the same network address. Each host also has an address that uniquely identifies it.
Depending on the scope of the network and the type of device, the address is either globally or locally unique. Devices that are visible to users outside the network (webservers, for example) must have a globally unique IP address. Devices that are visible only within the network must have locally unique IP addresses.

An IPv4 address is a series of four eight-bit binary numbers separated by a decimal point.
Example

Site	*Dot-decimal*	*Binary*
Google.com	*172.217.168.238*	*10101100.11011001.10101000.11101110*

IPv6

Internet Protocol version 6 is the most recent version of the Internet Protocol, the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet.

IPv4 to IPV6

In this method

1. **Convert the Decimal IPv4 address in to Binary representation.**
2. **Divide this Binary representation into four groups.**
3. **Make sure each group have the 8 bit binary number.**
4. **Divide this binary number in two parts which is 4 bits**
5. **Convert this 4 bit into hexadecimal number representation.**
6. **We get hexadecimal number i.e. 8 hexadecimal number**
7. **Group in to two parts i.e. 4 hexadecimal in in one group which is 16 bit binary number in the one group in IPv6 address.**

So we get two group out of 8 group of IPv6 address with the help of IPv4 address.

Example:

We have IPv4 : 192.168.25.234

- Converting this IPv4 into Binary representation.

  192 - 11000000

  168 - 10101000

  25 - 00011001

  134 - 11101010

  So, it is basically

  192.168.25.234

  11000000.10101000.00011001.11101010

- Dividing it into 4 groups

  11000000 10101000 00011001 11101010
- Dividing this binary number in two parts which is 4 bits

  1100 | 0000

  1010 | 1000

  0001 | 1001

  1110 | 1010
- Converting this 4 bit into hexadecimal number representation

1100 - C	0000 - 0
1010 - A	1000 - 8
0001 - 1	1001 - 9
1110 - E	1010 - A

- Now, Grouping it back of 4 bit

  C0A8 19EA
- Putting it together

  IPv6 : C0A8:19EA

IPv6 to IPV4

In this method , we will follow the reverse fashion

1. **Convert the hexadecimal IPv6 address in to Binary representation.**
2. **Divide this Binary representation into two groups.**
3. **Make sure each group have the 16 bit binary number.**
4. **Divide each group in two parts each consisting of 8 bits**
5. **Convert this 8 bit into decimal number representation.**
6. **We get two decimal representation each from two group, four in total.**

Example

Taking the same IPv6 : C0A8:19EA

- Converting the hexadecimal IPv6 address in to Binary representation.
  C - 1100
  0 - 0000
  A - 1010
  8 - 1000
  1 - 0010
  9 - 1001
  E - 1110
  A - 1010
- Dividing this Binary representation into two groups.

```
COA8                    19EA
1100  0000 1010 1000    0001 1001 1110 1010
```

- Dividing each group in two parts each consisting of 8 bits

```
1100  0000  |  1010 1000     0001 1001  |  1110 1010
```

- Converting this 8 bit into decimal number representation.
  1100 0000 - 192
  1010 1000 - 168
  0001 1001 - 25
  1110 1010 - 234
- Groupong it together.

```
1100  0000  |  1010 1000     0001 1001  |  1110 1010
192             168             25             234
```

- Seperate it with dot notation and we are done.
  IPv4 : 192.168.25.234

——————————————————————————————

Question 6

#!/bin/bash

Array= (Csed, Practical, Computing, UCS311,Tiet)

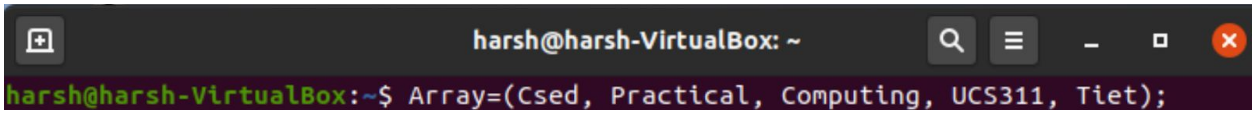
a. **Write a command to display all elements except first one**

b. what will be the output of

```
    i.  echo ${arr[0]:1}
   ii.  echo ${array[@]:1:3}
  iii.  echo ${array[1]:5:5}
```

Solution -

We can use arrayin terminal.
Array=(Csed, Practical, Computing, UCS311, Tiet);

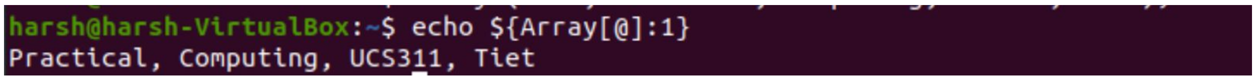


```
harsh@harsh-VirtualBox:~$ Array=(Csed, Practical, Computing, UCS311, Tiet);
```

We will create an array as arr

a. To display all elements except first one, we will print following command

```
echo ${arr[@]:1};
```



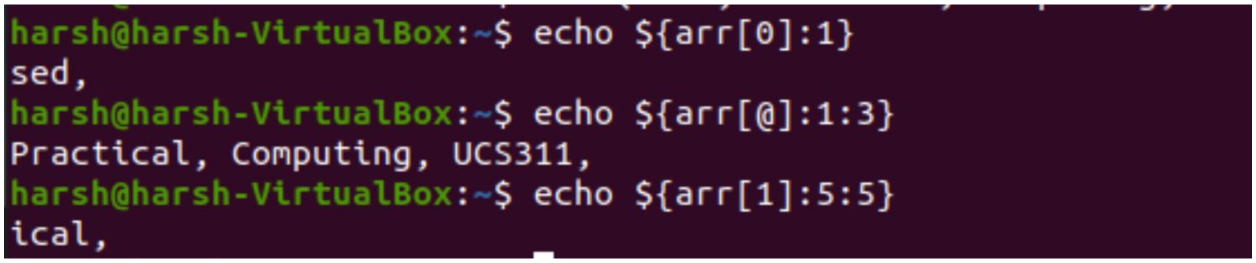
```
harsh@harsh-VirtualBox:~$ echo ${Array[@]:1}
Practical, Computing, UCS311, Tiet
```

b. Displaying Output

```
    i.  echo ${arr[0]:1}
```
Output - sed,

```
   ii.  echo ${array[@]:1:3}
```
Output - Practical, Computing, UCS311

```
  iii.  echo ${array[1]:5:5}
```
Output - ical,



```
harsh@harsh-VirtualBox:~$ echo ${arr[0]:1}
sed,
harsh@harsh-VirtualBox:~$ echo ${arr[@]:1:3}
Practical, Computing, UCS311,
harsh@harsh-VirtualBox:~$ echo ${arr[1]:5:5}
ical,
```

———————————————————————————————

Thank you...