# LINUX AND SHELL PROGRAMMING - LAB

**(Course Code: 23UPCSC1L01)**

**A laboratory record submitted to Periyar University, Salem**

**In partial fulfillment of the requirements for the degree of**

**MASTER OF COMPUTER APPLICATIONS**

**By**

**NAGARAJAN S**

**[Reg. No: U24PG507CAP016]**


**DEPARTMENT OF COMPUTER SCIENCE**

**PERIYAR UNIVERSITY**

**(NACC `A++` Grade with CGPA 3.61) – NIRF RANK 56 – ARIIA RANK 10**

**PERIYAR PALKALAI NAGAR,**

**SALEM – 636 011.**

**(OCTOBER - 2024)**

# CERTIFICATE

This is to certify that the Programming Laboratory entitled "**LINUX AND SHELL PROGRAMMING LAB (23UPCSC1L01)"** is a bonafide record work done by Mr. /Ms. _____Register No:_____as partial fulfillment of the requirements for the degree of Master of Computer Applications, in the Department of Computer Science, Periyar University, Salem, during the Academic Year 2024-2025.

Staff In-charge Head of the Department

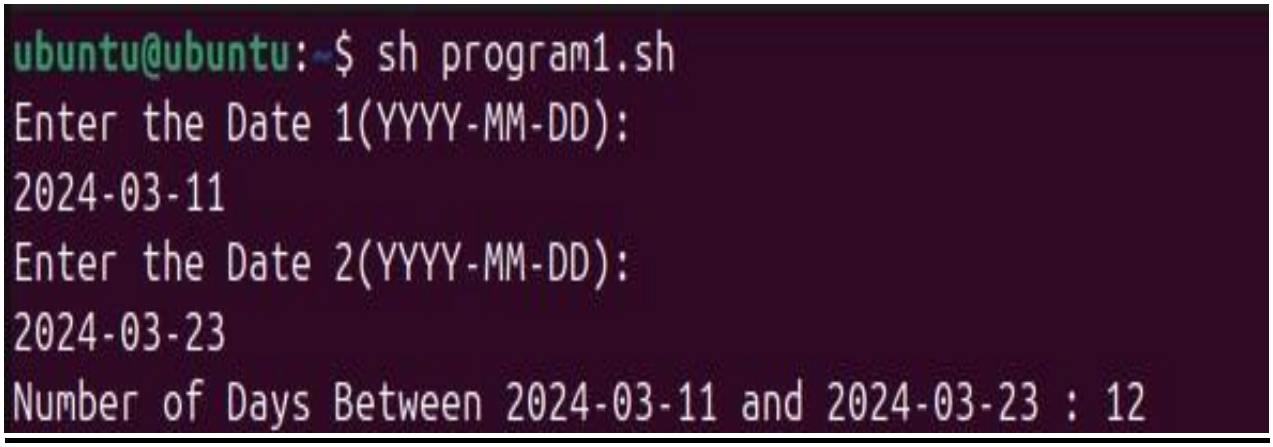
Submitted for the practical examination held on……..……

Internal Examiner External Examiner

## SOURCE CODE

```
echo "Enter the Date1 (YYYY-MM-DD):"
read date1
echo "Enter the Date2 (YYYY-MM-DD):"
read date2
datetosec1=$(date -d "$date1" +%s)
datetosec2=$(date -d "$date2" +%s)
diffsec=$((datetosec2-datetosec1))
diffdays=$((diffsec/86400))
echo "Number of Day between '$date1' and '$date2' is : $diffdays days"
```

**OUTPUT**


ubuntu@ubuntu:~$ sh program1.sh
Enter the Date 1(YYYY-MM-DD):
2024-03-11
Enter the Date 2(YYYY-MM-DD):
2024-03-23
Number of Days Between 2024-03-11 and 2024-03-23 : 12

## SOURCE CODE

```
echo "Enter the Base IP Address (e.g 192.168.1):"
read base_ip
echo "Enter the start of the IP range:"
read start_ip
echo "Enter the end of the IP range:"
read end_ip

if [ -z "$base_ip" ] || [ -z "$start_ip" ] || [ -z "$end_ip" ]
then
echo "Invalid IP Address,Please enter valid IP and range"
exit 1
fi

for ip in $(seq $start_ip $end_ip )
do
current_ip="$base_ip.$ip"
ping -c 1 -w 1 "$current_ip" &> /dev/null
if [ $? -eq 0 ]
then
echo "IP $current_ip is online"
else
echo "IP $current_ip is offline"
fi
done
```

# OUTPUT

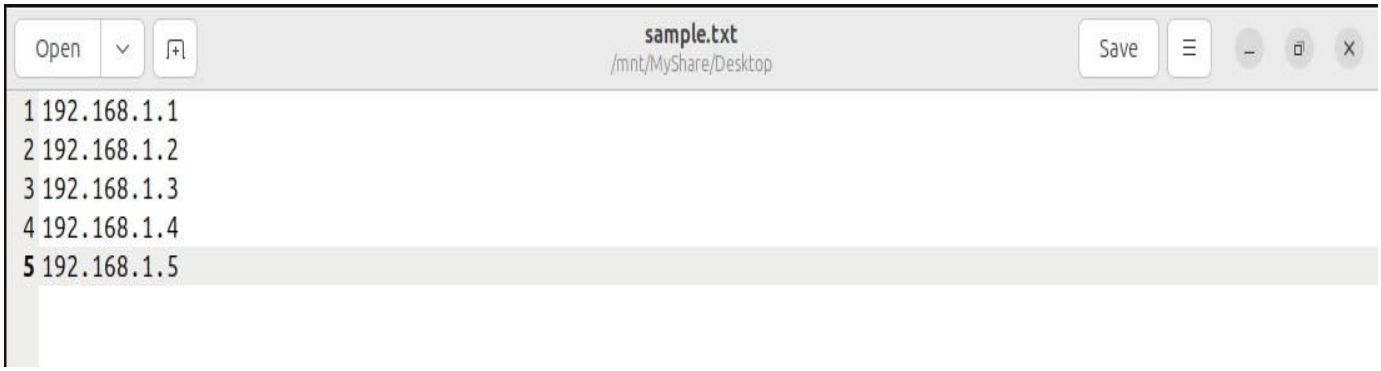
```
ubuntu@ubuntu:~$ sh program2.sh
Enter the Base IP Address (e.g 192.168.1):
192.168.1
Enter the start of the IP range:
1
Enter the end of the IP range:
7
IP 192.168.1.1 is online
IP 192.168.1.2 is online
IP 192.168.1.3 is online
IP 192.168.1.4 is online
IP 192.168.1.5 is online
IP 192.168.1.6 is online
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
PING 192.168.1.4 (192.168.1.4) 56(84) bytes of data.
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
PING 192.168.1.6 (192.168.1.6) 56(84) bytes of data.
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
IP 192.168.1.7 is online
ubuntu@ubuntu:~$ PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
PING 192.168.1.7 (192.168.1.7) 56(84) bytes of data.

--- 192.168.1.1 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

## SOURCE CODE

```
echo "Enter the filename contains IP Address:"
read filename
if [ ! -f "$filename" ]
then
echo "File not Found, PLease enter valid Filename"
exit 1
fi

while IFS= read -r ip
do
if [ -n "$ip" ]
then
  ping -c 1 -w 1 "$ip" &> /dev/null
  if [ $? -eq 0 ]
  then
  echo "IP $ip is online"
  else
  echo "IP $ip is offline"
 fi
fi
done <"$filename"
```

# OUTPUT

- Sample.txt file




```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ gedit program3.sh
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh program3.sh
Enter the filename contains IP Address:
sample.txt
IP 192.168.1.1 is online
IP 192.168.1.2 is online
IP 192.168.1.3 is online
IP 192.168.1.4 is online
IP 192.168.1.5 is online
ubuntu@ubuntu:/mnt/MyShare/Desktop$ PING 192.168.1.1 (192.168.1.1) 56(84) bytes
of data.
PING 192.168.1.4 (192.168.1.4) 56(84) bytes of data.
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
```

## SOURCE CODE

```
#!/bin/bash

trap "echo 'Signal caught! Exiting...'; exit" SIGINT

echo "This is a demonstration of script control commands."


echo "Enter a number (negative to exit the script):"
read num

if [ $num -lt 0 ]; then
    echo "You entered a negative number. Exiting..."
    exit 1
fi


echo "For loop demonstration with 'continue':"

for i in 1 2 3 4 5

do

    if [ $i -eq 3 ]; then

        echo "Skipping iteration $i using continue..."

        continue

    fi

    echo "Iteration $i"

done


echo "While loop demonstration with 'break':"

counter=5

while [ $counter -gt 0 ]; do

    echo "Counter is $counter"

    if [ $counter -eq 3 ]; then
```

```
        echo "Breaking the loop when counter is $counter"
        break
    fi
    counter=$((counter-1))
done

echo "Script execution completed."
```

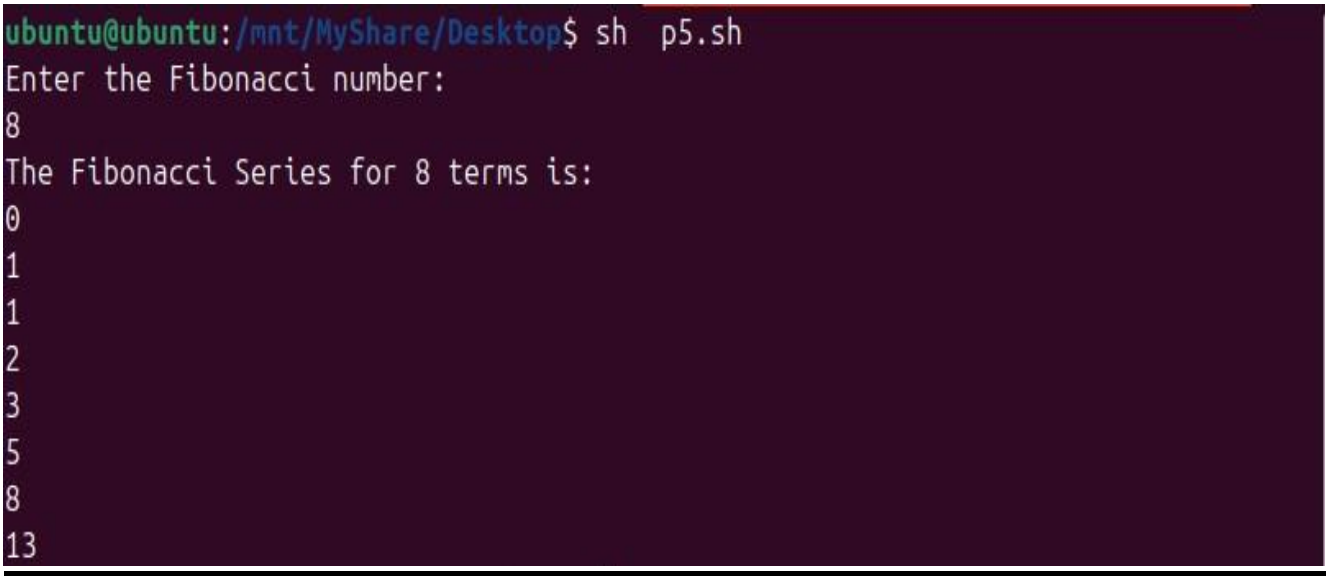
## OUTPUT

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh program4.sh
trap: SIGINT: bad trap
This is a demonstration of script control commands.
Enter a number (negative to exit the script):
-2
You entered a negative number. Exiting...
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh program4.sh
trap: SIGINT: bad trap
This is a demonstration of script control commands.
Enter a number (negative to exit the script):
4
For loop demonstration with 'continue':
Iteration 1
Iteration 2
Skipping iteration 3 using continue...
Iteration 4
Iteration 5
While loop demonstration with 'break':
Counter is 5
Counter is 4
Counter is 3
Breaking the loop when counter is 3
Script execution completed.
```

## SOURCE CODE

```
fibo(){
    a=0
    b=1
    c=0
    echo "The Fibonacci Series for $1 terms is:"
    for i in $(seq 1 $1)
    do
        echo "$c"
        c=$((a + b))
        b=$a
        a=$c
    done
}

echo "Enter the Fibonacci number: "
read n
fibo "$n"
```

# OUTPUT

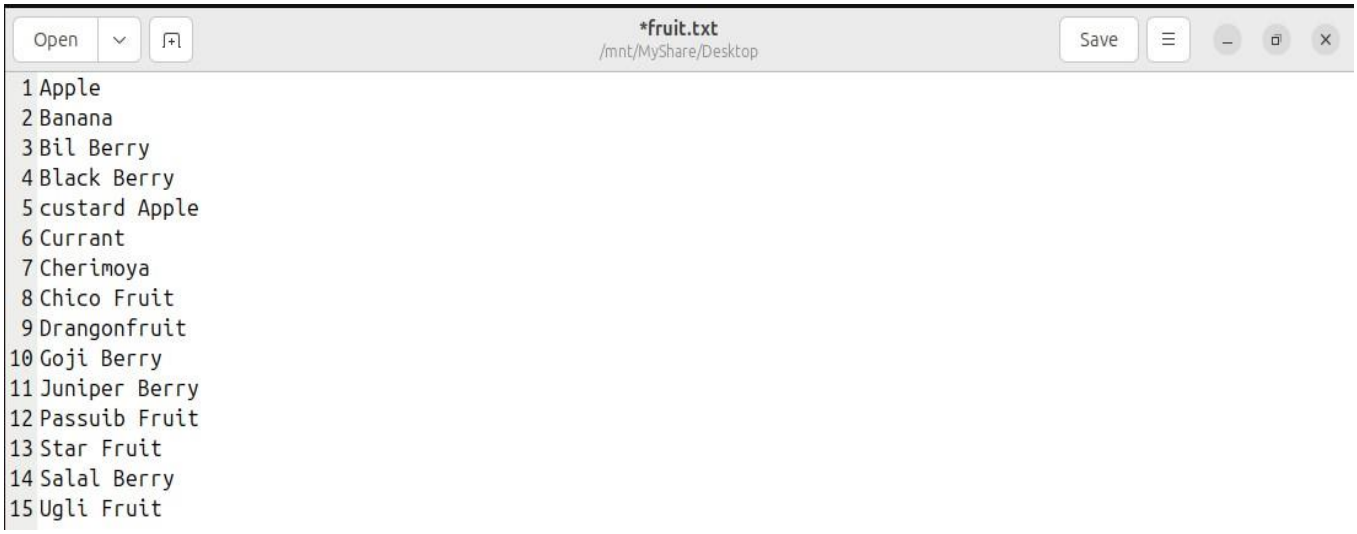

ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh p5.sh
Enter the Fibonacci number:
8
The Fibonacci Series for 8 terms is:
0
1
1
2
3
5
8
13

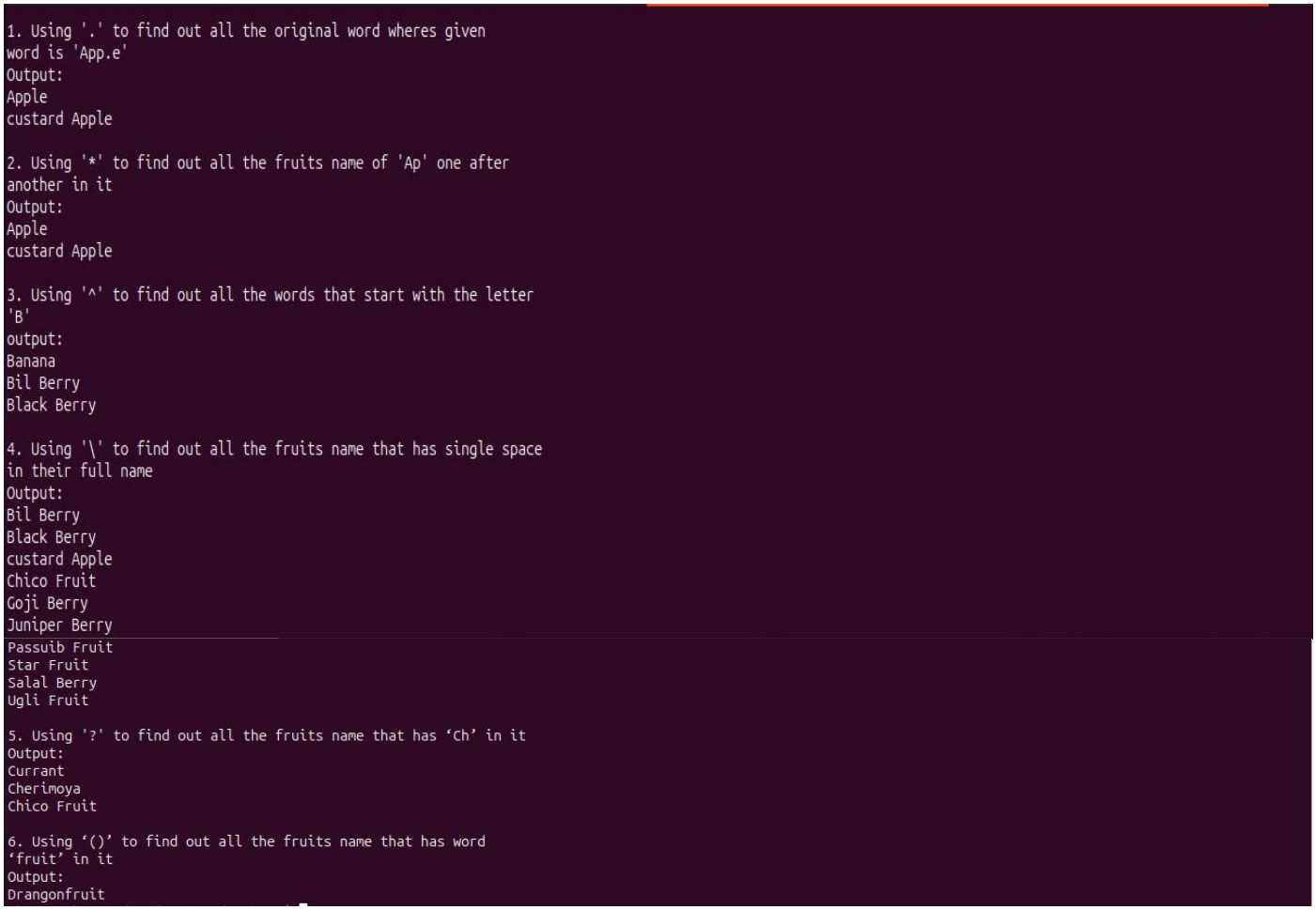
## SOURCE CODE

```
fruits_file=$(cat fruit.txt | grep App.e)
echo "\n1. Using '.' to find out all the original word wheres given word is 'App.e'"
echo "Output:\n$fruits_file"
fruits_file=$(cat fruit.txt | grep Ap*le)
echo "\n2. Using '*' to find out all the fruits name of 'Ap' one after another in it"
echo "Output:\n$fruits_file"
fruits_file=$(cat fruit.txt | grep ^B)
echo "\n3. Using '^' to find out all the words that start with the letter 'B'"
echo "output:\n$fruits_file"
fruits_file=$(cat fruit.txt | grep "\ ")
echo "\n4. Using '\' to find out all the fruits name that has single space in their full name"
echo "Output:\n$fruits_file"
fruits_file=$(cat fruit.txt | grep -E Ch?)
echo "\n5. Using '?' to find out all the fruits name that has ‘Ch’ in it"
echo "Output:\n$fruits_file"
fruits_file=$(cat fruit.txt | grep -E "(fruit)")
echo "\n6. Using ‘()’ to find out all the fruits name that has word ‘fruit’ in it"
echo "Output:\n$fruits_file"
```

# OUTPUT

- fruit.txt file



## SOURCE CODE (sed command):

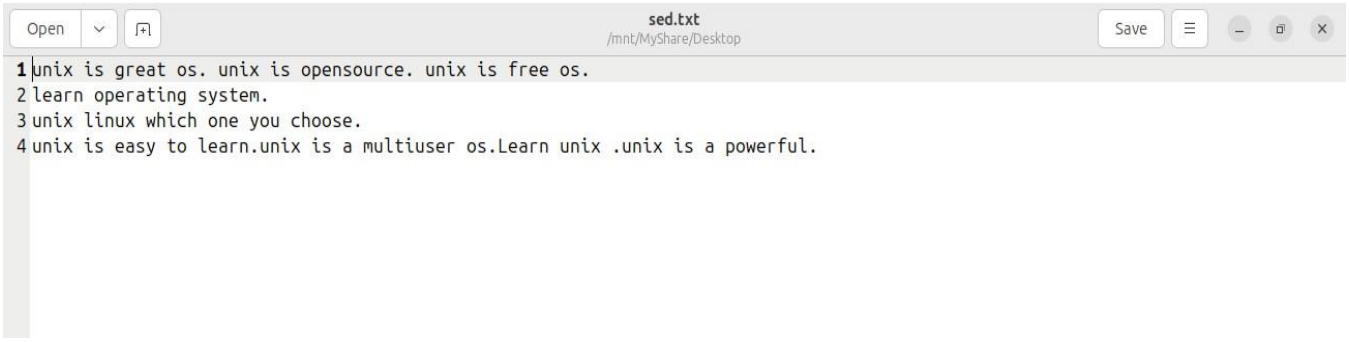
```
echo "1.Replacing or substituting string:"
echo "---------------------------------------"
sed 's/unix/linux/' sed.txt
echo
echo "2.Replacing the nth occurrence of a pattern in a line:"
echo "---------------------------------------------------------------"
sed 's/unix/linux/2' sed.txt
echo
echo "3.Replacing all the occurrence of the pattern in a line:"
echo "---------------------------------------------------------------"
sed 's/unix/linux/g' sed.txt
echo
echo "4.Replacing from nth occurrence to all occurrences in a line:"
echo "---------------------------------------------------------------------------"
sed 's/unix/linux/3g' sed.txt
echo
echo "5.Replacing string on a specific line number:"
echo "-----------------------------------------------------"
sed '3 s/unix/linux/' sed.txt
echo
echo "6.Duplicating the replaced line with /p flag:"
echo "-----------------------------------------------------"
sed 's/unix/linux/p' sed.txt
echo
```

```
echo "7.Printing only the replaced lines:"
echo "-------------------------------------"
sed -n 's/unix/linux/p' sed.txt
echo
echo "8.Replacing string on a range of lines:"
echo "-------------------------------------------"
sed '2,$ s/unix/linux/' sed.txt
echo
echo "9.Deleting lines from a particular file:"
echo "-------------------------------------------"
sed '2,4d' sed.txt
```

## SOURCE CODE (gawk command):

```
gawk 'BEGIN { print "Enter the mark:"
getline mark < "-"
if (mark >= 90) print "A+"
else if( mark >= 80) print "A"
else if( mark >= 70) print "B+"
else if( mark >= 60) print "B"
else if( mark >= 50) print "C+"
else print "Fail" }'
```

# OUTPUT (sed command):




```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh p7a.sh
1.Replacing or substituting string:
---------------------------------------
linux is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

2.Replacing the nth occurrence of a pattern in a line:
---------------------------------------------------------------
unix is great os. linux is opensource. unix is free os.
learn operating system.
unix linux which one you choose.
unix is easy to learn.linux is a multiuser os.Learn unix .unix is a powerful.

3.Replacing all the occurrence of the pattern in a line:
---------------------------------------------------------------
linux is great os. linux is opensource. linux is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn.linux is a multiuser os.Learn linux .linux is a powerful.

4.Replacing from nth occurrence to all occurrences in a line:
---------------------------------------------------------------------------
unix is great os. unix is opensource. linux is free os.
learn operating system.
unix linux which one you choose.
unix is easy to learn.unix is a multiuser os.Learn linux .linux is a powerful.

5.Replacing string on a specific line number:
-----------------------------------------------------
unix is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

6.Duplicating the replaced line with /p flag:
-----------------------------------------------------
linux is great os. unix is opensource. unix is free os.
linux is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

7.Printing only the replaced lines:
---------------------------------------
linux is great os. unix is opensource. unix is free os.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

8.Replacing string on a range of lines:
-----------------------------------------------
unix is great os. unix is opensource. unix is free os.
learn operating system.
linux linux which one you choose.
linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

9.Deleting lines from a particular file:
-----------------------------------------------
unix is great os. unix is opensource. unix is free os.
```

## OUTPUT (gawk command):

- Install gawk package

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sudo apt install gawk
```

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh program7b.sh
Enter the mark:
95
A+
```

## SOURCE CODE

```
DATE=$(date +%y%m%d)
read -p "Give name to the archive file:" file
FILE=$file$DATE.tgz
read -p "Enter the Filename: " SOURCE
read -p "Enter the Destination path: " des
DESTINATION=$des/$FILE
if [ -f $SOURCE ]
then
echo
else
echo "$SOURCE doesn't exist, BACKUP INCOMPLETE"
exit
fi
FILE_NO=1
exec < $SOURCE
read FILE_NAME
while [ $? -eq 0 ]
do
if [ -f $FILE_NAME ] || [ -d $FILE_NAME ]
then
FILE_LIST="$FILE_LIST $FILE_NAME"
else
echo "$FILE_NAME doesn’t exit, thus it is not included"
echo "BACKUP is still on process"
echo
```

```
fi
FILE_NO=$[$FILE_NO+1]
read FILE_NAME
done
echo "Starting Archive..."
tar -czf $DESTINATION $FILE_LIST 2>/dev/null
echo "Archive COMPLETED at $DESTINATION"
exit
```

# OUTPUT

- Creating the file to Store the backup file:

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ gedit backup.txt
```

- Backup Files:


- Running the Script to Backup the Files:

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ sh program8.sh
Give name to the archive file:NewBackup
Enter the Filename: backup.txt
Enter the Destination path: /home/ubuntu/Desktop

Starting Archive...
Archive COMPLETED at /home/ubuntu/Desktop/NewBackup241009.tgz
```

- Archive File(NewBackup241009.tgz):


- Listing of the Archive contents from a Terminal Prompt Type:

```
ubuntu@ubuntu:/mnt/MyShare/Desktop$ tar -tzf /home/ubuntu/Desktop/NewBackup241009.tgz
home/ubuntu/Documents/docfile
home/ubuntu/Downloads/downfile.txt
home/ubuntu/Music/musicfile.txt
ubuntu@ubuntu:/mnt/MyShare/Desktop$ tar -tzvf /home/ubuntu/Desktop/NewBackup241009.tgz
-rwxrwxrwx ubuntu/ubuntu    60 2024-10-09 21:31 home/ubuntu/Documents/docfile
-rwxrwxrwx ubuntu/ubuntu    60 2024-10-09 21:31 home/ubuntu/Downloads/downfile.txt
-rwxrwxrwx ubuntu/ubuntu    60 2024-10-09 21:31 home/ubuntu/Music/musicfile.txt
```

## SOURCE CODE

### A) Creating Text Menus

```
diskspace() {
clear
df -k
}
diskspace
whoseon() {
clear
who
}
whoseon
memusage() {
clear
cat /proc/meminfo
}
memusage
menu(){
clear
echo
echo "\t\t\t\tSys Admin Menu\n"
echo "\t\t1. Display disk space"
echo "\t\t2. Display logged on users"
echo "\t\t3. Display memory usage"
echo "\t\t0. Exit program\n\n"
echo
```

```
echo "\t\t\tEnter option: "
read option
echo
}
menu
while [ True ]
do
menu
case $option in
0)
break ;;
1)
diskspace ;;
2)
whoseon ;;
3)
memusage ;;
*)
clear
echo "Sorry, wrong selection";;
esac
echo "\n\n\t\t\tHit any key to continue"
read line
done
clear
```

**B)Text Window Widgets**

```
temp=$(mktemp -t test.XXXXXX)
temp2=$(mktemp -t test2.XXXXXX)
diskspace(){
clear
df -k> $temp
dialog --textbox $temp 20 50
}
whoseon() {
clear
who> $temp
dialog --textbox $temp 20 50
}
memusage(){
clear
cat /proc/meminfo> $temp
dialog --textbox $temp 20 50
}
while [ 1 ]
do
clear
dialog --menu "Sys Admin Menu" 20 30 10 1 "Display diskspace" 2
"Display users" 3 "Display memory usage" 2> $temp2
if [ $? -eq 1 ]
then
break
fi
```

```
selection=$(cat $temp2)
case $selection in
1) diskspace ;;
2) whoseon ;;
3) memusage ;;
*) dialog --msgbox "Sorry, invalid selection" 10 30
esac
done
clear
rm -f $temp 2> /dev/null
rm -f $temp2 2> /dev/null
```

# OUTPUT

## A) Text Menus

```
                        Sys Admin Menu

            1. Display disk space
            2. Display logged on users
            3. Display memory usage
            0. Exit program



                    Enter option:
```

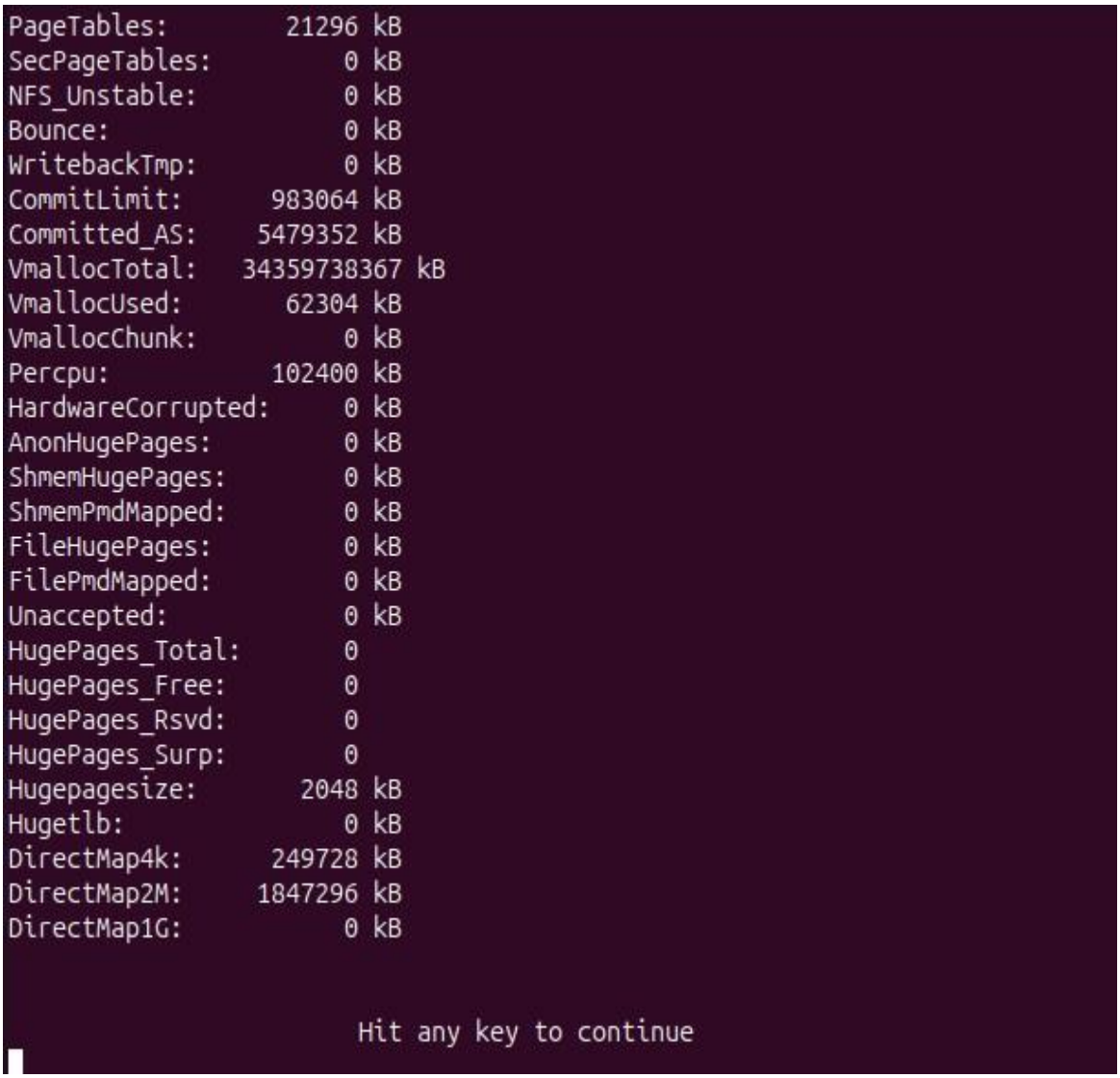
1. Display Disk Space

```
Filesystem     1K-blocks      Used Available Use% Mounted on
tmpfs             196616      2092    194524   2% /run
/dev/sr1         6057964   6057964         0 100% /cdrom
/cow              983064    271112    711952  28% /
tmpfs             983064         8    983056   1% /dev/shm
tmpfs               5120         8      5112   1% /run/lock
tmpfs             983064         4    983060   1% /tmp
tmpfs             196612       160    196452   1% /run/user/1000
/dev/sr0           90084     90084         0 100% /media/ubuntu/CDROM
vmhgfs-fuse    487730416 129648776 358081640  27% /mnt/hgfs
vmhgfs-fuse    487730416 129648776 358081640  27% /mnt/MyShare


                  Hit any key to continue
```

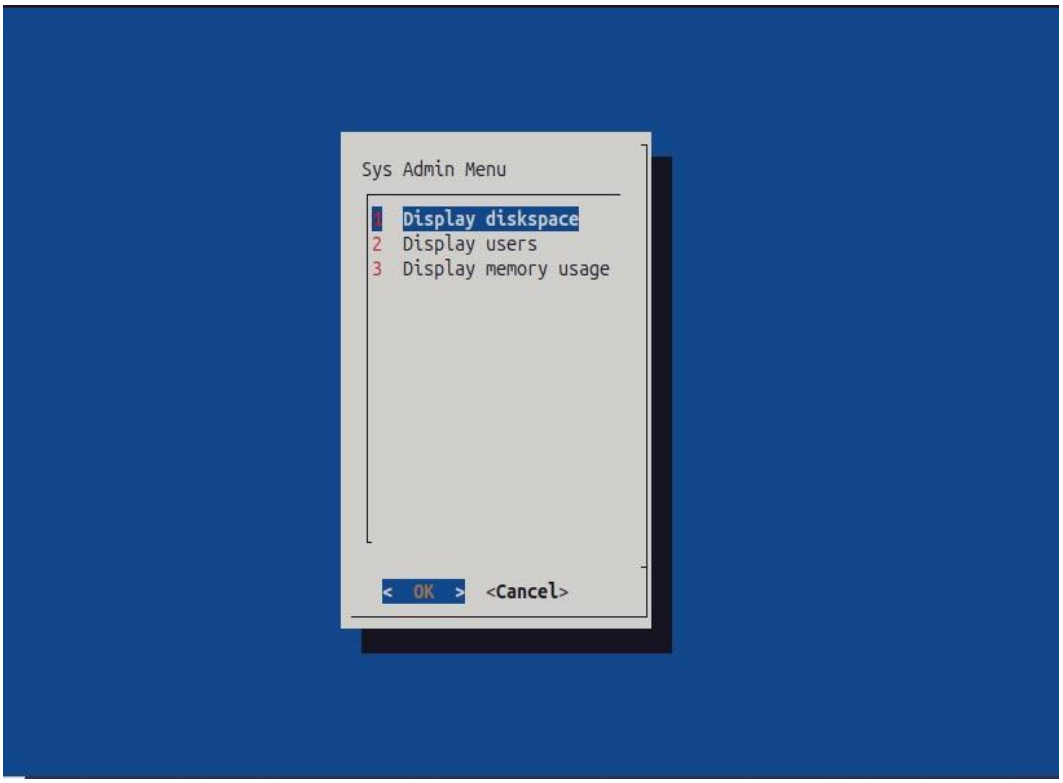
2. Displaying Logged Users

```
ubuntu   seat0        2024-10-09 16:27 (login screen)
ubuntu   :0           2024-10-09 16:28 (:0)


                    Hit any key to continue
```

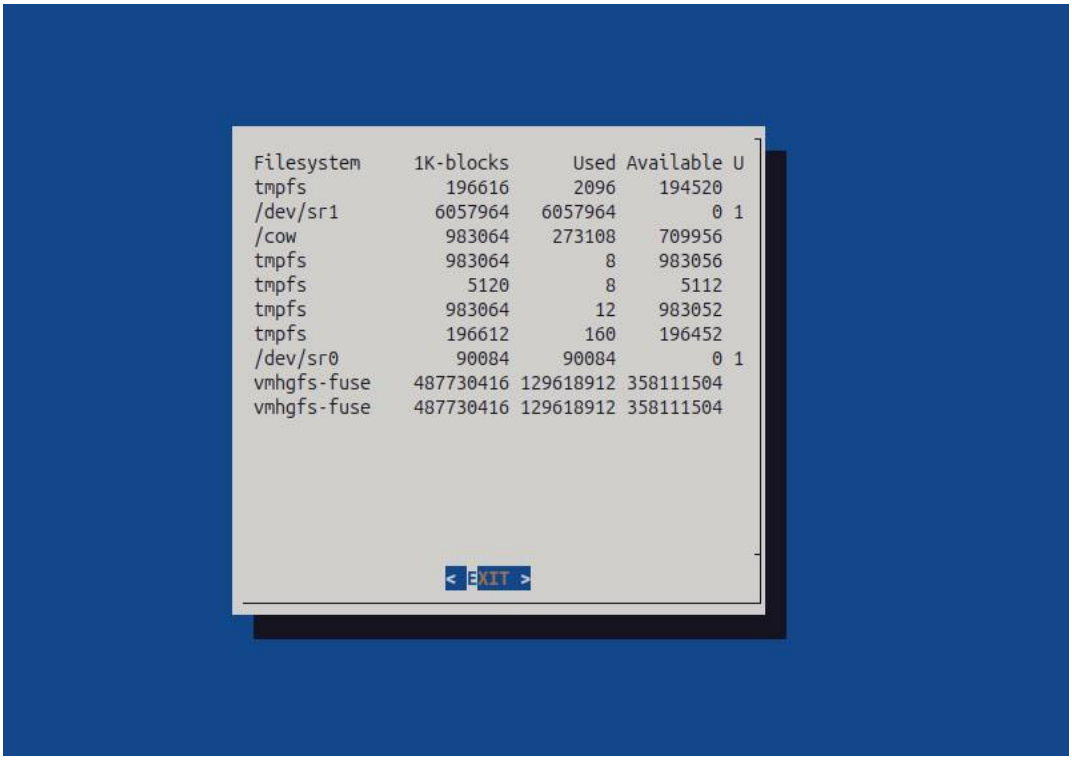
3. Displaying Memory Usage



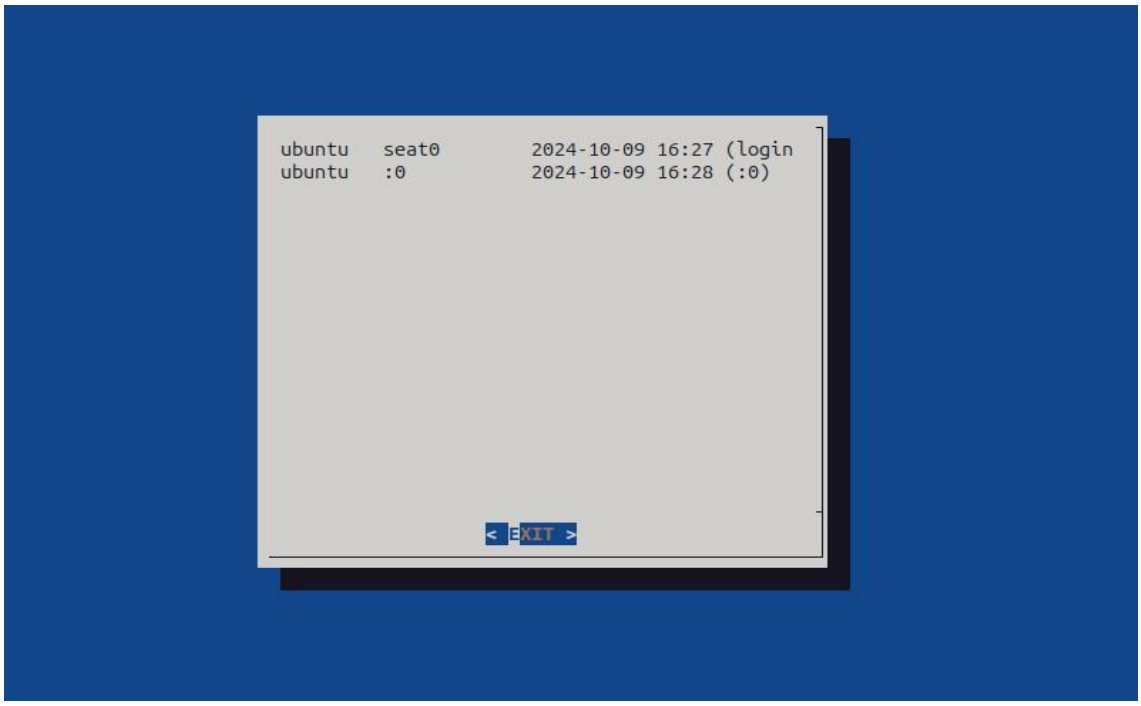
# B)Text Window Widgets



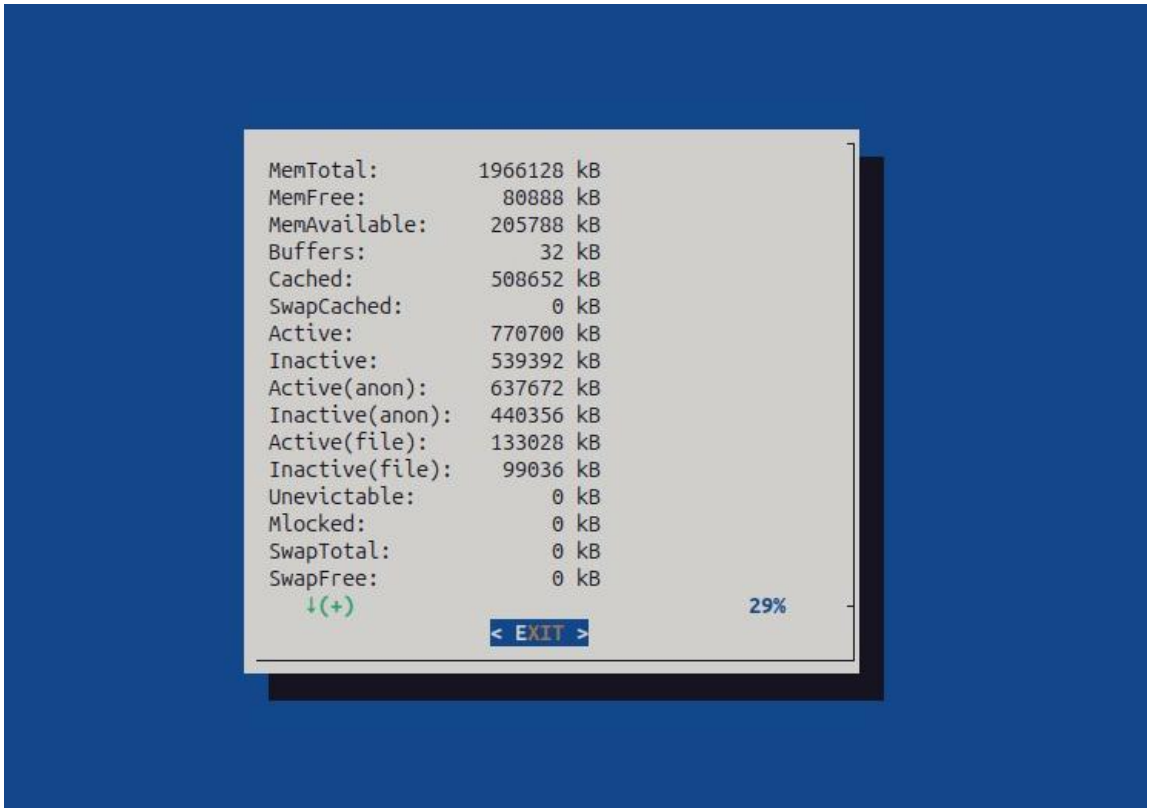

1. Display Disk Space



2. Displaying Logged Users




3. Displaying Memory Usage



# SOURCE CODE AND OUTPUTS

- To view the list of databases by using \l command:

```
ubuntu@ubuntu:~$ sudo -i -u postgres
postgres@ubuntu:~$ psql
psql (16.4 (Ubuntu 16.4-0ubuntu0.24.04.2))
Type "help" for help.

postgres=# \l
postgres=# \l
                                                List of databases
   Name    |  Owner   | Encoding | Locale Provider | Collate |  Ctype  | ICU Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+---------+---------+------------+-----------+-----------------------
 postgres  | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           |
 template0 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres          +
           |          |          |                 |         |         |            |           | postgres=CTc/postgres
(3 rows)
```

- Creating Databases:

```
postgres=# CREATE DATABASE bank_details;
CREATE DATABASE
```

- Listing the Database and Checking Database which Created by User:

```
                                                List of databases
    Name     |  Owner   | Encoding | Locale Provider | Collate |  Ctype  | ICU Locale | ICU Rules |   Access privileges

-------------+----------+----------+-----------------+---------+---------+------------+-----------+-----------------------
---
 bank_details | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           |
 postgres     | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           |
 template0    | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres
  +
              |          |          |                 |         |         |            |           | postgres=CTc/postgr
es
 template1    | postgres | UTF8     | libc            | C.UTF-8 | C.UTF-8 |            |           | =c/postgres
  +
              |          |          |                 |         |         |            |           | postgres=CTc/postgr
es
(4 rows)
```

- Changing Path to the Created Database(bank_details):

```
postgres-# \c bank_details;
You are now connected to database "bank_details" as user "postgres".
bank_details-#
```

- Creating Table:

```
bank_details=# CREATE TABLE BANKDETAILS(acc_no integer,name text,balance numeric,acc_type
text);
CREATE TABLE
```

- Inserting Values to the Table:

```
bank_details=# INSERT INTO BANKDETAILS VALUES(30201,'NAVEEN',1000.00,'Savings');
INSERT 0 1
bank_details=# SELECT *FROM BANKDETAILS;
 acc_no |  name  | balance | acc_type
--------+--------+---------+----------
  30201 | NAVEEN | 1000.00 | Savings
(1 row)
```

- Inserting Multiple Values to the Table:

```
bank_details=# INSERT INTO BANKDETAILS VALUES (30202,'Charan',500.00,'Savings'),(30203,'Si
va',6000.00,'Current'),(30204,'Sanjay',500.00,'Current'),(30205,'Tamil',4000.00,'Savings')
;
INSERT 0 4
bank_details=# SELECT * FROM BANKDETAILS;
 acc_no |  name  | balance | acc_type
--------+--------+---------+----------
  30201 | NAVEEN | 1000.00 | Savings
  30202 | Charan |  500.00 | Savings
  30203 | Siva   | 6000.00 | Current
  30204 | Sanjay |  500.00 | Current
  30205 | Tamil  | 4000.00 | Savings
(5 rows)
```

- Updating the Column in Table:

```
bank_details=# UPDATE BANKDETAILS SET balance=3000.00 WHERE balance=500.00;
UPDATE 2
bank_details=# SELECT * FROM BANKDETAILS;
 acc_no |  name  | balance | acc_type
--------+--------+---------+----------
  30201 | NAVEEN | 1000.00 | Savings
  30203 | Siva   | 6000.00 | Current
  30205 | Tamil  | 4000.00 | Savings
  30202 | Charan | 3000.00 | Savings
  30204 | Sanjay | 3000.00 | Current
(5 rows)
```

- Deleting the Column in Table:

```
bank_details=# DELETE FROM BANKDETAILS WHERE acc_no=30202;
DELETE 1
bank_details=# SELECT * FROM BANKDETAILS;
 acc_no |  name  | balance | acc_type
--------+--------+---------+----------
  30201 | NAVEEN | 1000.00 | Savings
  30203 | Siva   | 6000.00 | Current
  30205 | Tamil  | 4000.00 | Savings
  30204 | Sanjay | 3000.00 | Current
(4 rows)
```

- Deleting the Table:

```
bank_details=# DROP TABLE BANKDETAILS;
DROP TABLE
```

- Checking the Table if Exists or Not:

```
bank_details=# SELECT * FROM BANKDETAILS;
ERROR:  relation "bankdetails" does not exist
LINE 1: SELECT * FROM BANKDETAILS;
                      ^
```

- Deleting the Database and Listing of Databases:

```
bank_details=# \c postgres;
You are now connected to database "postgres" as user "postgres".
postgres=# DROP DATABASE bank_details;
DROP DATABASE
postgres=# \l
                                                List of databases
   Name    |  Owner   | Encoding | Locale Provider |  Collate   |   Ctype    | ICU Locale | ICU Rules |   Access privileges
-----------+----------+----------+-----------------+------------+------------+------------+-----------+-----------------------
 postgres  | postgres | UTF8     | libc            | C.UTF-8    | C.UTF-8    |            |           |
 template0 | postgres | UTF8     | libc            | C.UTF-8    | C.UTF-8    |            |           | =c/postgres          +
           |          |          |                 |            |            |            |           | postgres=CTc/postgres
 template1 | postgres | UTF8     | libc            | C.UTF-8    | C.UTF-8    |            |           | =c/postgres          +
           |          |          |                 |            |            |            |           | postgres=CTc/postgres
(3 rows)
```

- Quit from Database:

```
postgres=# \q
```

- Logout from psql:

```
postgres@ubuntu:~$ logout
ubuntu@ubuntu:~$
```