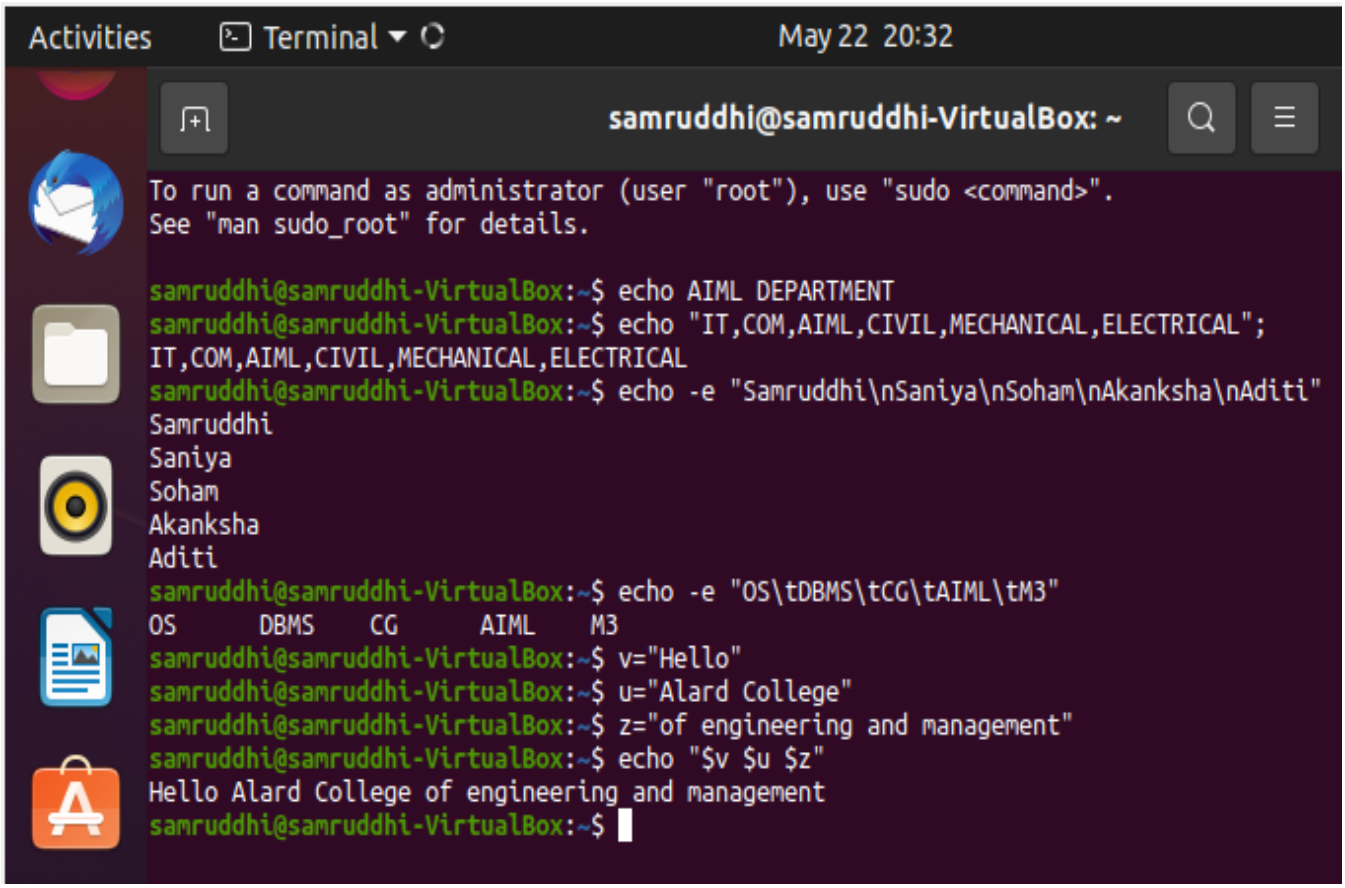
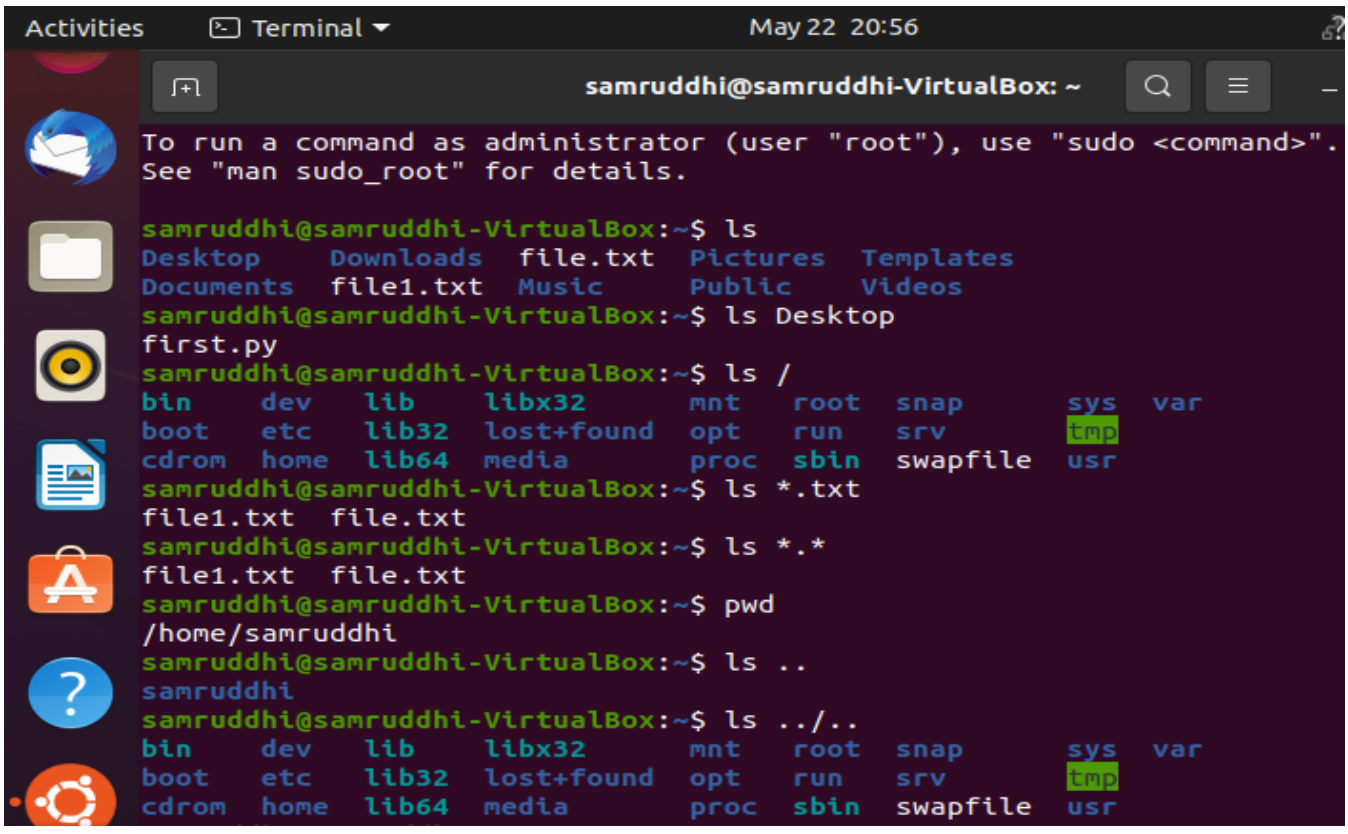
PRACTICAL NO. 1

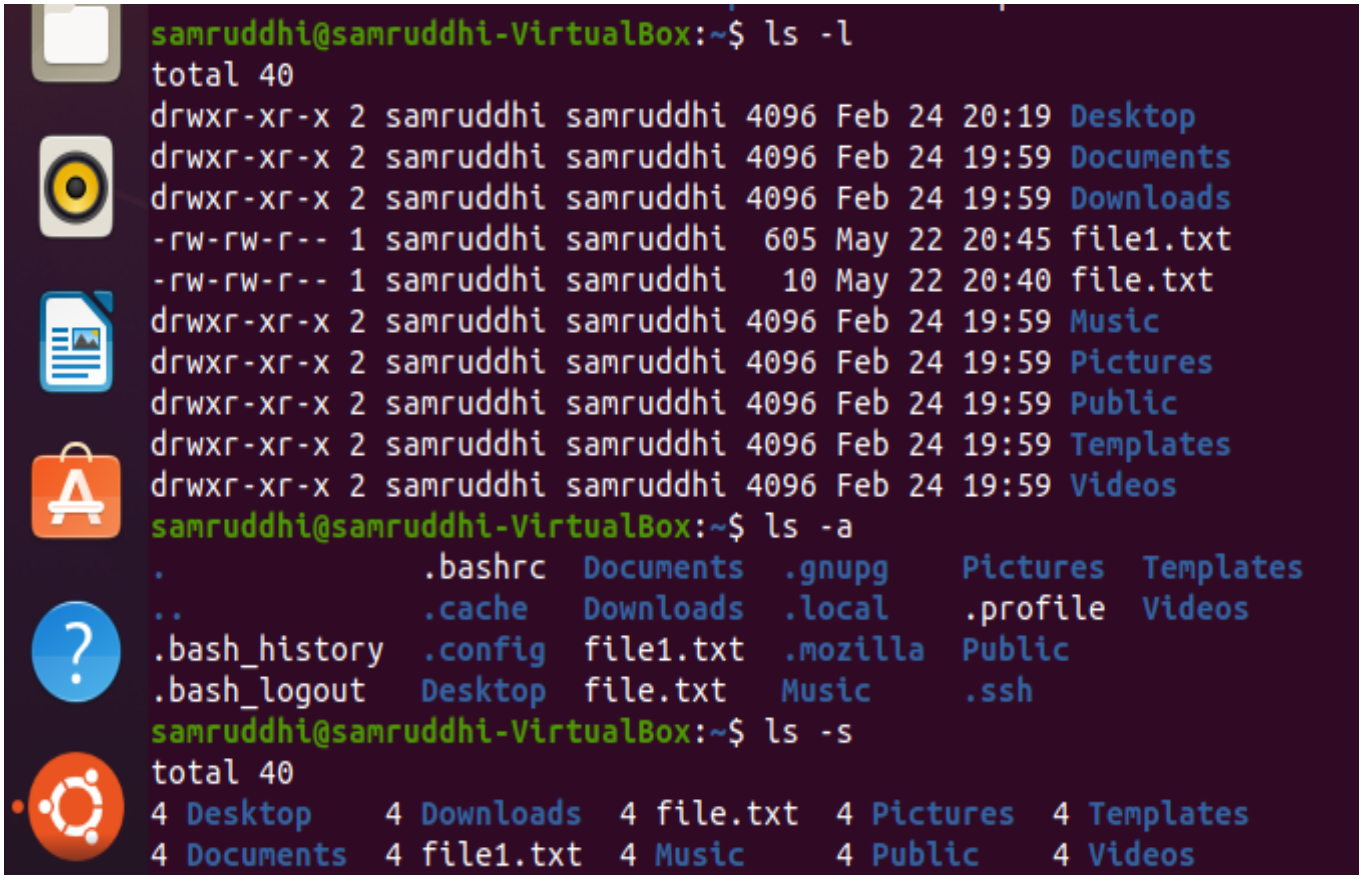
a. Study of Basic Linux Commands: echo, ls, read, cat, touch, test, loops, arithmetic comparison, conditional loops, grep, sed etc.

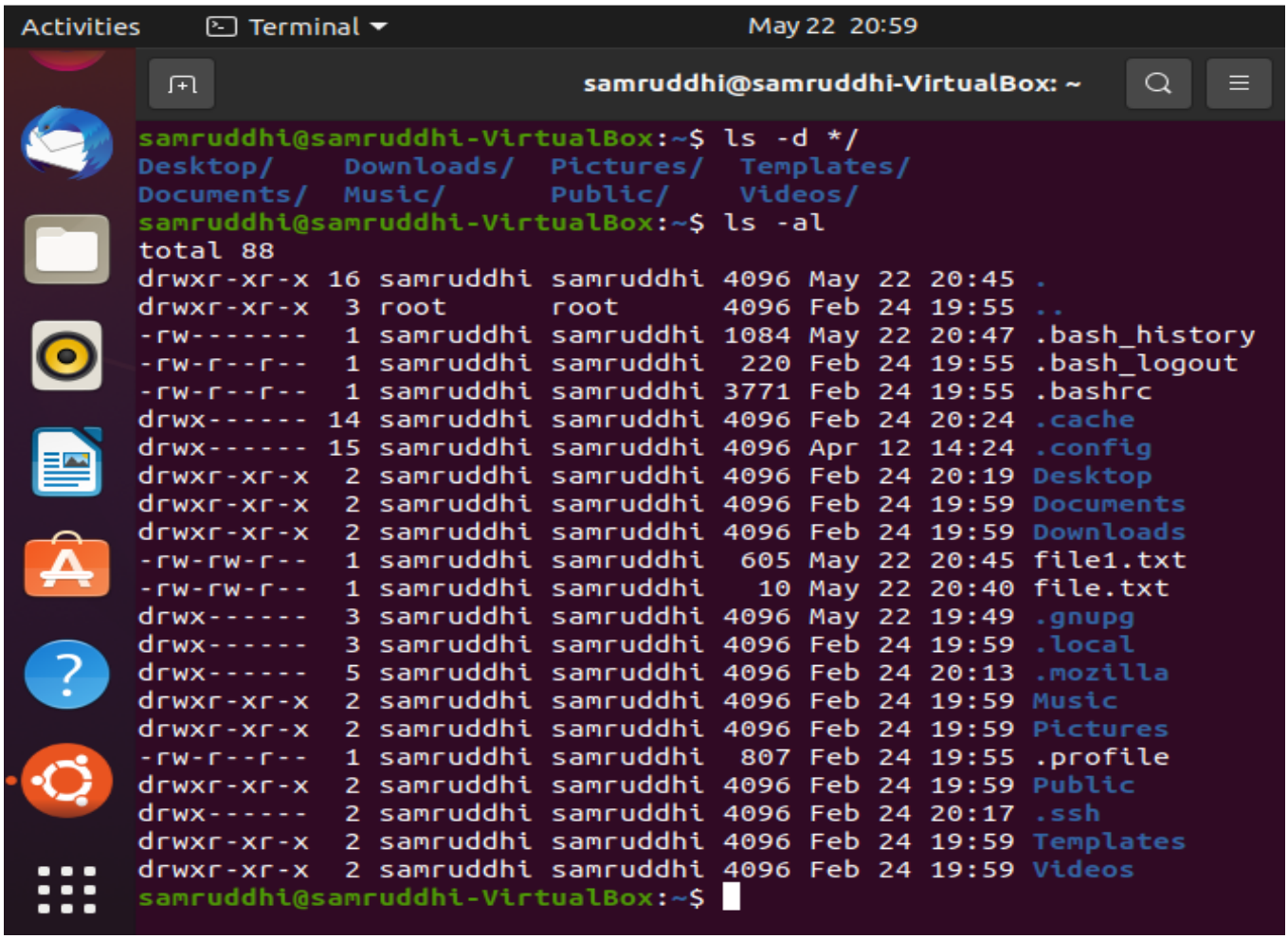
echo command:

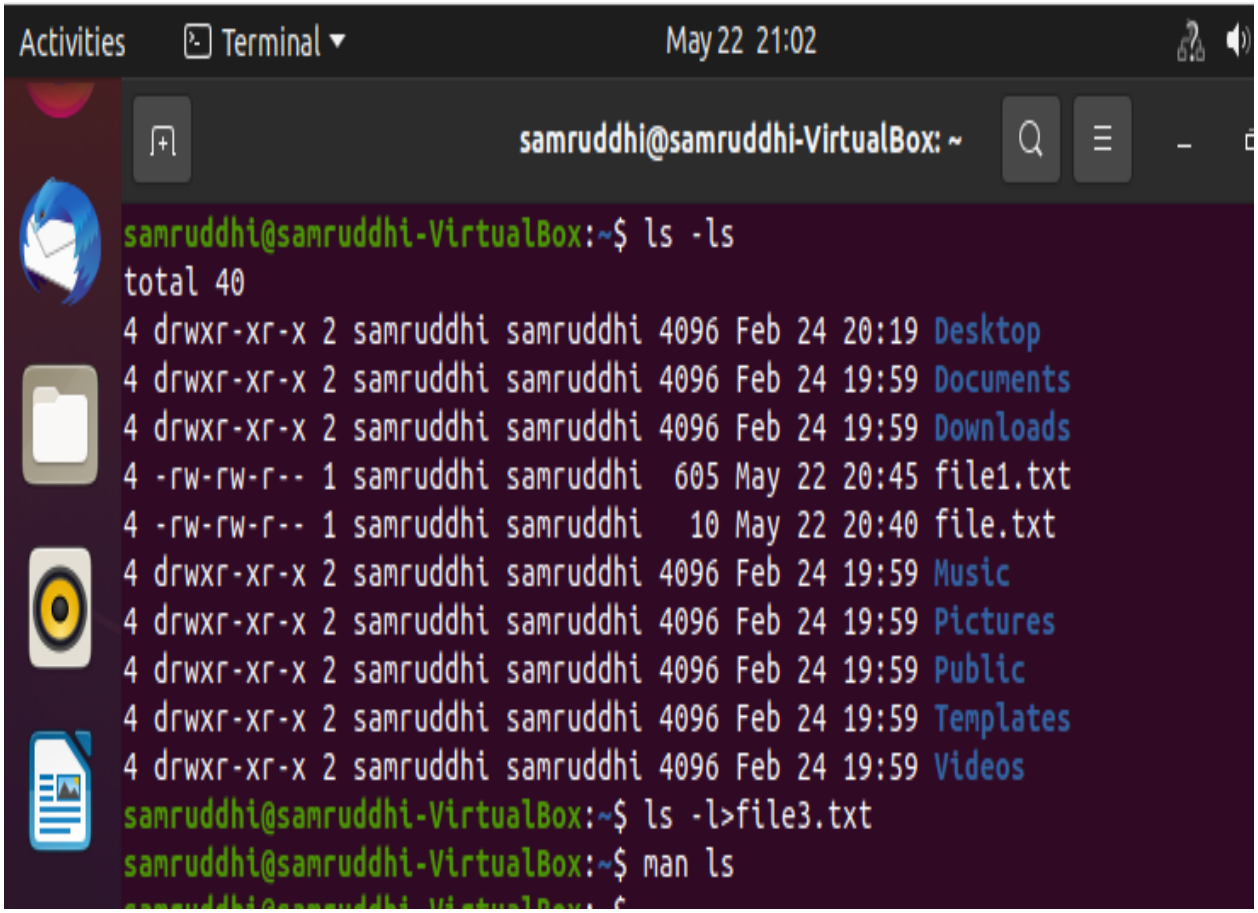


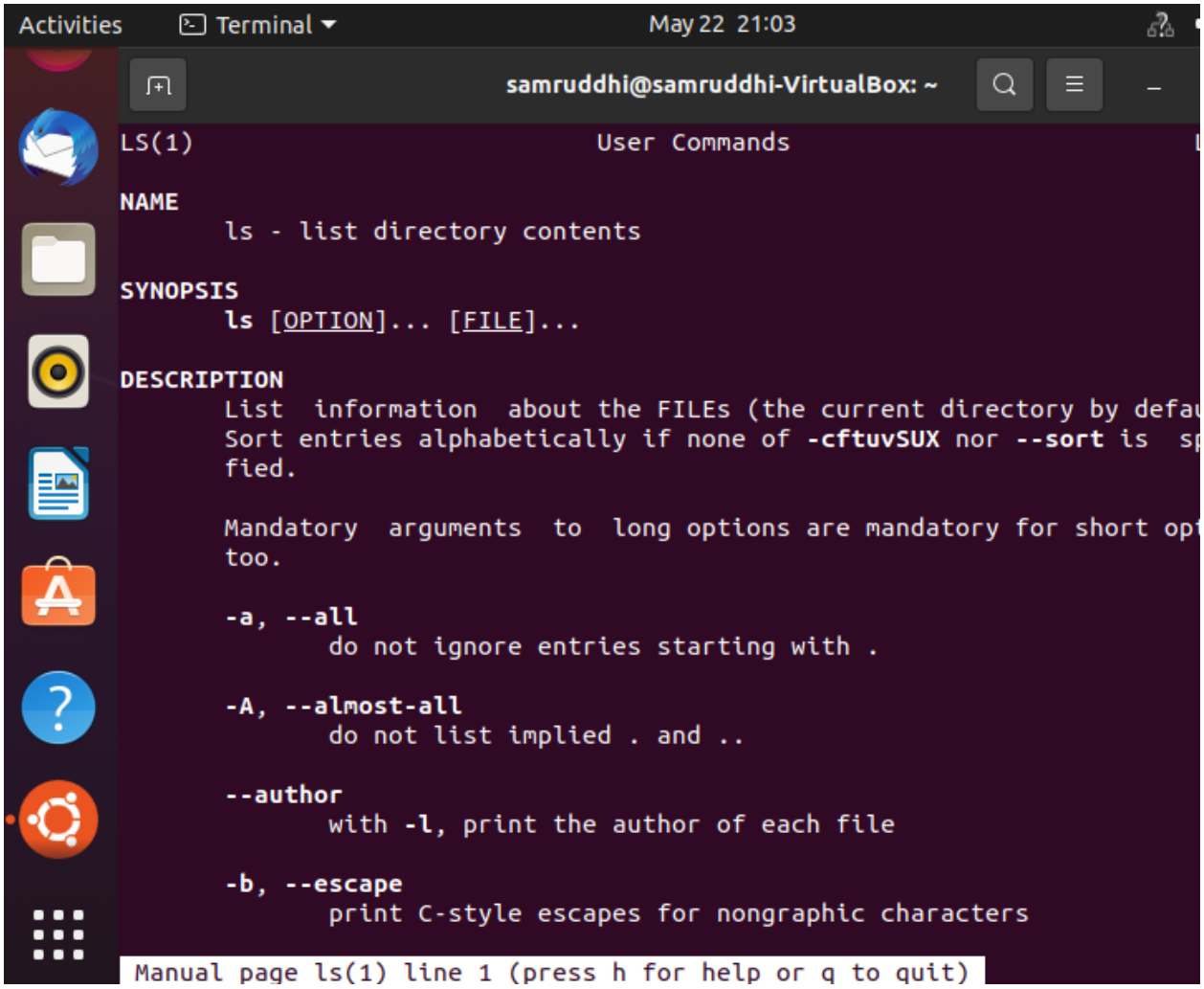
ls command:



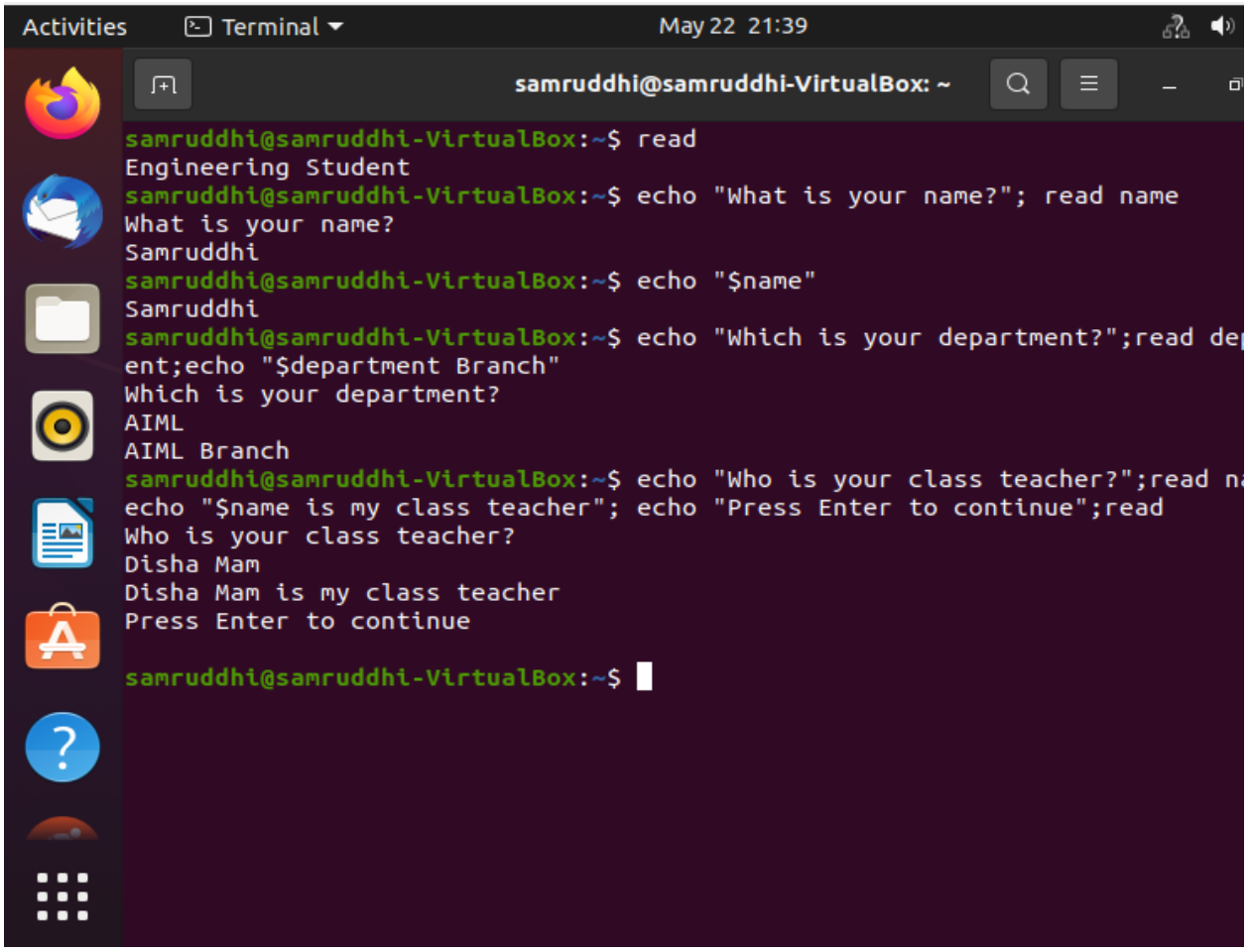

samruddhi@samruddhi-VirtualBox:~$ ls -l
total 40
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 20:19 Desktop
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Documents
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Downloads
-rw-rw-r-- 1 samruddhi samruddhi 605 May 22 20:45 file1.txt
-rw-rw-r-- 1 samruddhi samruddhi 10 May 22 20:40 file.txt
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Music
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Pictures
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Public
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Templates
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Videos
samruddhi@samruddhi-VirtualBox:~$ ls -a
. .bashrc Documents .gnupg Pictures Templates
.. .cache Downloads .local .profile Videos
.bash_history .config file1.txt .mozilla Public
.bash_logout Desktop file.txt Music .ssh
samruddhi@samruddhi-VirtualBox:~$ ls -s
total 40
4 Desktop 4 Downloads 4 file.txt 4 Pictures 4 Templates
4 Documents 4 file1.txt 4 Music 4 Public 4 Videos


Activities
Terminal
May 22 20:59
samruddhi@samruddhi-VirtualBox: ~
samruddhi@samruddhi-VirtualBox:~$ ls -d */
Desktop/ Downloads/ Pictures/ Templates/
Documents/ Music/ Public/ Videos/
samruddhi@samruddhi-VirtualBox:~$ ls -al
total 88
drwxr-xr-x 16 samruddhi samruddhi 4096 May 22 20:45 .
drwxr-xr-x 3 root root 4096 Feb 24 19:55 ..
-rw------- 1 samruddhi samruddhi 1084 May 22 20:47 .bash_history
-rw-r--r-- 1 samruddhi samruddhi 220 Feb 24 19:55 .bash_logout
-rw-r--r-- 1 samruddhi samruddhi 3771 Feb 24 19:55 .bashrc
drwx------ 14 samruddhi samruddhi 4096 Feb 24 20:24 .cache
drwx------ 15 samruddhi samruddhi 4096 Apr 12 14:24 .config
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 20:19 Desktop
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Documents
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Downloads
-rw-rw-r-- 1 samruddhi samruddhi 605 May 22 20:45 file1.txt
-rw-rw-r-- 1 samruddhi samruddhi 10 May 22 20:40 file.txt
drwx------ 3 samruddhi samruddhi 4096 May 22 19:49 .gnupg
drwx------ 3 samruddhi samruddhi 4096 Feb 24 19:59 .local
drwx------ 5 samruddhi samruddhi 4096 Feb 24 20:13 .mozilla
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Music
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Pictures
-rw-r--r-- 1 samruddhi samruddhi 807 Feb 24 19:55 .profile
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Public
drwx------ 2 samruddhi samruddhi 4096 Feb 24 20:17 .ssh
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Templates
drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Videos
samruddhi@samruddhi-VirtualBox:~$

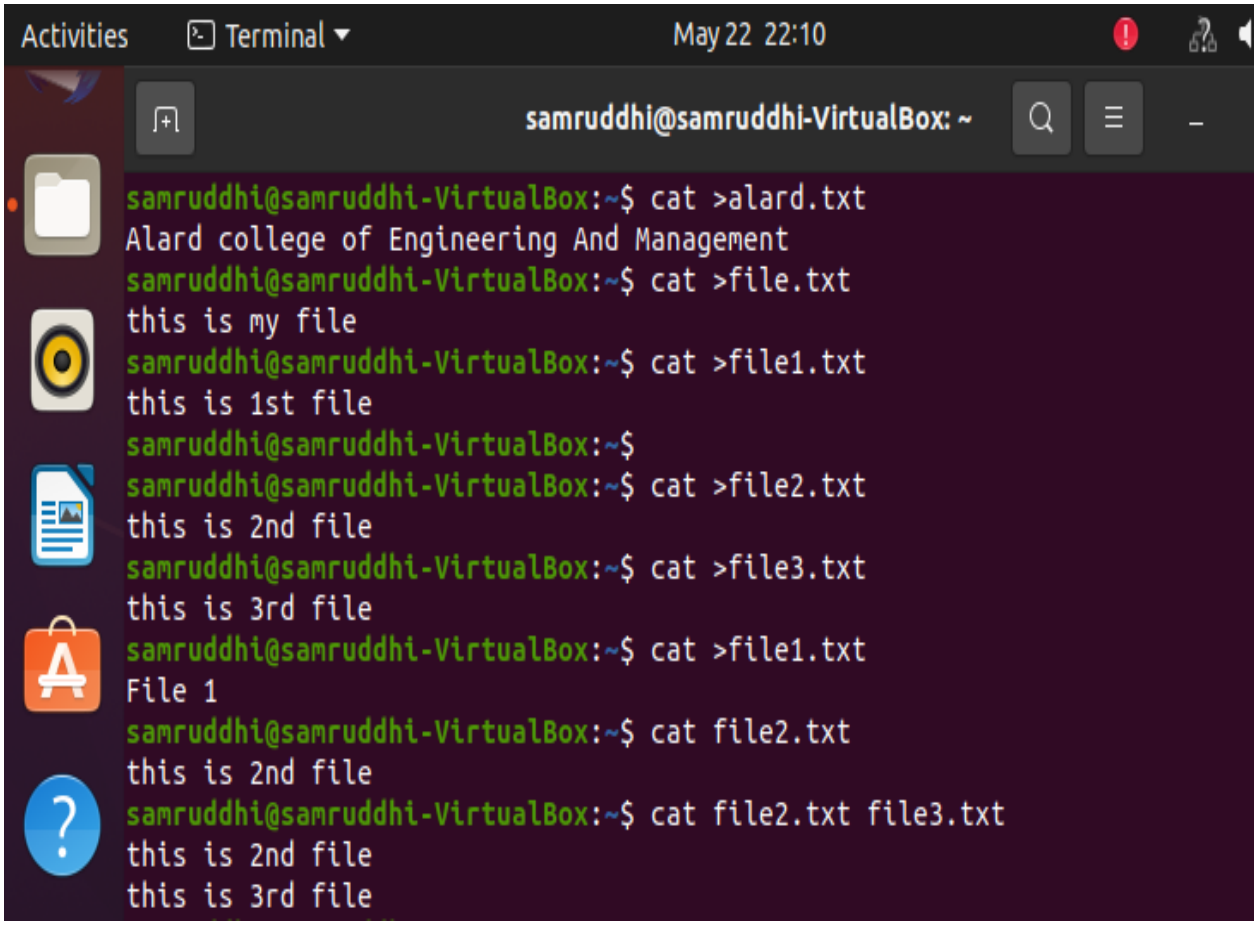

Activities
Terminal
May 22 21:02
samruddhi@samruddhi-VirtualBox: ~
samruddhi@samruddhi-VirtualBox:~$ ls -ls
total 40
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 20:19 Desktop
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Documents
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Downloads
4 -rw-rw-r-- 1 samruddhi samruddhi 605 May 22 20:45 file1.txt
4 -rw-rw-r-- 1 samruddhi samruddhi 10 May 22 20:40 file.txt
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Music
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Pictures
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Public
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Templates
4 drwxr-xr-x 2 samruddhi samruddhi 4096 Feb 24 19:59 Videos
samruddhi@samruddhi-VirtualBox:~$ ls -l>file3.txt
samruddhi@samruddhi-VirtualBox:~$ man ls

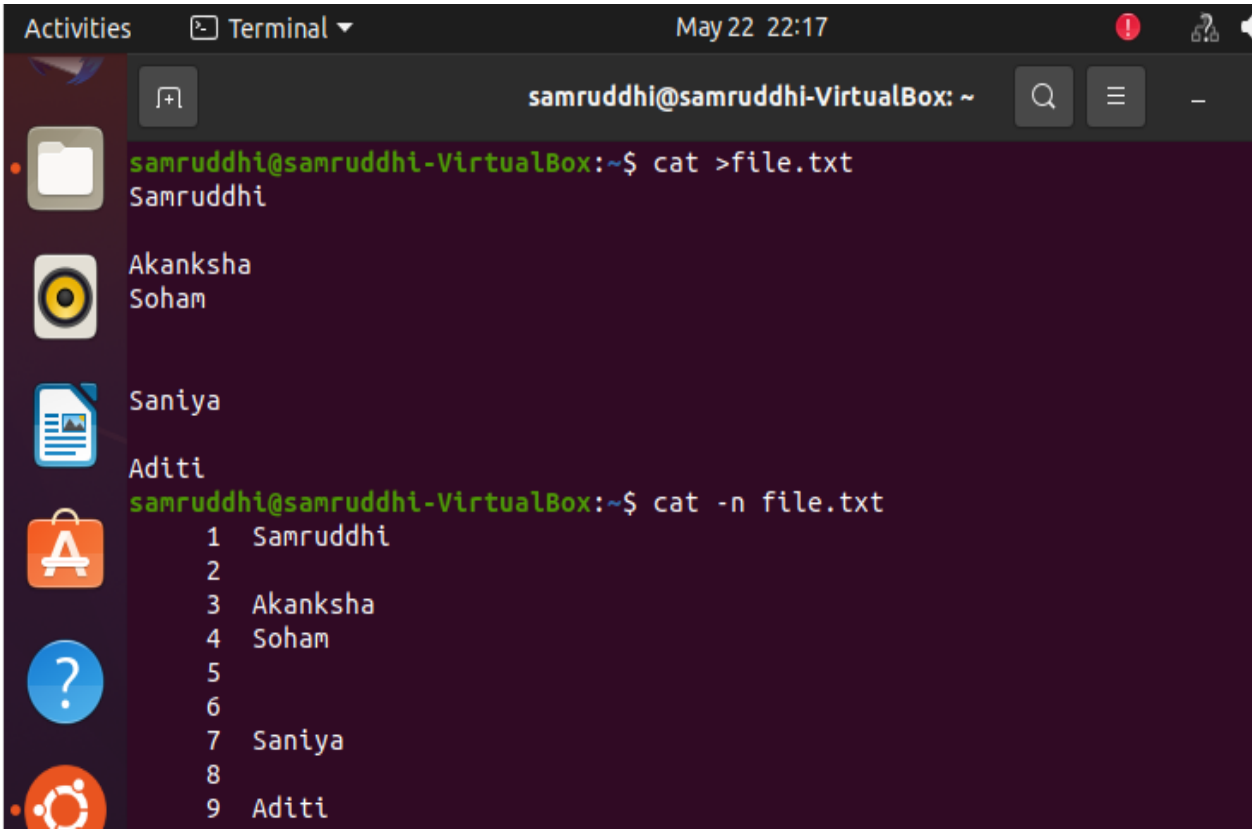

Activities
Terminal
May 22 21:03
samruddhi@samruddhi-VirtualBox: ~
LS(1) User Commands
NAME
ls - list directory contents
SYNOPSIS
ls [OPTION]... [FILE]...
DESCRIPTION
List information about the FILEs (the current directory by defau
Sort entries alphabetically if none of -cftuvSUX nor --sort is sp
fied.
Mandatory arguments to long options are mandatory for short opt
too.
-a, --all
do not ignore entries starting with .
-A, --almost-all
do not list implied . and ..
--author
with -l, print the author of each file
-b, --escape
print C-style escapes for nongraphic characters
Manual page ls(1) line 1 (press h for help or q to quit)

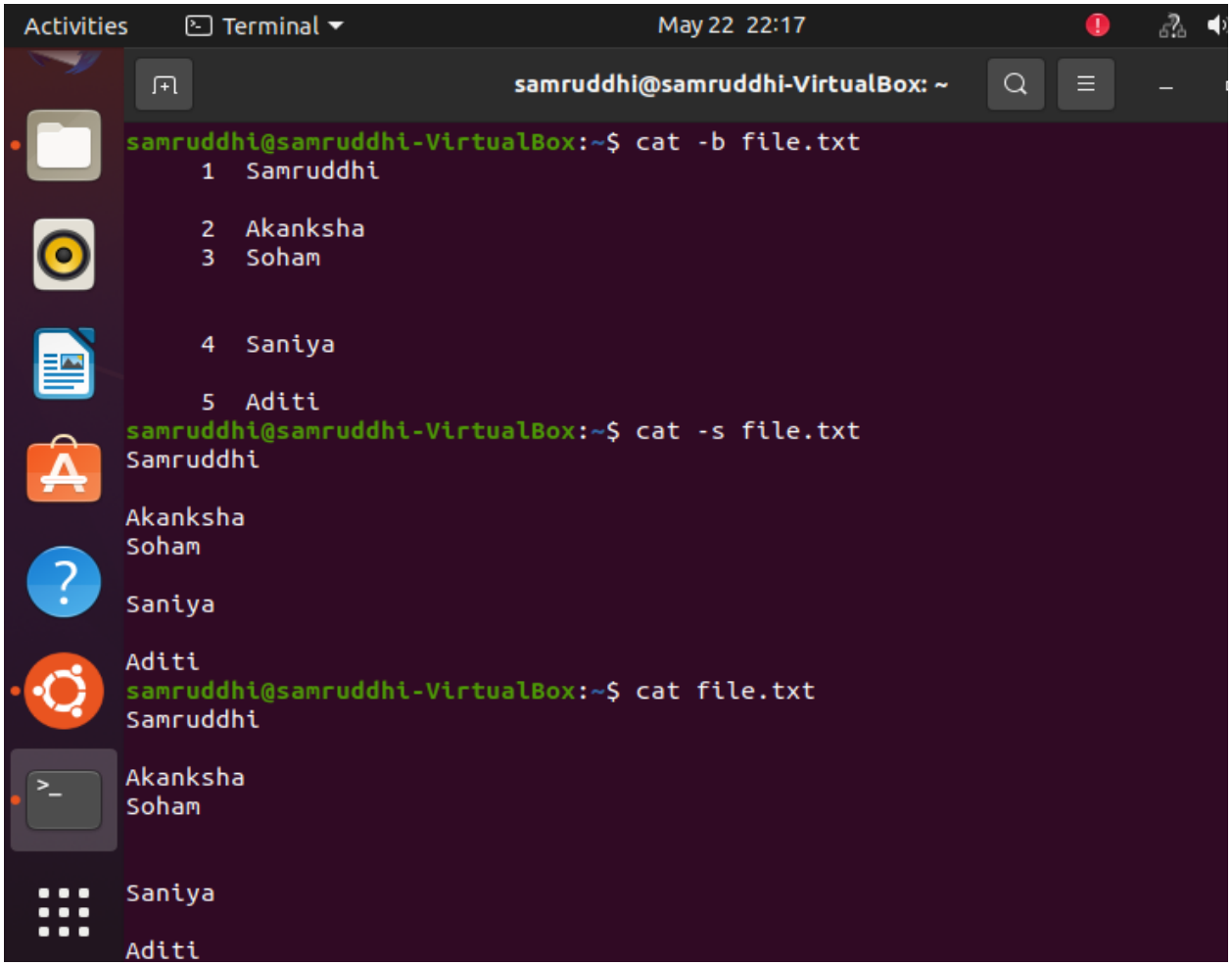
read command:

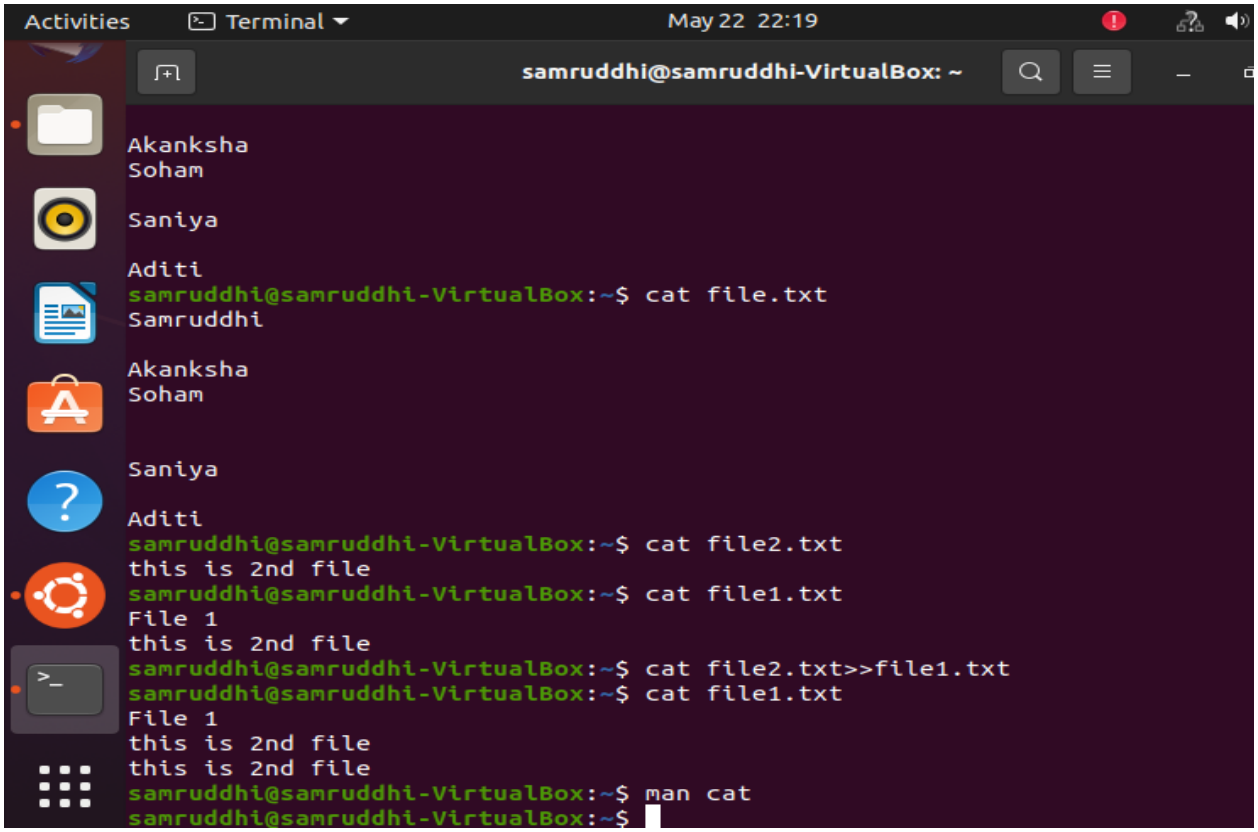


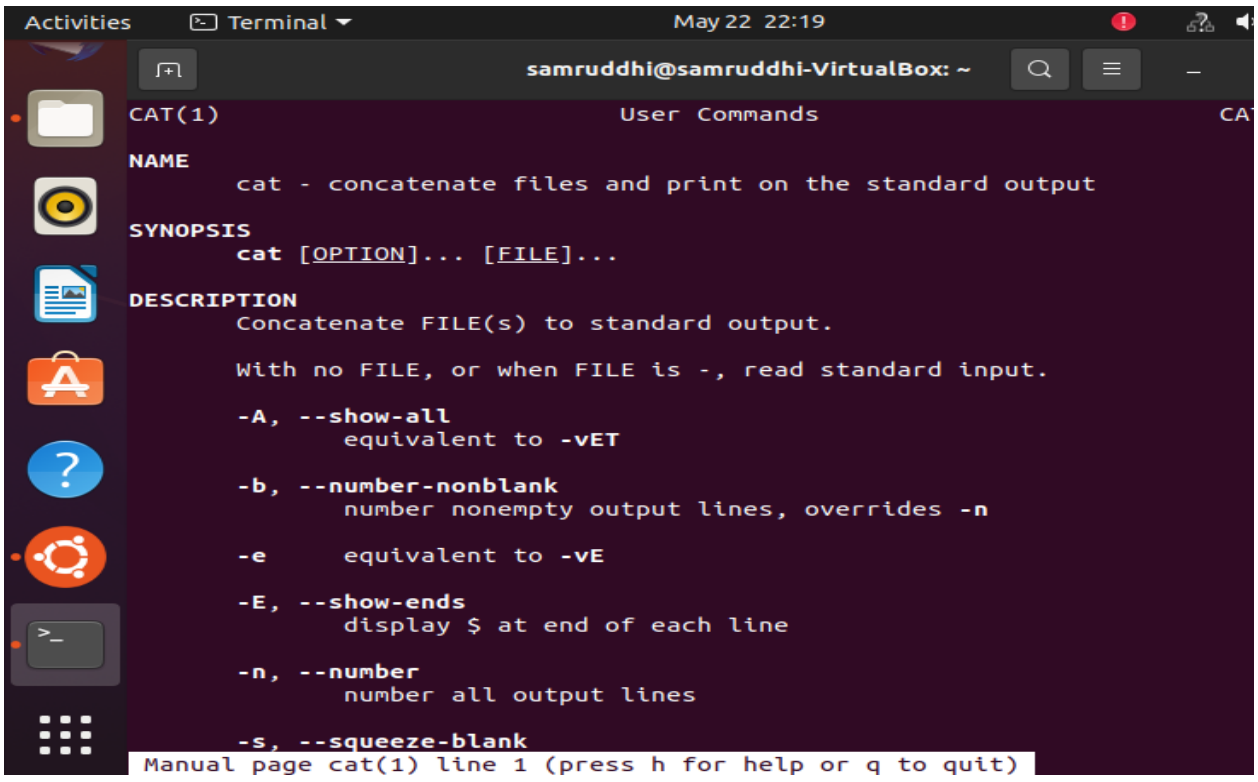
cat command:



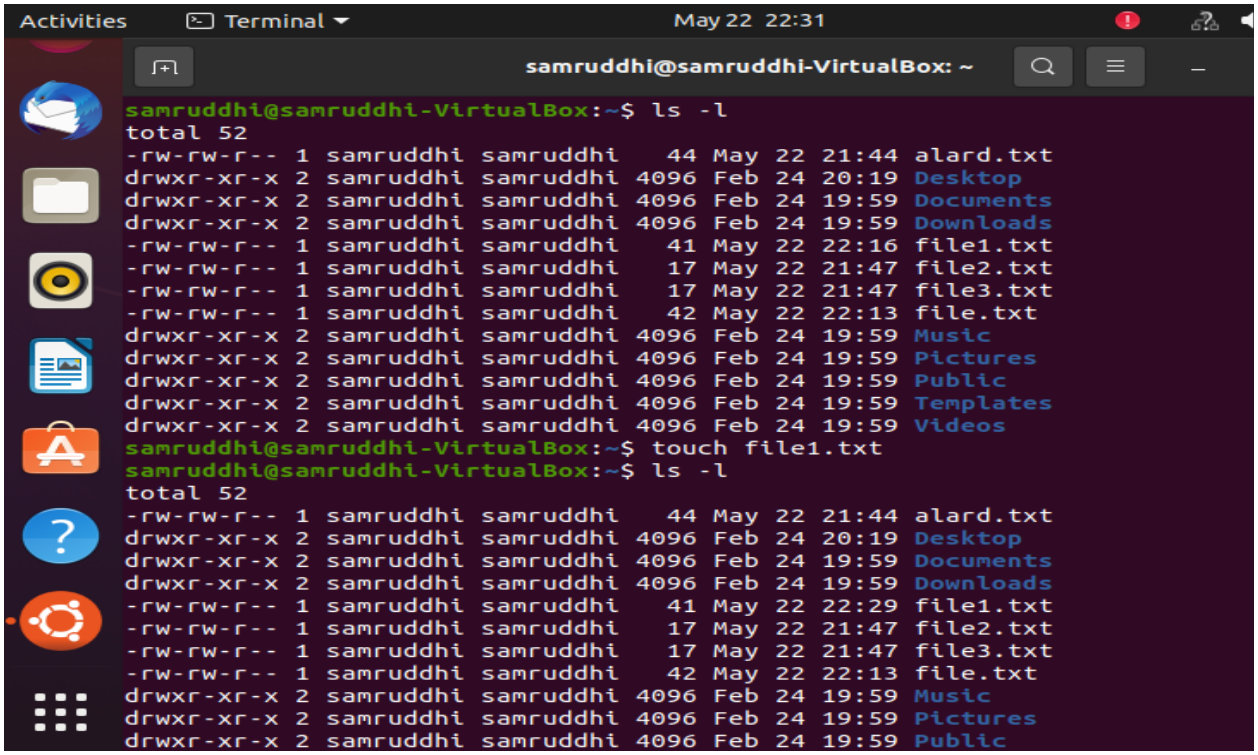

Activities
Terminal
May 22 22:17
samruddhi@samruddhi-VirtualBox: ~
samruddhi@samruddhi-VirtualBox:~$ cat >file.txt
Samruddhi
Akanksha
Soham
Saniya
Aditi
samruddhi@samruddhi-VirtualBox:~$ cat -n file.txt
1 Samruddhi
2
3 Akanksha
4 Soham
5
6
7 Saniya
8
9 Aditi

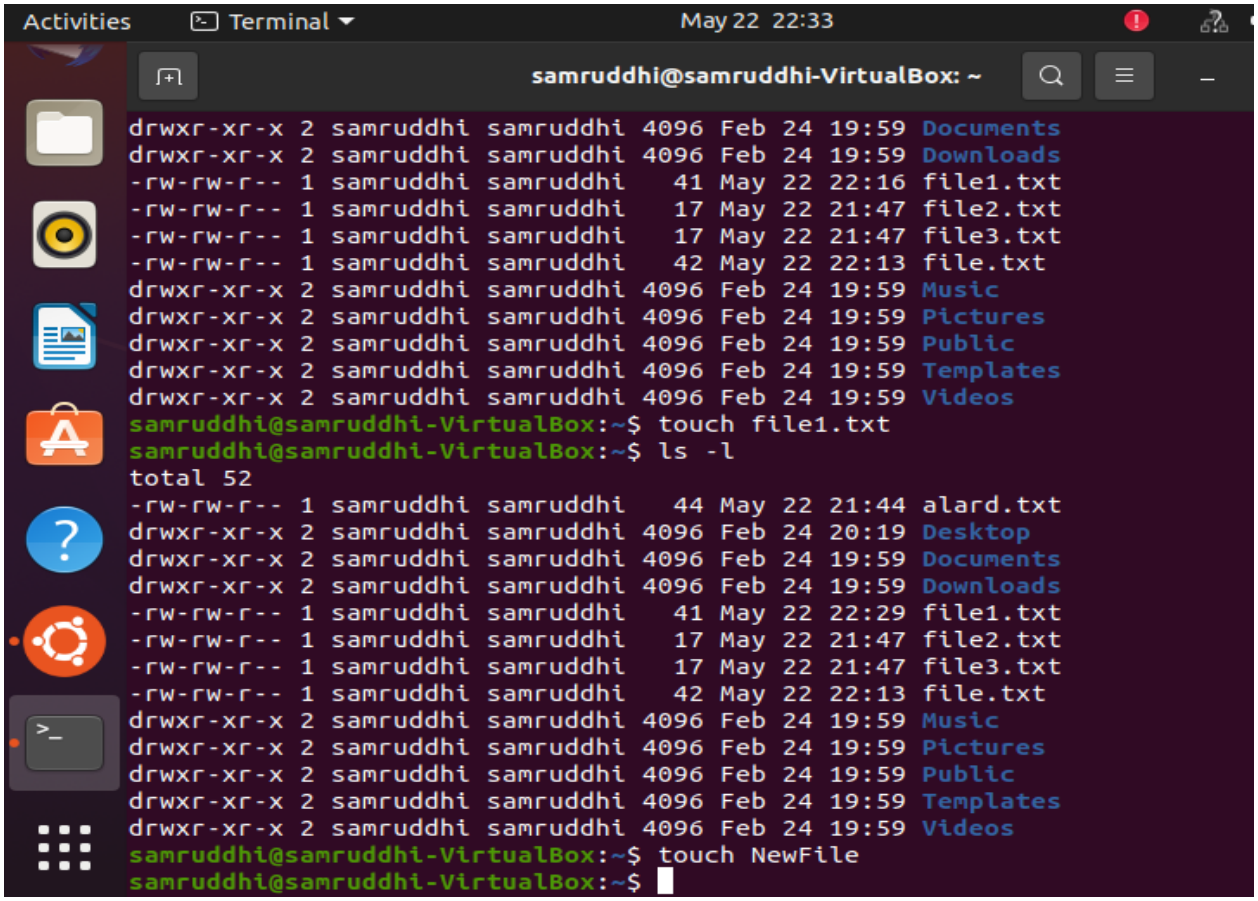

Activities
Terminal
May 22 22:17
samruddhi@samruddhi-VirtualBox: ~
samruddhi@samruddhi-VirtualBox:~$ cat -b file.txt
1 Samruddhi
2 Akanksha
3 Soham
4 Saniya
5 Aditi
samruddhi@samruddhi-VirtualBox:~$ cat -s file.txt
Samruddhi
Akanksha
Soham
Saniya
Aditi
samruddhi@samruddhi-VirtualBox:~$ cat file.txt
Samruddhi
Akanksha
Soham
Saniya
Aditi

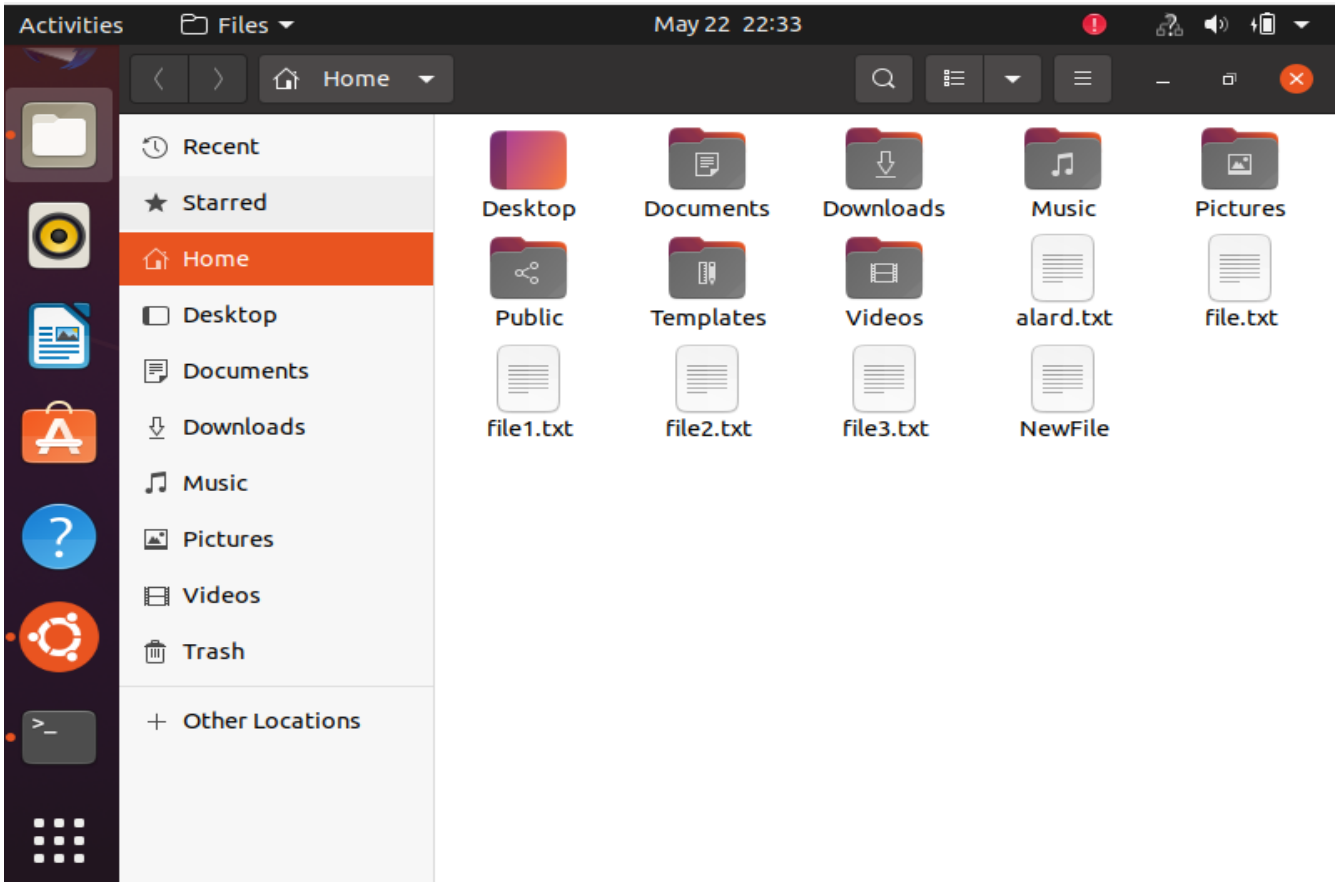

Activities
Terminal
May 22 22:19
samruddhi@samruddhi-VirtualBox: ~
Akanksha
Soham
Saniya
Aditi
samruddhi@samruddhi-VirtualBox:~$ cat file.txt
Samruddhi
Akanksha
Soham
Saniya
Aditi
samruddhi@samruddhi-VirtualBox:~$ cat file2.txt
this is 2nd file
samruddhi@samruddhi-VirtualBox:~$ cat file1.txt
File 1
this is 2nd file
samruddhi@samruddhi-VirtualBox:~$ cat file2.txt>>file1.txt
samruddhi@samruddhi-VirtualBox:~$ cat file1.txt
File 1
this is 2nd file
this is 2nd file
samruddhi@samruddhi-VirtualBox:~$ man cat
samruddhi@samruddhi-VirtualBox:~$

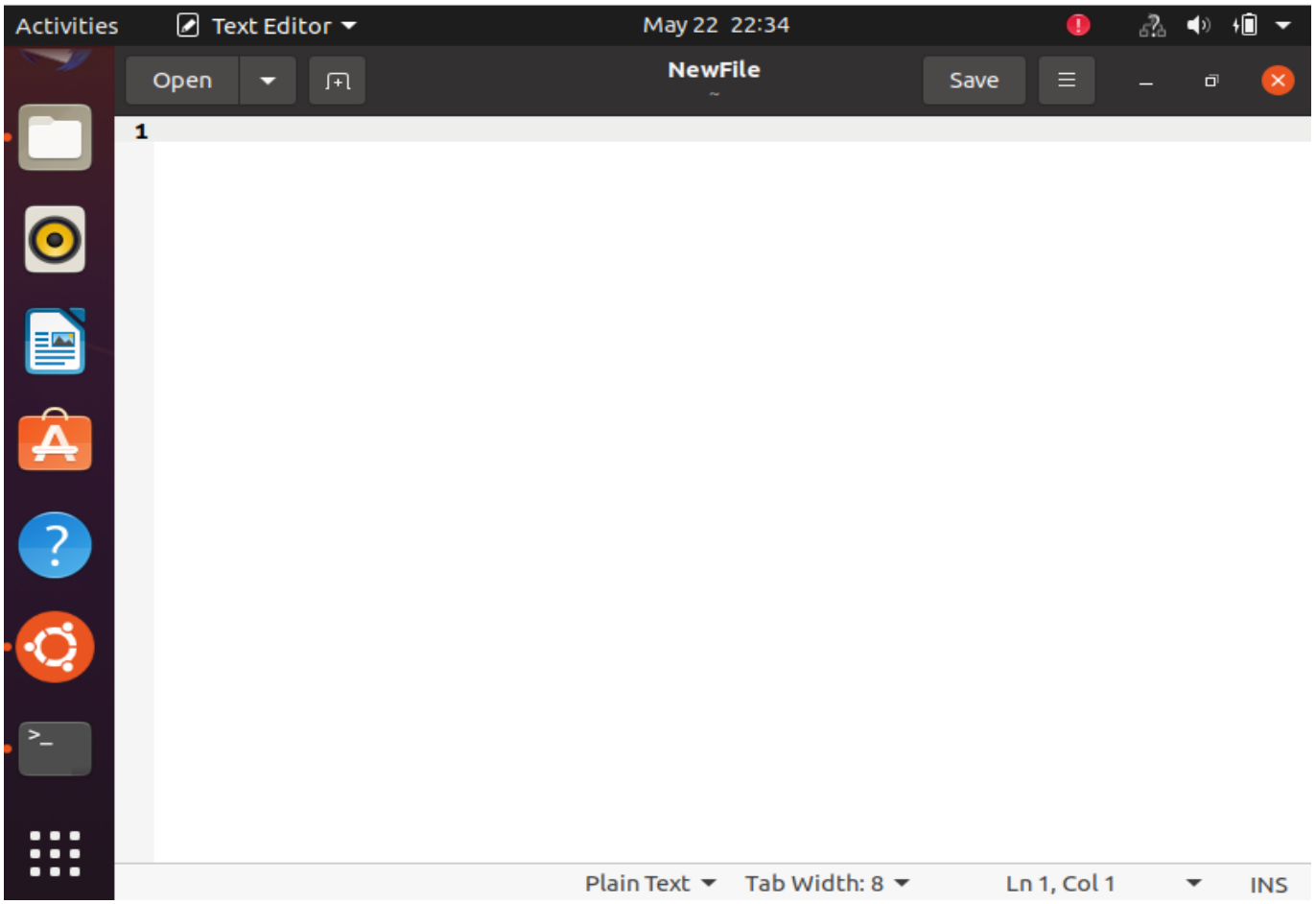

Activities
Terminal
May 22 22:19
samruddhi@samruddhi-VirtualBox: ~
CAT(1) User Commands CA
NAME
cat - concatenate files and print on the standard output
SYNOPSIS
cat [OPTION]... [FILE]...
DESCRIPTION
Concatenate FILE(s) to standard output.
With no FILE, or when FILE is -, read standard input.
-A, --show-all
equivalent to -vET
-b, --number-nonblank
number nonempty output lines, overrides -n
-e equivalent to -vE
-E, --show-ends
display $ at end of each line
-n, --number
number all output lines
-s, --squeeze-blank
Manual page cat(1) line 1 (press h for help or q to quit)

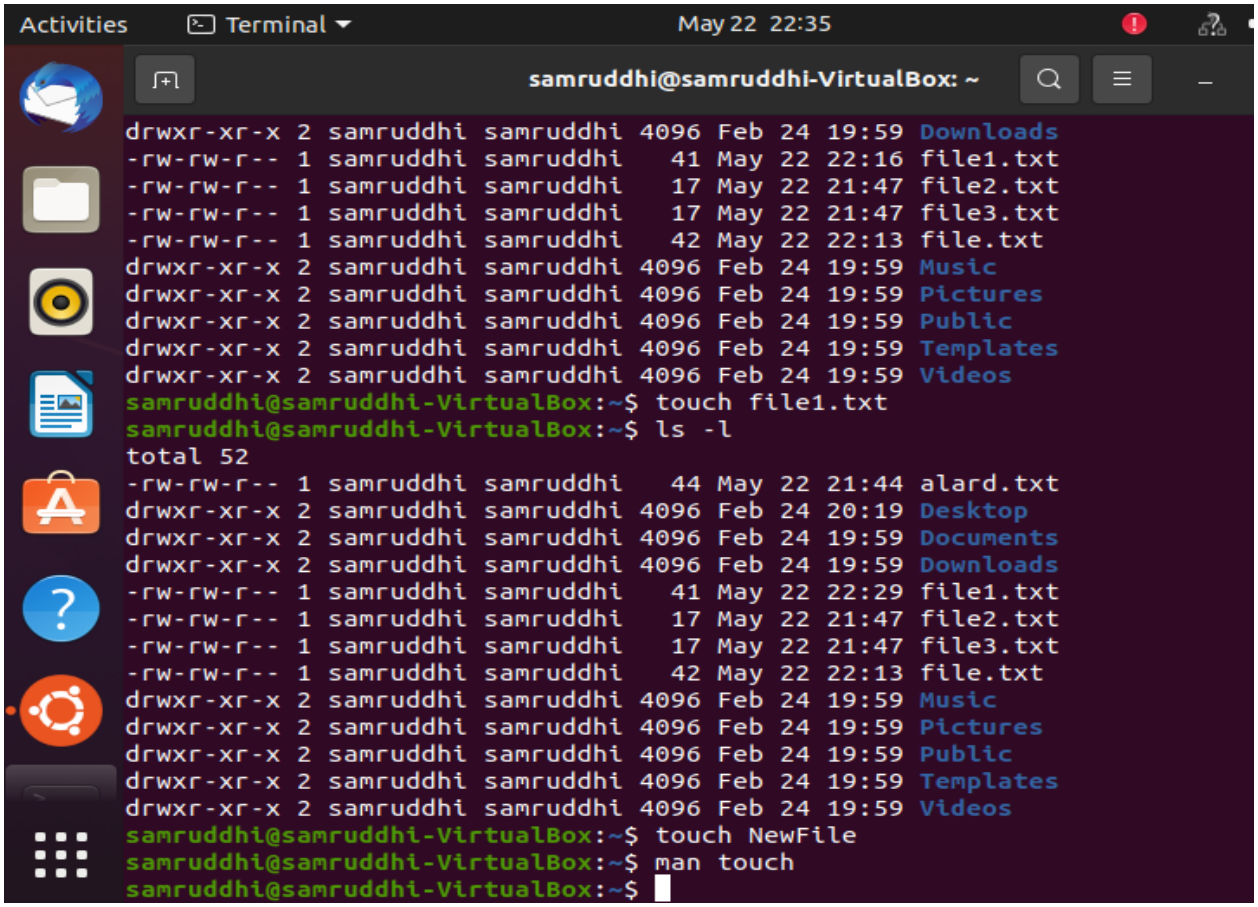
touch command:



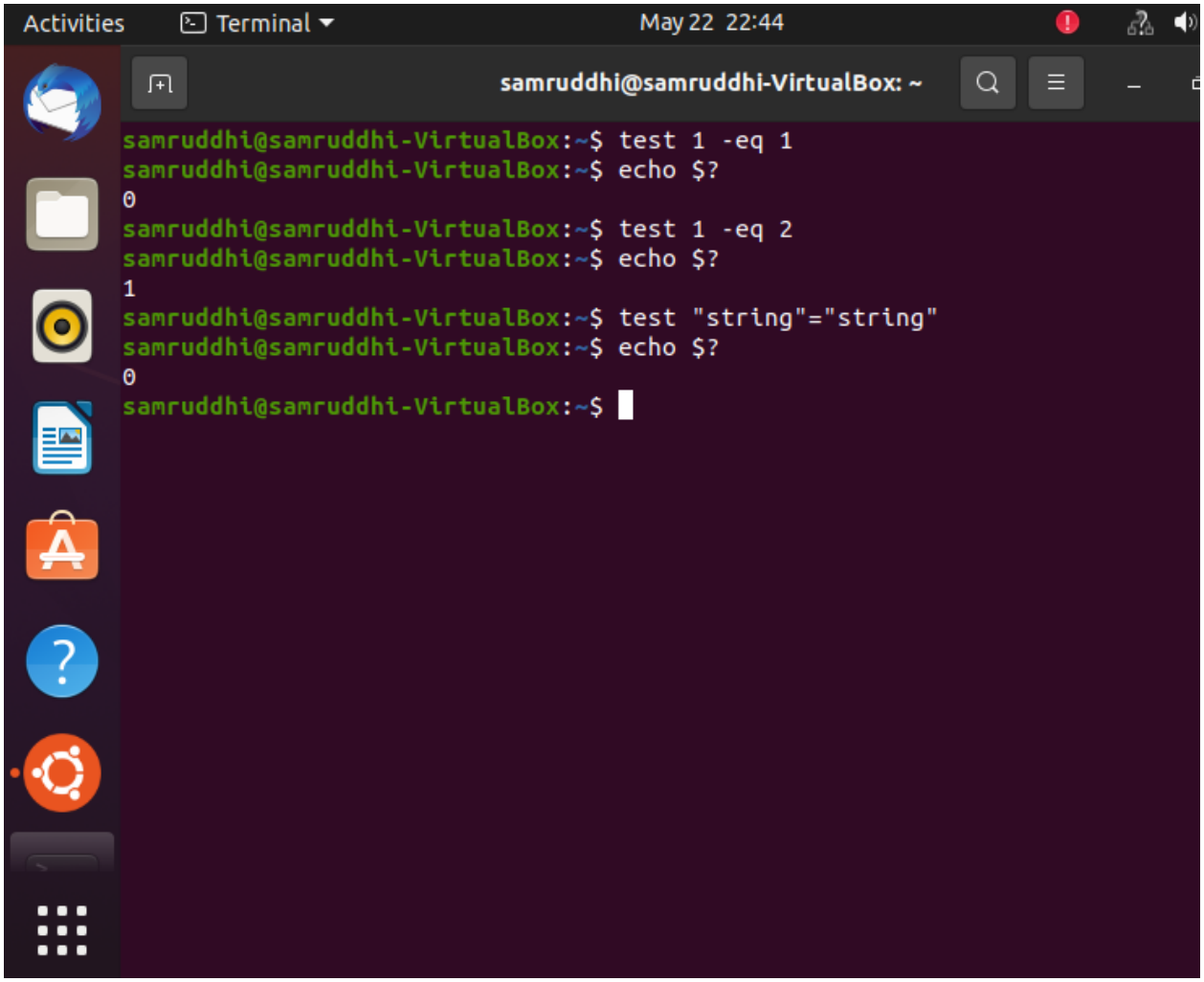



Activities
Files
May 22 22:33
Home
Recent
Starred
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
alard.txt
file.txt
file1.txt
file2.txt
file3.txt
NewFile


Activities
Text Editor
May 22 22:34
Open
NewFile
Save
1
Plain Text
Tab Width: 8
Ln 1, Col 1
INS



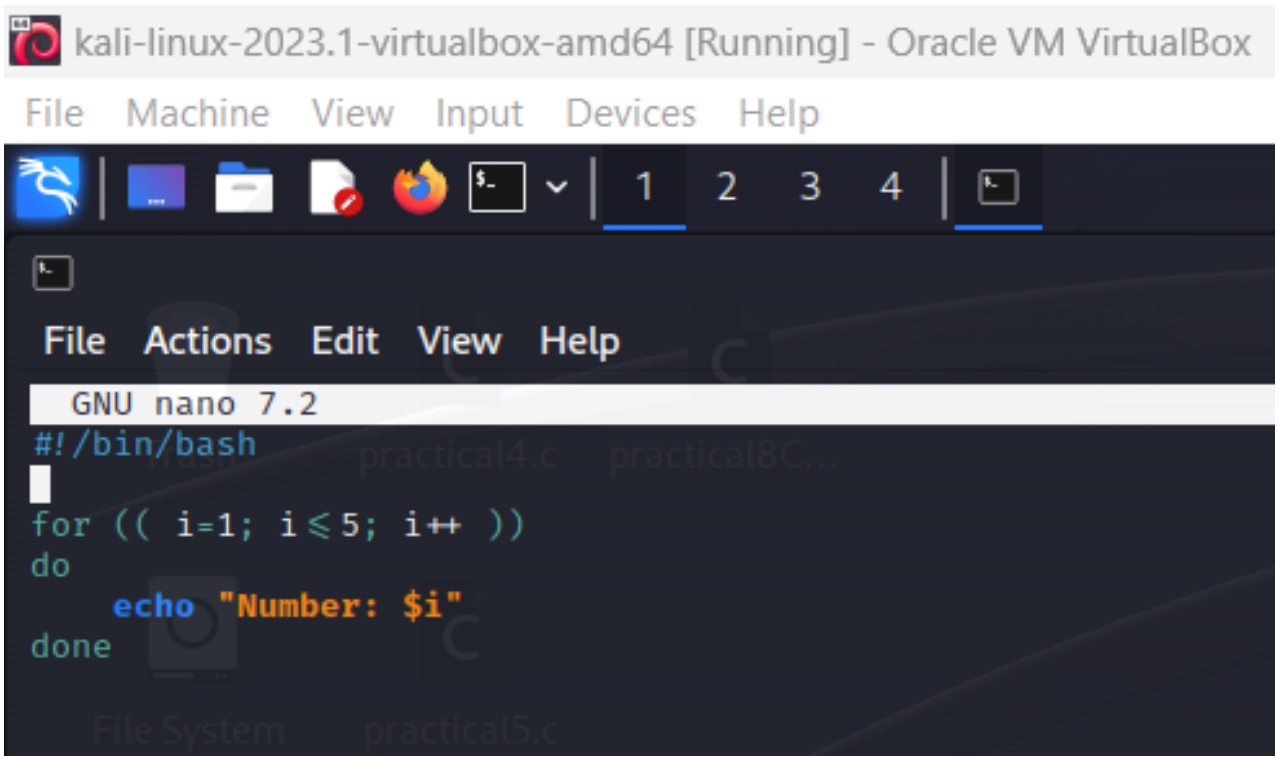
test command:

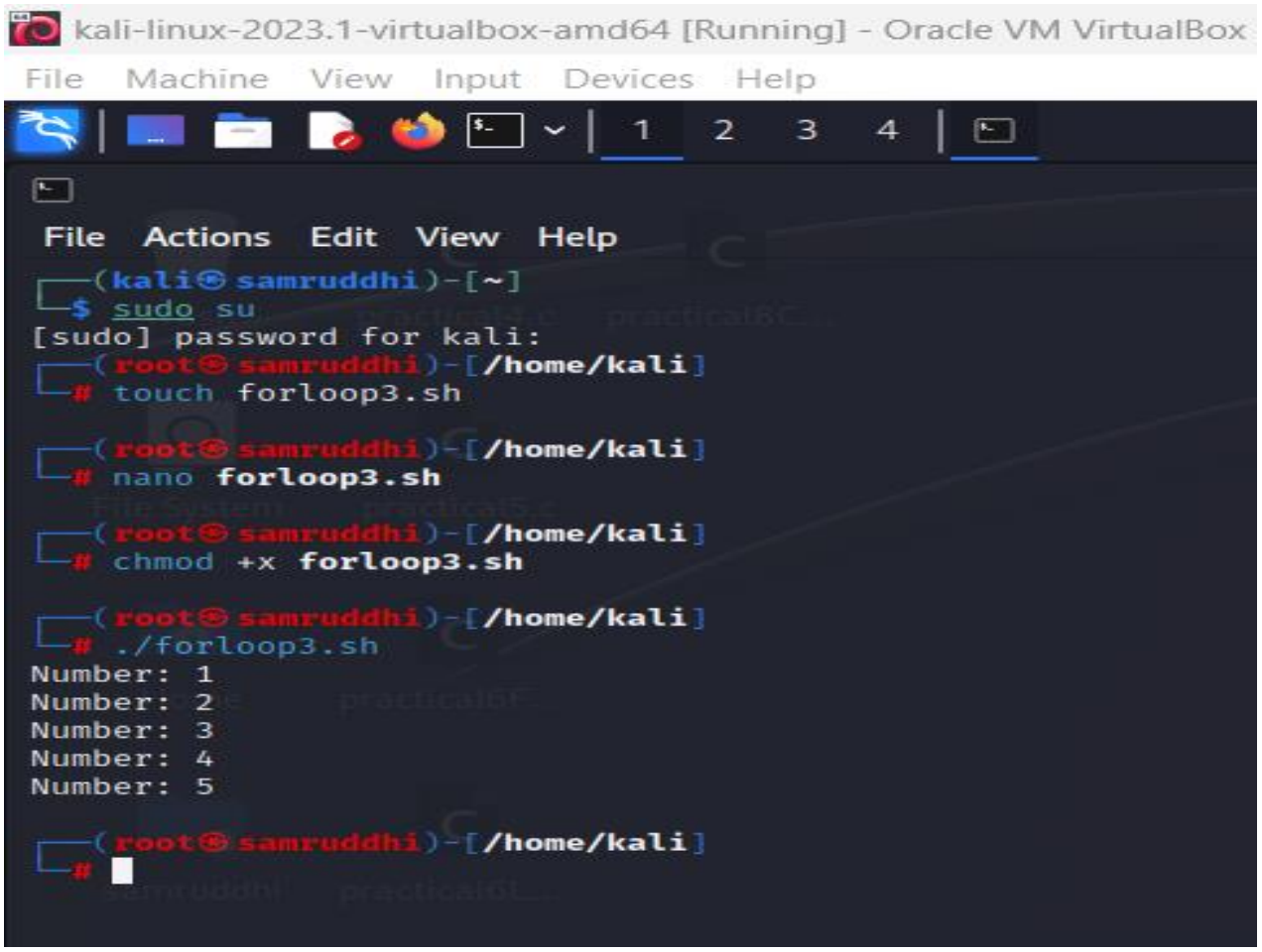


loops command:

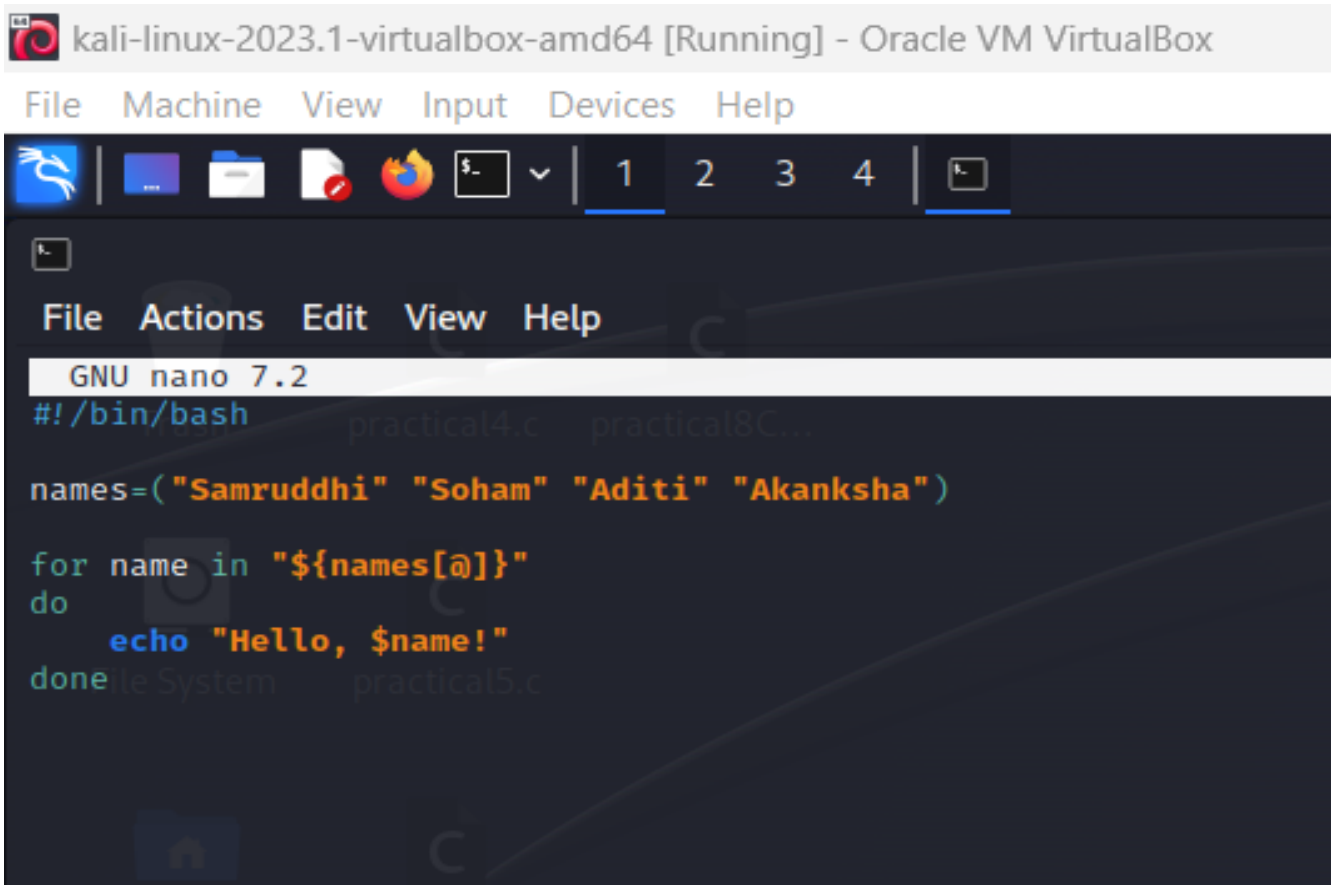
1) for loop

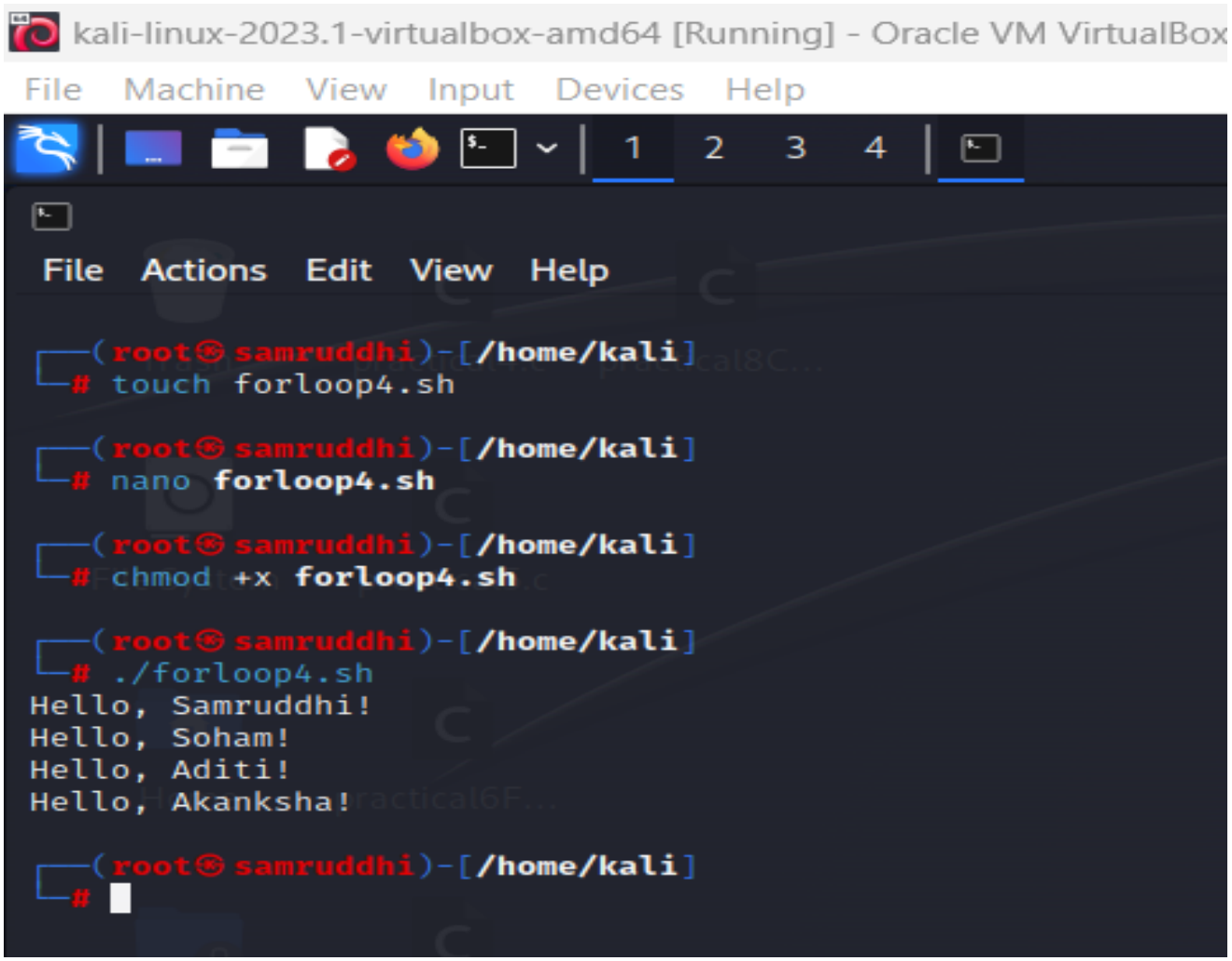
Example 1



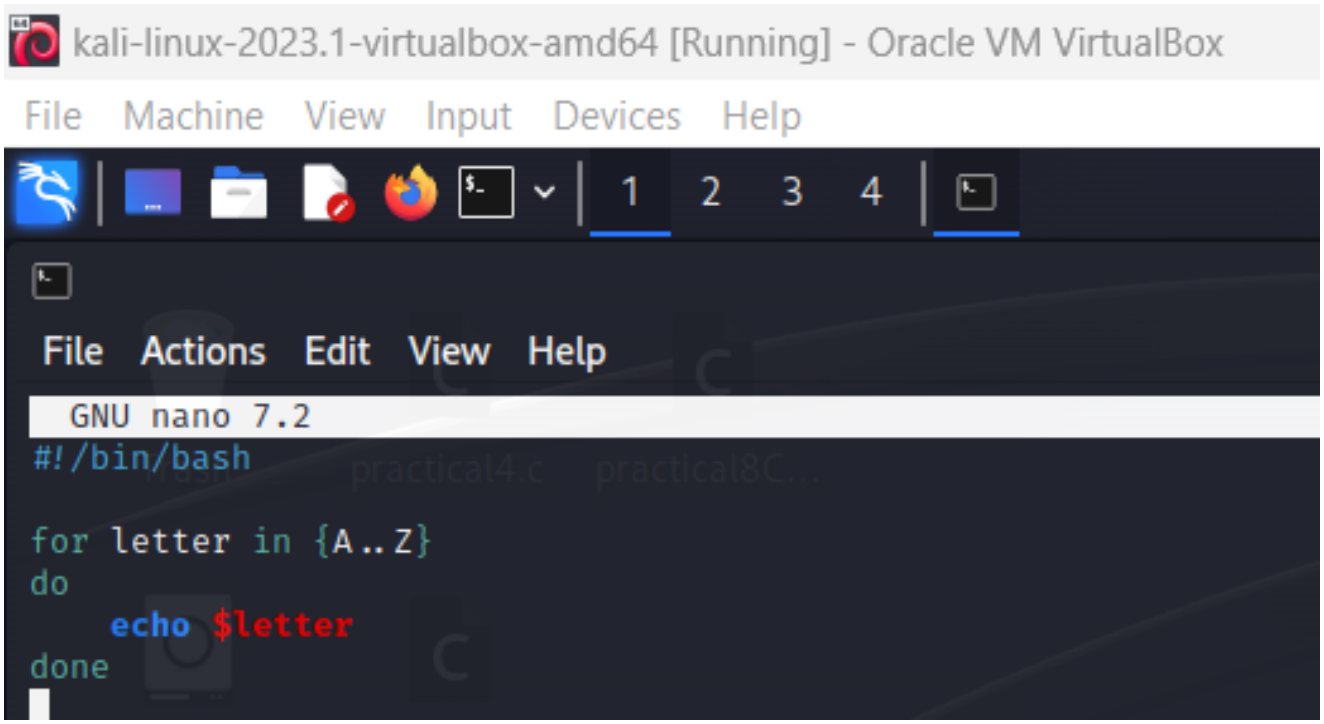


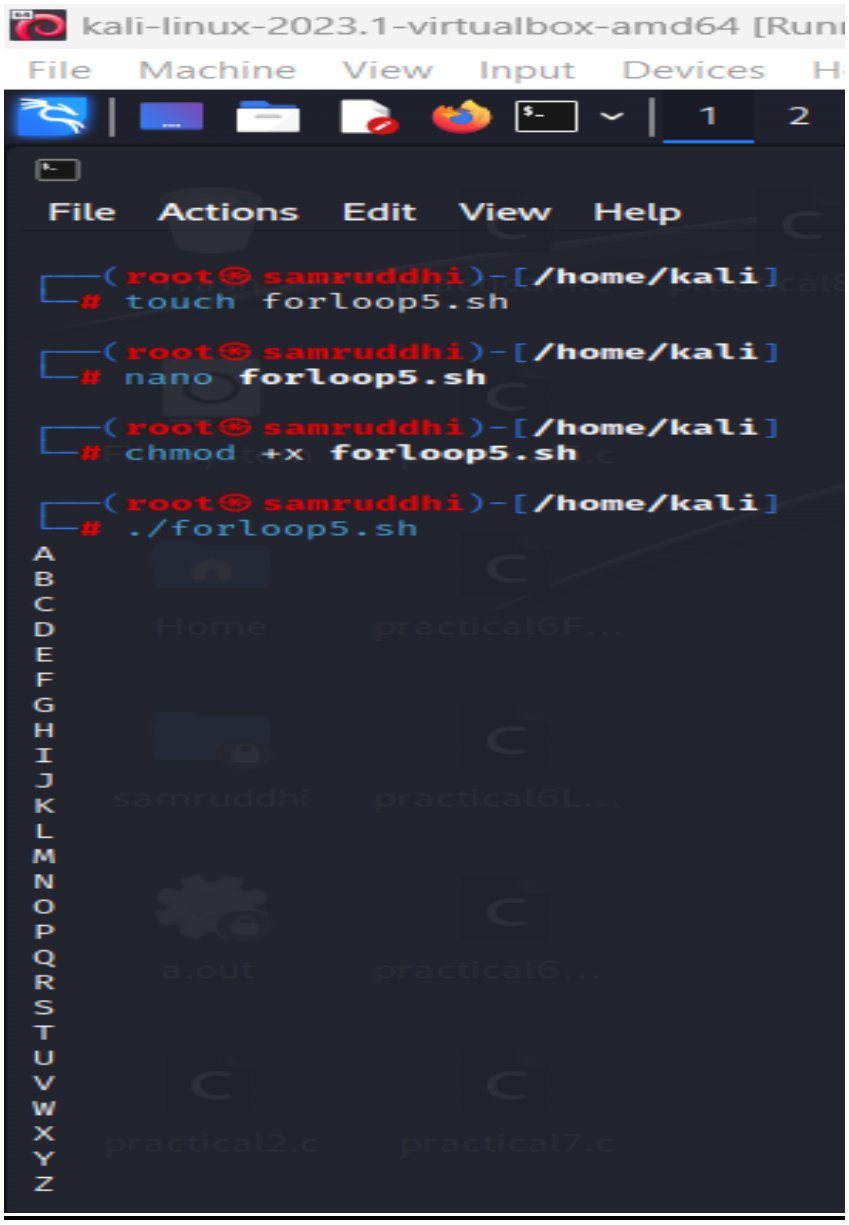
Example 2


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
names=("Samruddhi" "Soham" "Aditi" "Akanksha")
for name in "${names[@]}"
do
echo "Hello, $name!"
done


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
┌──(root㉿samruddhi)-[/home/kali]
└─# touch forloop4.sh
┌──(root㉿samruddhi)-[/home/kali]
└─# nano forloop4.sh
┌──(root㉿samruddhi)-[/home/kali]
└─# chmod +x forloop4.sh
┌──(root㉿samruddhi)-[/home/kali]
└─# ./forloop4.sh
Hello, Samruddhi!
Hello, Soham!
Hello, Aditi!
Hello, Akanksha!
┌──(root㉿samruddhi)-[/home/kali]
└─#

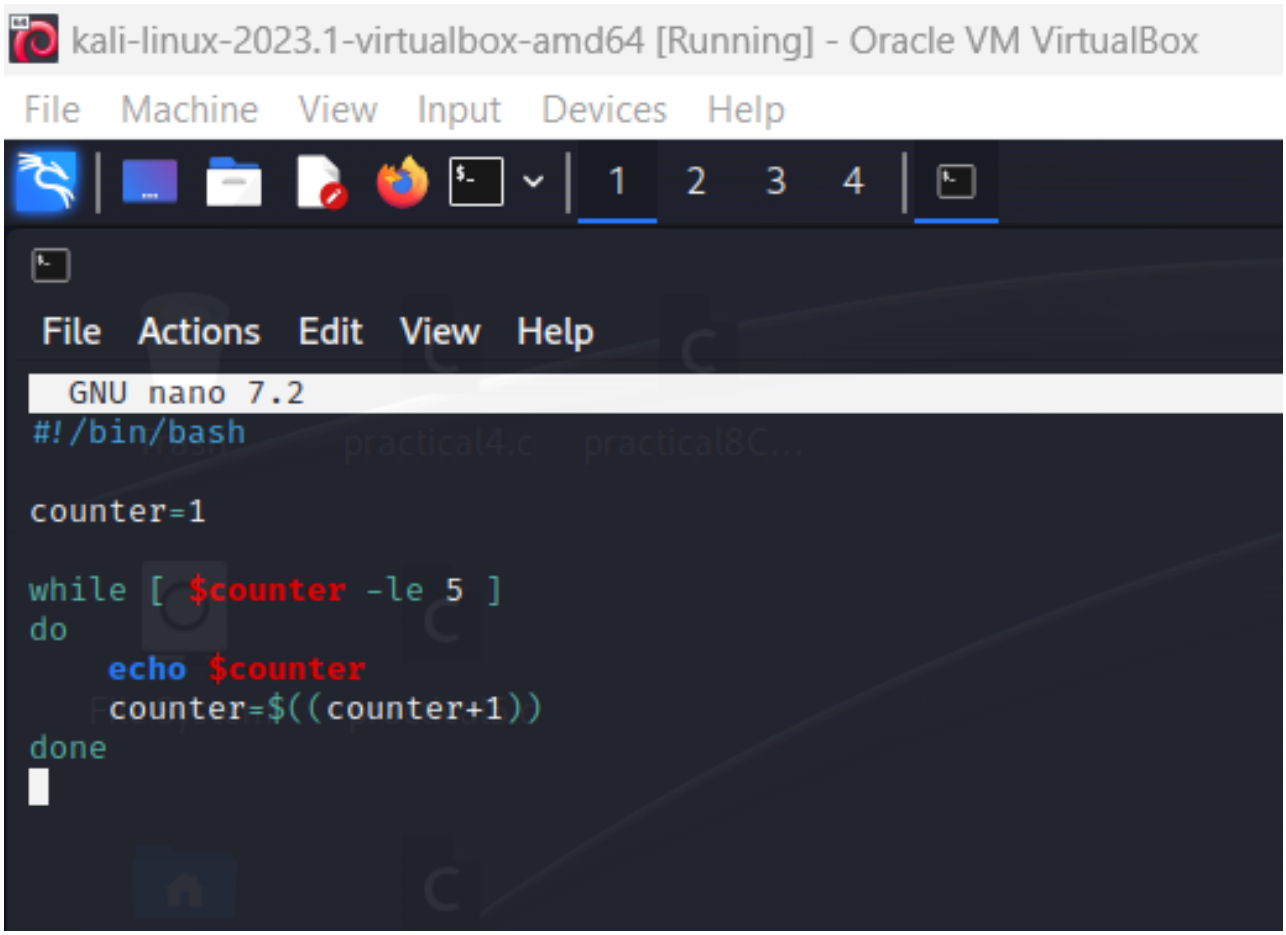
Example 3

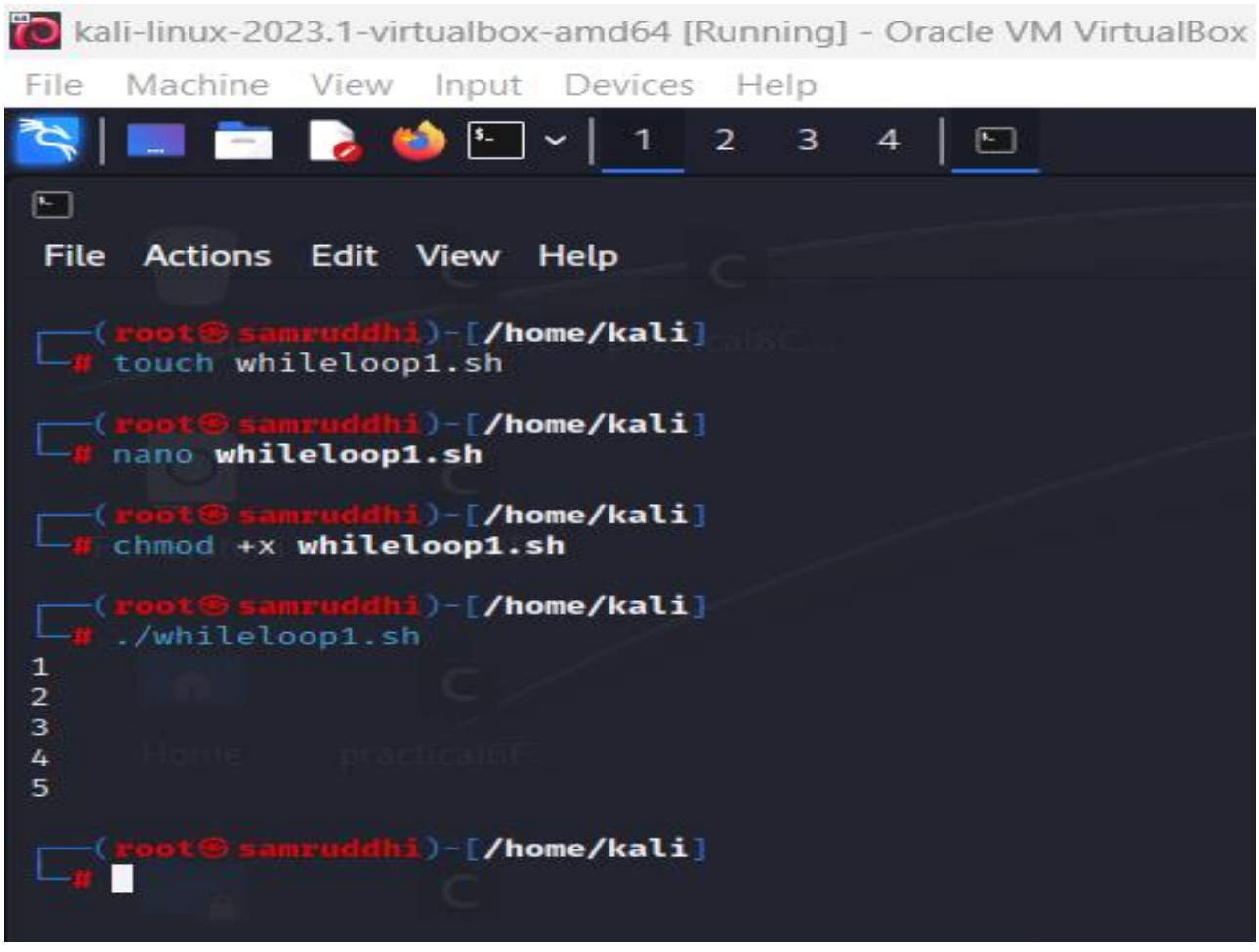

kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
for letter in {A..Z}
do
echo $letter
done


kali-linux-2023.1-virtualbox-amd64 [Runn
File Machine View Input Devices H
1 2
File Actions Edit View Help
(root㉿samruddhi)-[/home/kali]
touch forloop5.sh
(root㉿samruddhi)-[/home/kali]
nano forloop5.sh
(root㉿samruddhi)-[/home/kali]
chmod +x forloop5.sh
(root㉿samruddhi)-[/home/kali]
./forloop5.sh
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

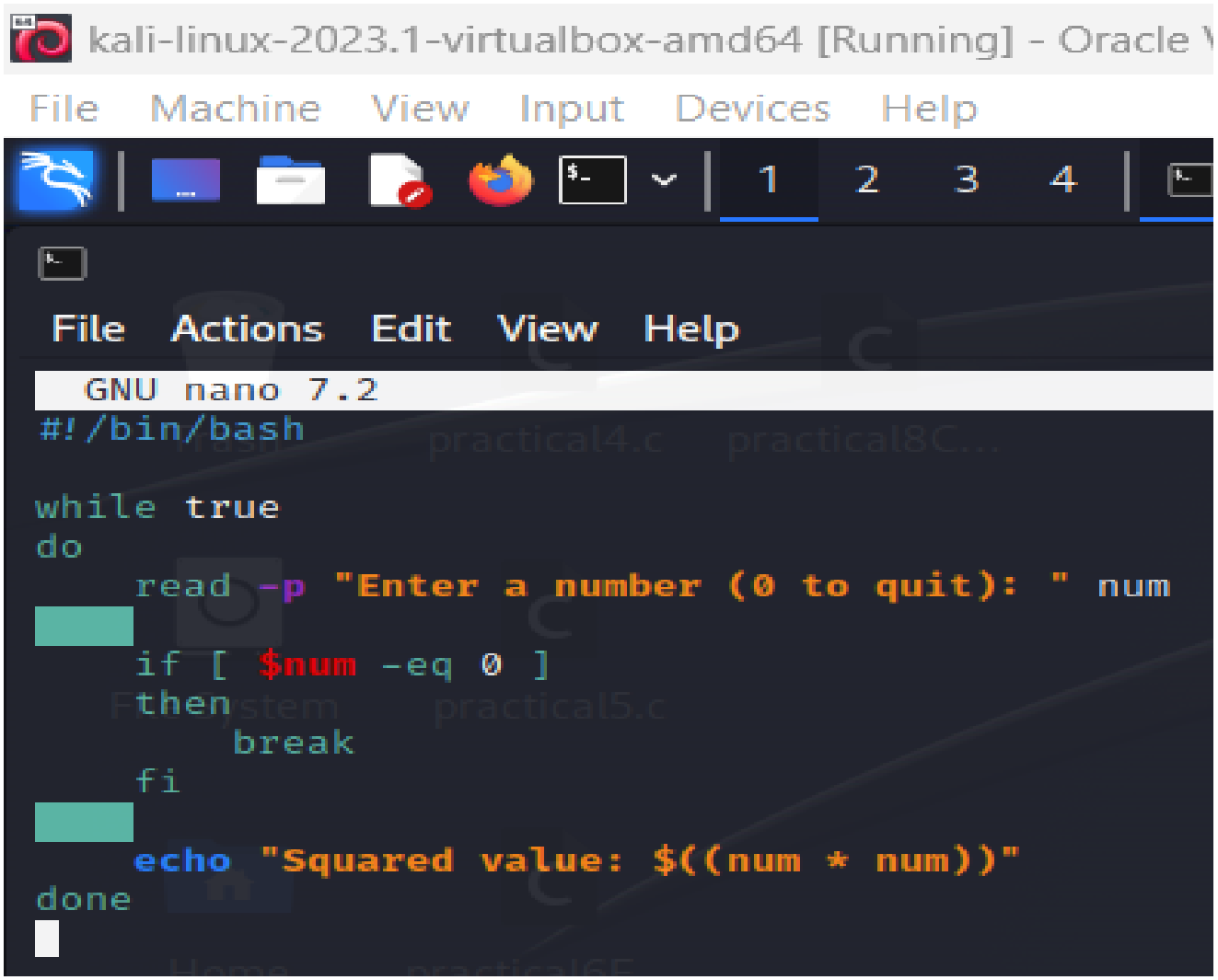
2) while loop

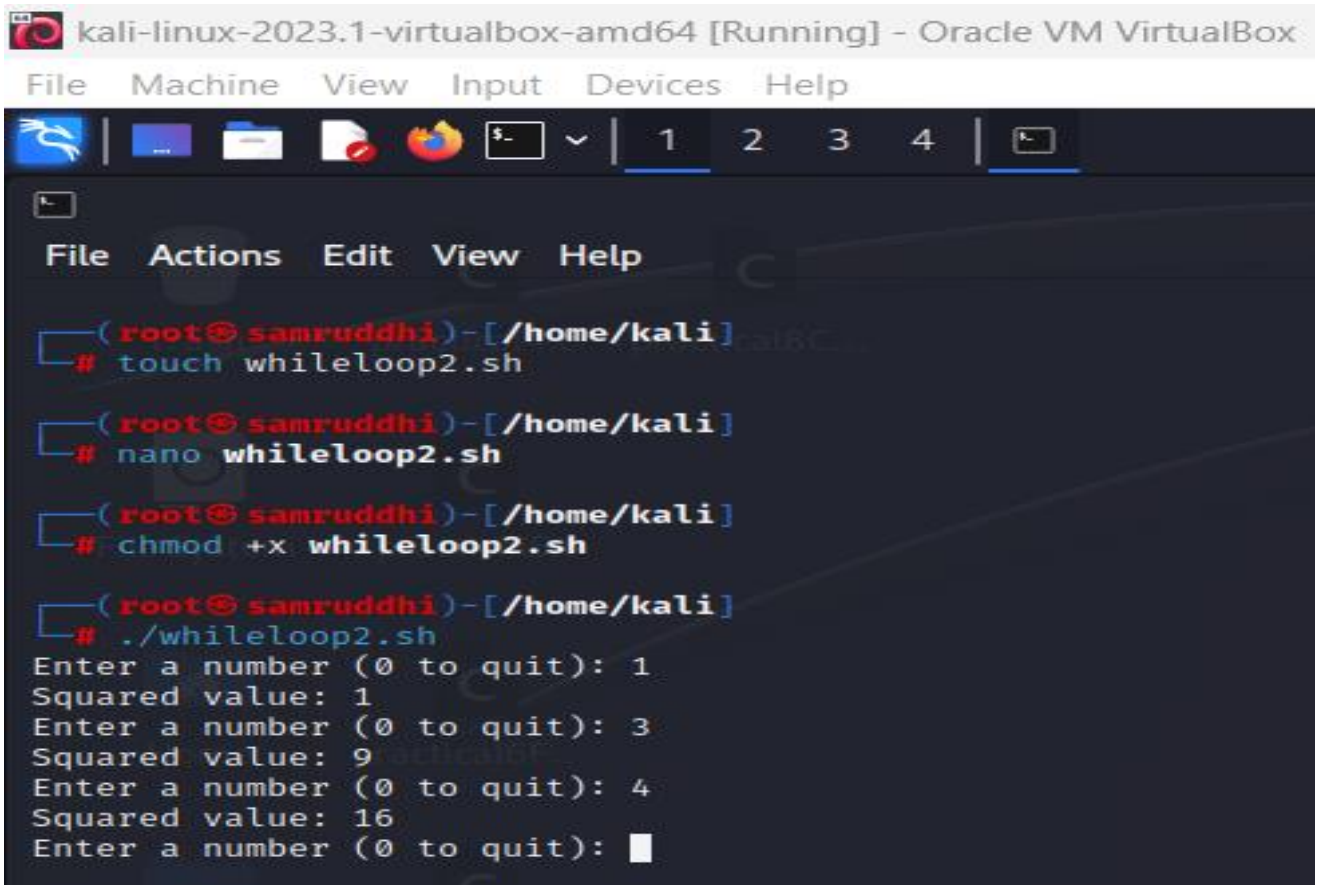
Example 1





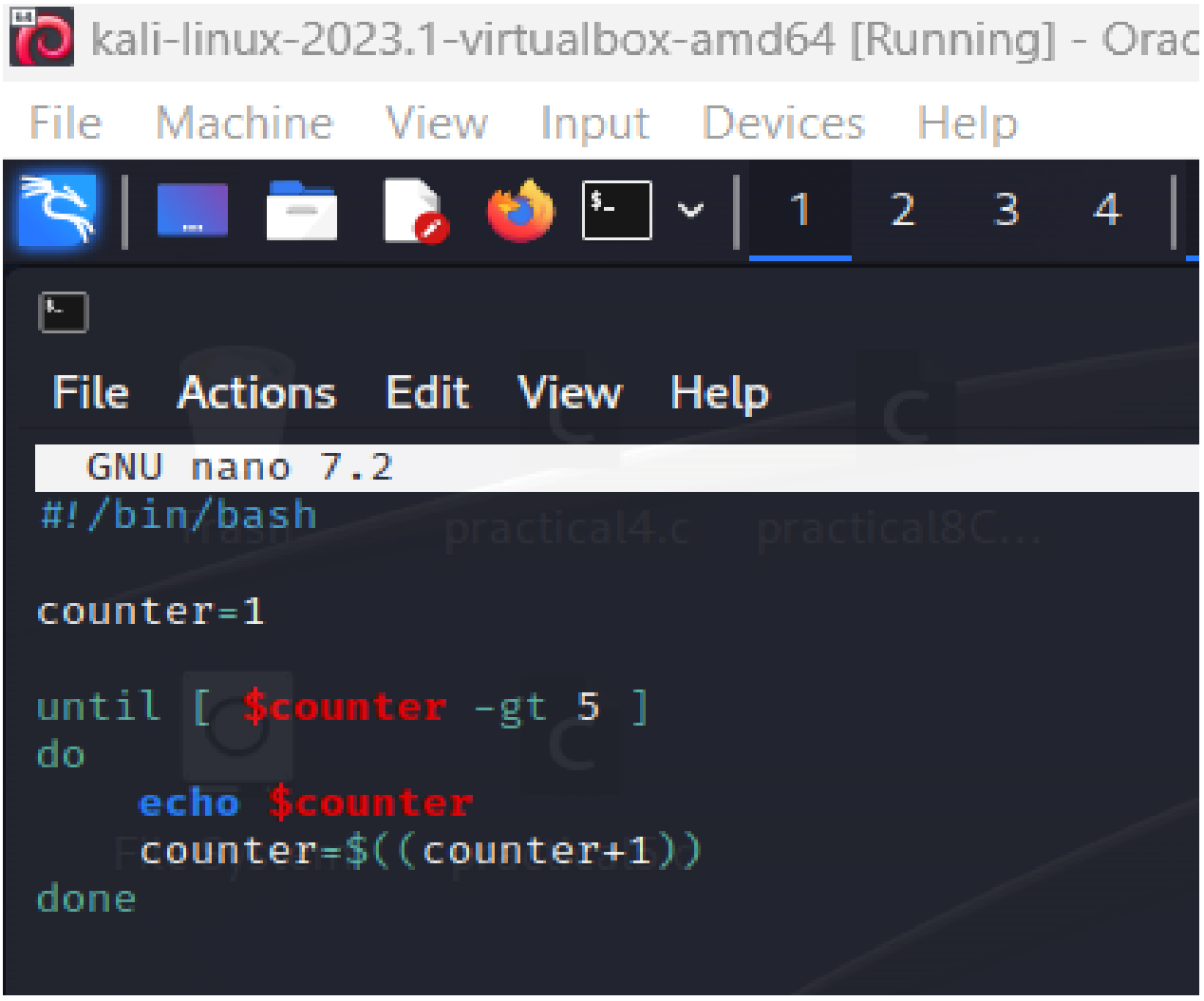
Example 2

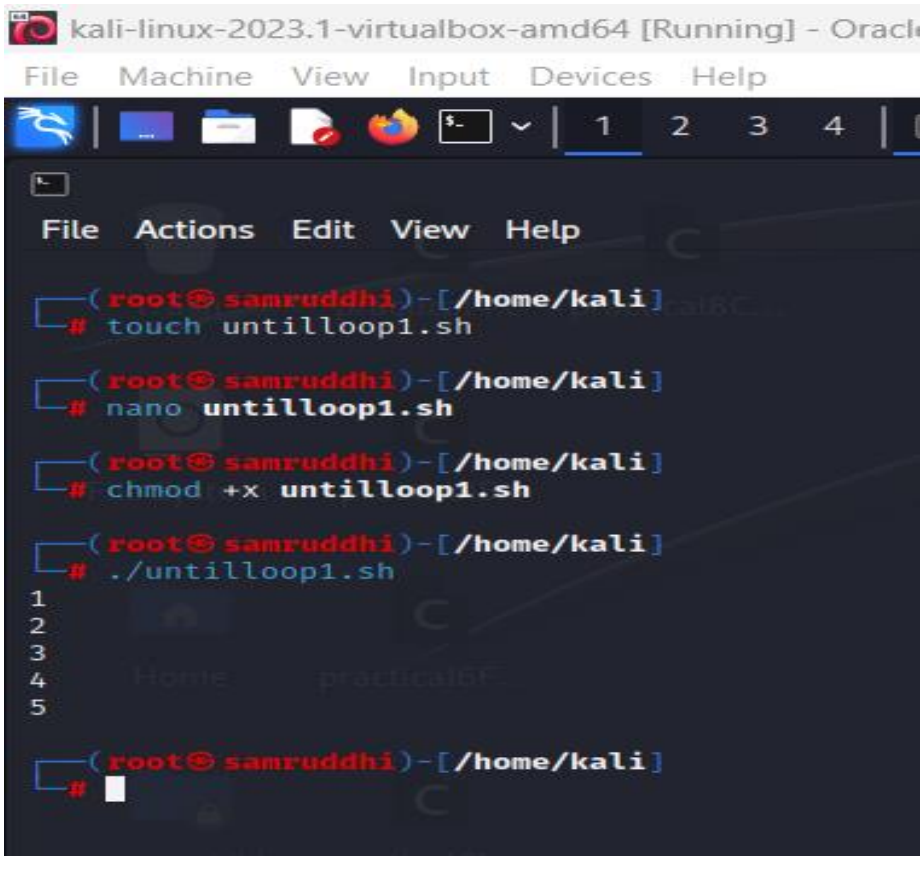

kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
while true
do
read -p "Enter a number (0 to quit): " num
if [$num -eq 0]
then
break
fi
echo "Squared value: $((num * num))"
done


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(root㉿samruddhi)-[/home/kali]
touch whileloop2.sh
(root㉿samruddhi)-[/home/kali]
nano whileloop2.sh
(root㉿samruddhi)-[/home/kali]
chmod +x whileloop2.sh
(root㉿samruddhi)-[/home/kali]
./whileloop2.sh
Enter a number (0 to quit): 1
Squared value: 1
Enter a number (0 to quit): 3
Squared value: 9
Enter a number (0 to quit): 4
Squared value: 16
Enter a number (0 to quit):

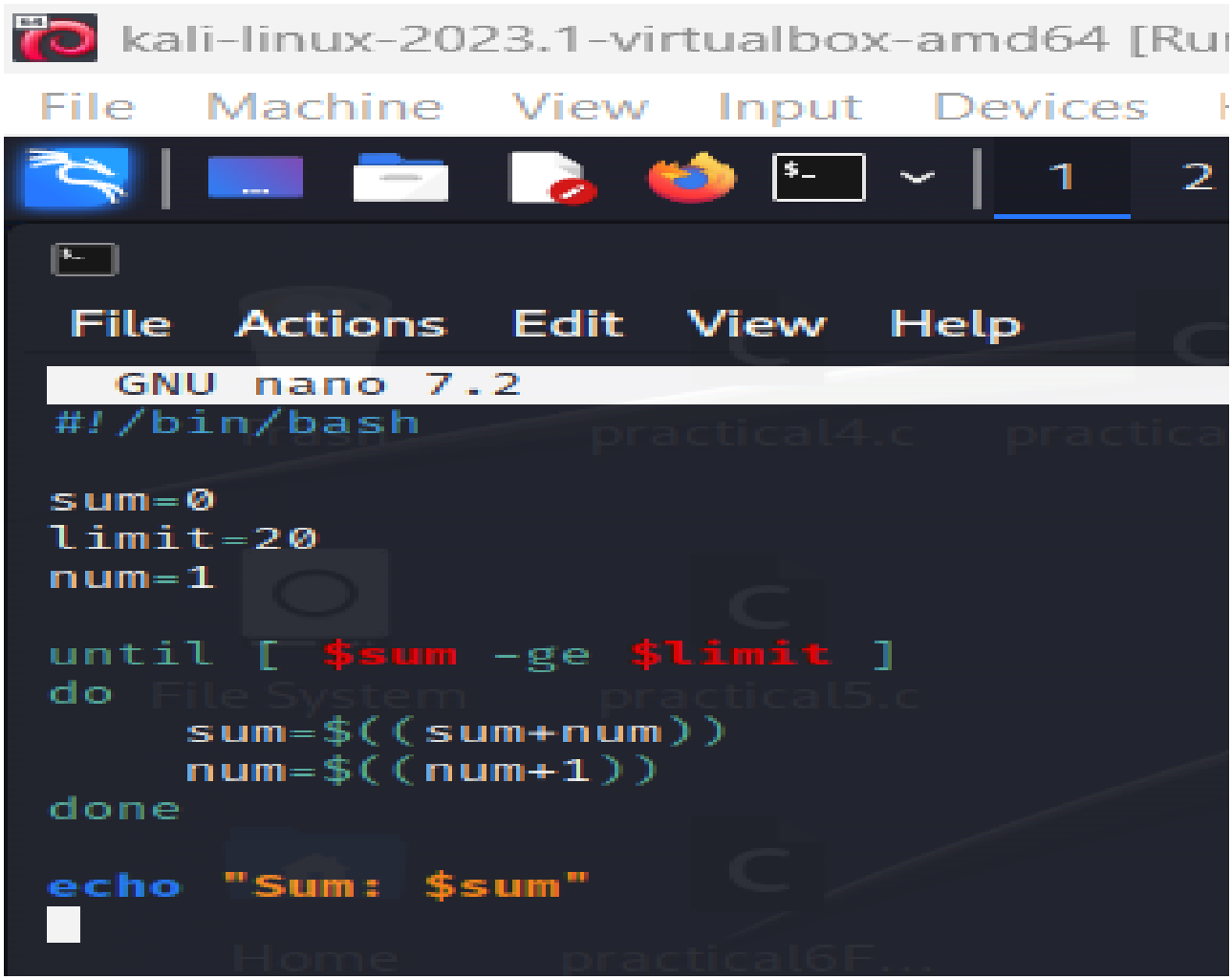
3) <u>until loop</u>

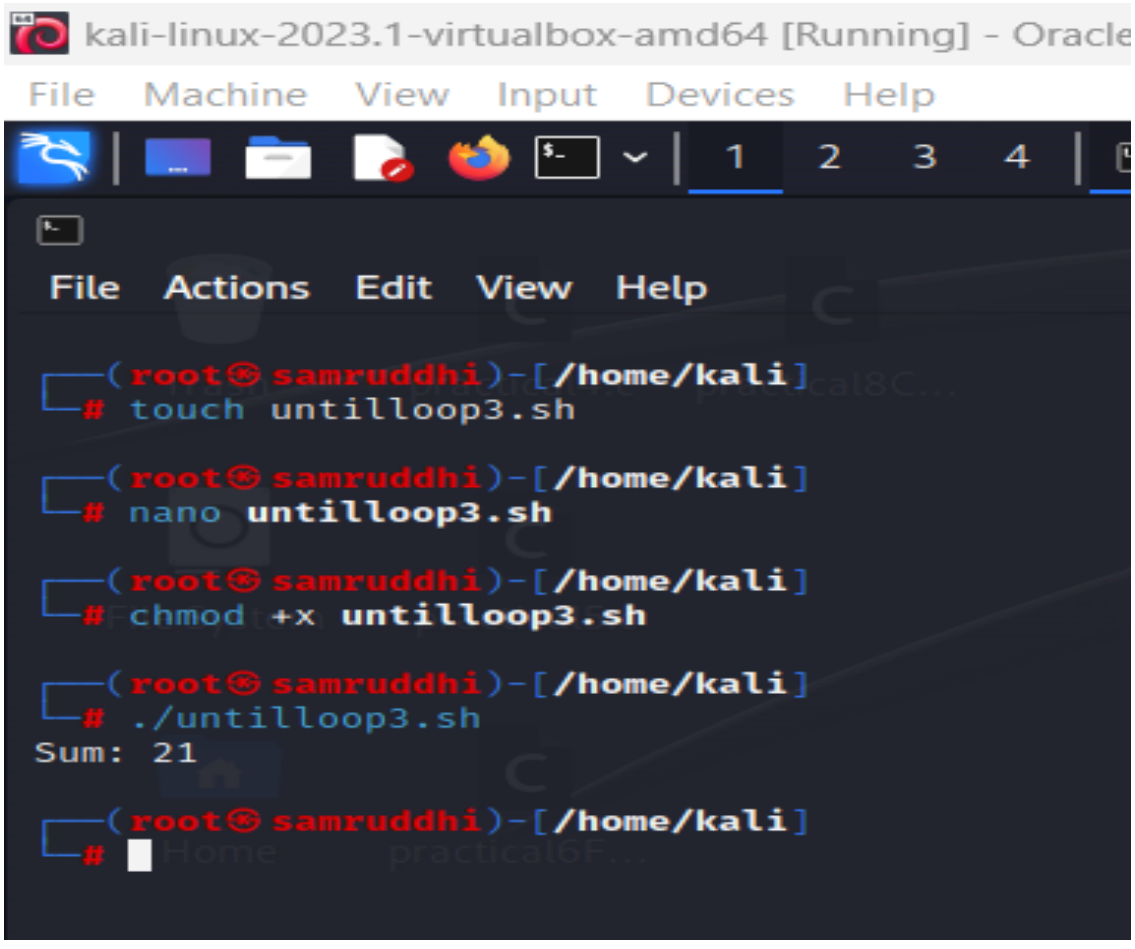
Example 1





Example 2

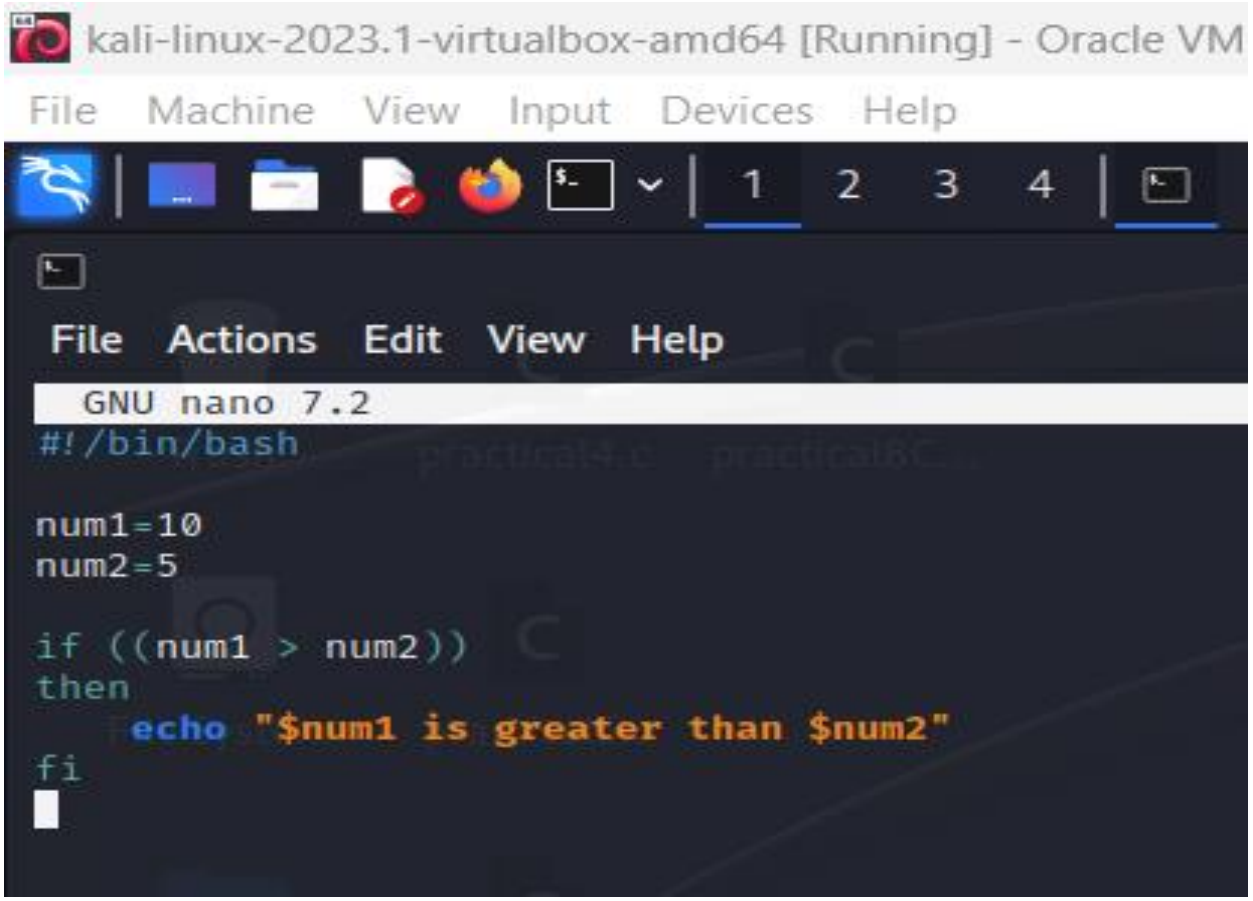

kali-linux-2023.1-virtualbox-amd64 [Ru
File Machine View Input Devices
1 2
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
sum=0
limit=20
num=1
until [$sum -ge $limit]
do
sum=$((sum+num))
num=$((num+1))
done
echo "Sum: $sum"

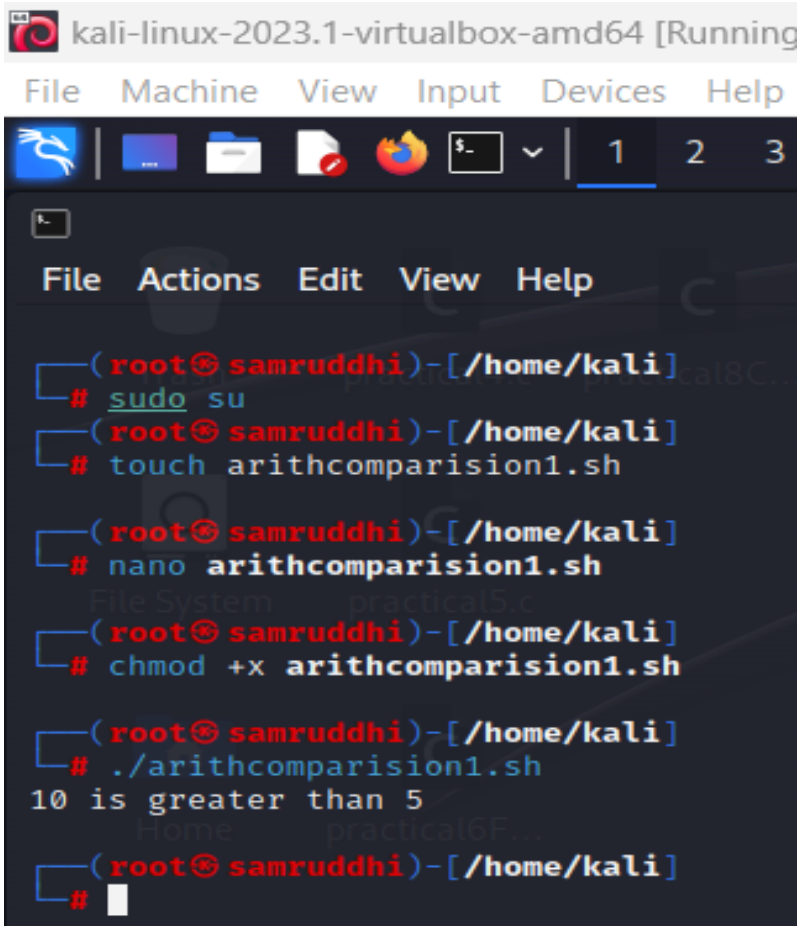

kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(root㉿samruddhi)-[/home/kali]
touch untilloop3.sh
(root㉿samruddhi)-[/home/kali]
nano untilloop3.sh
(root㉿samruddhi)-[/home/kali]
chmod +x untilloop3.sh
(root㉿samruddhi)-[/home/kali]
./untilloop3.sh
Sum: 21
(root㉿samruddhi)-[/home/kali]
#

arithmetic comparison:

1) > (Greater Than) Operator

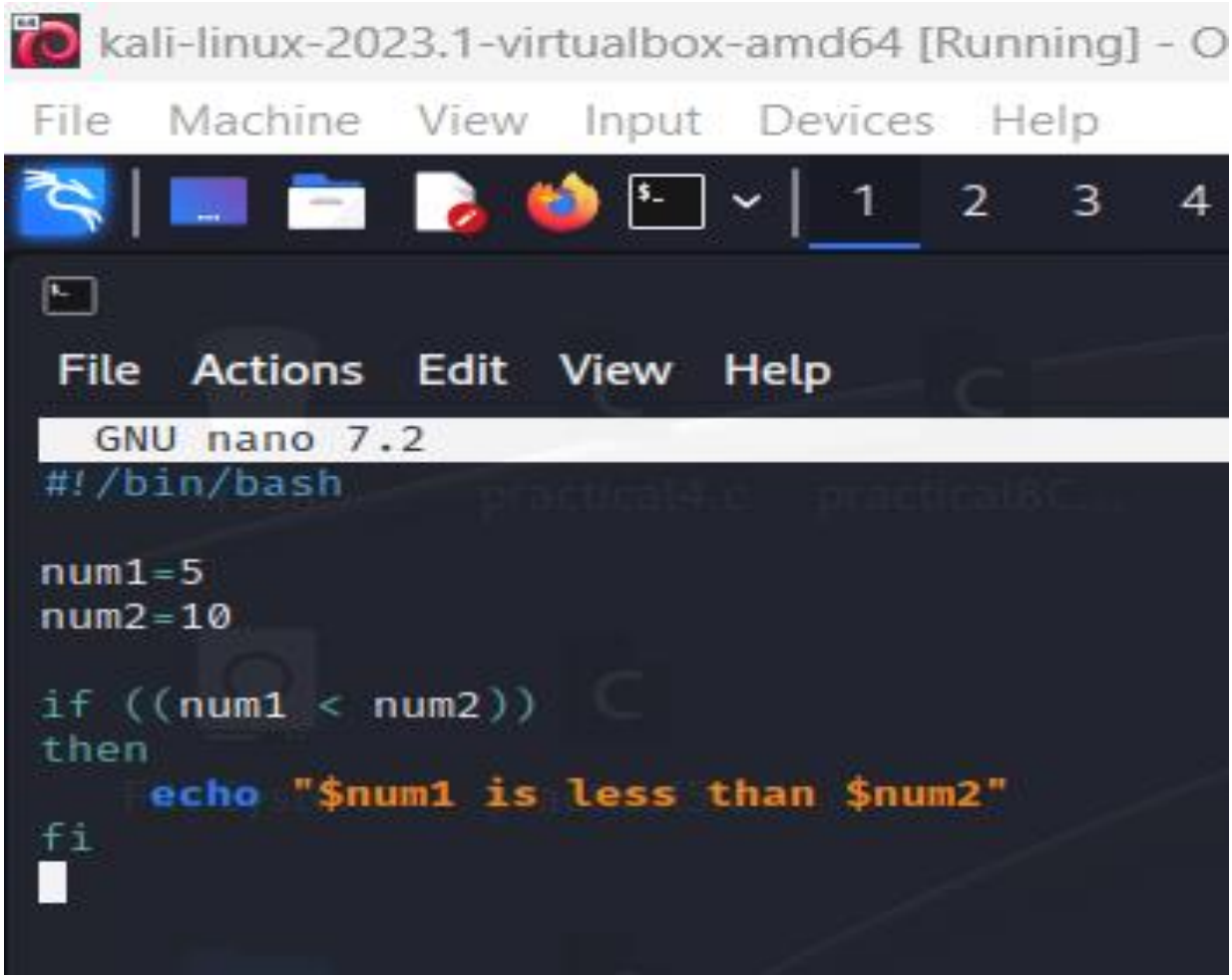
Example

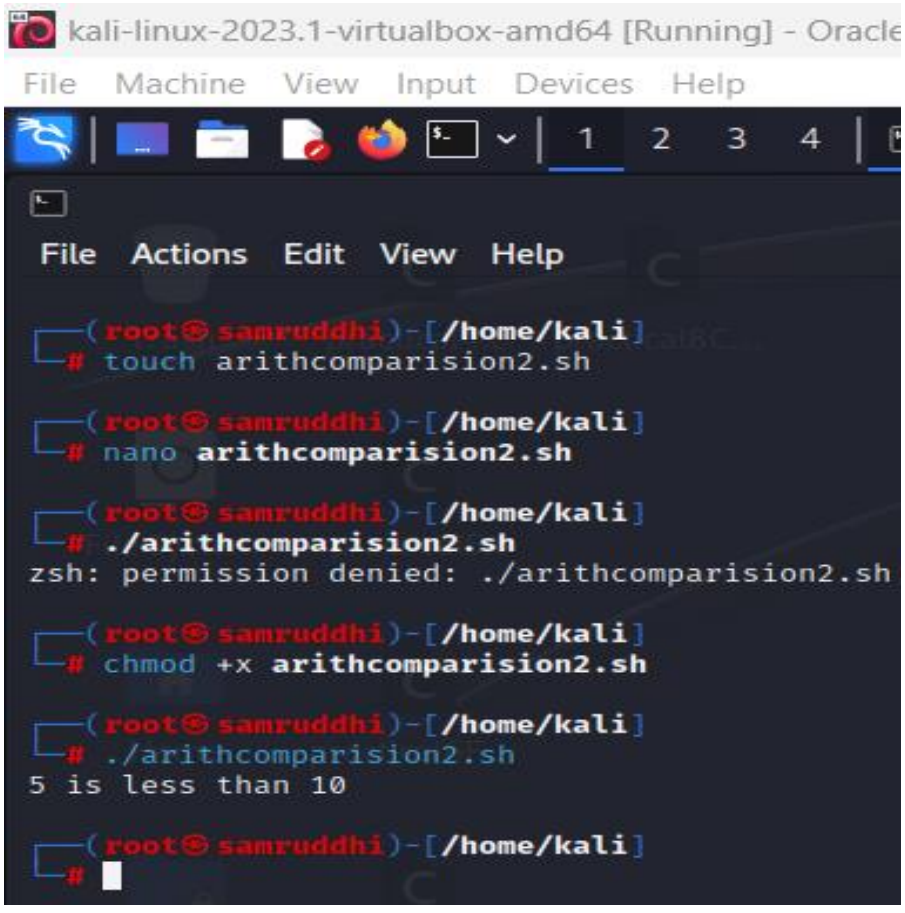




2) < (Less Than) Operator

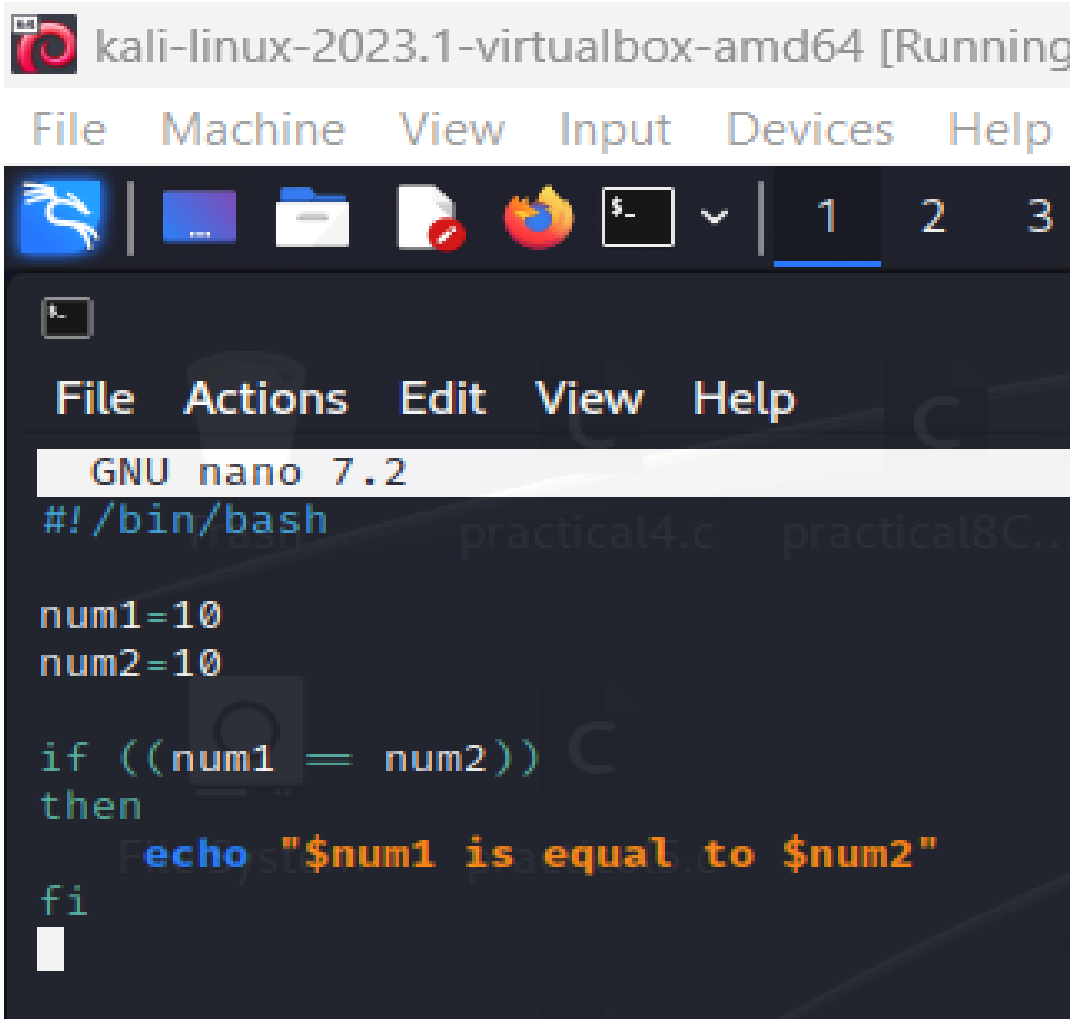
Example

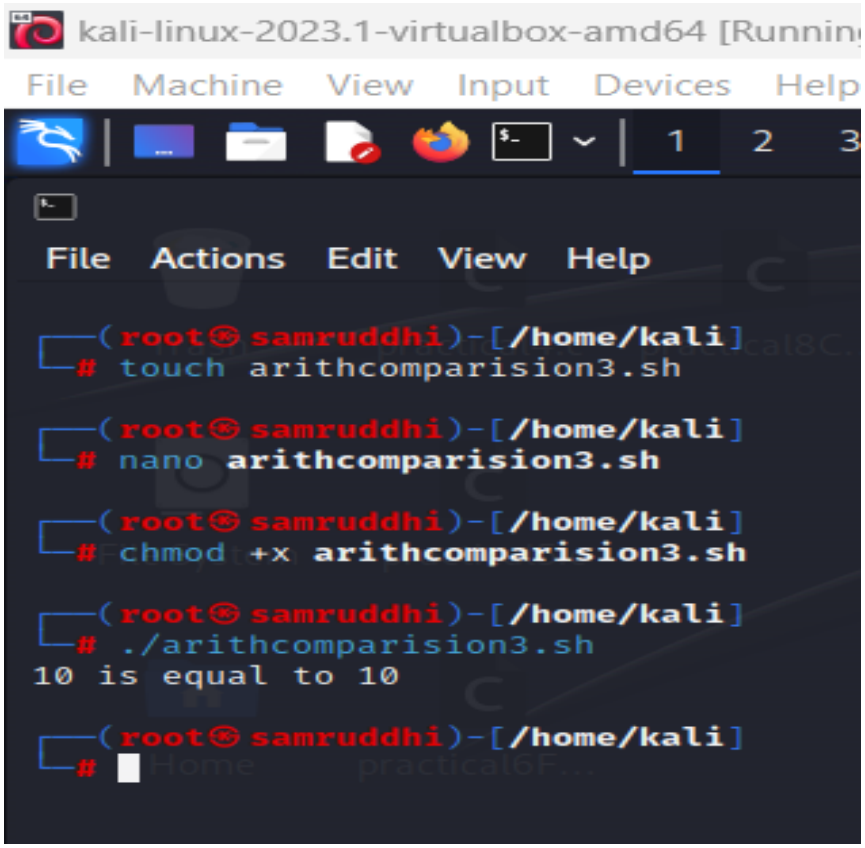




3) = = (Equal To) Operator

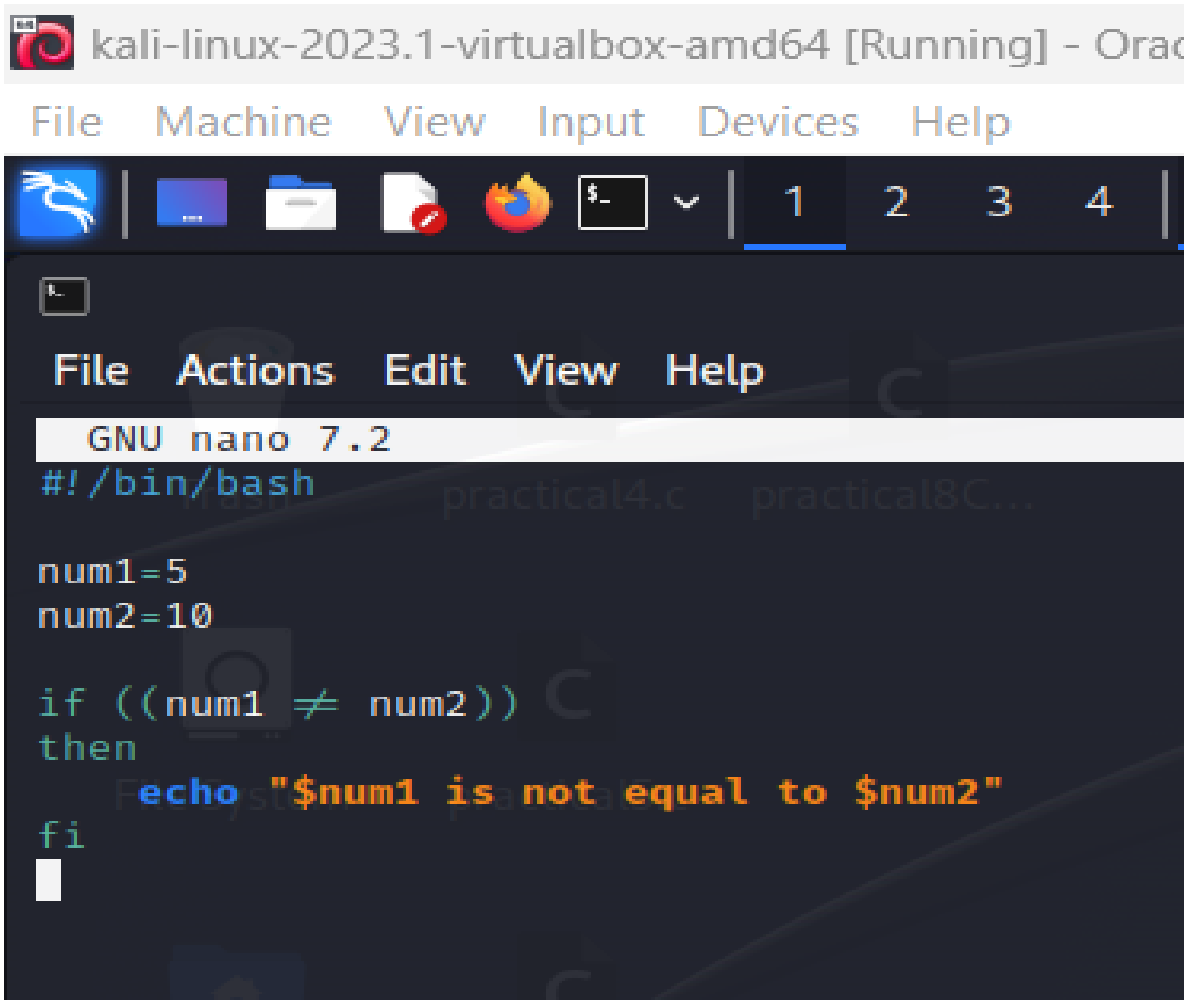
Example

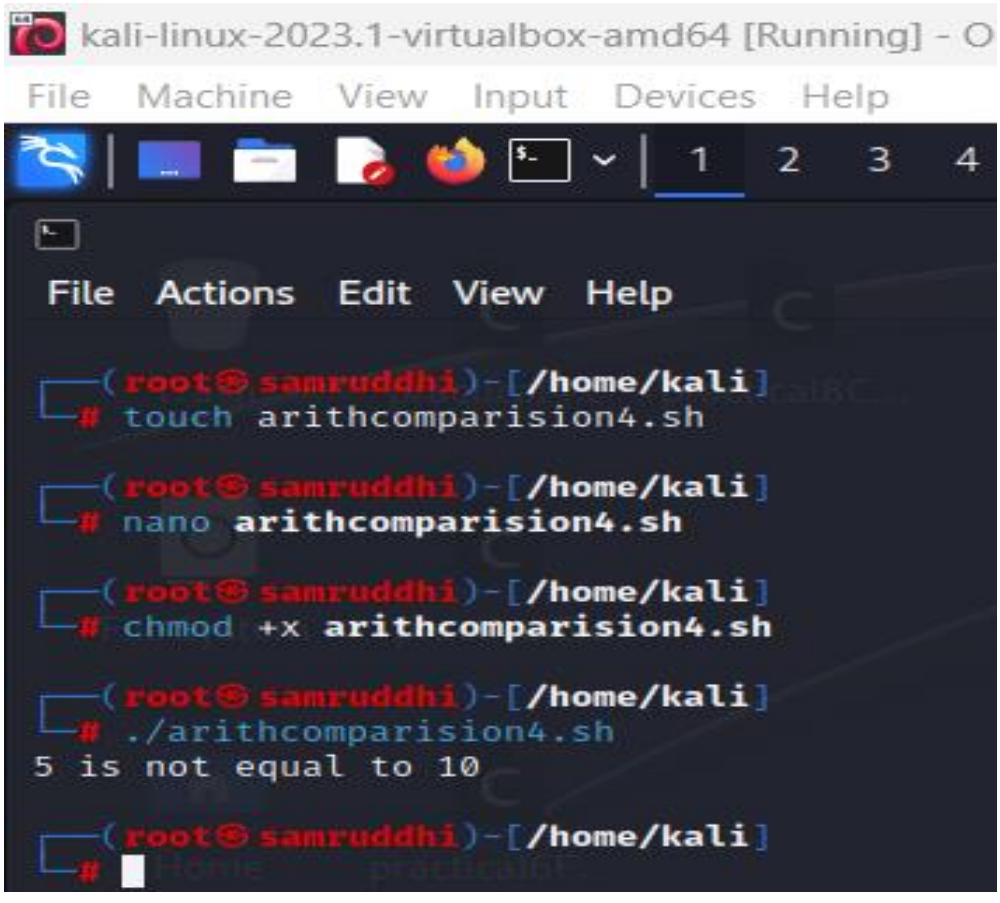




4) =! (Not Equal To) Operator

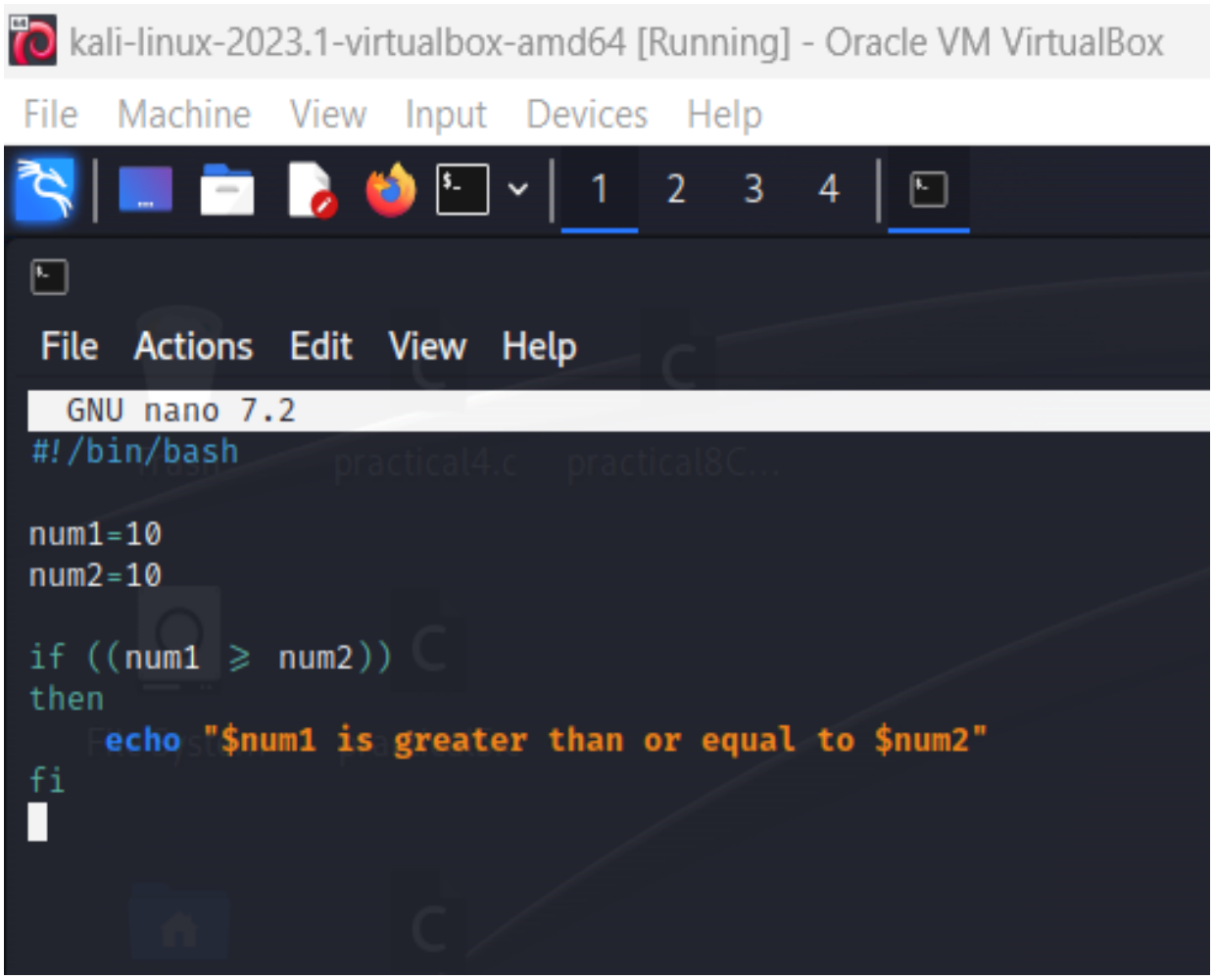
Example

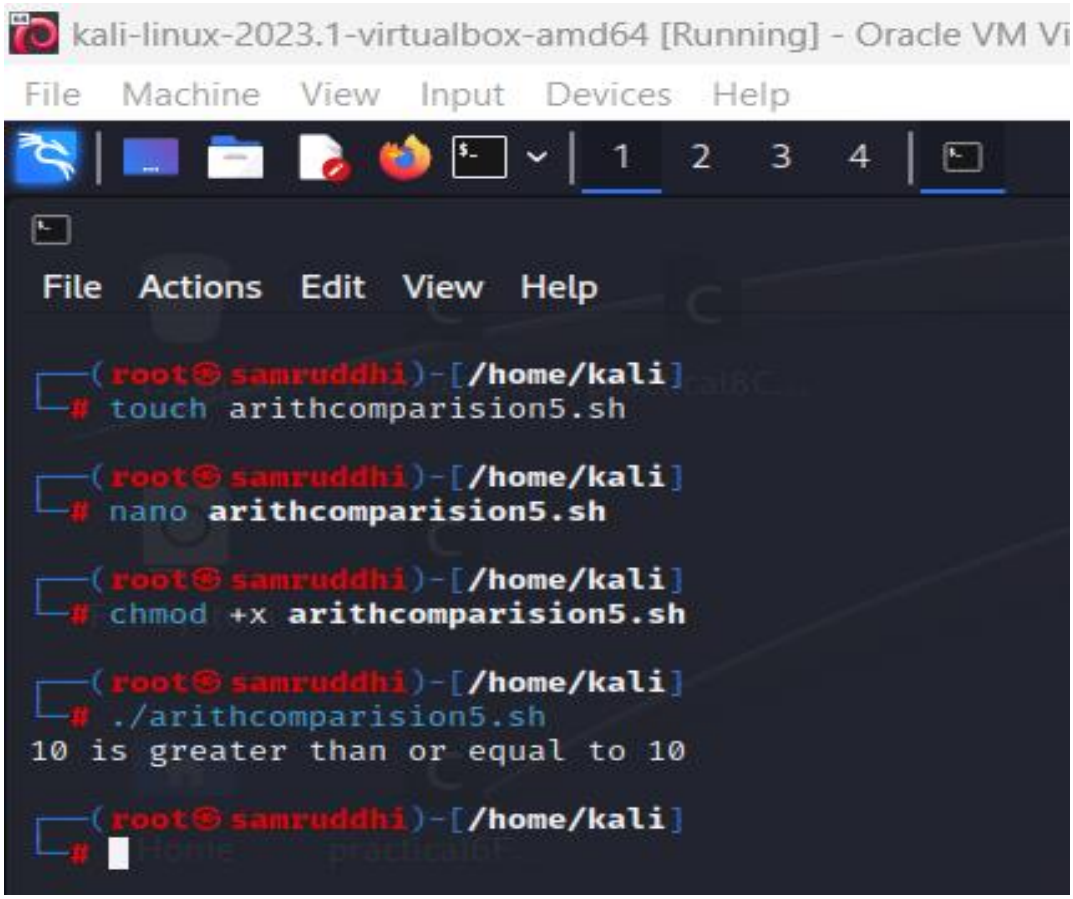




5) >= (Greater Than Equal) Operator

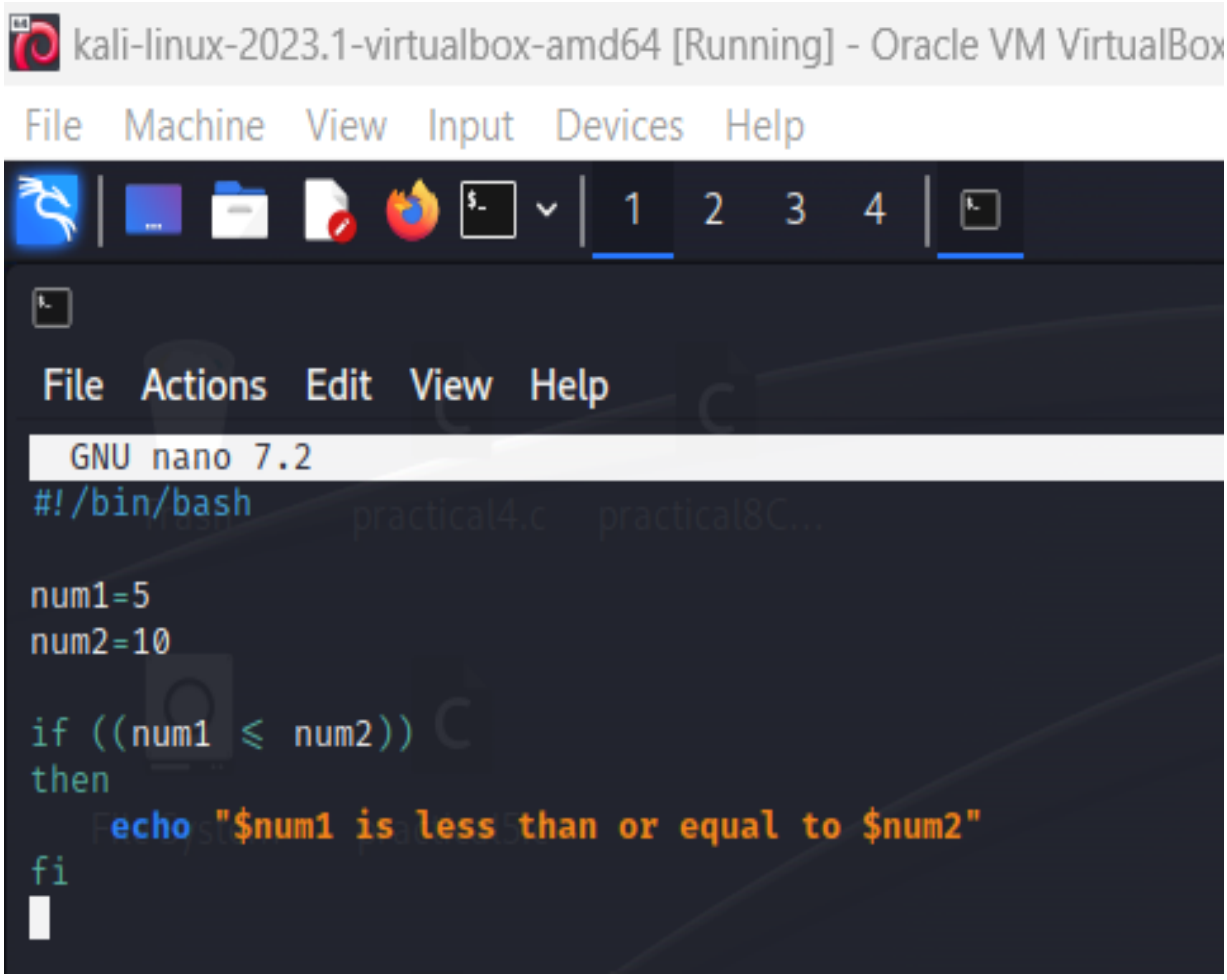
Example

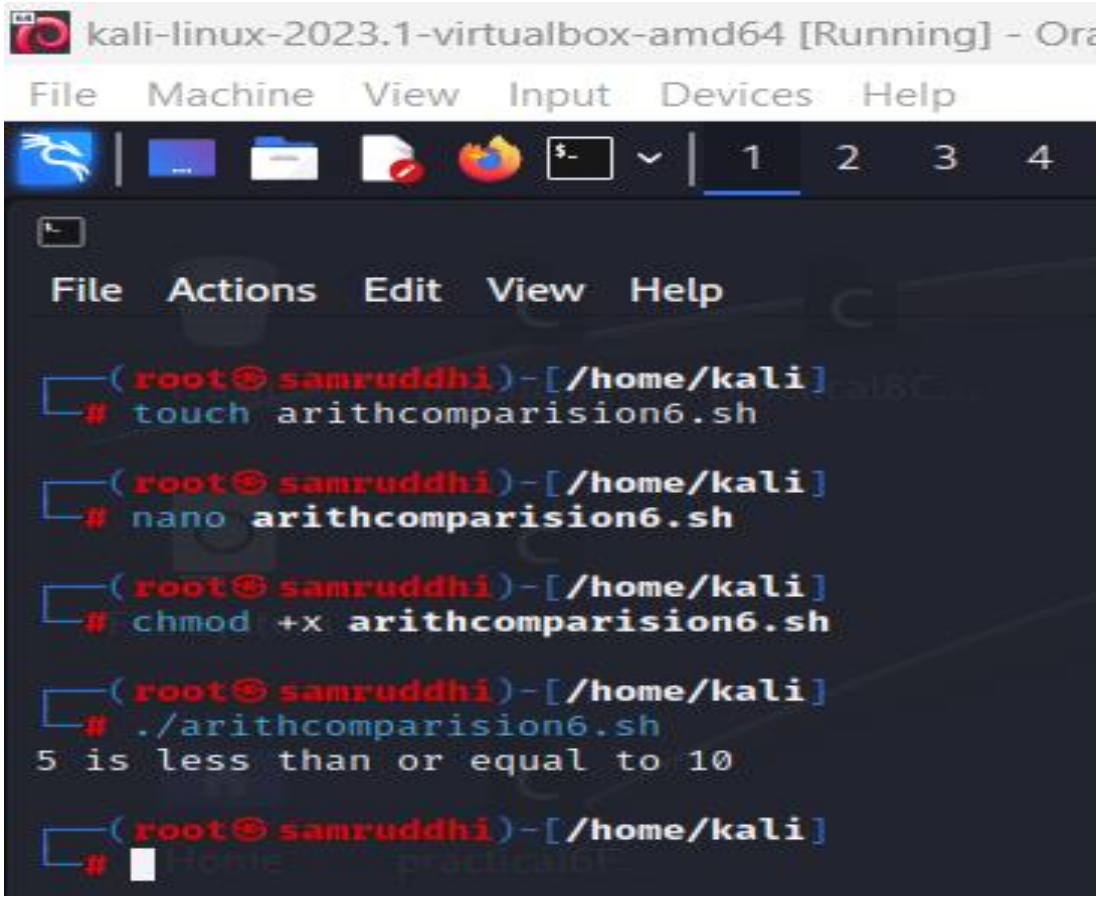




6) <= (Less Than Equal) Operator

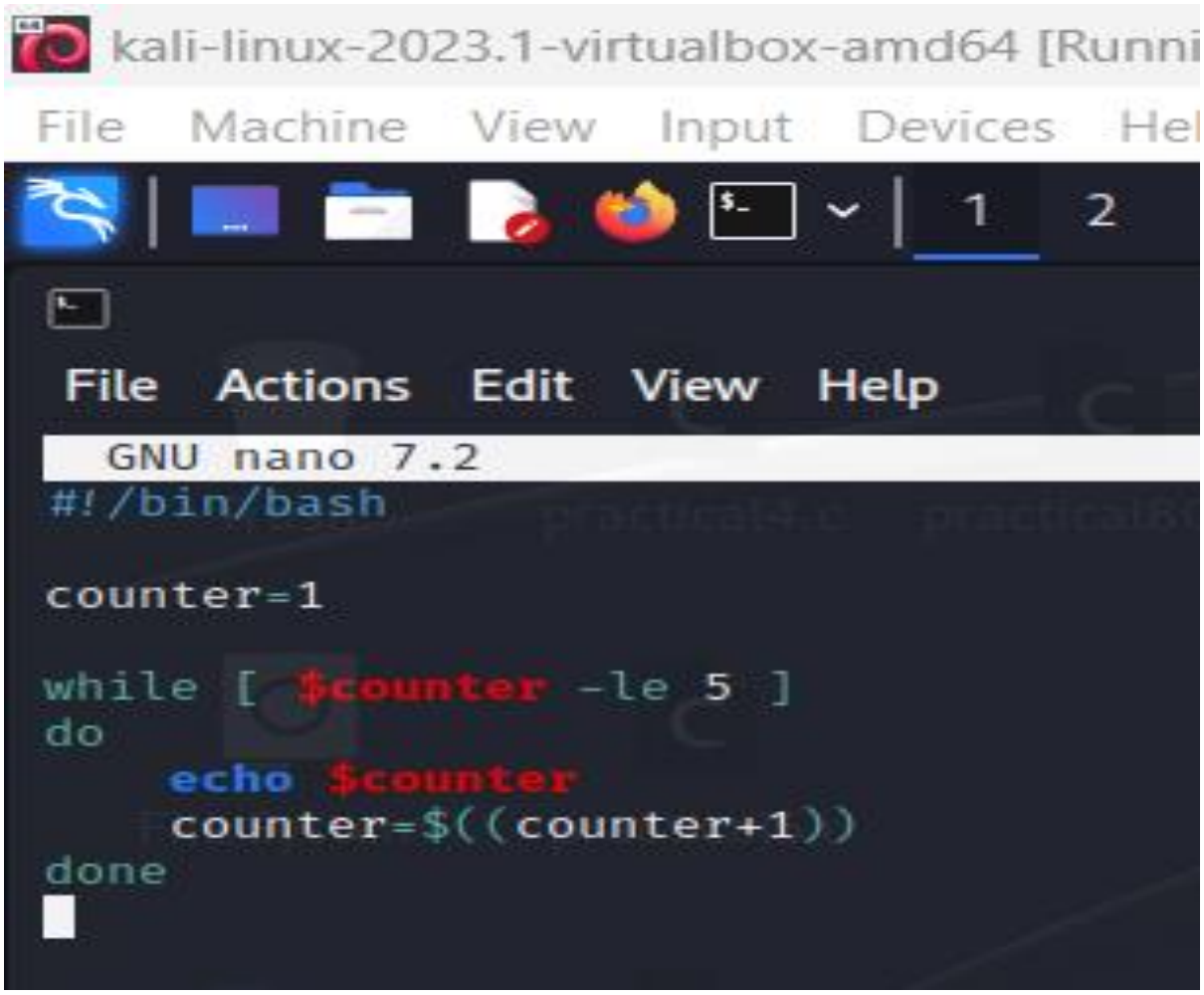
Example

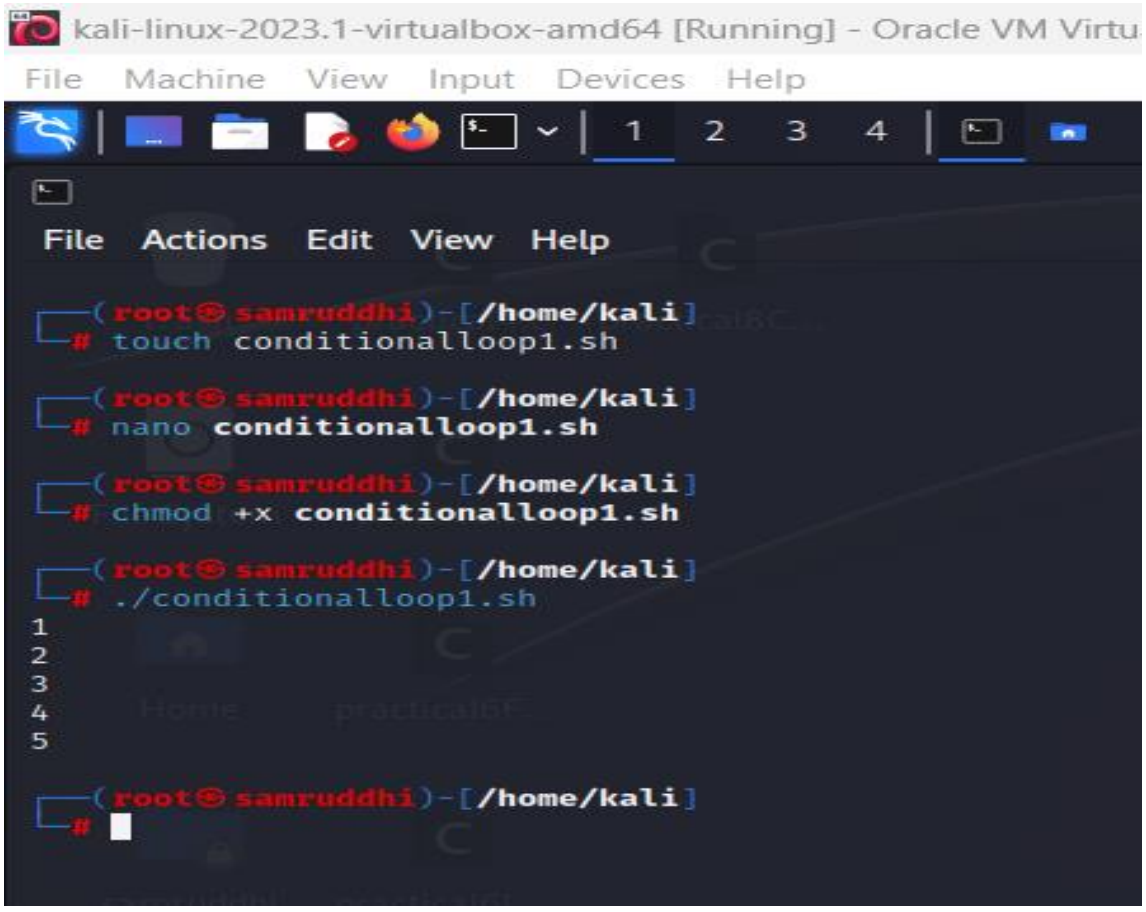




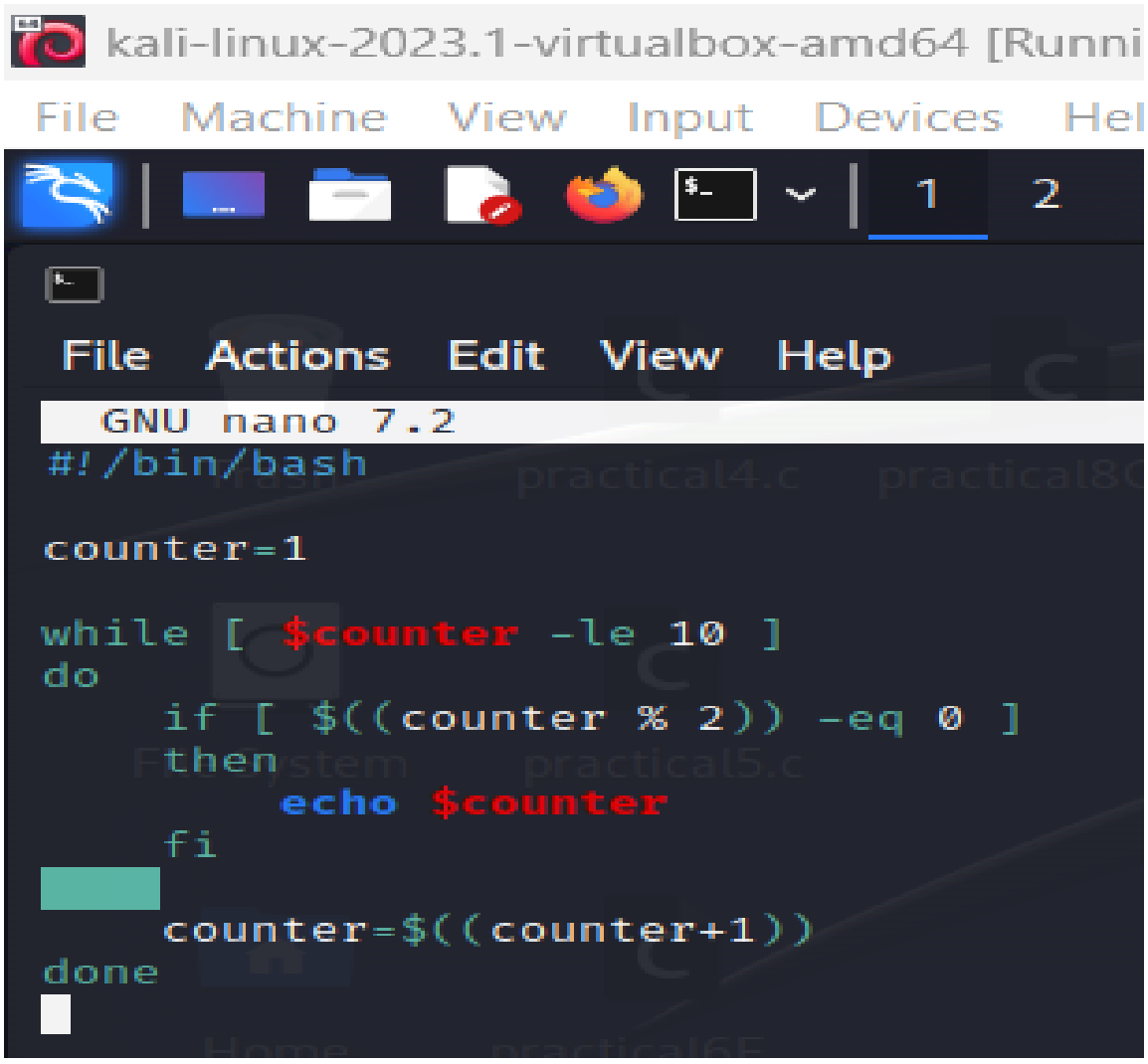
conditional loops:

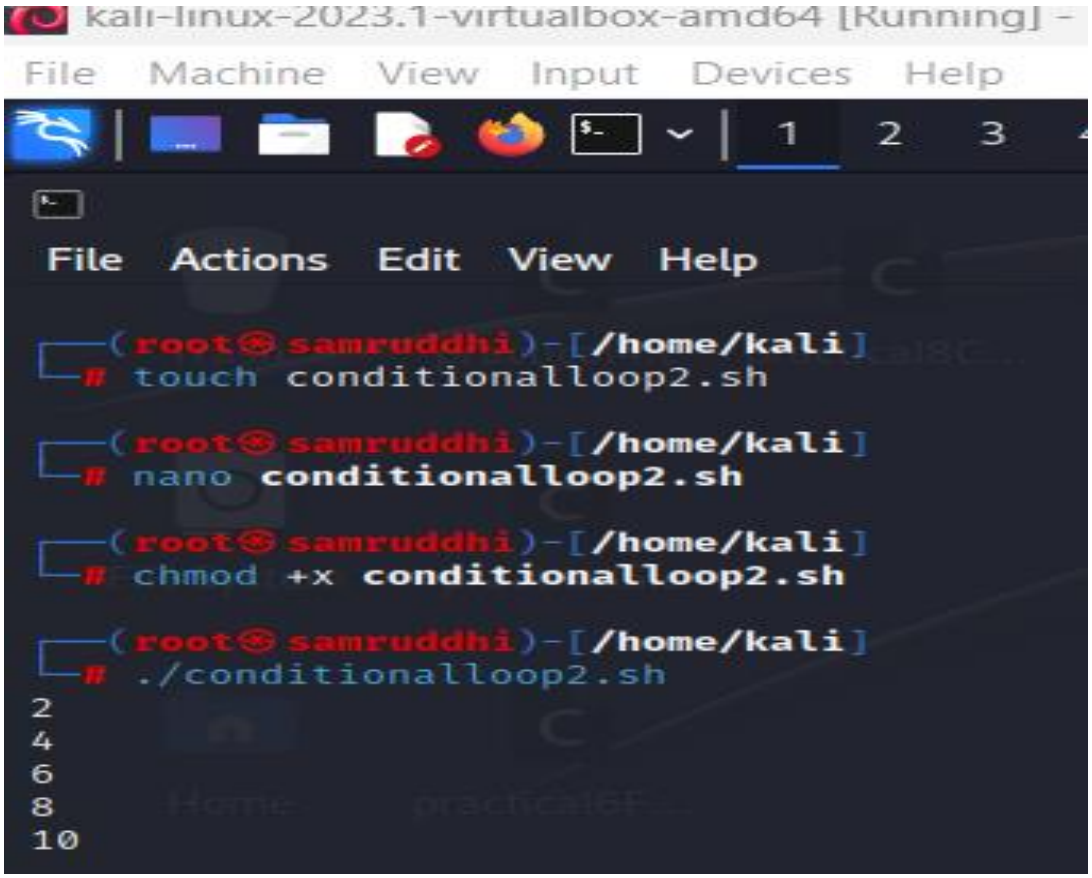
Example 1



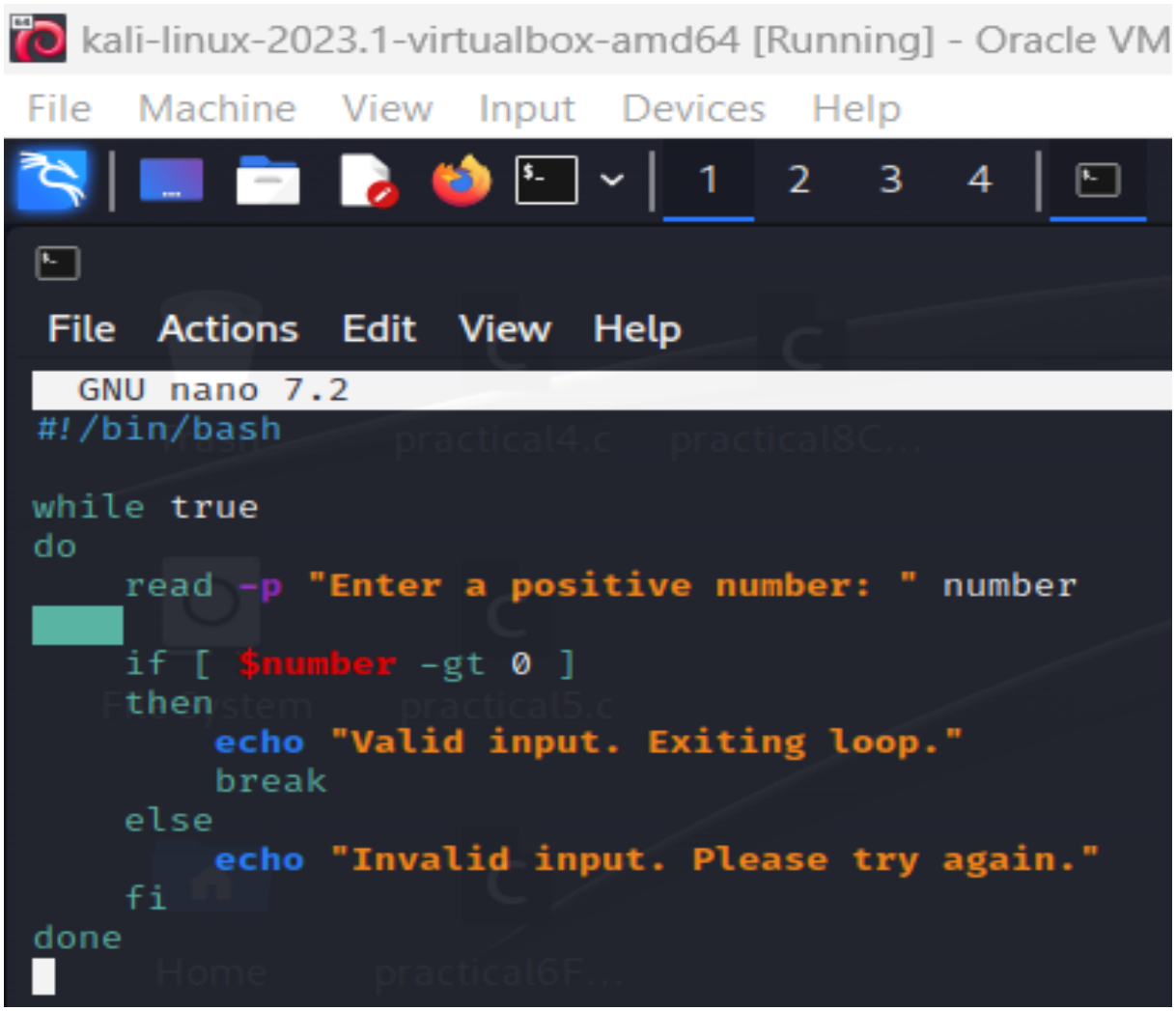


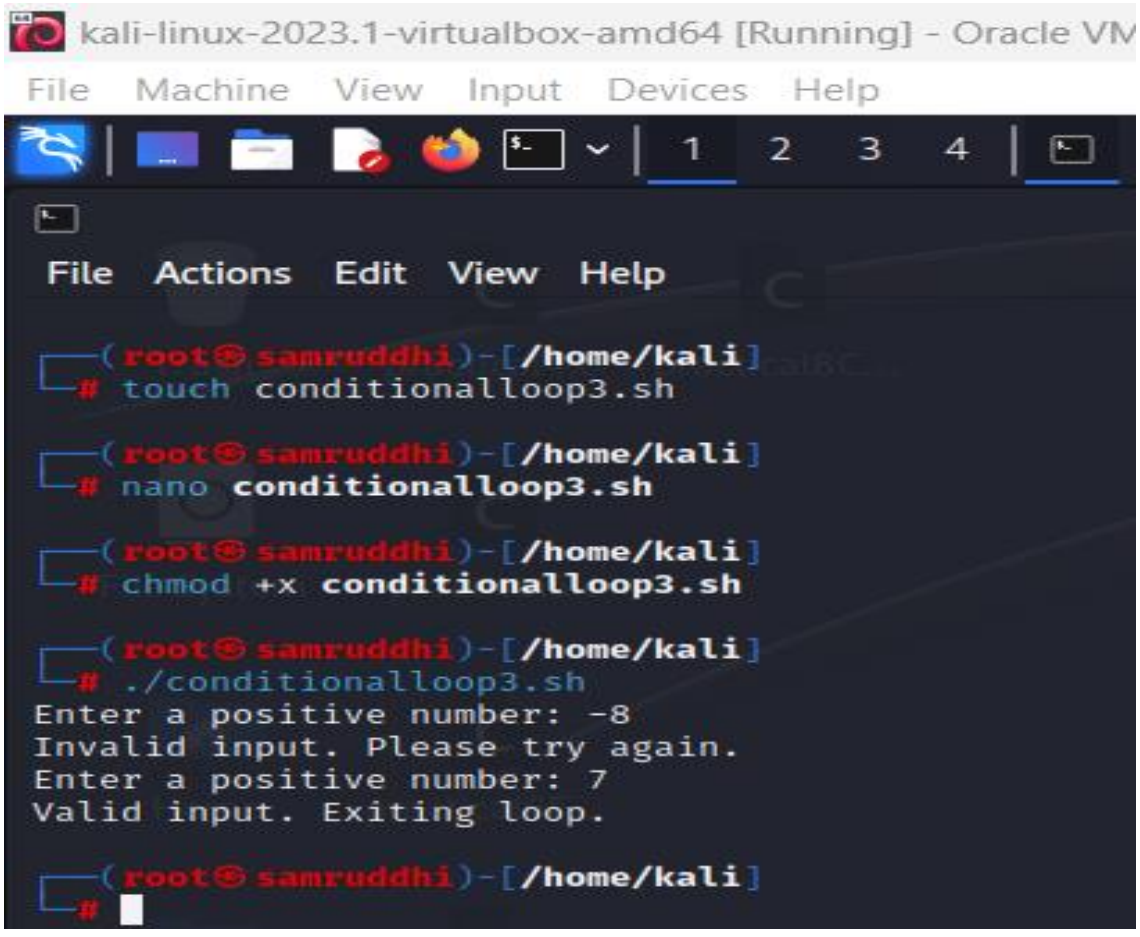
Example 2


kali-linux-2023.1-virtualbox-amd64 [Runni
File Machine View Input Devices Hel
1 2
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
counter=1
while [$counter -le 10]
do
if [$((counter % 2)) -eq 0]
then
echo $counter
fi
counter=$((counter+1))
done

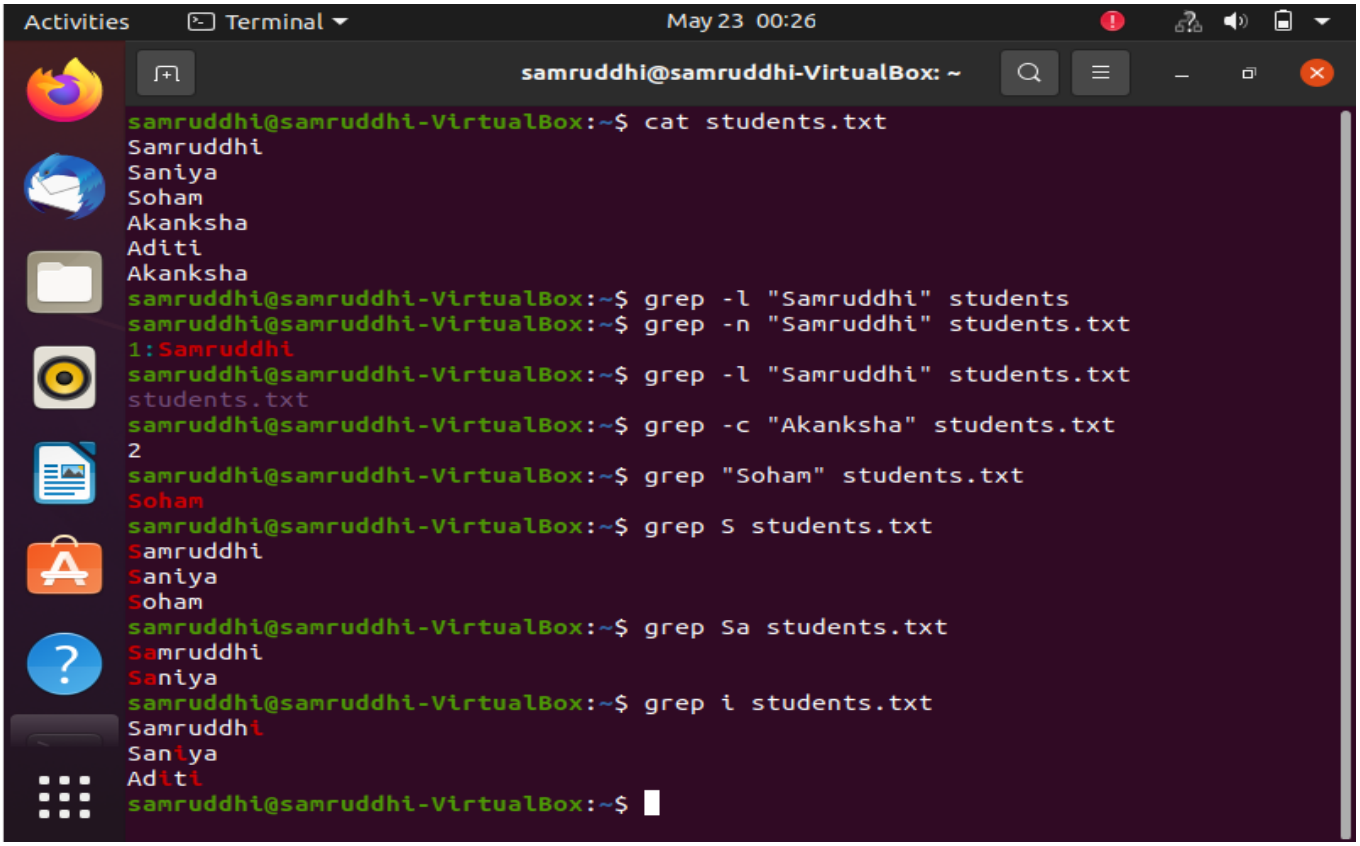

kali-linux-2023.1-virtualbox-amd64 [Running] -
File Machine View Input Devices Help
1 2 3
File Actions Edit View Help
(root㉿samruddhi)-[/home/kali]
touch conditionalloop2.sh
(root㉿samruddhi)-[/home/kali]
nano conditionalloop2.sh
(root㉿samruddhi)-[/home/kali]
chmod +x conditionalloop2.sh
(root㉿samruddhi)-[/home/kali]
./conditionalloop2.sh
2
4
6
8
10

Example 3

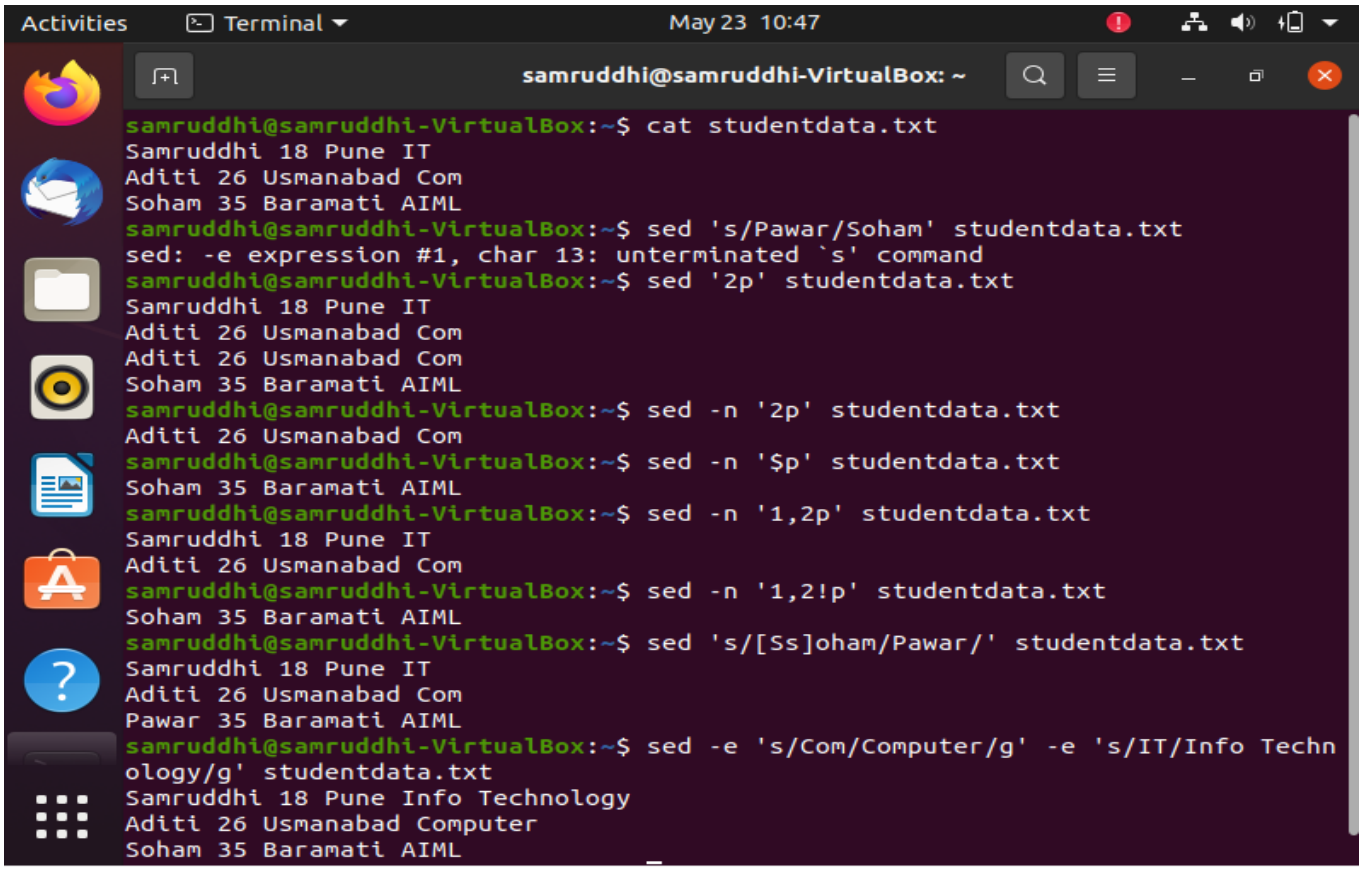

kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
GNU nano 7.2
#!/bin/bash
while true
do
read -p "Enter a positive number: " number
if [$number -gt 0]
then
echo "Valid input. Exiting loop."
break
else
echo "Invalid input. Please try again."
fi
done


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(root㉿samruddhi)-[/home/kali]
touch conditionalloop3.sh
(root㉿samruddhi)-[/home/kali]
nano conditionalloop3.sh
(root㉿samruddhi)-[/home/kali]
chmod +x conditionalloop3.sh
(root㉿samruddhi)-[/home/kali]
./conditionalloop3.sh
Enter a positive number: -8
Invalid input. Please try again.
Enter a positive number: 7
Valid input. Exiting loop.
(root㉿samruddhi)-[/home/kali]
#

grep command:



sed command:



PRACTICAL NO. 2

Process control system calls: The demonstration of FORK, EXECVE and WAIT system calls along with zombie and orphan states.

a. Implement the C program in which main program accepts the integers to be sorted. Main program uses the FORK system call to create a new process called a child process. Parent process sorts the integers using sorting algorithm and waits for child process using WAIT system call to sort the integers using any sorting algorithm. Also demonstrate zombie and orphan states.

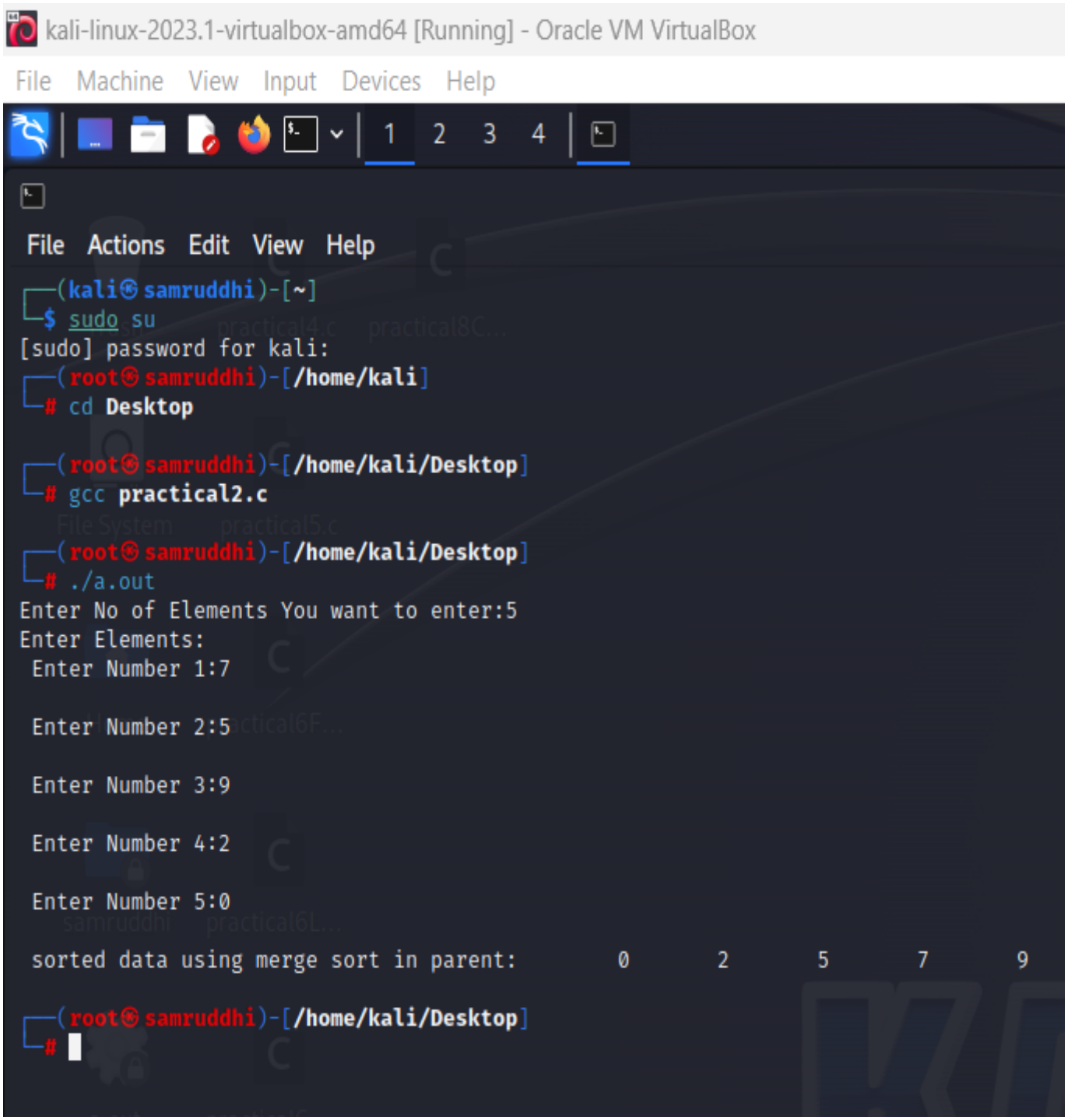
CODE:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/wait.h>

#define MAX 30

void merge_sort(int[],int,int);
void merge(int[],int,int,int);

int main()
{
        pid_t pid;
        int a[10],b[10],n,i;
        printf("Enter No of Elements You want to enter:");
        scanf("%d",&n);
        printf("Enter Elements:");
        for(i=0;i<n;i++)
        {
                printf("\n Enter Number %d:",i+1);
                scanf("%d",&a[i]);
                b[i]=a[i];
        }
        merge_sort(a,0,n-1);
```

```
        printf("\n sorted data using merge sort in parent:");
        for(i=0;i<n;i++)
        {
                printf("\t%d",a[i]);
        }
}
void merge_sort(int a[],int i,int j)
{
        int k;
        if(i<j)
        {
                k=(i+j)/2;
                merge_sort(a,i,k);
                merge_sort(a,k+1,j);
                merge(a,i,k,j);
        }
}
void merge(int a[],int l,int m,int u)
{
        int c[MAX];
        int i,j,k;
        i=l;
        j=m+1;
        k=0;
        while(i<=m && j<=u)
        {
                if(a[i]<a[j])
                {
                        c[k]=a[i];
                        k++;
                        i++;
```

```
            }
            else
            {
                c[k]=a[j];
                k++;
                j++;
            }
        }
        while(i<=m)
        {
            c[k]=a[i];
            i++;
            k++;
        }
        while(j<=u)
        {
            c[k]=a[j];
            k++;
            j++;
        }
        for(i=l,j=0;i<=u;i++,j++)
        a[i]=c[j];
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical2.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter No of Elements You want to enter:5
Enter Elements:
Enter Number 1:7
Enter Number 2:5
Enter Number 3:9
Enter Number 4:2
Enter Number 5:0
sorted data using merge sort in parent: 0 2 5 7 9
(root㉿samruddhi)-[/home/kali/Desktop]
#

PRACTICAL NO. 3

Implement the C program for CPU Scheduling Algorithms: Shortest Job First (Pre-emptive) and Round Robin with different arrival time.

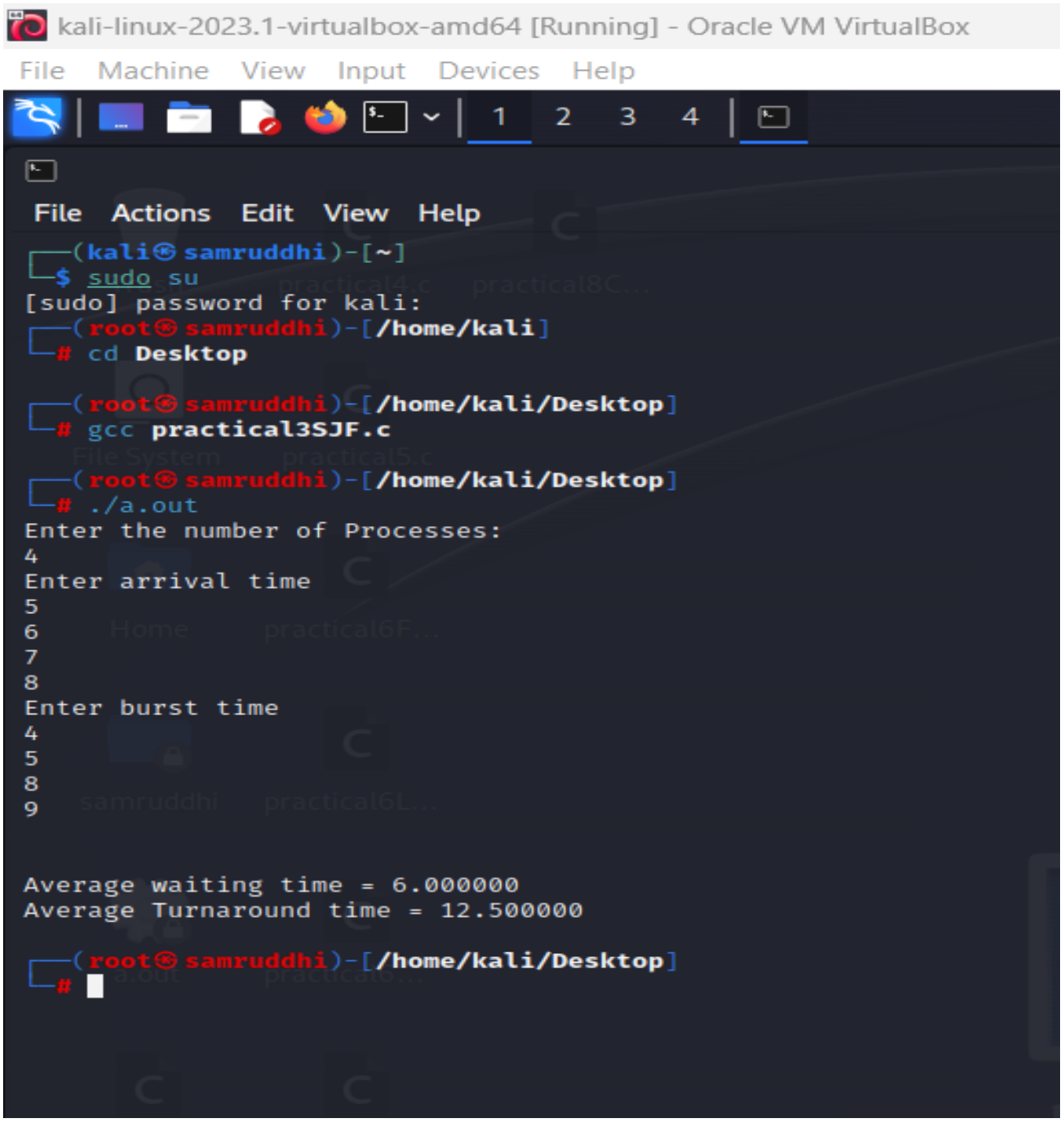
SHORTEST JOB FIRST (PRE-EMPTIVE)

CODE:

```
#include <stdio.h>
int main()
{
	int a[10],b[10],x[10],i,j,smallest,count=0,time,n;
	double avg=0,tt=0,end;
	printf("Enter the number of Processes:\n");
	scanf("%d",&n);
	printf("Enter arrival time\n");
	for(i=0;i<n;i++)
	scanf("%d",&a[i]);
	printf("Enter burst time\n");
	for(i=0;i<n;i++)
	scanf("%d",&b[i]);
	for(i=0;i<n;i++)
	x[i]=b[i];
	b[9]=9999;
	for(time=0;count!=n;time++)
	{
		smallest=9;
		for(i=0;i<n;i++)
		{
			if(a[i]<=time && b[i]<b[smallest] && b[i]>0 )
			smallest=i;
		}
		b[smallest]--;
```

```
            if(b[smallest]==0)
            {
                count++;
                end=time+1;
                avg=avg+end-a[smallest]-x[smallest];
                tt= tt+end-a[smallest];
            }
        }
        printf("\n\nAverage waiting time = %lf\n",avg/n);
        printf("Average Turnaround time = %lf",tt/n);
        return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical3SJF.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter the number of Processes:
4
Enter arrival time
5
6
7
8
Enter burst time
4
5
8
9
Average waiting time = 6.000000
Average Turnaround time = 12.500000
(root㉿samruddhi)-[/home/kali/Desktop]
#

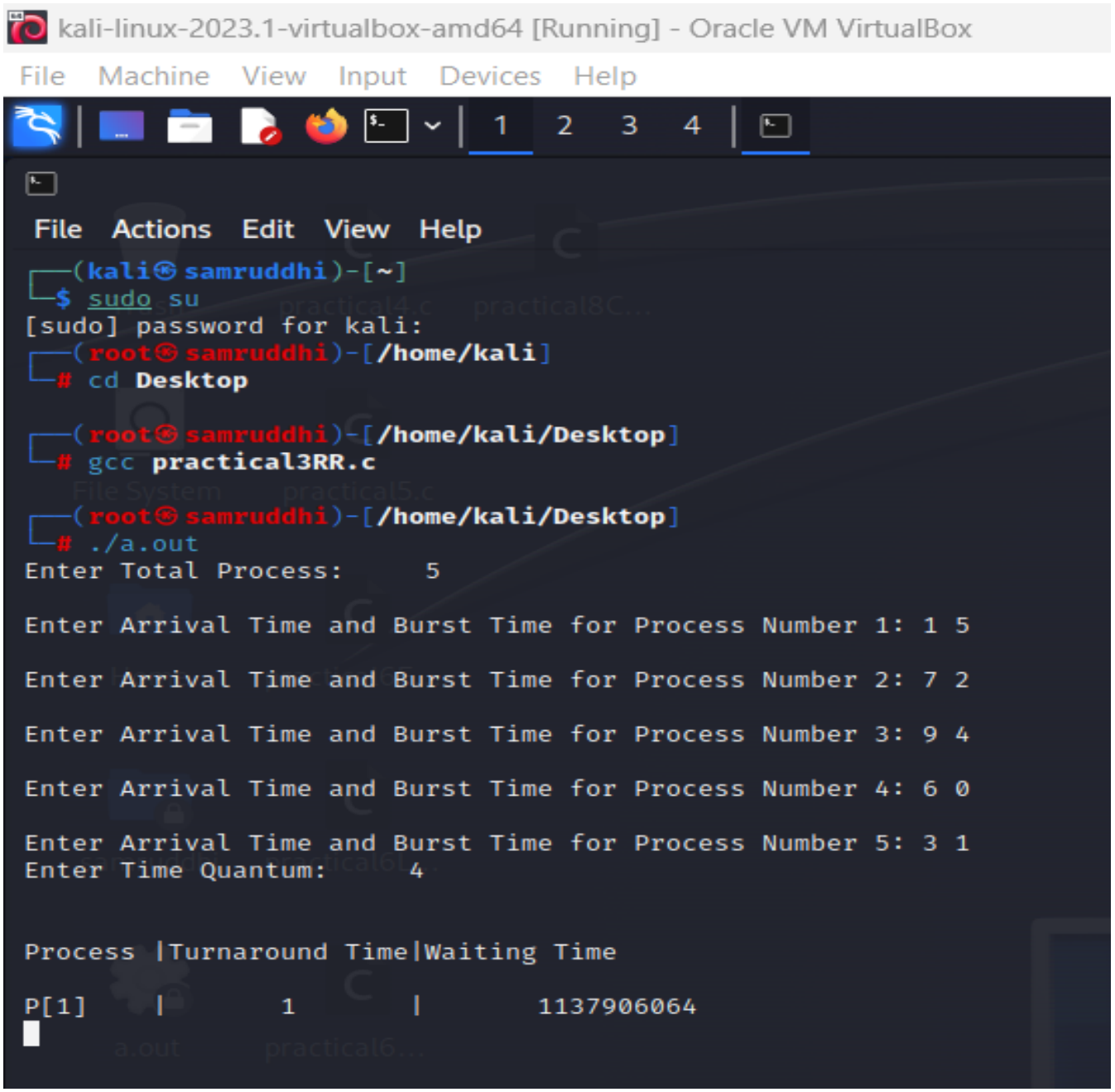
ROUND ROBIN

CODE:

```
#include <stdio.h>


int main()
{
   int cnt, j, n, t, remain, flag = 0, tq;
   float wt = 0, tat = 0, at[10], bt[10], rt[10];
   printf("Enter Total Process:\t ");
   scanf("%d", &n);
   remain = n;
   for (cnt = 0; cnt < n; cnt++)
   {
      printf("\nEnter Arrival Time and Burst Time for Process Number %d: ", cnt + 1);
      scanf("%f", &at[cnt]);
      scanf("%f", &bt[cnt]);
      rt[cnt] = bt[cnt];
   }
   printf("Enter Time Quantum:\t");
   scanf("%d", &tq);
   printf("\n\nProcess\t|Turnaround Time|Waiting Time\n\n");
   for (t = 0, cnt = 0; remain != 0;)
   {
      if (rt[cnt] <= tq && rt[cnt] > 0)
      {
         t += rt[cnt];
         rt[cnt] = 0;
         flag = 1;
      }
      else if (rt[cnt] > 0)
      {
```

```
            rt[cnt] -= tq;
            t += tq;
        }
        if (rt[cnt] == 0 && flag == 1)
        {
            remain--;
            printf("P[%d]\t|\t%d\t|\t%d\n", cnt + 1, t - at[cnt], t - at[cnt] - bt[cnt]);
            wt += t - at[cnt] - bt[cnt];
            tat += t - at[cnt];
            flag = 0;
        }
        if (cnt == n - 1)
            cnt = 0;
        else if (at[cnt + 1] <= t)
            cnt++;
        else
            cnt = 0;
    }
    printf("\nAverage Waiting Time= %.2f\n", wt / n);
    printf("Average Turnaround Time = %.2f", tat / n);
    return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical3RR.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter Total Process: 5
Enter Arrival Time and Burst Time for Process Number 1: 1 5
Enter Arrival Time and Burst Time for Process Number 2: 7 2
Enter Arrival Time and Burst Time for Process Number 3: 9 4
Enter Arrival Time and Burst Time for Process Number 4: 6 0
Enter Arrival Time and Burst Time for Process Number 5: 3 1
Enter Time Quantum: 4
Process |Turnaround Time|Waiting Time
P[1] | 1 | 1137906064

PRACTICAL NO. 4

a. Thread synchronization using counting semaphores. Application to demonstrate: producer-consumer problem with counting semaphores and mutex.

CODE:

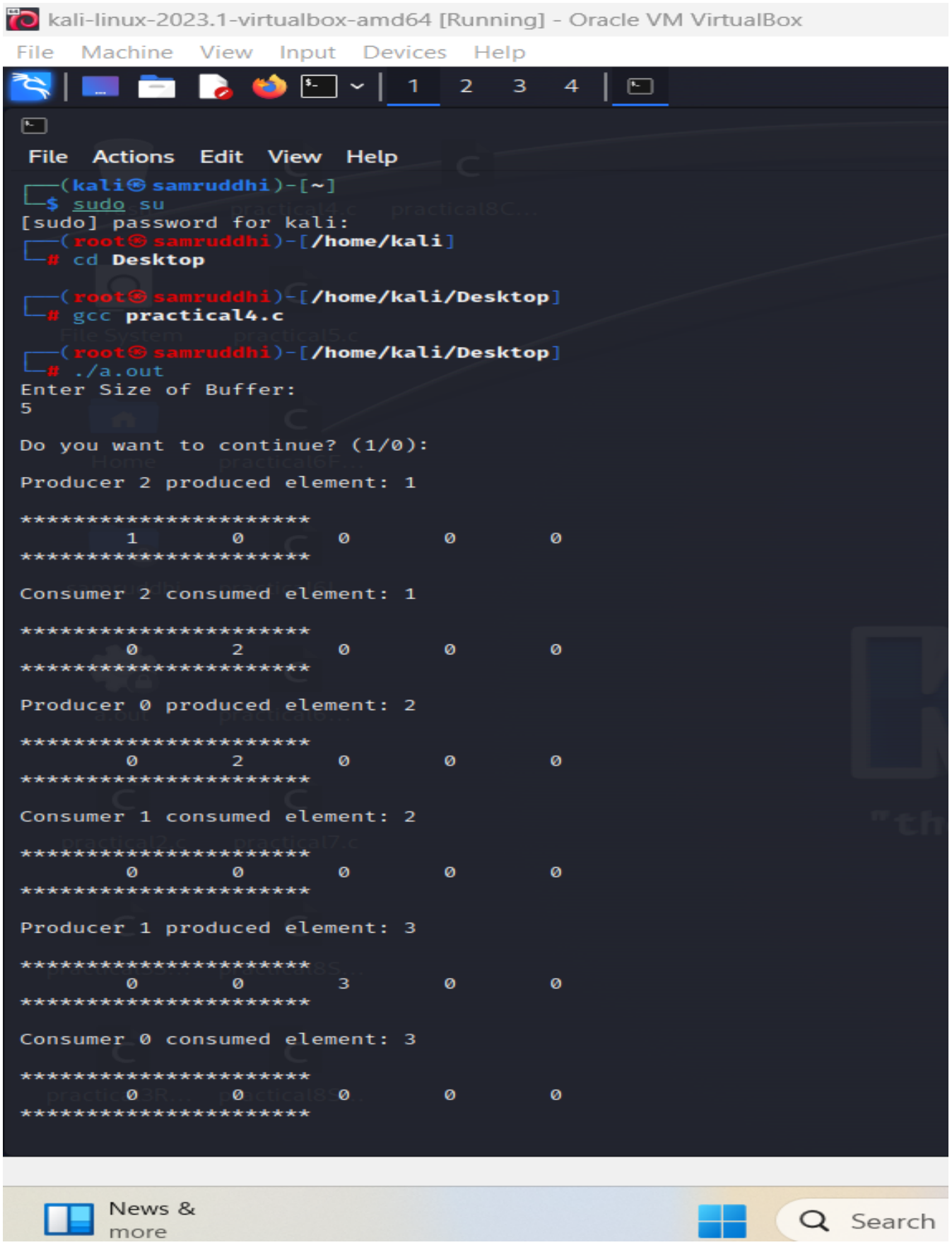
```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
#include <unistd.h>

sem_t mutex, full, empty;
int *buffer, front = 0, rear = 0, count = 1;
int sizeOfBuffer;
int stopProducing = 0;
int stopConsuming = 0;

void *producer(void *p);
void *consumer(void *p);

int main(int argc, char *argv[]) {
   int choice;
   int gProduce = 3;
   int gConsume = 3;
   int i;

   printf("Enter Size of Buffer:\n");
   scanf("%d", &sizeOfBuffer);

   buffer = (int *)malloc(sizeOfBuffer * sizeof(int));
```

```
sem_init(&mutex, 0, 1);
 sem_init(&full, 0, 0);
 sem_init(&empty, 0, sizeOfBuffer);

 pthread_t tid_p[gProduce];
 pthread_t tid_c[gConsume];

 while (1) {
    for (i = 0; i < gProduce; i++) {
       pthread_create(&tid_p[i], NULL, producer, (void *)(intptr_t)i);
    }

    for (i = 0; i < gConsume; i++) {
       pthread_create(&tid_c[i], NULL, consumer, (void *)(intptr_t)i);
    }

    printf("\nDo you want to continue? (1/0):\n");
    scanf("%d", &choice);

    if (choice == 0)
       break;
 }

 stopProducing = 1;
 stopConsuming = 1;

 for (i = 0; i < gProduce; i++) {
    pthread_join(tid_p[i], NULL);
 }

 for (i = 0; i < gConsume; i++) {
```

```
        pthread_join(tid_c[i], NULL);
    }

    free(buffer);
    sem_destroy(&mutex);
    sem_destroy(&full);
    sem_destroy(&empty);

    return 0;
}

void *producer(void *p) {
    int t = (int)(intptr_t)p;

    while (!stopProducing) {
        sem_wait(&empty);
        sem_wait(&mutex);

        buffer[rear] = count;
        printf("\nProducer %d produced element: %d\n", t, buffer[rear]);
        rear = (rear + 1) % sizeOfBuffer;
        count++;

        sem_post(&mutex);
        sem_post(&full);

        printf("\n**********************\n");
        for (int i = 0; i < sizeOfBuffer; i++)
            printf("\t%d", buffer[i]);
        printf("\n**********************\n");
```

```
        sleep(1);
    }

    pthread_exit(0);
}

void *consumer(void *p) {
    int t = (int)(intptr_t)p;

    while (!stopConsuming) {
        sem_wait(&full);
        sem_wait(&mutex);

        printf("\nConsumer %d consumed element: %d\n", t, buffer[front]);
        buffer[front] = 0;
        front = (front + 1) % sizeOfBuffer;

        sem_post(&mutex);
        sem_post(&empty);

        printf("\n**********************\n");
        for (int i = 0; i < sizeOfBuffer; i++)
            printf("\t%d", buffer[i]);
        printf("\n**********************\n");

        sleep(1);
    }

    pthread_exit(0);
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical4.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter Size of Buffer:
5
Do you want to continue? (1/0):
Producer 2 produced element: 1
****
1 0 0 0 0
****
Consumer 2 consumed element: 1
****
0 2 0 0 0
****
Producer 0 produced element: 2
****
0 2 0 0 0
****
Consumer 1 consumed element: 2
****
0 0 0 0 0
****
Producer 1 produced element: 3
****
0 0 3 0 0
****
Consumer 0 consumed element: 3
****
0 0 0 0 0
****
News & more
Search

PRACTICAL NO. 5

Implement the C program for Deadlock Avoidance Algorithm: Bankers Algorithm.

BANKERS ALGORITHM

CODE:

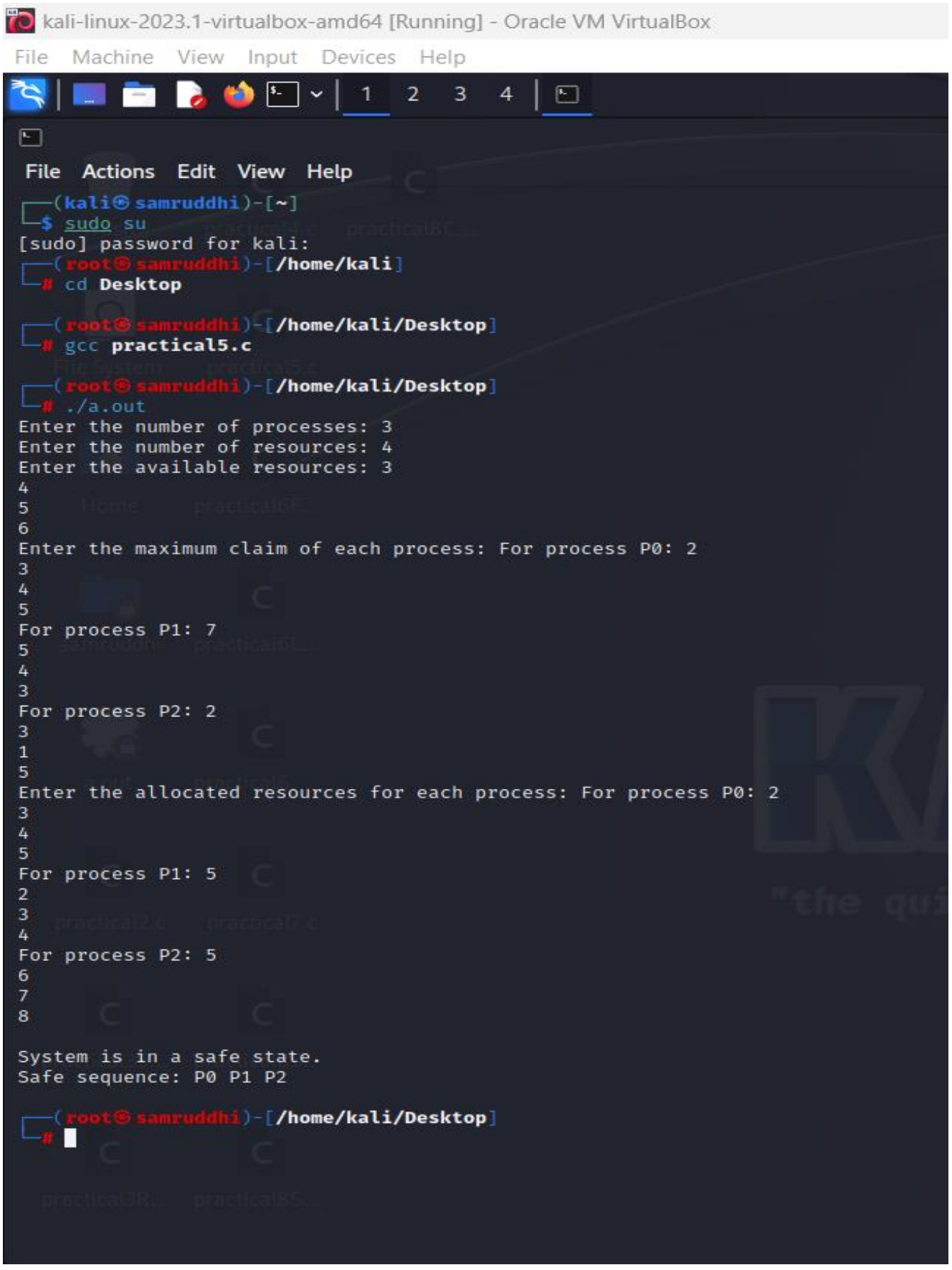
```
#include <stdio.h>
#define MAX_PROCESSES 10
#define MAX_RESOURCES 10
int main()
{
   int num_processes, num_resources;
   int available[MAX_RESOURCES];
   int max_claim[MAX_PROCESSES][MAX_RESOURCES];
   int allocation[MAX_PROCESSES][MAX_RESOURCES];
   int need[MAX_PROCESSES][MAX_RESOURCES];
   int work[MAX_RESOURCES];
   int finish[MAX_PROCESSES];
   printf("Enter the number of processes: ");
   scanf("%d", &num_processes);
   printf("Enter the number of resources: ");
   scanf("%d", &num_resources);
   printf("Enter the available resources: ");
   for (int i = 0; i < num_resources; i++)
   {
        scanf("%d", &available[i]);
   }
   printf("Enter the maximum claim of each process: ");
   for (int i = 0; i < num_processes; i++)
   {
        printf("For process P%d: ", i);
        for (int j = 0; j < num_resources; j++)
```

```
        {
            scanf("%d", &max_claim[i][j]);
        }
    }
    printf("Enter the allocated resources for each process: ");
    for (int i = 0; i < num_processes; i++)
    {
        printf("For process P%d: ", i);
        for (int j = 0; j < num_resources; j++)
        {
            scanf("%d", &allocation[i][j]);
            need[i][j] = max_claim[i][j] - allocation[i][j];
        }
        finish[i] = 0;
    }
    for (int i = 0; i < num_resources; i++)
    {
        work[i] = available[i];
    }
    int count = 0;
    int safe_sequence[MAX_PROCESSES];
    while (count < num_processes)
    {
        int found = 0;
        for (int i = 0; i < num_processes; i++)
        {
            if (finish[i] == 0)
            {
                int j;
                for (j = 0; j < num_resources; j++)
                {
```

```
            if (need[i][j] > work[j])
            {
                break;
            }
        }
        if (j == num_resources)
        {
            for (int k = 0; k < num_resources; k++)
            {
                work[k] += allocation[i][k];
            }
            safe_sequence[count++] = i;
            finish[i] = 1;
            found = 1;
        }
    }
    if (found == 0)
    {
        break;
    }
}
if (count < num_processes)
{
    printf("\nSystem is in an unsafe state. Deadlock detected.\n");
}
else
{
    printf("\nSystem is in a safe state.\nSafe sequence: ");
    for (int i = 0; i < num_processes; i++)
    {
```

```
            printf("P%d ", safe_sequence[i]);
        }
        printf("\n");
    }
    return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical5.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter the number of processes: 3
Enter the number of resources: 4
Enter the available resources: 3
4
5
6
Enter the maximum claim of each process: For process P0: 2
3
4
5
For process P1: 7
5
4
3
For process P2: 2
3
1
5
Enter the allocated resources for each process: For process P0: 2
3
4
5
For process P1: 5
2
3
4
For process P2: 5
6
7
8
System is in a safe state.
Safe sequence: P0 P1 P2
(root㉿samruddhi)-[/home/kali/Desktop]
#

PRACTICAL NO. 6

Implement the C program for Page Replacement Algorithms: FCFS, LRU, and Optimal for frame size as minimum three.

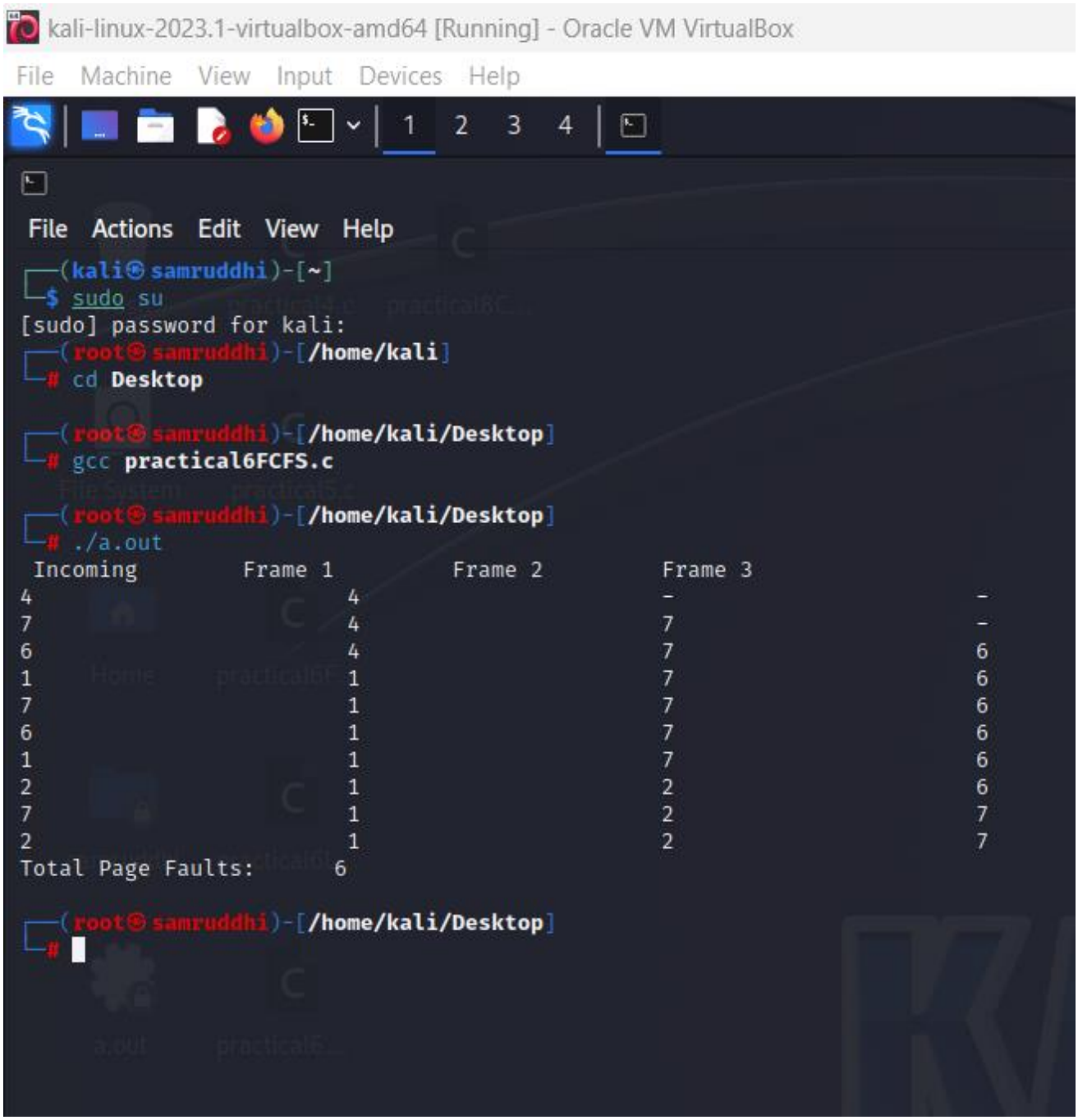
1) <u>FCFS</u>

CODE:

```
#include<stdio.h>
int main()
{
	int incomingStream[] = {4,7,6,1,7,6,1,2,7,2};
	int pageFaults = 0;
	int frames = 3;
	int m, n, s, pages;
	pages = sizeof(incomingStream)/sizeof(incomingStream[0]);
	printf(" Incoming \t Frame 1 \t Frame 2 \t Frame 3 ");
	int temp[ frames ];
	for(m = 0; m < frames; m++)
	{
		temp[m] = -1;
	}
	for(m = 0; m < pages; m++)
	{
		s = 0;
		for(n = 0; n < frames; n++)
		{
			if(incomingStream[m] == temp[n])
			{
				s++;
				pageFaults--;
			}
		}
```

```
                pageFaults++;
                if((pageFaults <= frames) && (s == 0))
                {
                        temp[m] = incomingStream[m];
                }
                else if(s == 0)
                {
                        temp[(pageFaults - 1) % frames] = incomingStream[m];
                }
                printf("\n");
                printf("%d\t\t\t",incomingStream[m]);
                for(n = 0; n < frames; n++)
                {
                        if(temp[n] != -1)
                        printf(" %d\t\t\t", temp[n]);
                        else
                        printf(" - \t\t\t");
                }
        }
        printf("\nTotal Page Faults:\t%d\n", pageFaults);
        return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical6FCFS.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Incoming Frame 1 Frame 2 Frame 3
4 4 - -
7 4 7 -
6 4 7 6
1 1 7 6
7 1 7 6
6 1 7 6
1 1 7 6
2 1 2 6
7 1 2 7
2 1 2 7
Total Page Faults: 6
(root㉿samruddhi)-[/home/kali/Desktop]
#

2) LRU

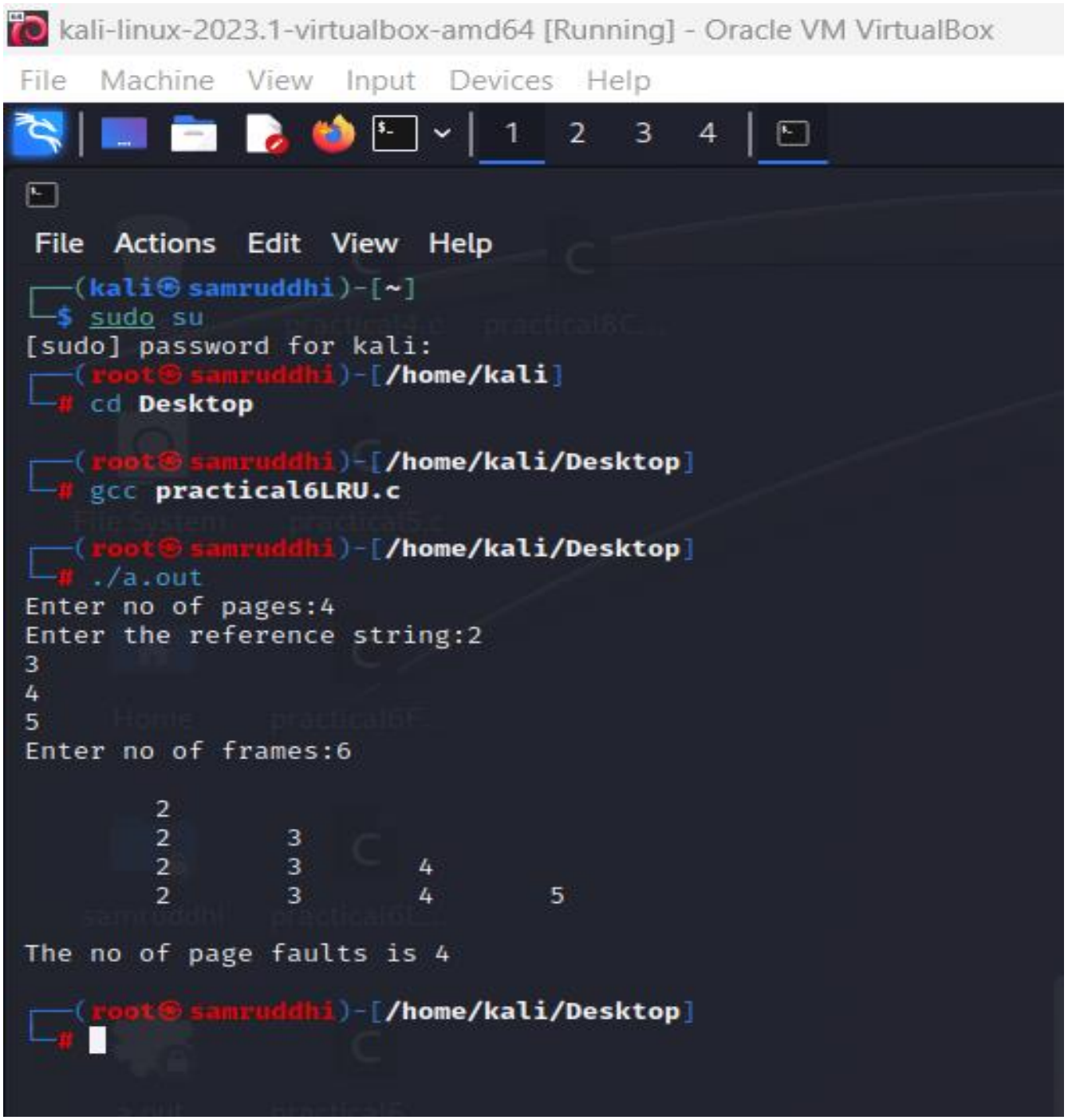
CODE:

```
#include<stdio.h>
int main()
{
	int q[20],p[50],c=0,c1,d,f,i,j,k=0,n,r,t,b[20],c2[20];
	printf("Enter no of pages:");
	scanf("%d",&n);
	printf("Enter the reference string:");
	for(i=0;i<n;i++)
	scanf("%d",&p[i]);
	printf("Enter no of frames:");
	scanf("%d",&f);
	q[k]=p[k];
	printf("\n\t%d\n",q[k]);
	c++;
	k++;
	for(i=1;i<n;i++)
	{
		c1=0;
		for(j=0;j<f;j++)
		{
			if(p[i]!=q[j])
			c1++;
		}
		if(c1==f)
		{
			c++;
			if(k<f)
			{
				q[k]=p[i];
```

```
            k++;
            for(j=0;j<k;j++)
            printf("\t%d",q[j]);
            printf("\n");
        }
        else
        {
            for(r=0;r<f;r++)
            {
                c2[r]=0;
                for(j=i-1;j<n;j--)
                {
                    if(q[r]!=p[j])
                    c2[r]++;
                    else
                    break;
                }
            }
            for(r=0;r<f;r++)
            b[r]=c2[r];
            for(r=0;r<f;r++)
            {
                for(j=r;j<f;j++)
                {
                    if(b[r]<b[j])
                    {
                        t=b[r];
                        b[r]=b[j];
                        b[j]=t;
                    }
                }
```

```
                        }
                        for(r=0;r<f;r++)
                        {
                                if(c2[r]==b[0])
                                q[r]=p[i];
                                printf("\t%d",q[r]);
                        }
                        printf("\n");
                }
        }
    }
    printf("\nThe no of page faults is %d",c);
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical6LRU.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter no of pages:4
Enter the reference string:2
3
4
5
Enter no of frames:6
2
2 3
2 3 4
2 3 4 5
The no of page faults is 4
(root㉿samruddhi)-[/home/kali/Desktop]
#

3) OPTIMAL

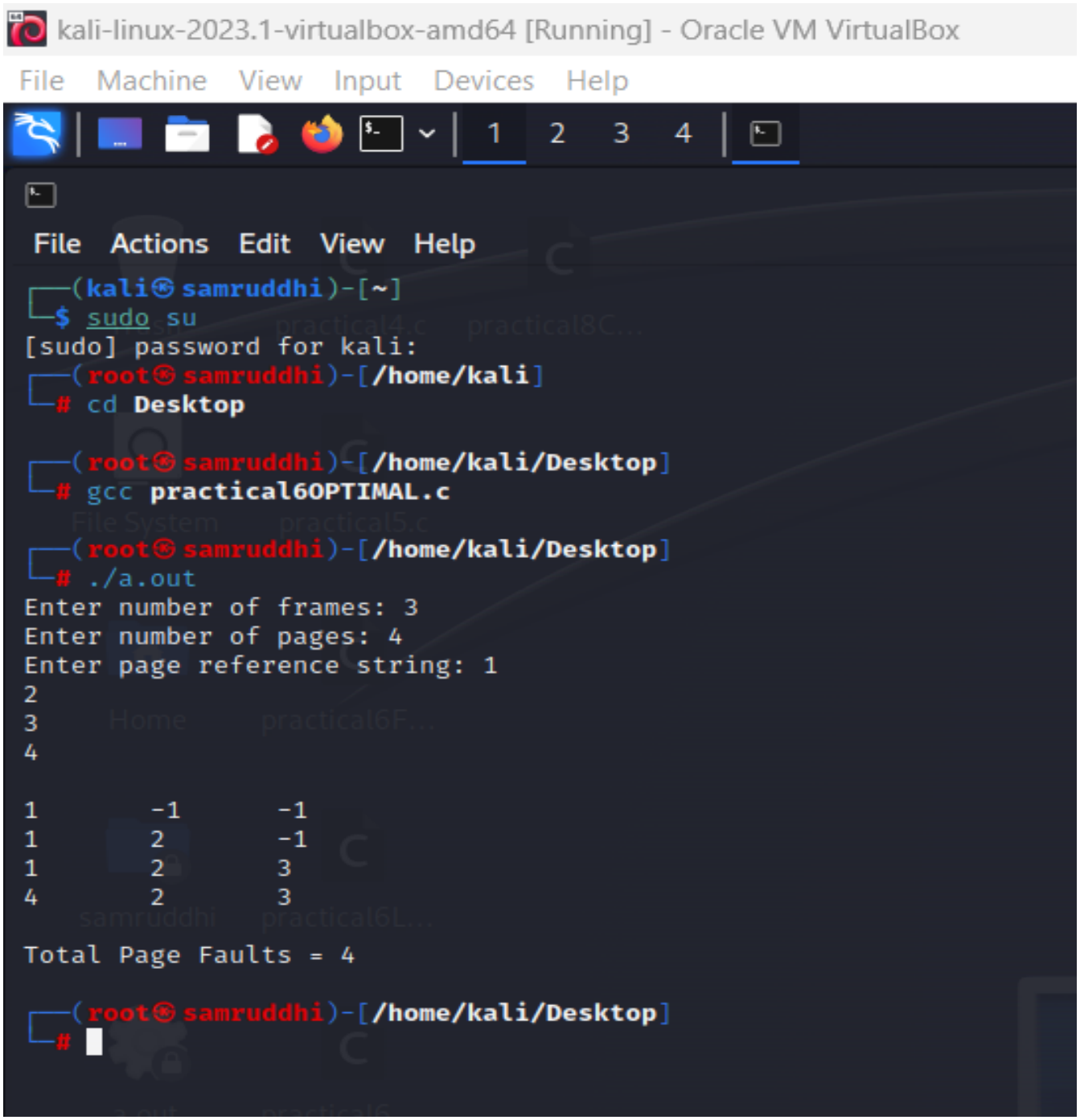
CODE:

```
#include<stdio.h>
int main()
{
 int no_of_frames, no_of_pages, frames[10], pages[30], temp[10];
 int flag1, flag2, flag3, i, j, k, pos, max, faults = 0;
 printf("Enter number of frames: ");
 scanf("%d", &no_of_frames);
 printf("Enter number of pages: ");
 scanf("%d", &no_of_pages);
 printf("Enter page reference string: ");
 for(i = 0; i < no_of_pages; ++i)
 {
        scanf("%d", &pages[i]);
 }
 for(i = 0; i < no_of_frames; ++i)
 {
        frames[i] = -1;
 }
 for(i = 0; i < no_of_pages; ++i)
 {
        flag1 = flag2 = 0;
        for(j = 0; j < no_of_frames; ++j)
        {
                if(frames[j] == pages[i])
                {
                        flag1 = flag2 = 1;
                        break;
                }
        }
```

```
if(flag1 == 0)
{
	for(j = 0; j < no_of_frames; ++j)
	{
		if(frames[j] == -1)
		{
			faults++;
			frames[j] = pages[i];
			flag2 = 1;
			break;
		}
	}
	if(flag2 == 0)
	{
		flag3 = 0;
		for(j = 0; j < no_of_frames; ++j)
		{
			temp[j] = -1;
		}
		for(k = i + 1; k < no_of_pages; ++k)
		{
			for(j = 0; j < no_of_frames; ++j)
			{
				if(frames[j] == pages[k])
				{
					temp[j] = k;
					break;
				}
			}
		}
		for(j = 0; j < no_of_frames; ++j)
```

```
                {
                        if(temp[j] == -1)
                        {
                                pos = j;
                                flag3 = 1;
                                break;
                        }
                }
                if(flag3 == 0)
                {
                        max = temp[0];
                        pos = 0;
                        for(j = 1; j < no_of_frames; ++j)
                        {
                                if(temp[j] > max)
                                {
                                        max = temp[j];
                                        pos = j;
                                }}}
                frames[pos] = pages[i];
                faults++;
        }
    }
    printf("\n");
    for(j = 0; j < no_of_frames; ++j)
    {
        printf("%d\t", frames[j]);
    } }
    printf("\n\nTotal Page Faults = %d", faults);
    return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical6OPTIMAL.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter number of frames: 3
Enter number of pages: 4
Enter page reference string: 1
2
3
4
1 -1 -1
1 2 -1
1 2 3
4 2 3
Total Page Faults = 4
(root㉿samruddhi)-[/home/kali/Desktop]
#

PRACTICAL NO. 7

Inter process communication in Linux using following

a. FIFOs: Full duplex communication between two independent processes. First process accepts sentences and writes on one pipe to be read by second process and second process counts number of characters, number of words and number of lines in accepted sentences, writes this output in a text file and writes the contents of the file on second pipe to be read by first process and displays on standard output.

FIFO

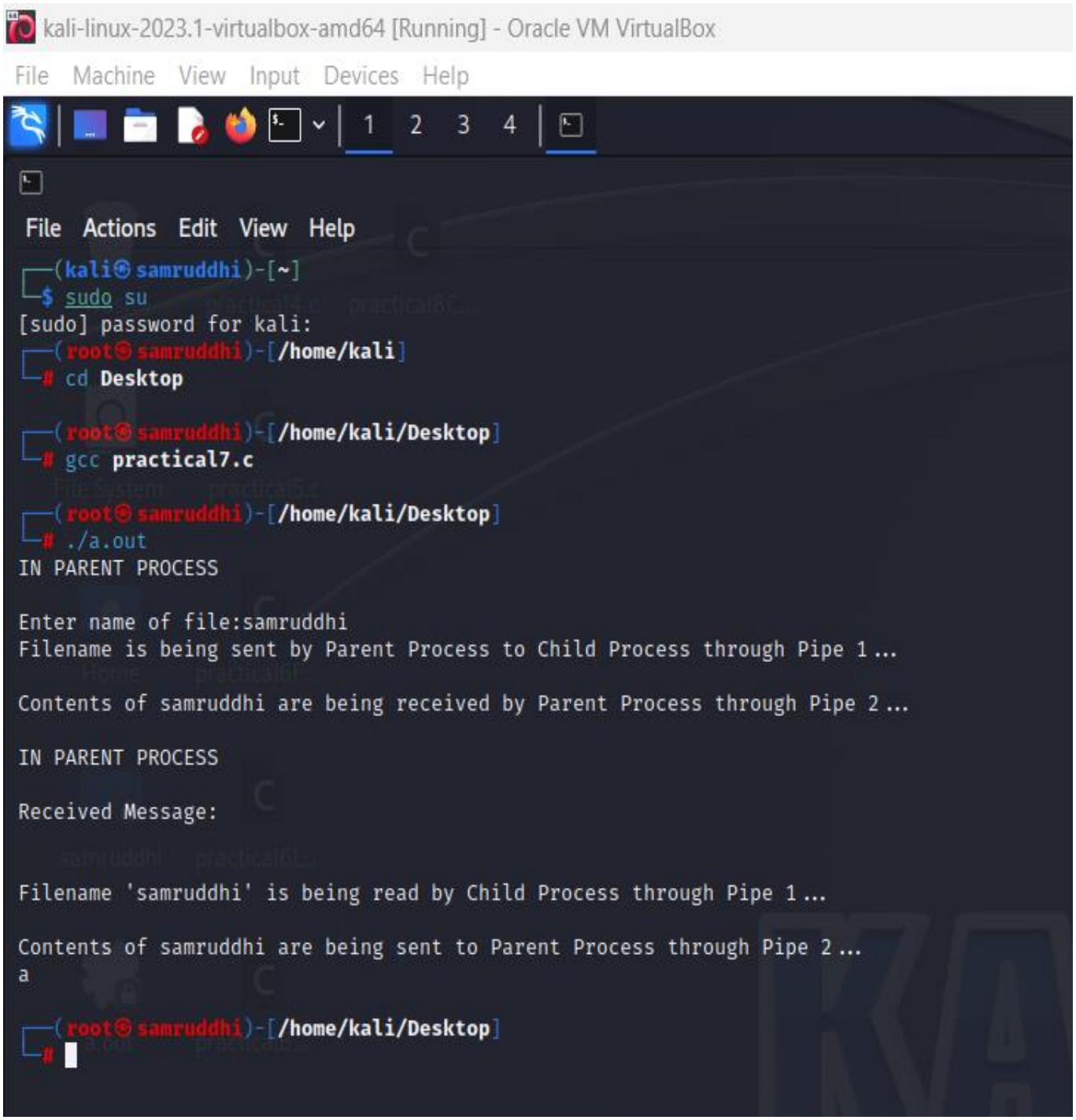
CODE:

```
#include<stdio.h>
#include<unistd.h>
#include<string.h>
#include<stdlib.h>
int main()
{
        int fd1[2], nbytes=1,fd2[2],a=0;
        pid_t pid;
        char string[80];
        char readbuffer[80];
        char ch='a',ch1='\n';
        FILE *fp;
        pipe(fd1);
        pipe(fd2);
        if((pid = fork()) == -1)
        {
                perror("fork");
                exit(1);
        }
        if(pid == 0)
        {
                close(fd1[1]);
```

```
            read(fd1[0], readbuffer,sizeof(readbuffer));
            printf("\nFilename '%s' is being read by Child Process through Pipe 1...\n",readbuffer);
            fp=fopen(readbuffer,"r");
            close(fd1[0]);
            close(fd2[0]);
            printf("\nContents of %s are being sent to Parent Process through Pipe 2...\n",readbuffer);
            while(a!=-1)
            {
                  a=fscanf(fp,"%c",&ch);
                  write(fd2[1], &ch,sizeof(ch));
            }
            close(fd2[1]);
            exit(0);
      }
      else
      {
            close(fd1[0]);
            printf("IN PARENT PROCESS\n" );
            printf("\nEnter name of file:");
            scanf("%s",string);
            printf("Filename is being sent by Parent Process to Child Process through Pipe 1...\n");
            write(fd1[1],string, (strlen(string)+1));
            close(fd1[1]);
            close(fd2[1]);
            printf("\nContents of %s are being received by Parent Process through Pipe 2...\n\n",string);
            printf("IN PARENT PROCESS\n" );
            printf("\nReceived Message:\n");
            while(nbytes!=0)
```

```
                {
                        printf("%c",ch1);
                        nbytes = read(fd2[0], &ch1, sizeof(ch1));
                }
                close(fd2[0]);
        }
        return(0);
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical7.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
IN PARENT PROCESS
Enter name of file:samruddhi
Filename is being sent by Parent Process to Child Process through Pipe 1 ...
Contents of samruddhi are being received by Parent Process through Pipe 2 ...
IN PARENT PROCESS
Received Message:
Filename 'samruddhi' is being read by Child Process through Pipe 1 ...
Contents of samruddhi are being sent to Parent Process through Pipe 2 ...
a
(root㉿samruddhi)-[/home/kali/Desktop]
#

PRACTICAL NO. 8

Implement the C program for Disk Scheduling Algorithms: SSTF, SCAN, C-Look considering the initial head position moving away from the spindle.

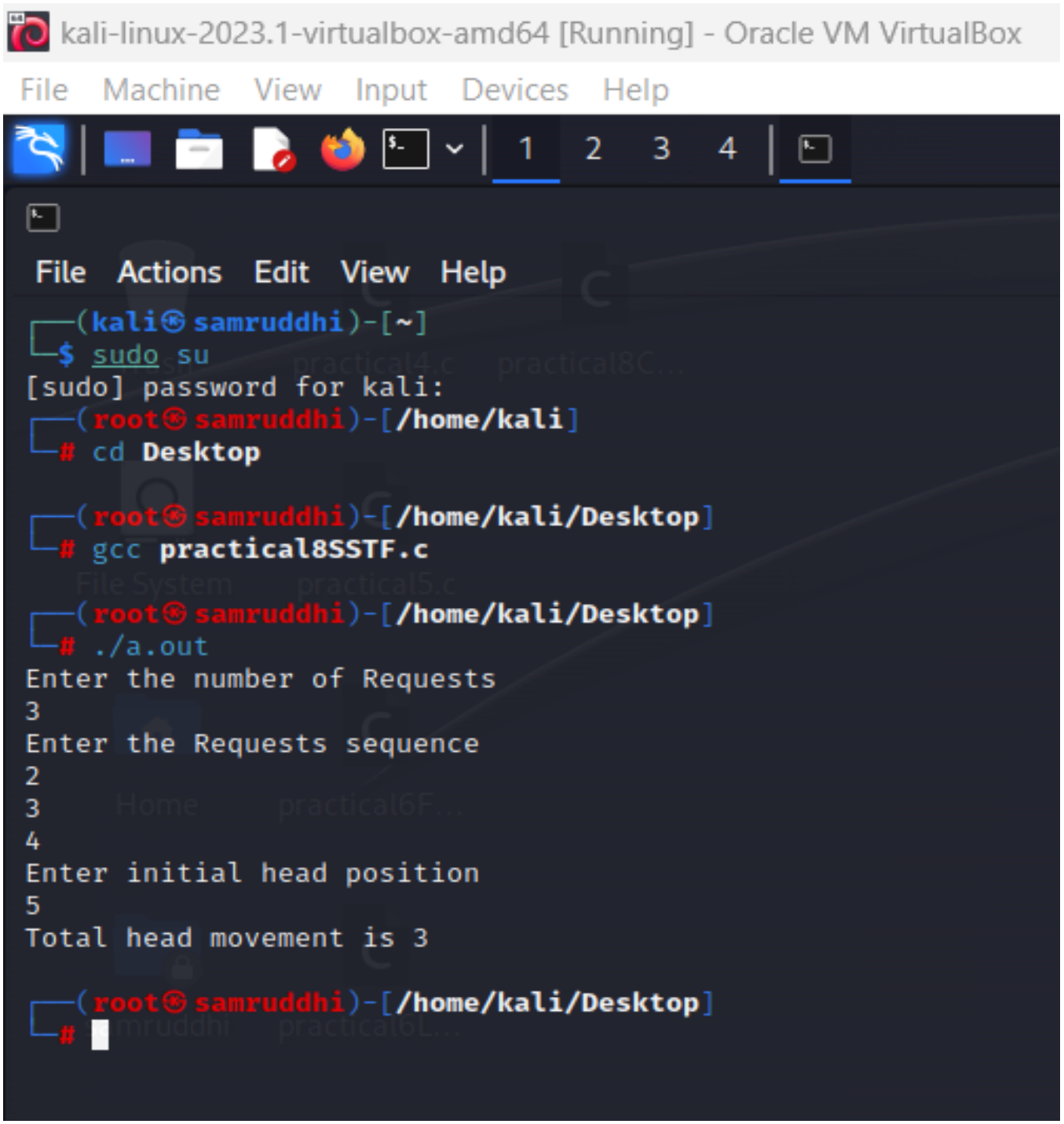
1) SSTF

CODE:

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
	int RQ[100],i,n,TotalHeadMoment=0,initial,count=0;
	printf("Enter the number of Requests\n");
	scanf("%d",&n);
	printf("Enter the Requests sequence\n");
	for(i=0;i<n;i++)
	scanf("%d",&RQ[i]);
	printf("Enter initial head position\n");
	scanf("%d",&initial);
	while(count!=n)
	{
		int min=1000,d,index;
		for(i=0;i<n;i++)
		{
			d=abs(RQ[i]-initial);
			if(min>d)
			{
				min=d;
				index=i;
			}
		}
		TotalHeadMoment=TotalHeadMoment+min;
```

```
            initial=RQ[index];
            RQ[index]=1000;
            count++;
        }
        printf("Total head movement is %d",TotalHeadMoment);
        return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical8SSTF.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter the number of Requests
3
Enter the Requests sequence
2
3
4
Enter initial head position
5
Total head movement is 3
(root㉿samruddhi)-[/home/kali/Desktop]
#

2) SCAN

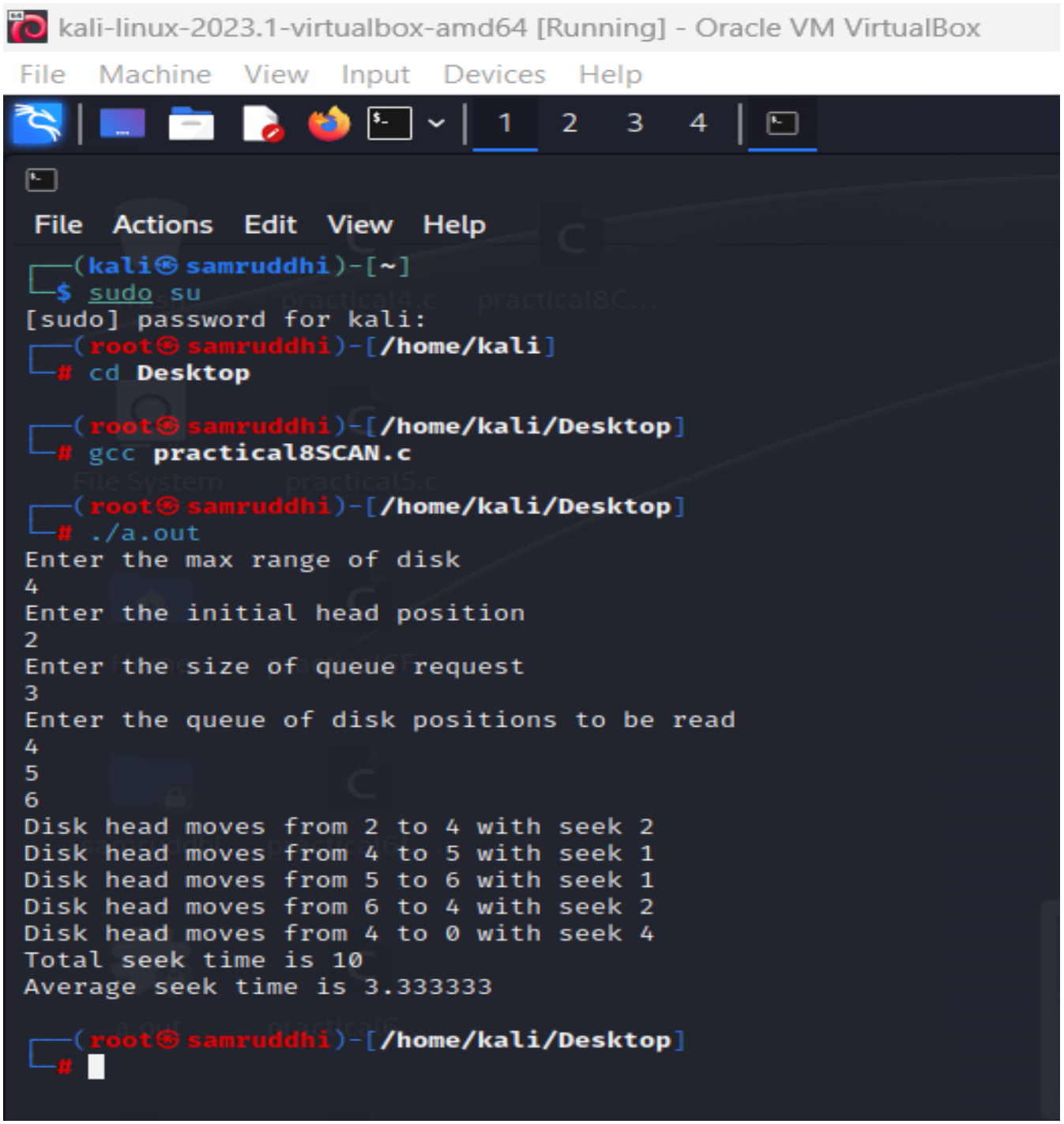
CODE:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main()
{
        int queue[20], n, head, i, j, k, seek = 0, max, diff, temp, queue1[20],
        queue2[20], temp1 = 0, temp2 = 0;
        float avg;
        printf("Enter the max range of disk\n");
        scanf("%d", &max);
        printf("Enter the initial head position\n");
        scanf("%d", &head);
        printf("Enter the size of queue request\n");
        scanf("%d", &n);
        printf("Enter the queue of disk positions to be read\n");
        for (i = 1; i <= n; i++)
        {
                scanf("%d", &temp);
                if (temp >= head)
                {
                        queue1[temp1] = temp;
                        temp1++;
                }
                else
                {
                        queue2[temp2] = temp;
                        temp2++;
                }
        }
```

```
for (i = 0; i < temp1 - 1; i++)
{
        for (j = i + 1; j < temp1; j++)
        {
                if (queue1[i] > queue1[j])
                {
                        temp = queue1[i];
                        queue1[i] = queue1[j];
                        queue1[j] = temp;
                }
        }
}
for (i = 0; i < temp2 - 1; i++)
{
        for (j = i + 1; j < temp2; j++)
        {
                if (queue2[i] < queue2[j])
                {
                        temp = queue2[i];
                        queue2[i] = queue2[j];
                        queue2[j] = temp;
                }
        }
}
for (i = 1, j = 0; j < temp1; i++, j++)
queue[i] = queue1[j];
queue[i] = max;
for (i = temp1 + 2, j = 0; j < temp2; i++, j++)
queue[i] = queue2[j];
queue[i] = 0;
queue[0] = head;
```

```
    for (j = 0; j <= n + 1; j++)
    {
        diff = abs(queue[j + 1] - queue[j]);
        seek += diff;
        printf("Disk head moves from %d to %d with seek %d\n", queue[j],
        queue[j + 1], diff);
    }
    printf("Total seek time is %d\n", seek);
    avg = seek / (float)n;
    printf("Average seek time is %f\n", avg);
    return 0;
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical8SCAN.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter the max range of disk
4
Enter the initial head position
2
Enter the size of queue request
3
Enter the queue of disk positions to be read
4
5
6
Disk head moves from 2 to 4 with seek 2
Disk head moves from 4 to 5 with seek 1
Disk head moves from 5 to 6 with seek 1
Disk head moves from 6 to 4 with seek 2
Disk head moves from 4 to 0 with seek 4
Total seek time is 10
Average seek time is 3.333333
(root㉿samruddhi)-[/home/kali/Desktop]
#

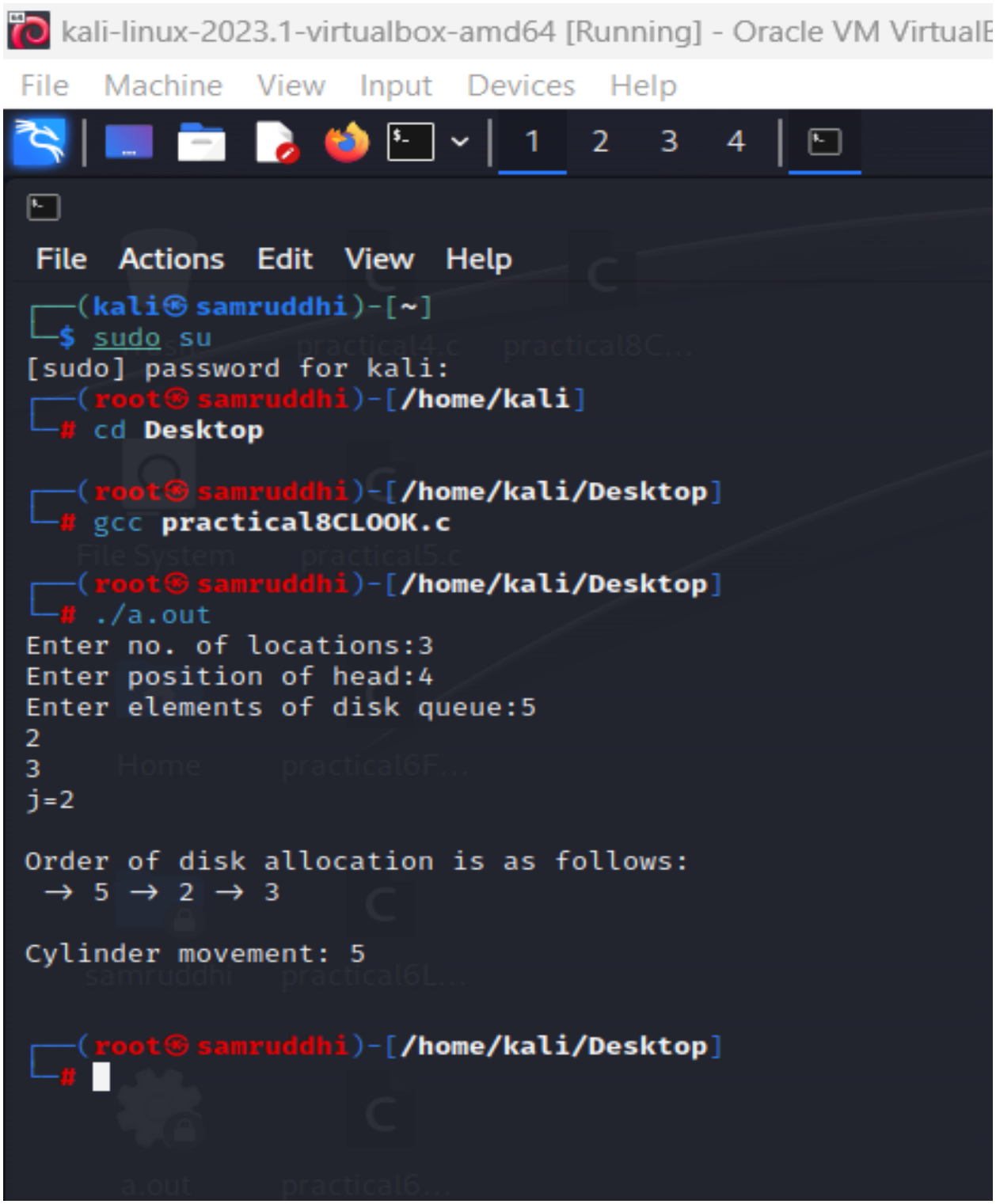
3) C-Look

CODE:

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
	int n, i, j, head, item[20], dst[20];
	int cylinders=0;
	printf("Enter no. of locations:");
	scanf("%d",&n);
	printf("Enter position of head:");
	scanf("%d",&head);
	printf("Enter elements of disk queue:");
	for(i=0;i<n;i++)
	{
		scanf("%d",&item[i]);
		dst[i]=(head-item[i]);
	}
	for(i=0;i<n-1;i++)
	{
		for(j=i+1;j<n;j++)
		{
			if(dst[j]>dst[i])
			{
				int temp=dst[j];
				dst[j]=dst[i];
				dst[i]=temp;
				temp=item[i];
				item[i]=item[j];
				item[j]=temp;
			}
```

```
            }
        }
        for(i=0;i<n;i++)
        {
            if(item[i]>=head)
            {
                j=i;
                break;
            }
        }
        printf("j=%d", j);
        printf("\n\nOrder of disk allocation is as follows:\n");
        for(i=j;i<n;i++)
        {
            printf(" -> %d", item[i]);
            cylinders+= abs(head-item[i]);
            head=item[i];
        }
        for(i=0;i<j;i++)
        {
            printf(" -> %d", item[i]);
            cylinders+= abs(head-item[i]);
            head=item[i];
        }
        printf("\n\nCylinder movement: %d\n\n", cylinders );
}
```

OUTPUT:


kali-linux-2023.1-virtualbox-amd64 [Running] - Oracle VM VirtualB
File Machine View Input Devices Help
1 2 3 4
File Actions Edit View Help
(kali㉿samruddhi)-[~]
$ sudo su
[sudo] password for kali:
(root㉿samruddhi)-[/home/kali]
cd Desktop
(root㉿samruddhi)-[/home/kali/Desktop]
gcc practical8CLOOK.c
(root㉿samruddhi)-[/home/kali/Desktop]
./a.out
Enter no. of locations:3
Enter position of head:4
Enter elements of disk queue:5
2
3
j=2
Order of disk allocation is as follows:
→ 5 → 2 → 3
Cylinder movement: 5
(root㉿samruddhi)-[/home/kali/Desktop]
#