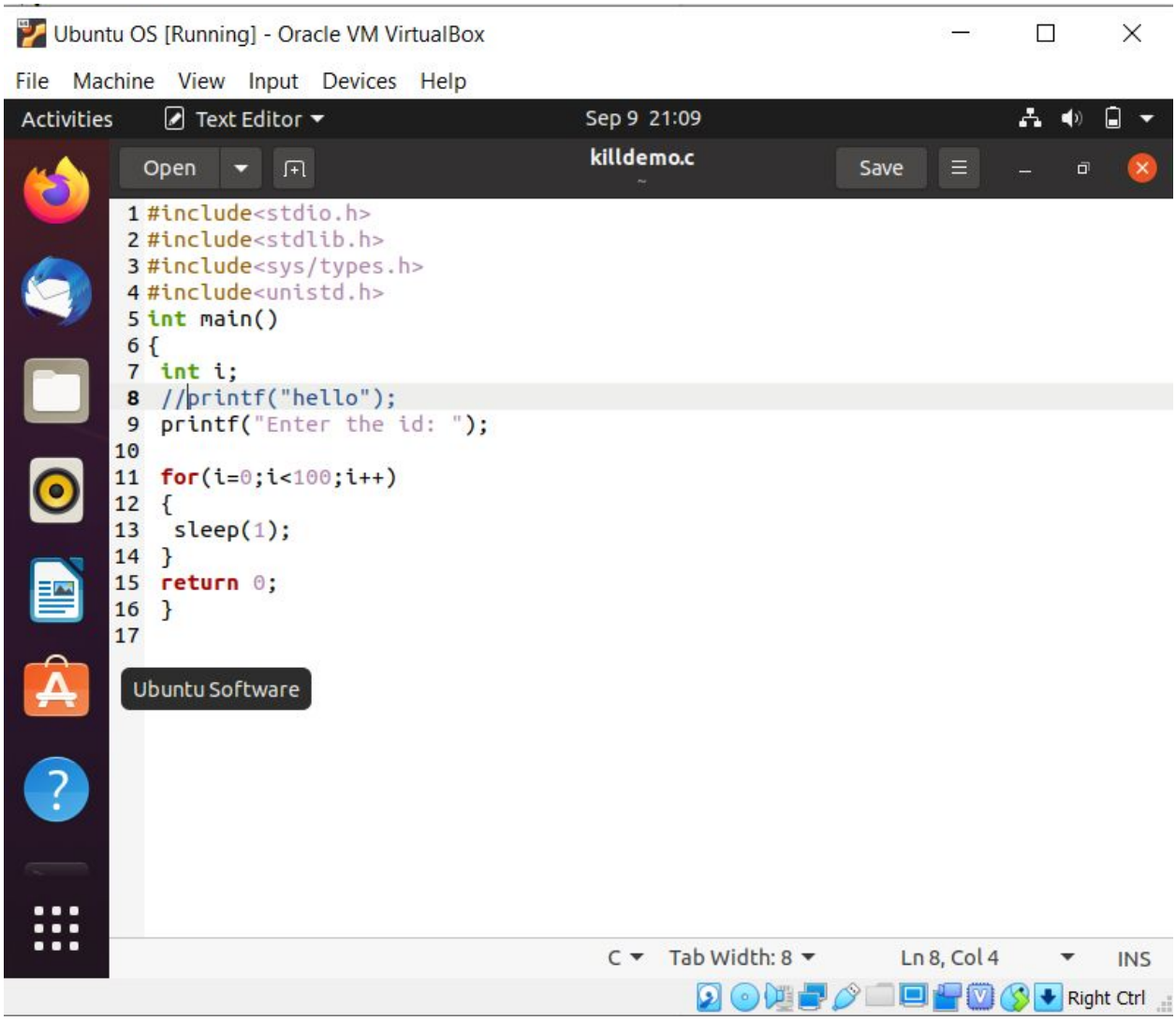
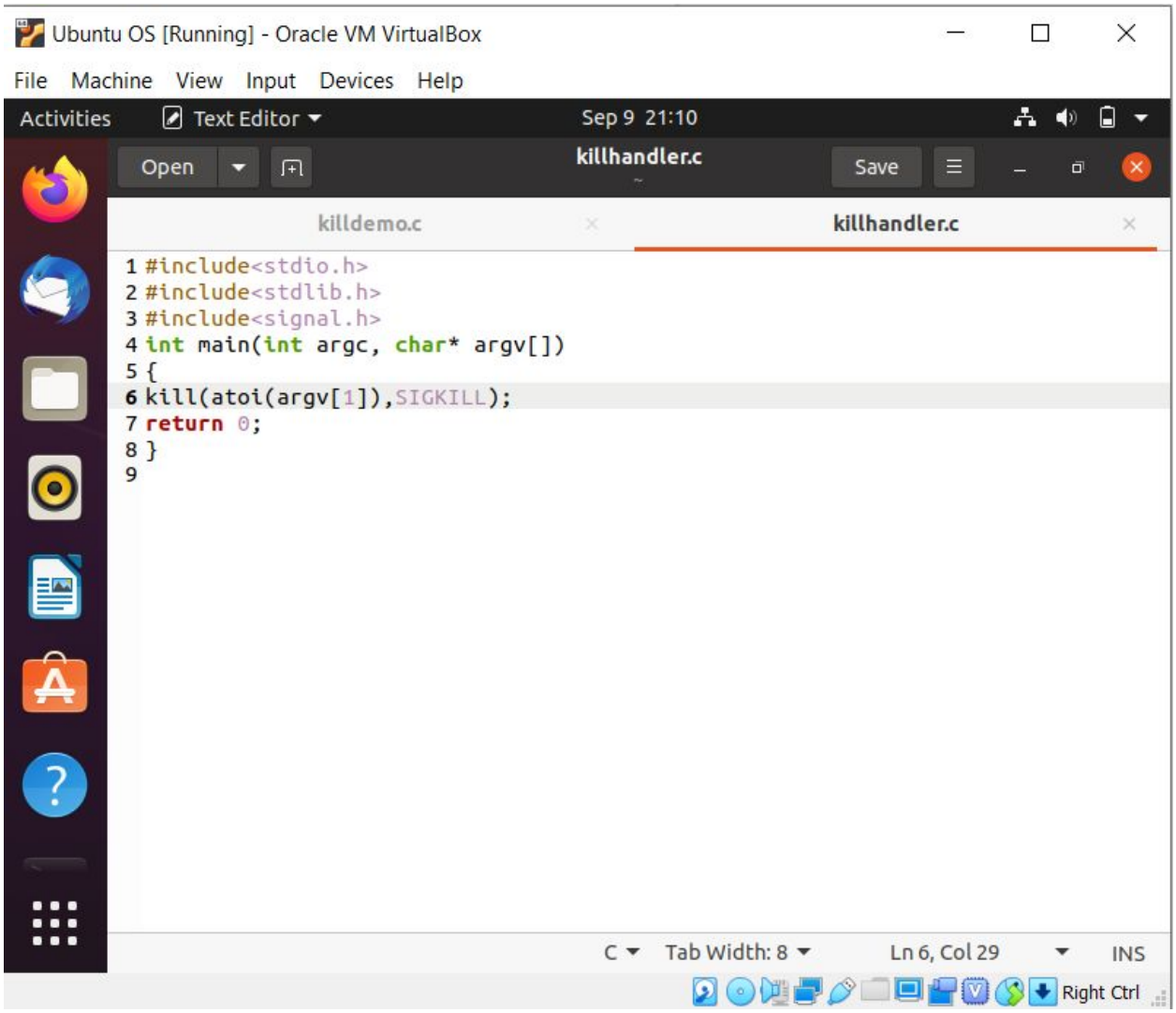
OS-LAB-DA-2

NAME: PRANAV K KADAMBI
REG.NO: 19BCE0964

1)KILL PROCESS:

CODE:(1st terminal):

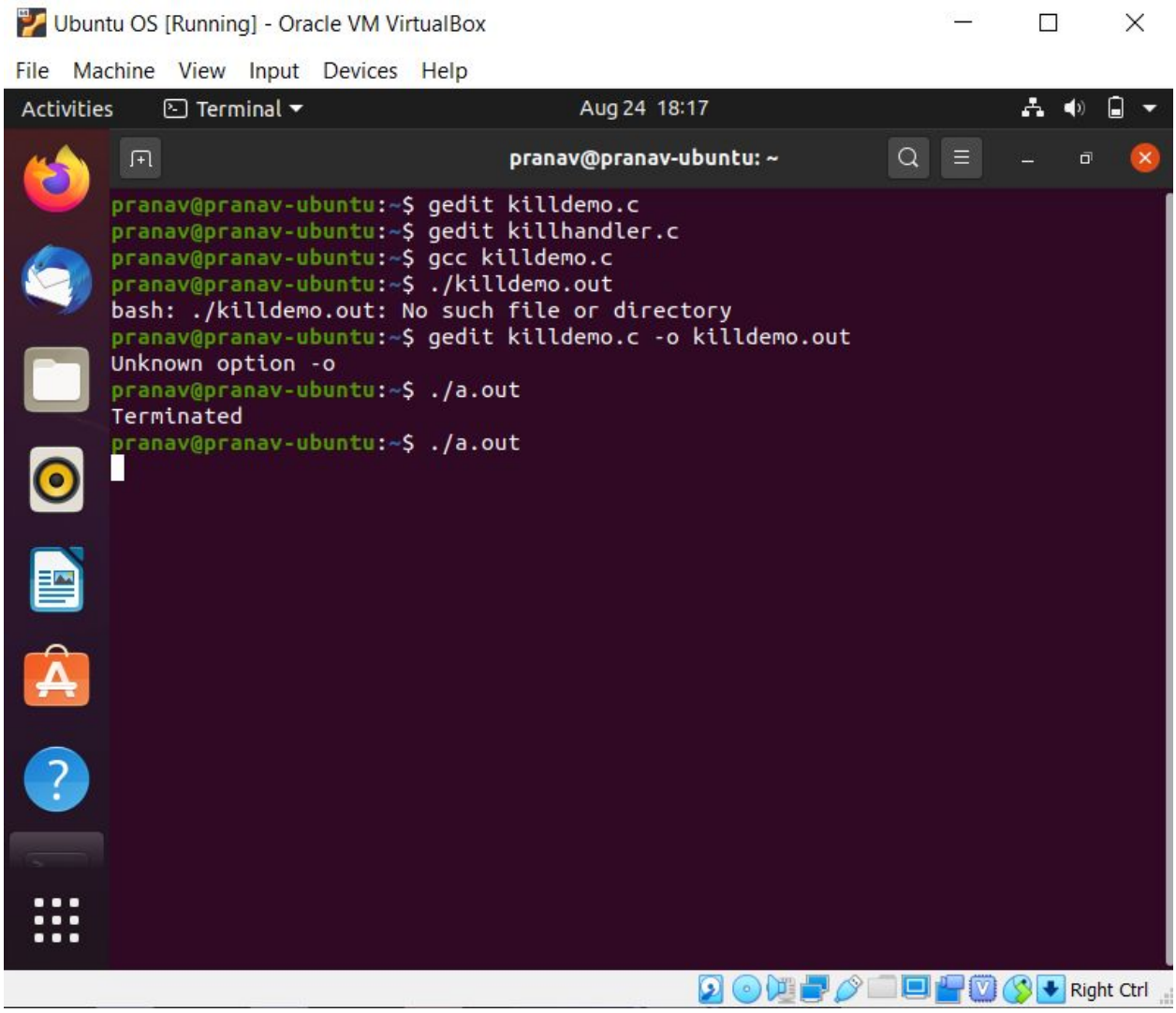




CODE (2nd terminal) :

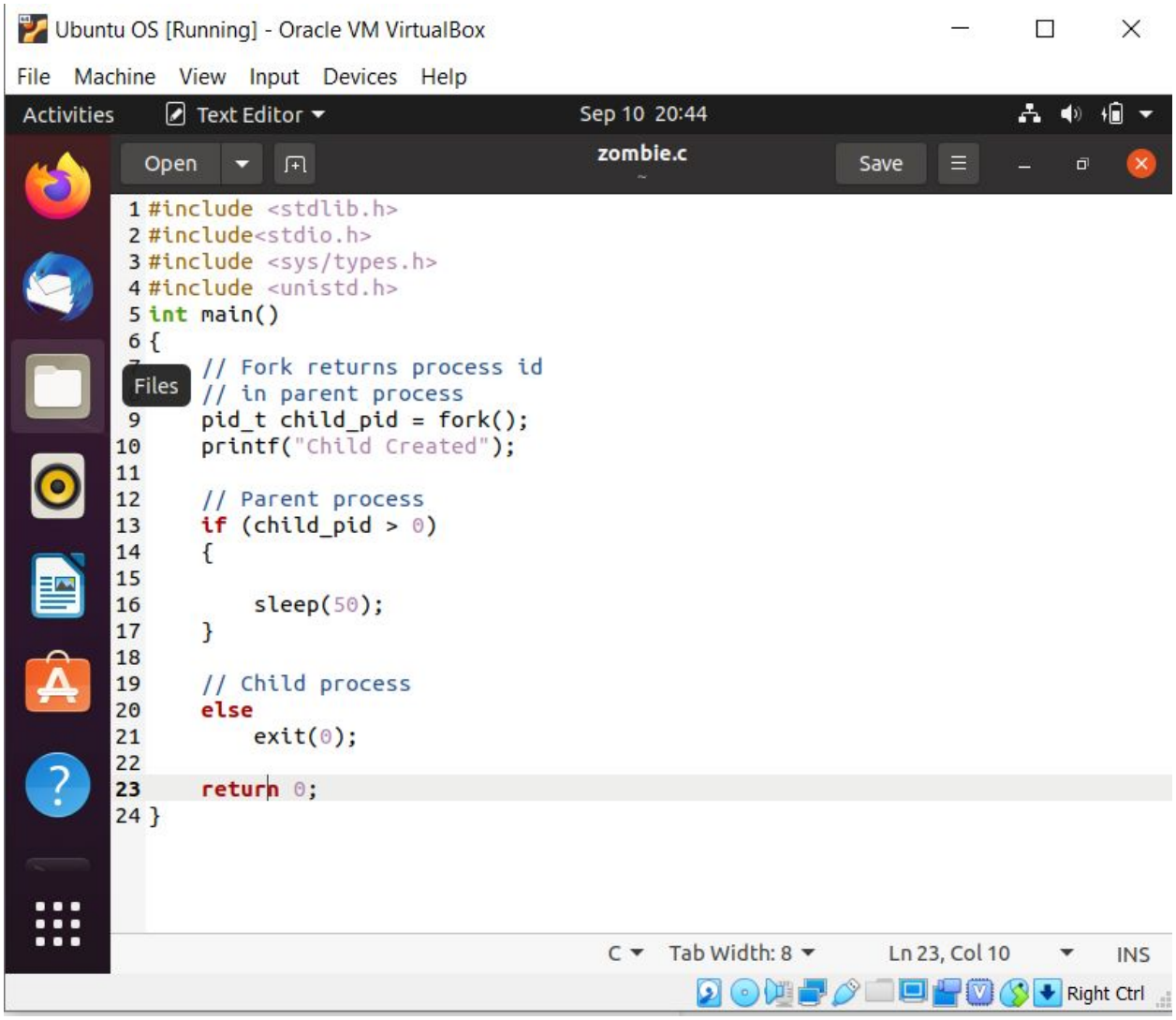
Ps -aux
kill (process number)

OUTPUT:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Aug 24 18:17
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gedit killdemo.c
pranav@pranav-ubuntu:~$ gedit killhandler.c
pranav@pranav-ubuntu:~$ gcc killdemo.c
pranav@pranav-ubuntu:~$ ./killdemo.out
bash: ./killdemo.out: No such file or directory
pranav@pranav-ubuntu:~$ gedit killdemo.c -o killdemo.out
Unknown option -o
pranav@pranav-ubuntu:~$ ./a.out
Terminated
pranav@pranav-ubuntu:~$ ./a.out
Right Ctrl

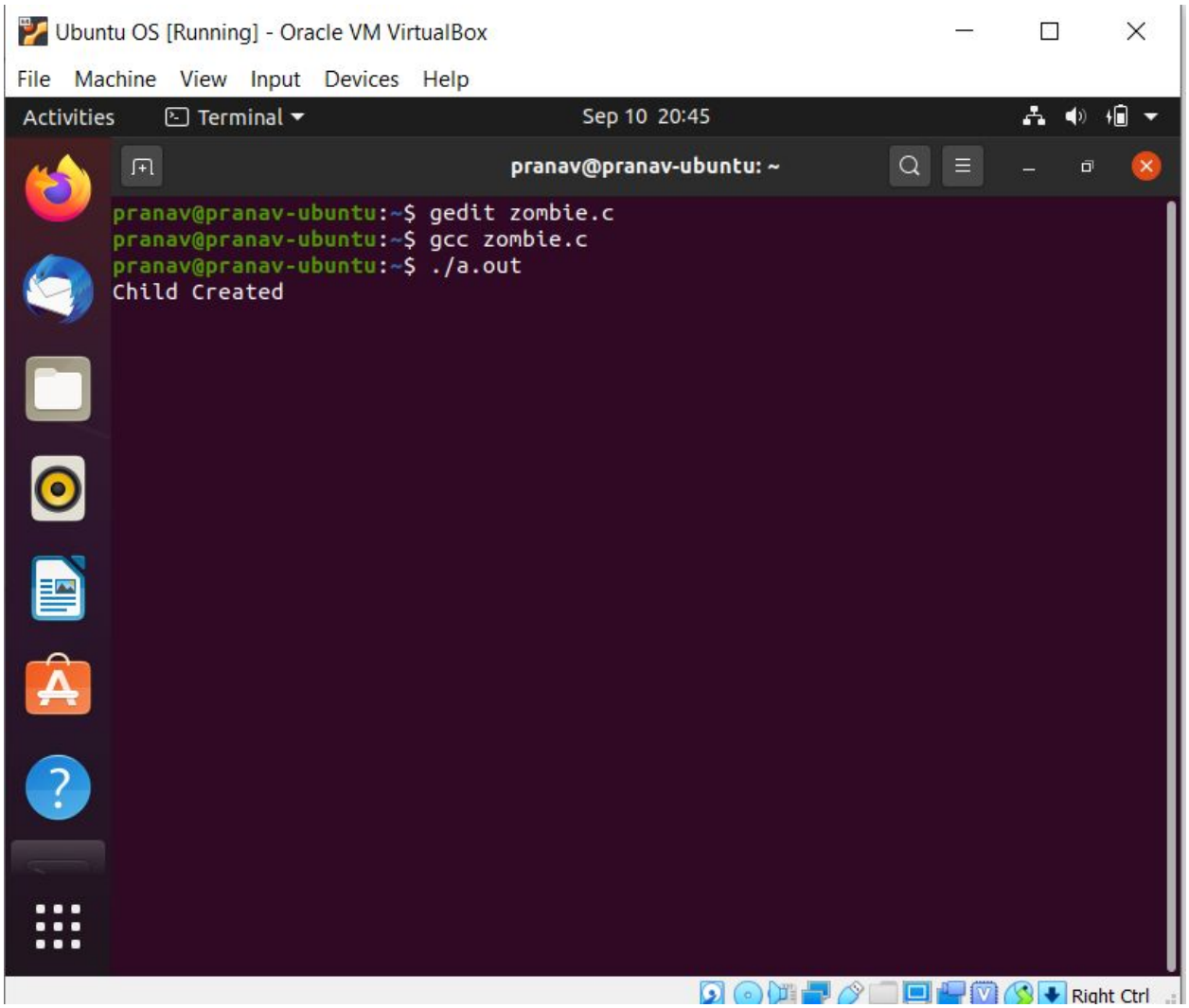
2)Zombie Process:

CODE:




```c
#include <stdlib.h>
#include<stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{
    // Fork returns process id
    // in parent process
    pid_t child_pid = fork();
    printf("Child Created");

    // Parent process
    if (child_pid > 0)
    {

        sleep(50);
    }

    // Child process
    else
        exit(0);

    return 0;
}
```

OUTPUT:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 10 20:45
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gedit zombie.c
pranav@pranav-ubuntu:~$ gcc zombie.c
pranav@pranav-ubuntu:~$ ./a.out
Child Created
Right Ctrl

3)JOB SCHEDULING:

CODE:

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<limits.h>
#define MAX 100
typedef struct
{
int pid;
int arrival_time;
int burst_time;
int waiting_time;
int turnaround_time;
int completion_time;
int priority;
int rem_bt;
}Process;

void print_table(Process p[], int n);

void FCFS()
{
```

```
// get values and calculate the average waiting time and
turnaround time
Process p[MAX];
int n,i,j,sum_waiting_time=0,sum_turnaround_time=0;
int temp[MAX];
printf("Enter number of processes: ");
scanf("%d",&n);
printf("Enter the values:\n");
for(i=0;i<n;i++)
{
   printf("Enter pid: ");
   scanf("%d",&p[i].pid);
   printf("\n");
   printf("Enter Arrival time: ");
   scanf("%d",&p[i].arrival_time);
   printf("\n");
   printf("Enter Burst time: ");
   scanf("%d",&p[i].burst_time);
   printf("\n");

}
temp[0]=0;
for(j=0;j<n;j++)
{
   p[j].waiting_time=0;
   p[j].turnaround_time=0;
```

```
    temp[j+1]=temp[j] + p[j].burst_time;
    p[j].waiting_time=temp[j] - p[j].arrival_time;
    p[j].turnaround_time= p[j].turnaround_time +
p[j].waiting_time+p[j].burst_time;
    sum_waiting_time=sum_waiting_time + p[j].waiting_time;
    sum_turnaround_time=sum_turnaround_time  +
p[j].turnaround_time;
    p[j].completion_time=p[j].turnaround_time +
p[j].arrival_time;

}


// print table
puts(""); // Empty line
print_table(p, n);
puts(""); // Empty Line
printf("Total Waiting Time     : %d\n", sum_waiting_time);
printf("Average Waiting Time       : %f\n",
(double)sum_waiting_time / (double) n);
printf("Total Turnaround Time : %d\n", sum_turnaround_time);
printf("Average Turnaround Time : %f\n",
(double)sum_turnaround_time / (double) n);
```

```
}


void print_table(Process p[], int n)
{
int i;
puts("| PID | Burst Time | Waiting Time | Turnaround Time |
Completion Time|");
for(i=0; i<n; i++)
{
printf("| %2d | %2d |       %2d |       %2d |       %2d |\n", p[i].pid,
p[i].burst_time, p[i].waiting_time,
p[i].turnaround_time,p[i].completion_time );
}

}


void SJF()
{

Process p[MAX];
int n,i,j,t,sum_waiting_time=0,sum_turnaround_time=0;
int temp[MAX];
printf("Enter number of processes: ");
scanf("%d",&n);
```

```
printf("Enter the values:\n");
for(i=0;i<n;i++)
{
   printf("Enter pid: ");
   scanf("%d",&p[i].pid);
   printf("\n");
   printf("Enter Burst time: ");
   scanf("%d",&p[i].burst_time);
   printf("\n");

}
for(i=0;i<n;i++)
{
   for(j=0;j<n-i-1;j++) //bubble sort for burst time
   {
      if(p[j].burst_time>p[j+1].burst_time)
      {
         t=p[j].burst_time;
         p[j].burst_time=p[j+1].burst_time;
         p[j+1].burst_time=t;

         t=p[j].pid;
         p[j].pid=p[j+1].pid;
         p[j+1].pid=t;
      }
   }
```

```
}
for(i=0;i<n;i++)
{
   p[i].waiting_time=0;
   p[i].turnaround_time=0;
   for(j=0;j<i;j++)
   {
      p[i].waiting_time=p[i].waiting_time+p[j].burst_time;
   }

    p[i].turnaround_time=p[i].waiting_time+p[i].burst_time;
    sum_waiting_time=sum_waiting_time + p[i].waiting_time;
    sum_turnaround_time=sum_turnaround_time  +
p[i].turnaround_time;
    p[i].completion_time=p[i].turnaround_time +
p[i].arrival_time;
}
puts(""); // Empty line
print_table(p, n);
puts(""); // Empty Line
printf("Total Waiting Time     : %-2d\n", sum_waiting_time);
printf("Average Waiting Time       : %-2.2lf\n",
(double)sum_waiting_time / (double) n);
printf("Total Turnaround Time : %-2d\n",
sum_turnaround_time);
```

```
printf("Average Turnaround Time : %-2.2lf\n",
(double)sum_turnaround_time / (double) n);



}
void SRTF()
{
Process p[MAX];
int n,i,j,sum_waiting_time=0,sum_turnaround_time=0;
int rt[20];
int complete = 0, t = 0, minm = INT_MAX,check=0,shortest =
0, finish_time;



printf("Enter number of processes: ");
scanf("%d",&n);
printf("Enter the values:\n");
for(i=0;i<n;i++)
{
   printf("Enter pid: ");
   scanf("%d",&p[i].pid);
   printf("\n");
   printf("Enter Burst time: ");
```

```
    scanf("%d",&p[i].burst_time);
    printf("Enter Arrival time: ");
    scanf("%d",&p[i].arrival_time);
    printf("\n");

}
for(i = 0; i < n; i++)
{
    rt[i] = p[i].burst_time;
}
while (complete != n) {

        // Find process with minimum
        // remaining time among the
        // processes that arrives till the
        // current time`
        for (j = 0; j < n; j++) {
            if ((p[j].arrival_time <= t) && (rt[j] < minm) && rt[j] >
0)
            {
                minm = rt[j];
                shortest = j;
                check = 1;
            }
        }
```

```
if (check == 0) {
    t++;
    continue;
}

// Reduce remaining time by one
rt[shortest]--;

// Update minimum
minm = rt[shortest];
if (minm == 0)
    minm = INT_MAX;

// If a process gets completely
// executed
if (rt[shortest] == 0) {

    // Increment complete
    complete++;
    check = 0;

    // Find finish time of current
    // process
    finish_time = t + 1;

    // Calculate waiting time
```

```
            p[shortest].waiting_time = finish_time -
                        p[shortest].burst_time -
                        p[shortest].arrival_time;

            if (p[shortest].waiting_time < 0)
                p[shortest].waiting_time = 0;
        }
        // Increment time
        t++;
    }
    for(i=0;i<n;i++)
{



    p[i].turnaround_time=p[i].waiting_time+p[i].burst_time;
    sum_waiting_time=sum_waiting_time + p[i].waiting_time;
    sum_turnaround_time=sum_turnaround_time  +
p[i].turnaround_time;
    p[i].completion_time=p[i].turnaround_time ;
}
puts(""); // Empty line
print_table(p, n);
puts(""); // Empty Line
printf("Total Waiting Time     : %-2d\n", sum_waiting_time);
```

```
printf("Average Waiting Time       : %-2.2lf\n",
(double)sum_waiting_time / (double) n);
printf("Total Turnaround Time : %-2d\n",
sum_turnaround_time);
printf("Average Turnaround Time : %-2.2lf\n",
(double)sum_turnaround_time / (double) n);



}



void Priority()
{
Process p[MAX];
int
n,i,j,t,b=0,min,k=1,sum_waiting_time=0,sum_turnaround_time=
0;
int temp[MAX];
printf("Enter number of processes: ");
scanf("%d",&n);
printf("Enter the values:\n");
for(i=0;i<n;i++)
{
```

```
    printf("Enter pid: ");
    scanf("%d",&p[i].pid);
    printf("\n");
    printf("Enter Burst time: ");
    scanf("%d",&p[i].burst_time);
    printf("\n");
    printf("Enter Arrival time: ");
    scanf("%d",&p[i].arrival_time);
    printf("Enter priority: ");
    scanf("%d",&p[i].priority);
    printf("\n");



}
for(i=0;i<n;i++)
{
    for(j=0;j<n;j++)
    {
      if(p[i].arrival_time<p[j].arrival_time)
      {


       t=p[j].arrival_time;
       p[j].arrival_time=p[i].arrival_time;
       p[i].arrival_time=t;
```

```
        t=p[j].burst_time;
        p[j].burst_time=p[i].burst_time;
        p[i].burst_time=t;

       }

    }
}
for(j=0;i<n;j++)
{
    b=b+p[j].burst_time;
    min=p[k].burst_time;
    for(i=k;i<n;i++)
    {
        min=p[k].priority;
        if(b>=p[i].arrival_time)
        {
            if(p[i].priority<min)
            {
                t=p[k].arrival_time;
                p[k].arrival_time=p[i].arrival_time;
                p[i].arrival_time=t;

                t=p[k].burst_time;
                p[k].burst_time=p[i].burst_time;
```

```
            p[i].burst_time=t;

            t=p[k].priority;
            p[k].priority=p[i].priority;
            p[i].priority=t;
        }
    }
  }
  k++;
}
temp[0]=0;
for(i=0;i<n;i++)
{
  p[i].waiting_time=0;
  p[i].turnaround_time=0;
  temp[i+1]=temp[i]+p[i].burst_time;
  p[i].waiting_time=temp[i] - p[i].arrival_time;
  p[i].turnaround_time=p[i].waiting_time+p[i].burst_time;
  sum_waiting_time=sum_waiting_time + p[i].waiting_time;
  sum_turnaround_time=sum_turnaround_time  +
p[i].turnaround_time;
  p[i].completion_time=p[i].turnaround_time +
p[i].arrival_time;



}
```

```
puts(""); // Empty line
print_table(p, n);
puts(""); // Empty Line
printf("Total Waiting Time     : %-2d\n", sum_waiting_time);
printf("Average Waiting Time       : %-2.2lf\n",
(double)sum_waiting_time / (double) n);
printf("Total Turnaround Time : %-2d\n",
sum_turnaround_time);
printf("Average Turnaround Time : %-2.2lf\n",
(double)sum_turnaround_time / (double) n);

}

void RR()
{
Process p[MAX];
int
n,i,j,sum_waiting_time=0,sum_turnaround_time=0,count=0,tem
p,qt,sq=0;
printf("Enter number of processes: ");
scanf("%d",&n);
printf("Enter the values:\n");
for(i=0;i<n;i++)
{
   printf("Enter pid: ");
   scanf("%d",&p[i].pid);
```

```
    printf("\n");
    printf("Enter Burst time: ");
    scanf("%d",&p[i].burst_time);
    p[i].rem_bt=p[i].burst_time;
    printf("\n");

}
printf("Enter quantum time: ");
scanf("%d",&qt);
while(1)
{

    for(i=0,count=0;i<n;i++)
    {
        temp=qt;
        if(p[i].rem_bt==0)
        {
            count++;
            continue;
        }
        if(p[i].rem_bt>qt)
            p[i].rem_bt=p[i].rem_bt-qt;
        else
        {
```

```
            if(p[i].rem_bt>=0)
            {
                temp=p[i].rem_bt;
                p[i].rem_bt=0;

            }
        }

            sq=sq+temp;
            p[i].turnaround_time=sq;


    }
    if(n==count)
        break;
}

for(i=0;i<n;i++)
{
    p[i].waiting_time=p[i].turnaround_time-p[i].burst_time;
    sum_waiting_time=sum_waiting_time+p[i].waiting_time;

sum_turnaround_time=sum_turnaround_time+p[i].turnaround_ti
me;
    p[i].completion_time=p[i].turnaround_time;
```

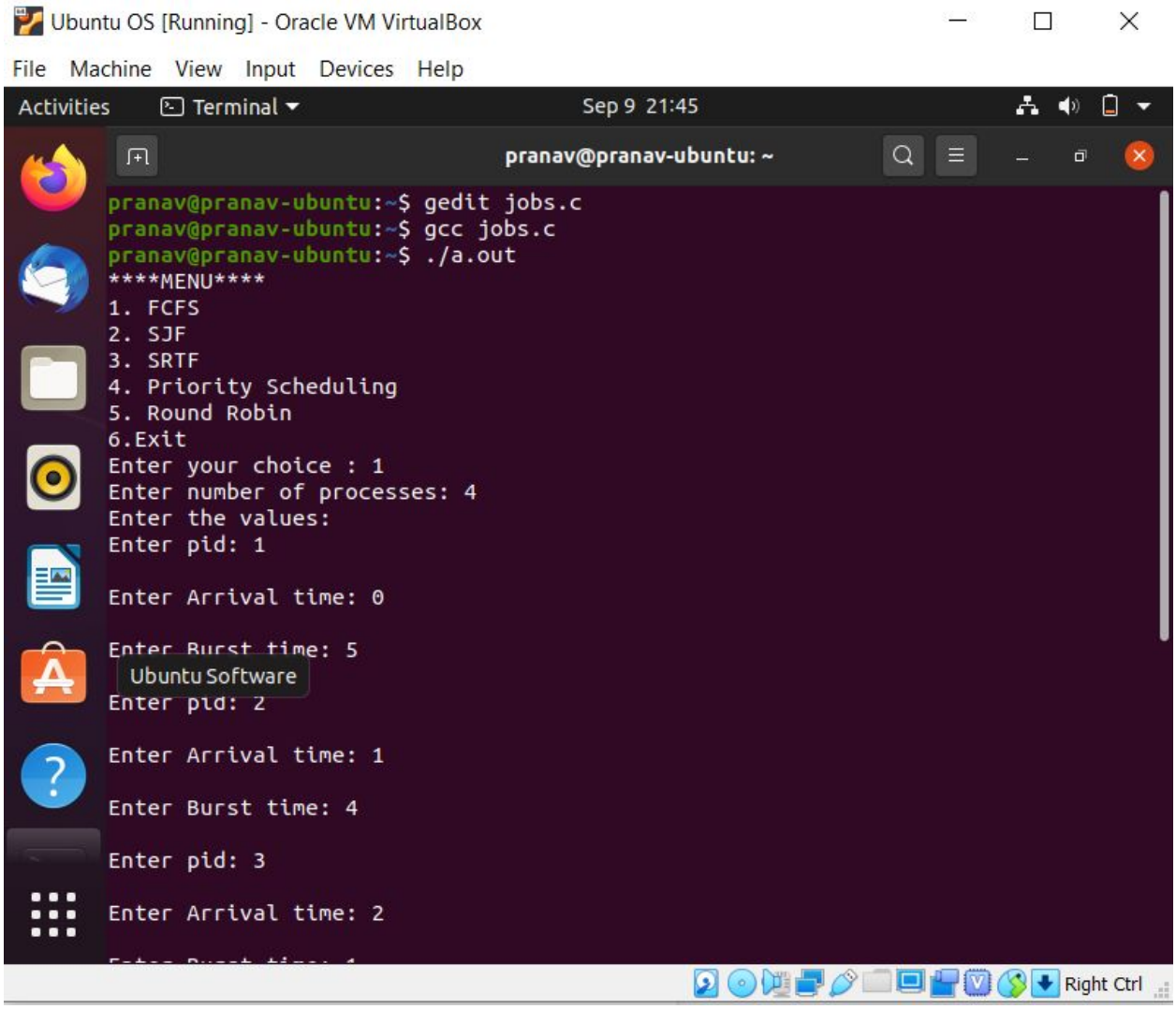
```
}
puts(""); // Empty line
print_table(p, n);
puts(""); // Empty Line
printf("Total Waiting Time    : %-2d\n", sum_waiting_time);
printf("Average Waiting Time       : %-2.2lf\n",
(double)sum_waiting_time / (double) n);
printf("Total Turnaround Time : %-2d\n",
sum_turnaround_time);
printf("Average Turnaround Time : %-2.2lf\n",
(double)sum_turnaround_time / (double) n);




}


int main()

{


printf("****MENU****\n");
printf("1. FCFS\n");
printf("2. SJF\n");
```

```
printf("3. SRTF\n");
printf("4. Priority Scheduling\n");
printf("5. Round Robin\n");
printf("6.Exit\n");
printf("Enter your choice : ");
int ch;
scanf("%d",&ch);

switch(ch)
{
case 1:
FCFS();
break;
case 2:
SJF();
break;
case 3:
SRTF();
break;
case 4:
Priority();
break;
case 5:
RR();
break;
case 6:
```

```
exit(0);
default:
printf("Invalid input!");
}
return 0;
}
```

OUTPUT:

1)FCFS:



Ubuntu OS [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

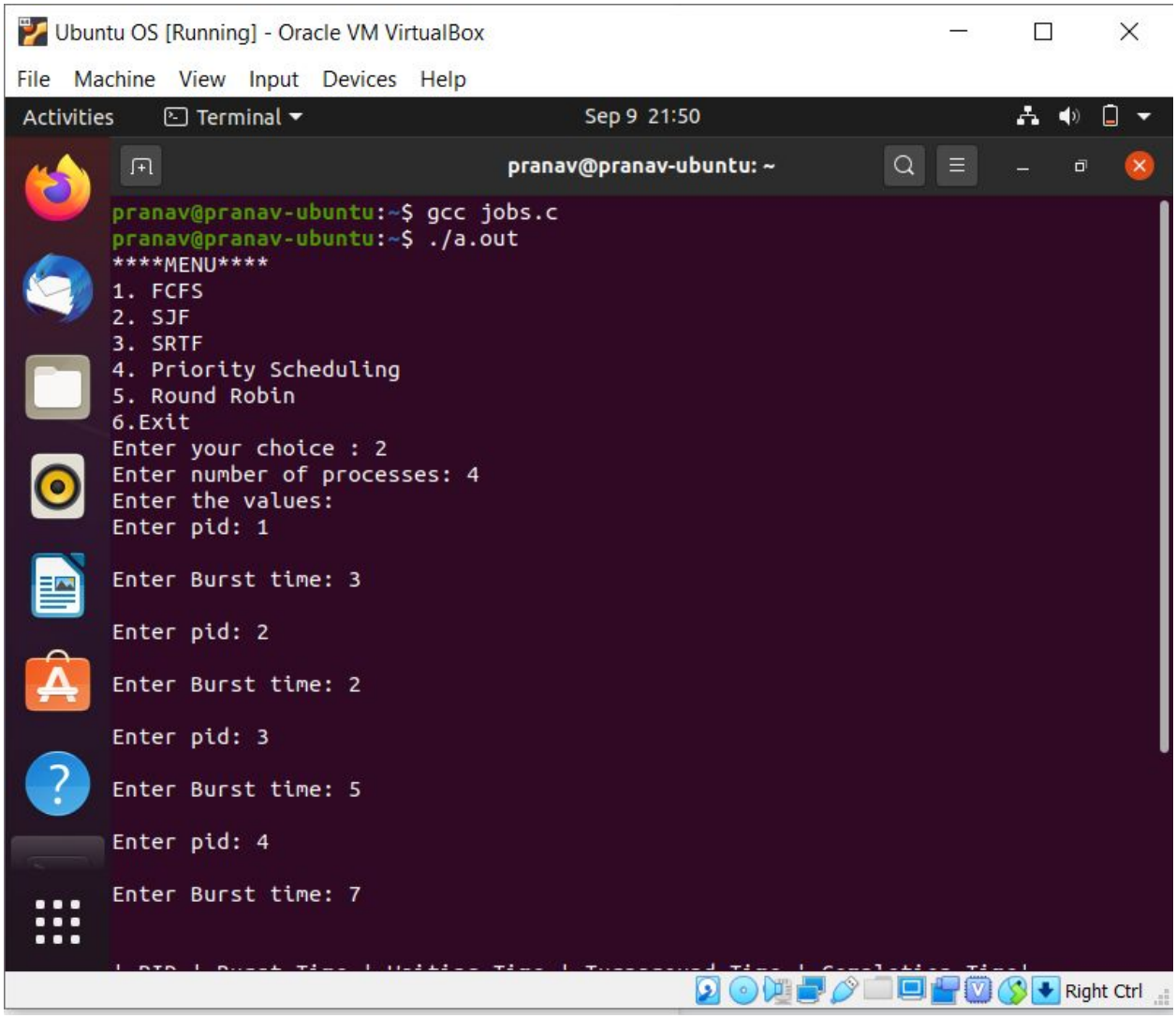
Activities Terminal Sep 9 21:46

pranav@pranav-ubuntu: ~

```
Enter Arrival time: 1

Enter Burst time: 4

Enter pid: 3

Enter Arrival time: 2

Enter Burst time: 1

Enter pid: 4

Enter Arrival time: 3

Enter Burst time: 3


| PID | Burst Time | Waiting Time | Turnaround Time | Completion Time|
|  1 |   5      |        0      |        5      |        5      |
|  2 |   4      |        4      |        8      |        9      |
|  3 |   1      |        7      |        8      |       10      |
|  4 |   3      |        7      |       10      |       13      |

Total Waiting Time      : 18
Average Waiting Time    : 4.500000
Total Turnaround Time : 31
Average Turnaround Time : 7.750000
pranav@pranav-ubuntu:~$
```

Right Ctrl

2)SJF:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 9 21:50
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gcc jobs.c
pranav@pranav-ubuntu:~$ ./a.out
****MENU****
1. FCFS
2. SJF
3. SRTF
4. Priority Scheduling
5. Round Robin
6.Exit
Enter your choice : 2
Enter number of processes: 4
Enter the values:
Enter pid: 1
Enter Burst time: 3
Enter pid: 2
Enter Burst time: 2
Enter pid: 3
Enter Burst time: 5
Enter pid: 4
Enter Burst time: 7
Right Ctrl

Ubuntu OS [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

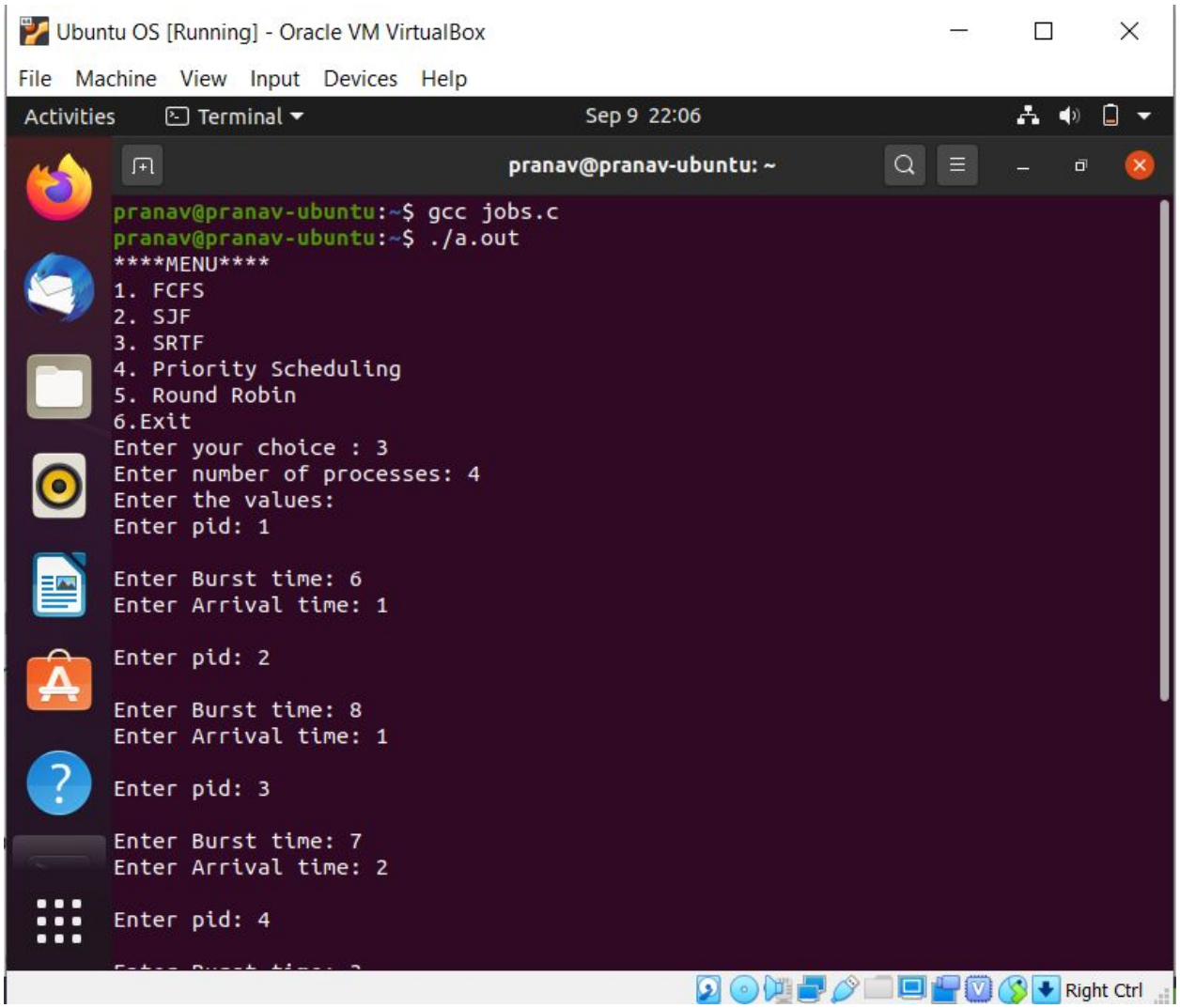
Activities Terminal Sep 9 21:55

pranav@pranav-ubuntu: ~

```
Enter the values:
Enter pid: 1

Enter Burst time: 3

Enter pid: 2

Enter Burst time: 2

Enter pid: 3

Enter Burst time: 5

Enter pid: 4

Enter Burst time: 7


| PID | Burst Time | Waiting Time | Turnaround Time | Completion Time|
|  2 |   2      |        0      |        2      |        2      |
|  1 |   3      |        2      |        5      |        5      |
|  3 |   5      |        5      |       10      |       10      |
|  4 |   7      |       10      |       17      |       17      |

Total Waiting Time      : 17
Average Waiting Time    : 4.25
Total Turnaround Time : 34
Average Turnaround Time : 8.50
pranav@pranav-ubuntu:~$
```

Right Ctrl

3)SRTF:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 9 22:06
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gcc jobs.c
pranav@pranav-ubuntu:~$ ./a.out
****MENU****
1. FCFS
2. SJF
3. SRTF
4. Priority Scheduling
5. Round Robin
6.Exit
Enter your choice : 3
Enter number of processes: 4
Enter the values:
Enter pid: 1
Enter Burst time: 6
Enter Arrival time: 1
Enter pid: 2
Enter Burst time: 8
Enter Arrival time: 1
Enter pid: 3
Enter Burst time: 7
Enter Arrival time: 2
Enter pid: 4
Right Ctrl

Ubuntu OS [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

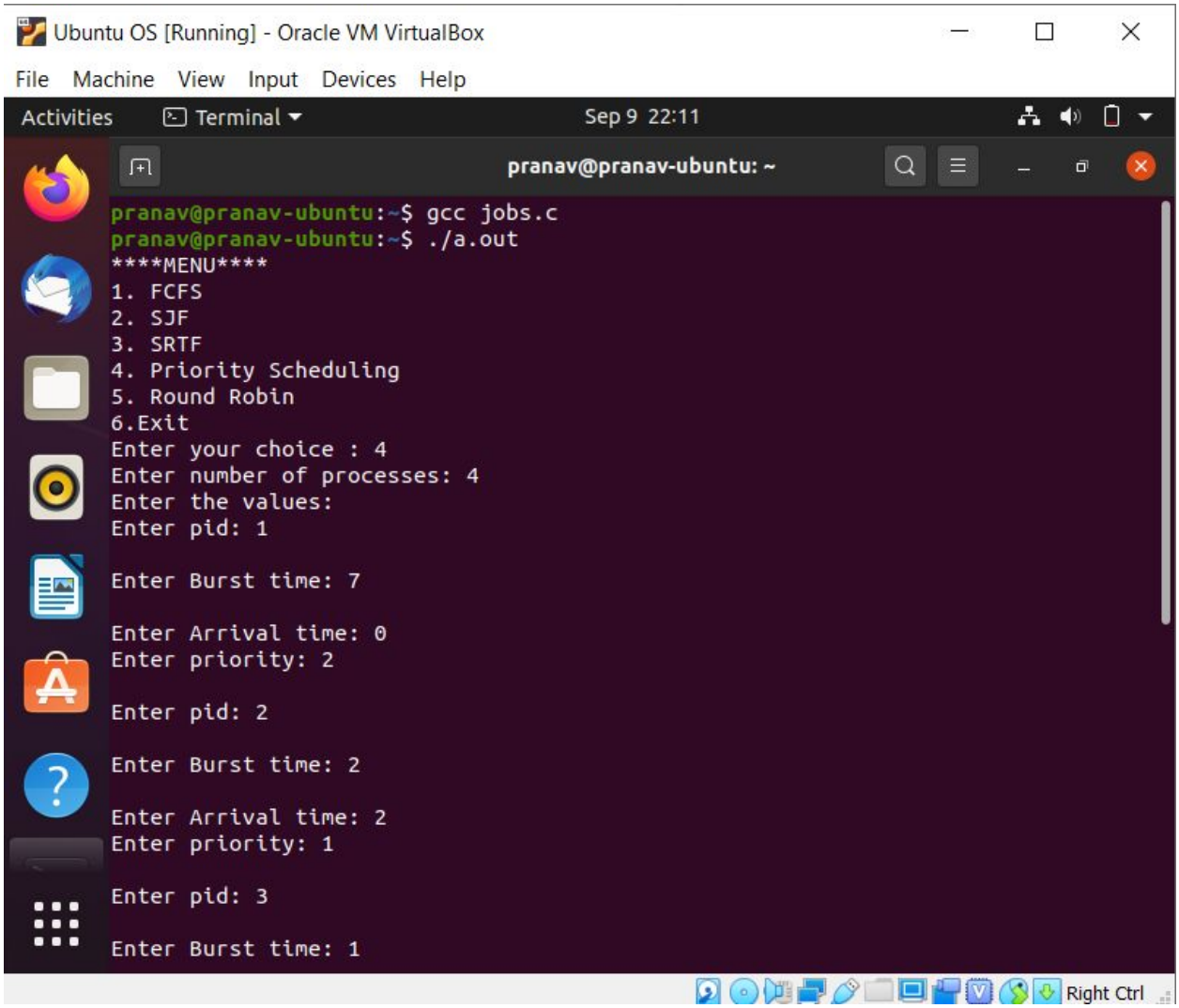
Activities Terminal Sep 9 22:07

pranav@pranav-ubuntu: ~

```
Enter Arrival time: 1

Enter pid: 2

Enter Burst time: 8
Enter Arrival time: 1

Enter pid: 3

Enter Burst time: 7
Enter Arrival time: 2

Enter pid: 4

Enter Burst time: 3
Enter Arrival time: 3


| PID | Burst Time | Waiting Time | Turnaround Time | Completion Time|
|  1 |   6      |        3      |        9      |       10      |
|  2 |   8      |       16      |       24      |       25      |
|  3 |   7      |        8      |       15      |       17      |
|  4 |   3      |        0      |        3      |        6      |

Total Waiting Time      : 27
Average Waiting Time    : 6.75
Total Turnaround Time : 51
Average Turnaround Time : 12.75
pranav@pranav-ubuntu:~$
```

Right Ctrl

4)PRIORITY:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Sep 9 22:11
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gcc jobs.c
pranav@pranav-ubuntu:~$ ./a.out
****MENU****
1. FCFS
2. SJF
3. SRTF
4. Priority Scheduling
5. Round Robin
6.Exit
Enter your choice : 4
Enter number of processes: 4
Enter the values:
Enter pid: 1
Enter Burst time: 7
Enter Arrival time: 0
Enter priority: 2
Enter pid: 2
Enter Burst time: 2
Enter Arrival time: 2
Enter priority: 1
Enter pid: 3
Enter Burst time: 1
Right Ctrl

Ubuntu OS [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

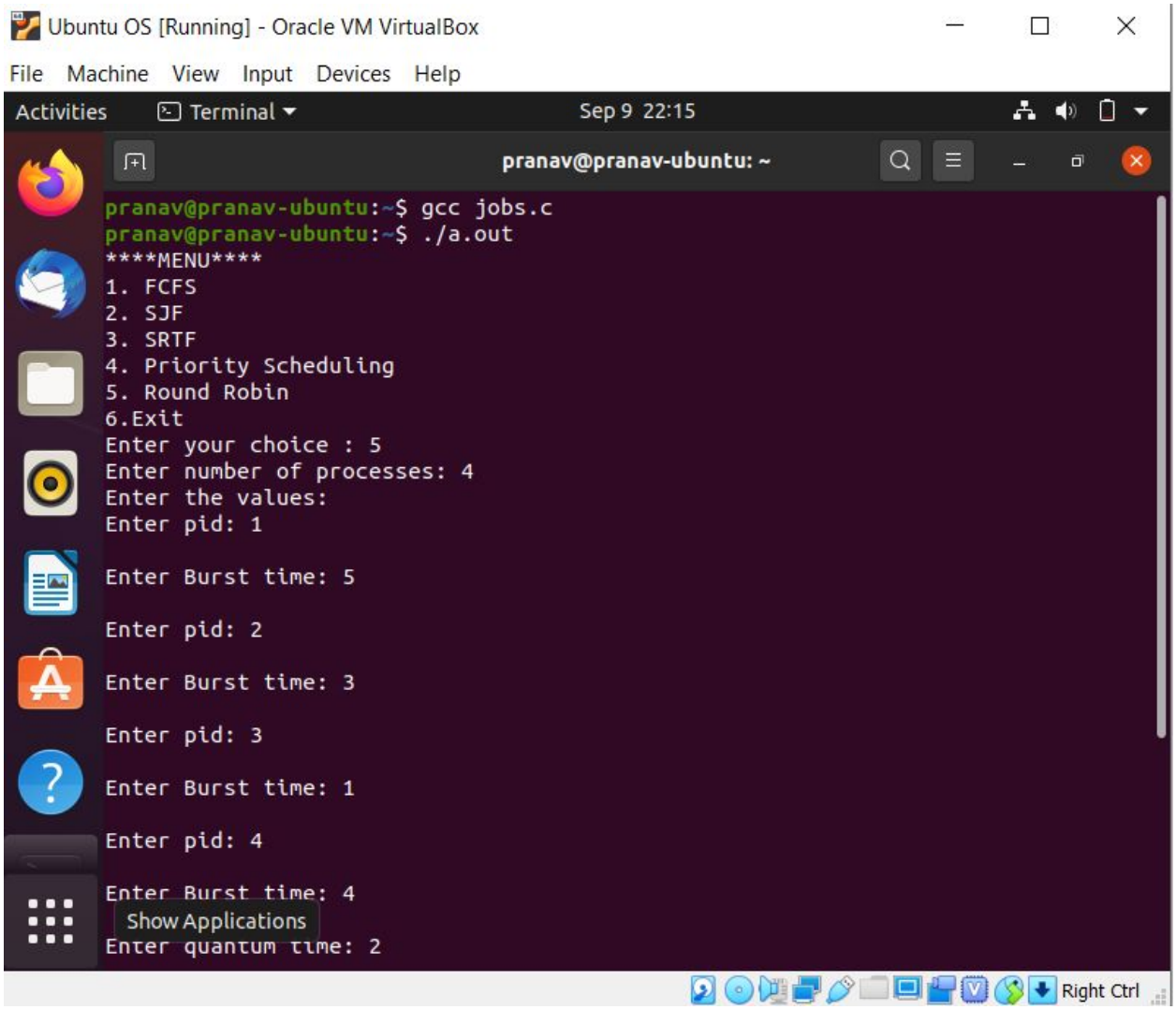
Activities Terminal Sep 9 22:11

pranav@pranav-ubuntu: ~

```
Enter Arrival time: 2
Enter priority: 1

Enter pid: 3

Enter Burst time: 1

Enter Arrival time: 4
Enter priority: 3

Enter pid: 4

Enter Burst time: 5

Enter Arrival time: 6
Enter priority: 4


| PID | Burst Time | Waiting Time | Turnaround Time | Completion Time|
|  1 |   7      |        0      |        7      |        7      |
|  2 |   2      |        5      |        7      |        9      |
|  3 |   1      |        5      |        6      |       10      |
|  4 |   5      |        4      |        9      |       15      |

Total Waiting Time      : 14
Average Waiting Time    : 3.50
Total Turnaround Time : 29
Average Turnaround Time : 7.25
pranav@pranav-ubuntu:~$
```

Right Ctrl

5)ROUND ROBIN:



Ubuntu OS [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Terminal Sep 9 22:16

pranav@pranav-ubuntu: ~

```
Enter pid: 1

Enter Burst time: 5

Enter pid: 2

Enter Burst time: 3

Enter pid: 3

Enter Burst time: 1

Enter pid: 4

Enter Burst time: 4

Enter quantum time: 2

| PID | Burst Time | Waiting Time | Turnaround Time | Completion Time|
|  1 |   5      |        8      |       13      |       13      |
|  2 |   3      |        7      |       10      |       10      |
|  3 |   1      |        4      |        5      |        5      |
|  4 |   4      |        8      |       12      |       12      |

Total Waiting Time      : 27
Average Waiting Time    : 6.75
Total Turnaround Time : 40
Average Turnaround Time : 10.00
pranav@pranav-ubuntu:~$
```

Right Ctrl

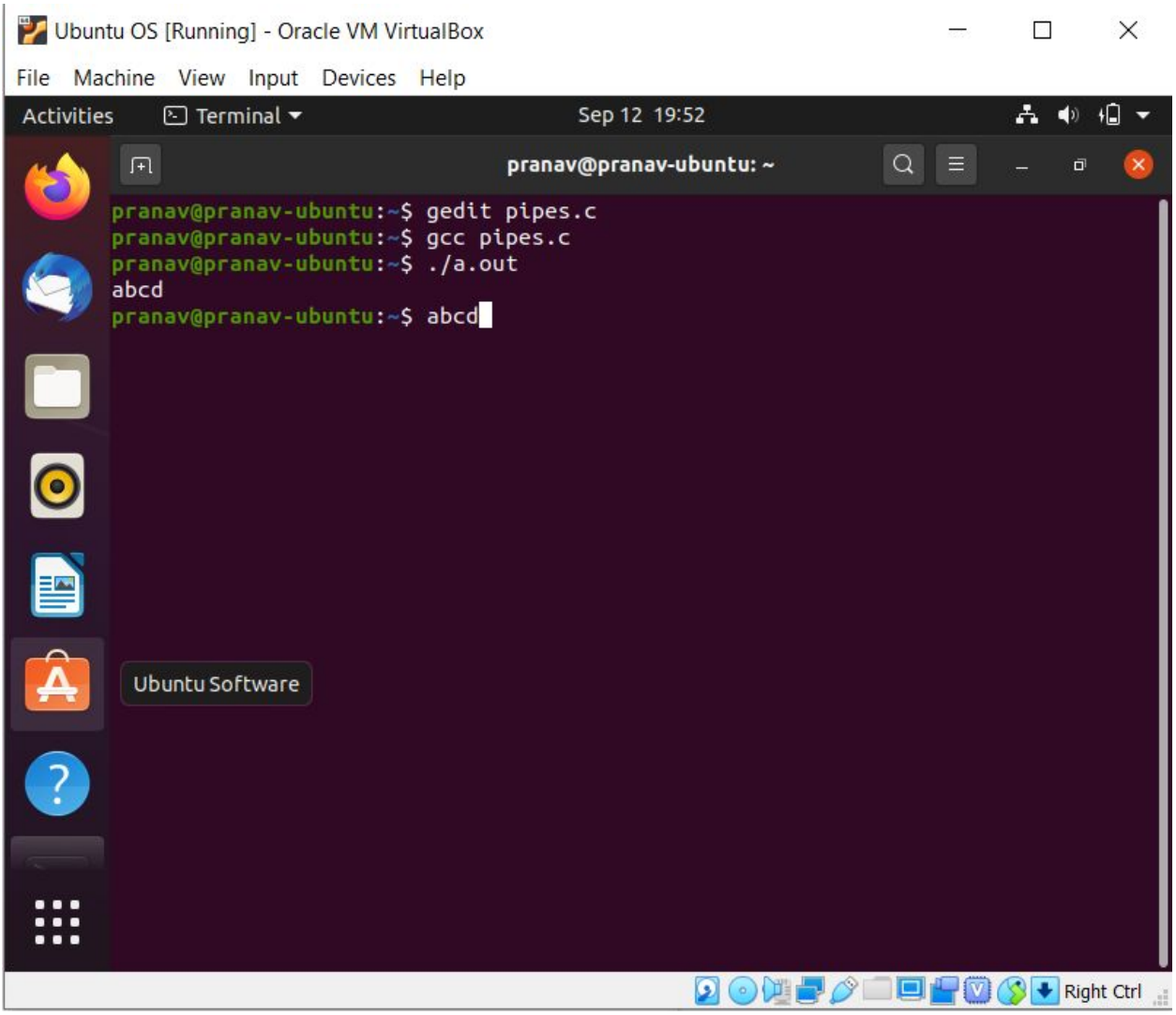
4)IPC:

CODE:

```
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
int main()
{
char msg[25];
char read_m[25];
int fd[2];
pid_t pid;
if (pipe(fd)==-1){
fprintf(stderr, "pipe failed");
return 1;
}
pid=fork();
if(pid>0){
scanf("%s",msg);
close(fd[0]);
write(fd[1],msg,strlen(msg)+1);
close(fd[1]);
read(fd[0],msg,25);
```

```
}
else{
close(fd[1]);
read(fd[0],read_m,25);
close(fd[0]);
write(fd[1],read_m,strlen(read_m));
close(fd[1]);
printf("%s",read_m);
}
return 0;
}
```

OUTPUT:


Ubuntu OS [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities
Terminal
Sep 12 19:52
pranav@pranav-ubuntu: ~
pranav@pranav-ubuntu:~$ gedit pipes.c
pranav@pranav-ubuntu:~$ gcc pipes.c
pranav@pranav-ubuntu:~$ ./a.out
abcd
pranav@pranav-ubuntu:~$ abcd
Ubuntu Software
Right Ctrl