ITE2002-OPERATING SYSTEM

LAB

WINTER SEM 20-21

Assessment - 1

Name :Pravin G

Reg No :19BIT0393

Slot :L41-42

1. Basic Linux commands

1.cd

```
PRAVIN@DESKTOP-B2LB8FB ~
$ cd oslab
```

2.ls

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab
$ ls
20210209  20210223  FIFO.c  FIFO.exe  da1
```

3.mkdir

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ mkdir temp
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cd temp
```

4.cd ..

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/temp
$ cd ..
```

5.rmdir

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ rmdir temp

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name1
```

6.mv

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ mv name1 name2

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name2
```

7.pwd
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ pwd
/home/PRAVIN/oslab/da1
```

8.date
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date
Mon Mar 15 07:47:06 IST 2021
```

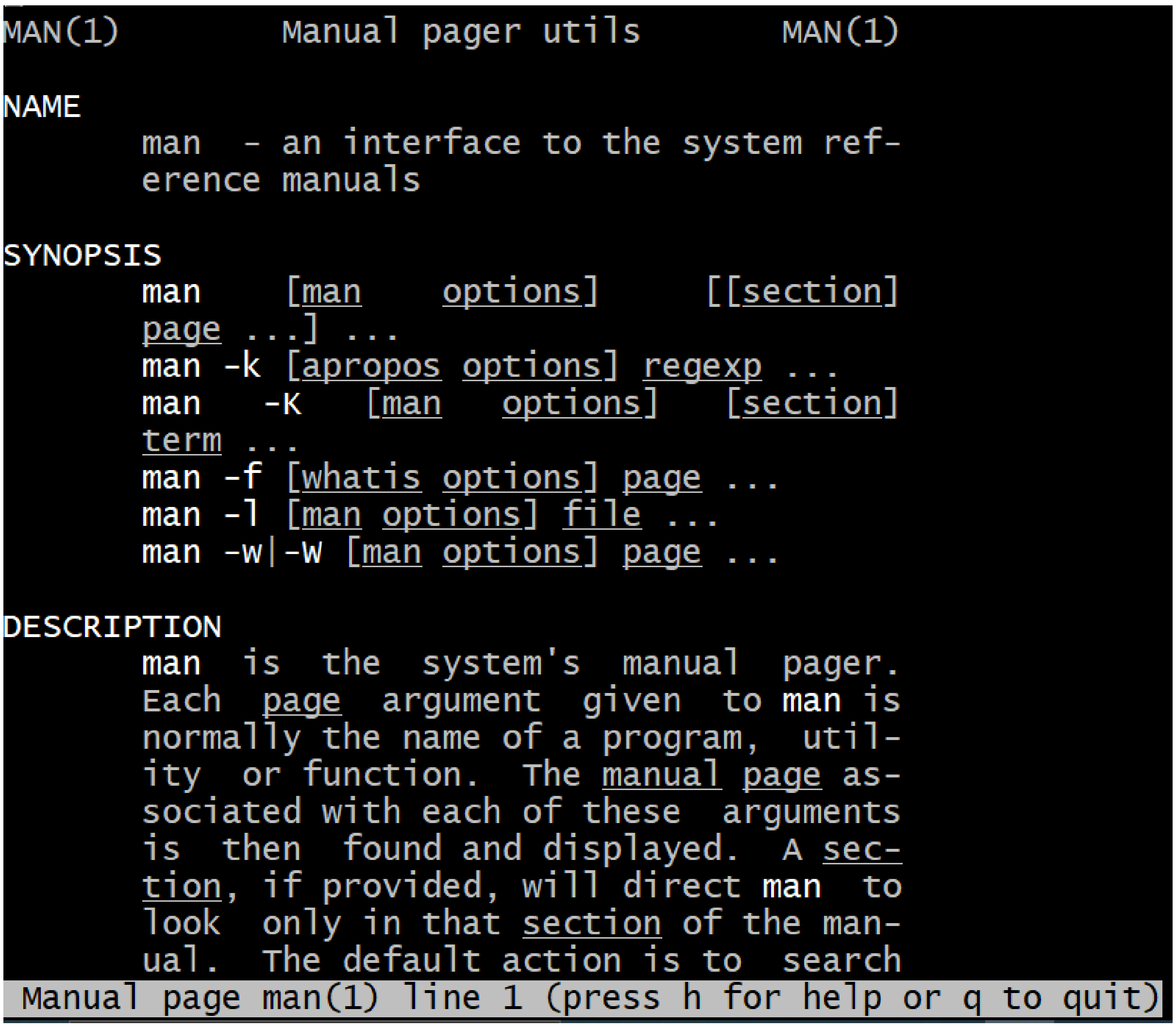
9.history
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ history
    1  vim
    2  ls
    3  mkdir oslab
    4  cd oslab
   .
   .
   .
   65  ls
   66  mv name1 name2
   67  ls
   68  pwd
   69  date
   70  history
```

10.cal
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cal march 2021
     March 2021
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
```

```
28 29 30 31
```

11.man

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ man man
```



```
MAN(1)          Manual pager utils          MAN(1)

NAME
       man  - an interface to the system ref-
       erence manuals

SYNOPSIS
       man    [man    options]     [[section]
       page ...] ...
       man -k [apropos options] regexp ...
       man   -K   [man   options]   [section]
       term ...
       man -f [whatis options] page ...
       man -l [man options] file ...
       man -w|-W [man options] page ...

DESCRIPTION
       man  is  the  system's  manual  pager.
       Each  page  argument  given  to man is
       normally the name of a program,  util-
       ity  or function.  The manual page as-
       sociated with each of these  arguments
       is  then  found and displayed.  A sec-
       tion, if provided, will direct man  to
       look  only in that section of the man-
       ual.  The default action is to  search
 Manual page man(1) line 1 (press h for help or q to quit)
```

12.tty

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ tty
/dev/pty0
```

13.uname

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
```

```
$ uname
CYGWIN_NT-10.0

14.cat
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cat osdemo.txt
small text
for
os demo
class
for da1
so
this
is
txt
file
only
for
demo

15.cal
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cal
     March 2021
Su Mo Tu We Th Fr Sa
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cal 04 2021
     April 2021
Su Mo Tu We Th Fr Sa
             1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
```

```
18 19 20 21 22 23 24
25 26 27 28 29 30
16.who
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ who

17.time
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ time
real    0m0.002s
user    0m0.000s
sys     0m0.000s

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name2  osdemo.txt

18.cp
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cp osdemo.txt osdemo2.txt

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name2  osdemo.txt  osdemo2.txt

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cat osdemo2.txt
small text
for
os demo
class
for da1
so
this
is
txt
file
only
```

```
for
demo

19.rm
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ rm osdemo2.txt

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name2  osdemo.txt

20.mv
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ mv osdemo.txt osdemo1.txt

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ls
name2  osdemo1.txt

21. cat <filename> |wc
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cat osdemo1.txt |wc
     12      16      86


22. cat <filename> |wc -l
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cat osdemo1.txt |wc -l
12

23. cat <filename> |wc -c
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ cat osdemo1.txt |wc -c
86

24. cd <path>
PRAVIN@DESKTOP-B2LB8FB ~
$ cd oslab/da1
```

25.date
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date
Mon Mar 15 13:17:/57 IST 2021
```

26.date +%m
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date +%m
03
```

27.date +%h
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date +%h
Mar
```

28.date +%m%h
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date +%m%h
03Mar
```

29. date -u
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date -u
Mon Mar 15 07:48:44 UTC 2021
```

30.date +%r
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ date +%r
01:18:53 PM
```

31.head
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ head osdemo1.txt
small text
for
os demo
```

```
class
for da1
so
this
is
txt
file
```

32.tail

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ tail osdemo1.txt
class
for da1
so
this
is
txt
file
only
for
demo
```

33.echo $$

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ echo $$
1356
```

34.ps

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1
$ ps
      PID    PPID    PGID     WINPID   TTY         UID    STIME COMMAND
     1375    1356    1375       1280  pty0      197609 13:26:10 /usr/bin/ps
     1355       1    1355       5860  ?         197609 13:03:21 /usr/bin/mintty
     1356    1355    1356       5424  pty0      197609 13:03:23 /usr/bin/bash
```

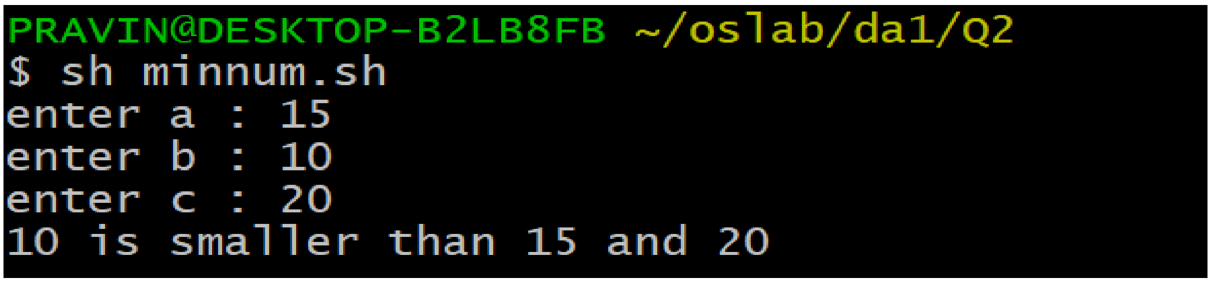
35.clear

```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ clear
```

2. Shell Programming

a. Find the smallest of three numbers

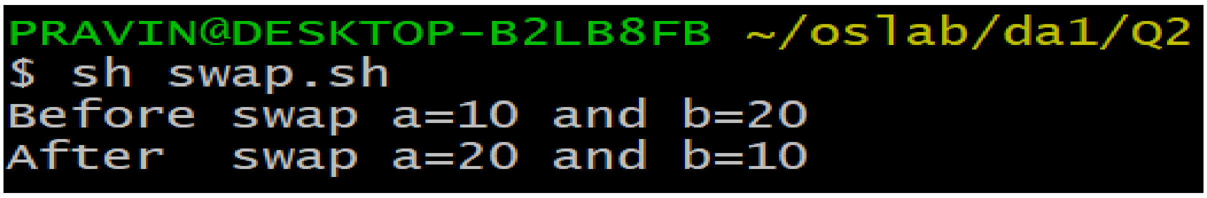
```
echo -n "enter a : "
read a
echo -n "enter b : "
read b
echo -n "enter c : "
read c
if [ $a -lt $b -a $a -lt $c ]
then
    echo "$a is smaller than $b and $c"
elif [ $b -lt $c ]
then
    echo "$b is smaller than $a and $c"
else
    echo "$c is smaller than $a and $b"
fi
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh minnum.sh
enter a : 15
enter b : 10
enter c : 20
10 is smaller than 15 and 20
```

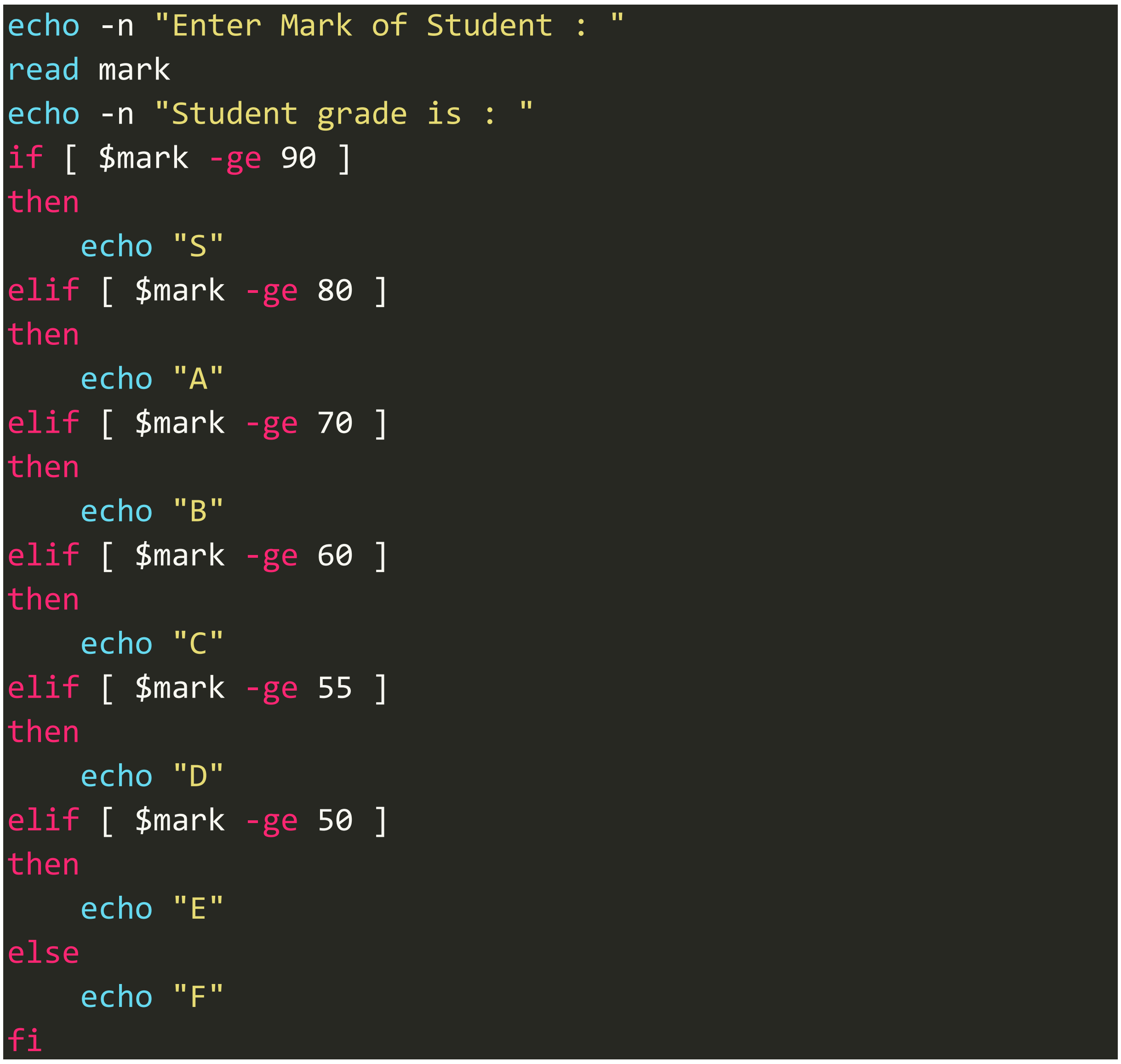
b. Swapping of two numbers without using third variable

```
a=10
b=20
echo "Before swap a=$a and b=$b"
a=`expr $a + $b`
b=`expr $a - $b`
a=`expr $a - $b`
echo "After  swap a=$a and b=$b"
```

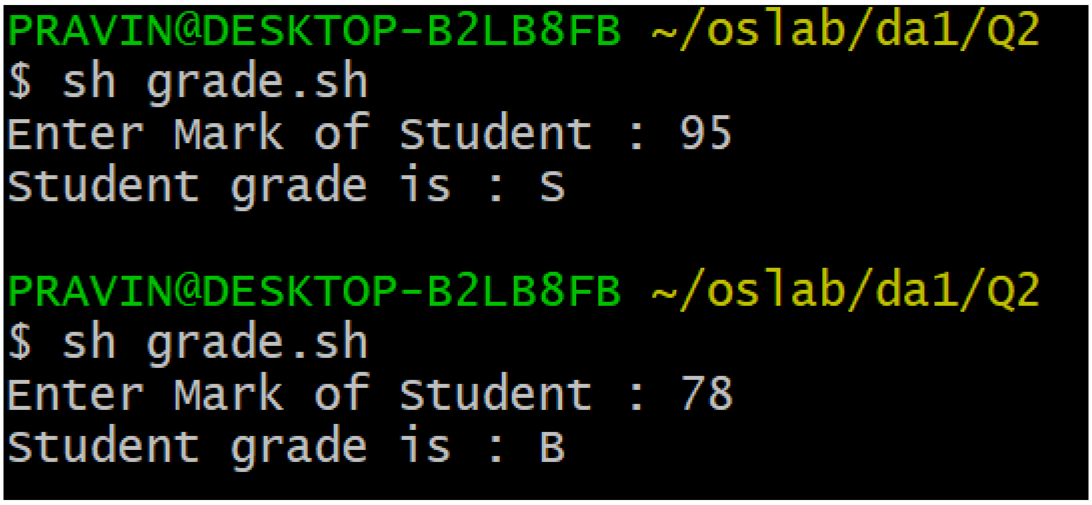


```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh swap.sh
Before swap a=10 and b=20
After  swap a=20 and b=10
```

c. Check the grade of the students based on marks using elif



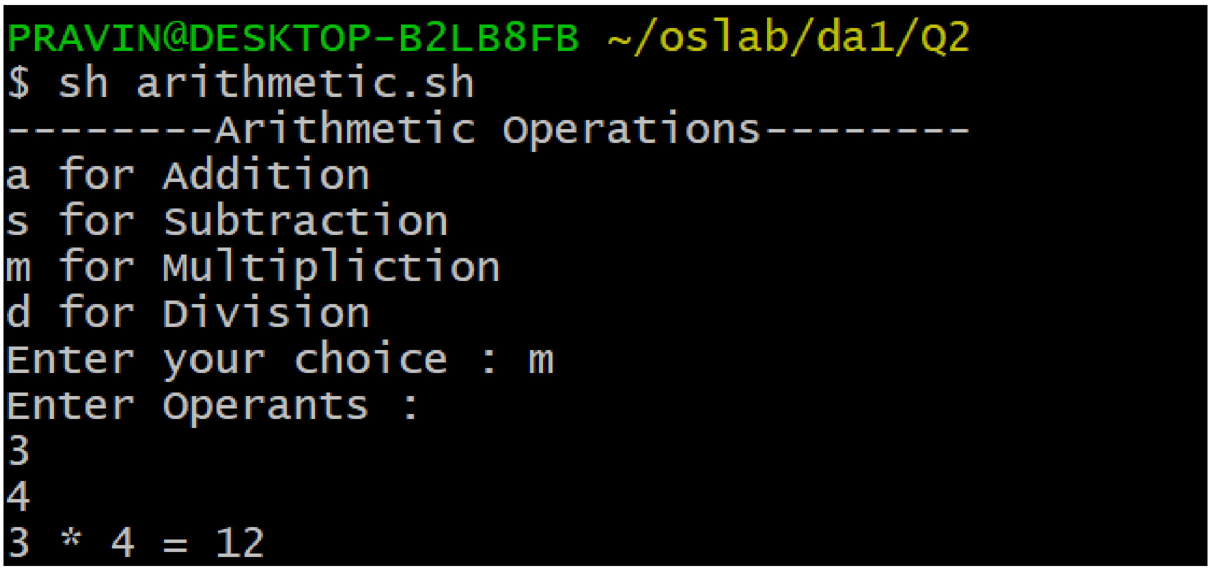
```
echo -n "Enter Mark of Student : "
read mark
echo -n "Student grade is : "
if [ $mark -ge 90 ]
then
    echo "S"
elif [ $mark -ge 80 ]
then
    echo "A"
elif [ $mark -ge 70 ]
then
    echo "B"
elif [ $mark -ge 60 ]
then
    echo "C"
elif [ $mark -ge 55 ]
then
    echo "D"
elif [ $mark -ge 50 ]
then
    echo "E"
else
    echo "F"
fi
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh grade.sh
Enter Mark of Student : 95
Student grade is : S

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh grade.sh
Enter Mark of Student : 78
Student grade is : B
```

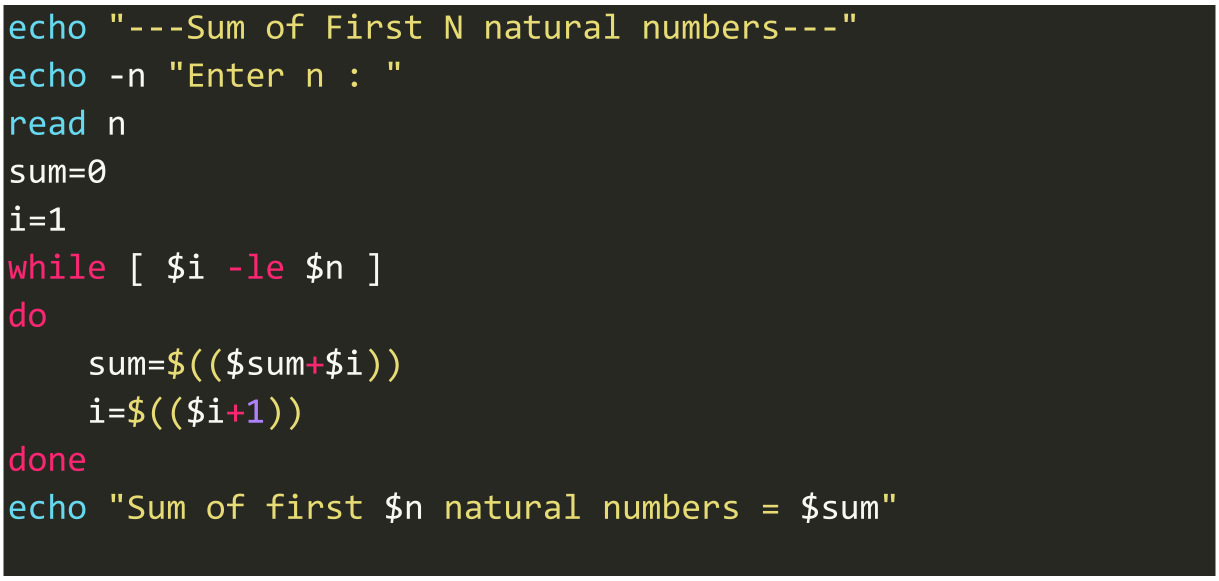
d. Perform basic arithmetic operations based on user choice (Case)

```
echo "--------Arithmetic Operations--------"
echo "a for Addition"
echo "s for Subtraction"
echo "m for Multipliction"
echo "d for Division"
echo -n "Enter your choice : "
read ch
echo "Enter Operants : "
read a
read b
case "$ch" in
    "a")
        echo "$a + $b = $(($a+$b))"
        ;;
    "s")
        echo "$a - $b = $(($a-$b))"
        ;;
    "m")
        echo "$a * $b = $(($a*$b))"
        ;;
    "d")
        echo "$a / $b = $(($a/$b))"
        ;;
esac
```

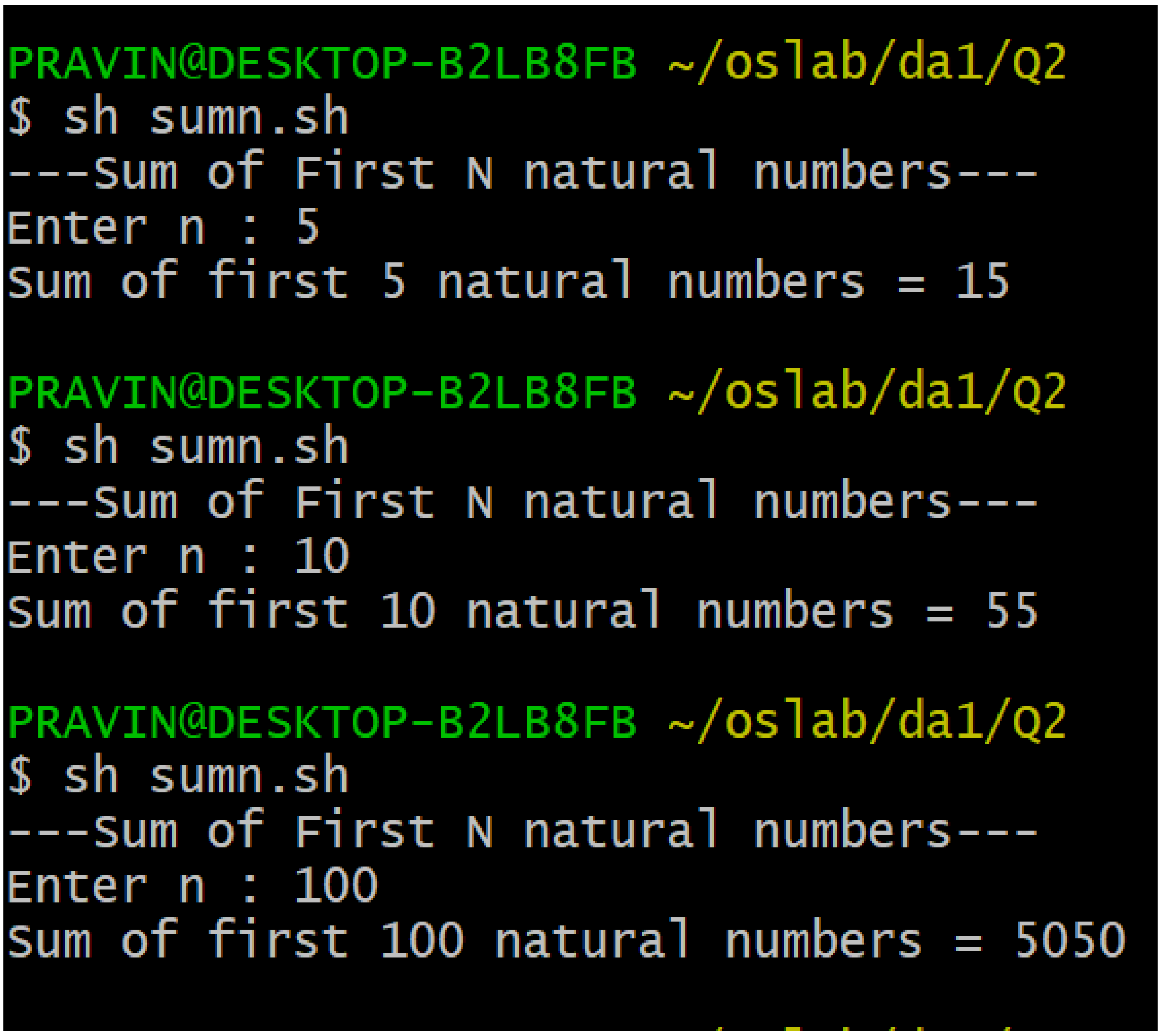


```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh arithmetic.sh
--------Arithmetic Operations--------
a for Addition
s for Subtraction
m for Multipliction
d for Division
Enter your choice : m
Enter Operants :
3
4
3 * 4 = 12
```

e.Find the sum of first n natural numbers



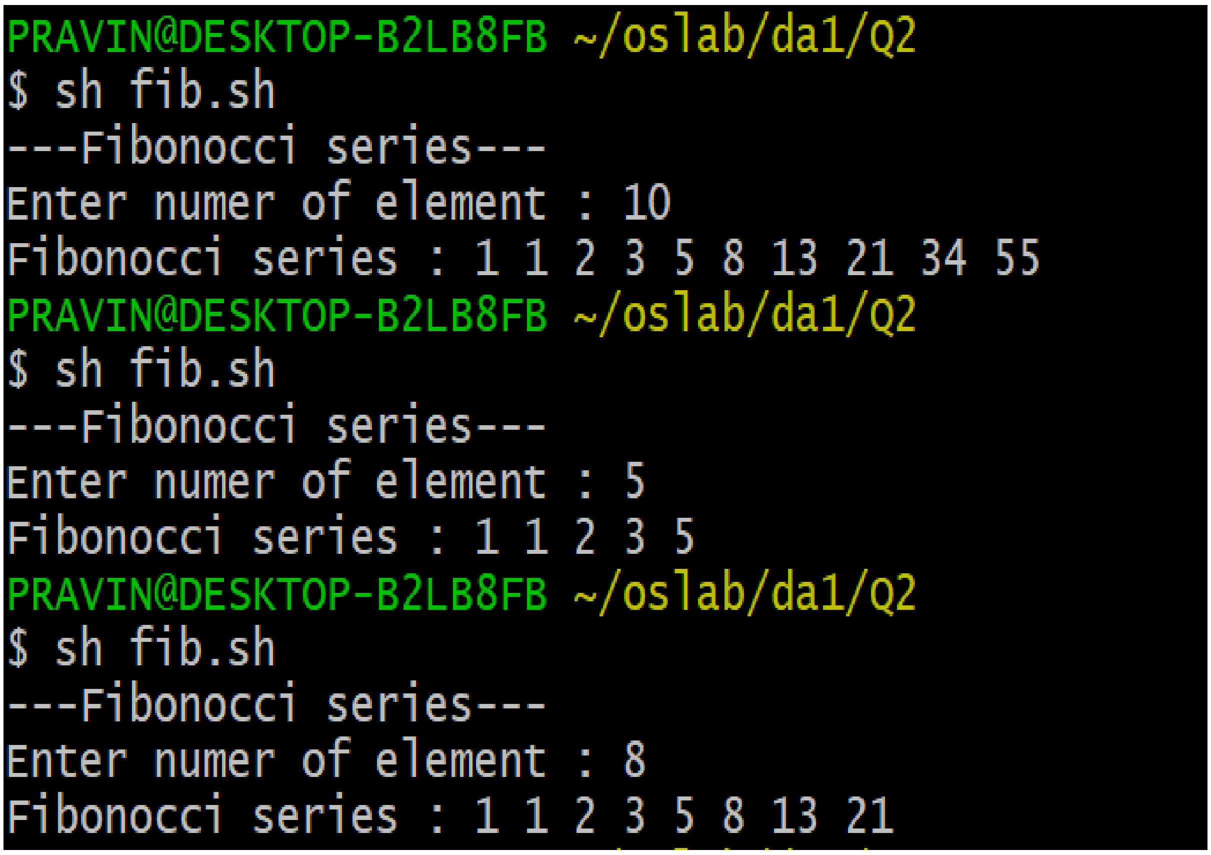
```
echo "---Sum of First N natural numbers---"
echo -n "Enter n : "
read n
sum=0
i=1
while [ $i -le $n ]
do
    sum=$(($sum+$i))
    i=$(($i+1))
done
echo "Sum of first $n natural numbers = $sum"
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh sumn.sh
---Sum of First N natural numbers---
Enter n : 5
Sum of first 5 natural numbers = 15

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh sumn.sh
---Sum of First N natural numbers---
Enter n : 10
Sum of first 10 natural numbers = 55

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh sumn.sh
---Sum of First N natural numbers---
Enter n : 100
Sum of first 100 natural numbers = 5050
```

f. Print the Fibonacci series

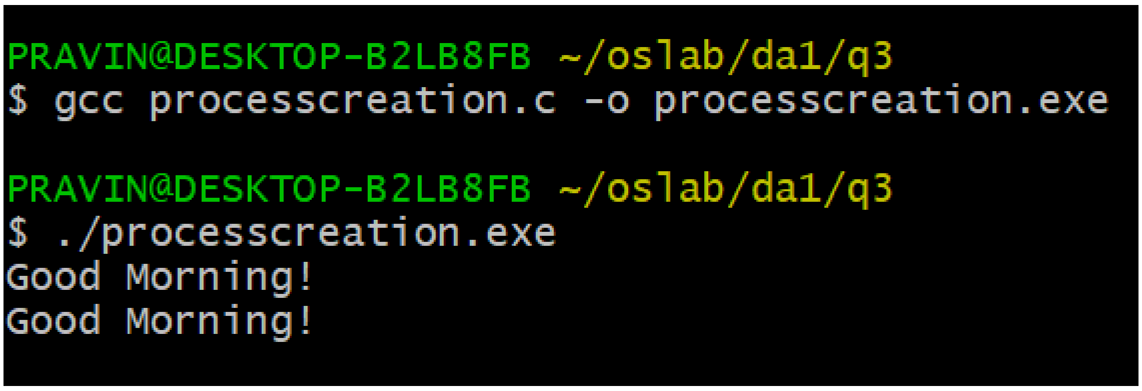
```
echo "---Fibonocci series---"
echo -n "Enter numer of element : "
read n
a=0
b=1
echo -n "Fibonocci series : $b"
i=2
while [ $i -le $n ]
do
    c=$(($a+$b))
    echo -n " $c"
    a=$b
    b=$c
    i=$(($i+1))
done
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh fib.sh
---Fibonocci series---
Enter numer of element : 10
Fibonocci series : 1 1 2 3 5 8 13 21 34 55
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh fib.sh
---Fibonocci series---
Enter numer of element : 5
Fibonocci series : 1 1 2 3 5
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/Q2
$ sh fib.sh
---Fibonocci series---
Enter numer of element : 8
Fibonocci series : 1 1 2 3 5 8 13 21
```

3. Process Creation CPU scheduling algorithms

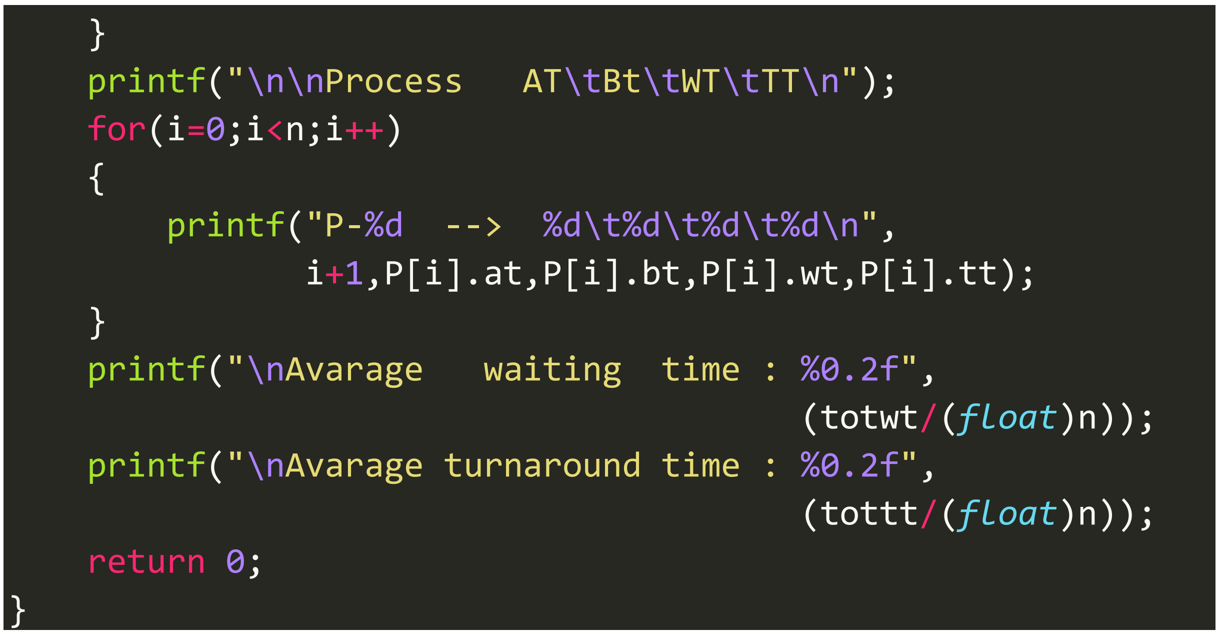
```c
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
int main()
{
    fork();
    printf("Good Morning!\n");
    return 0;
}
```



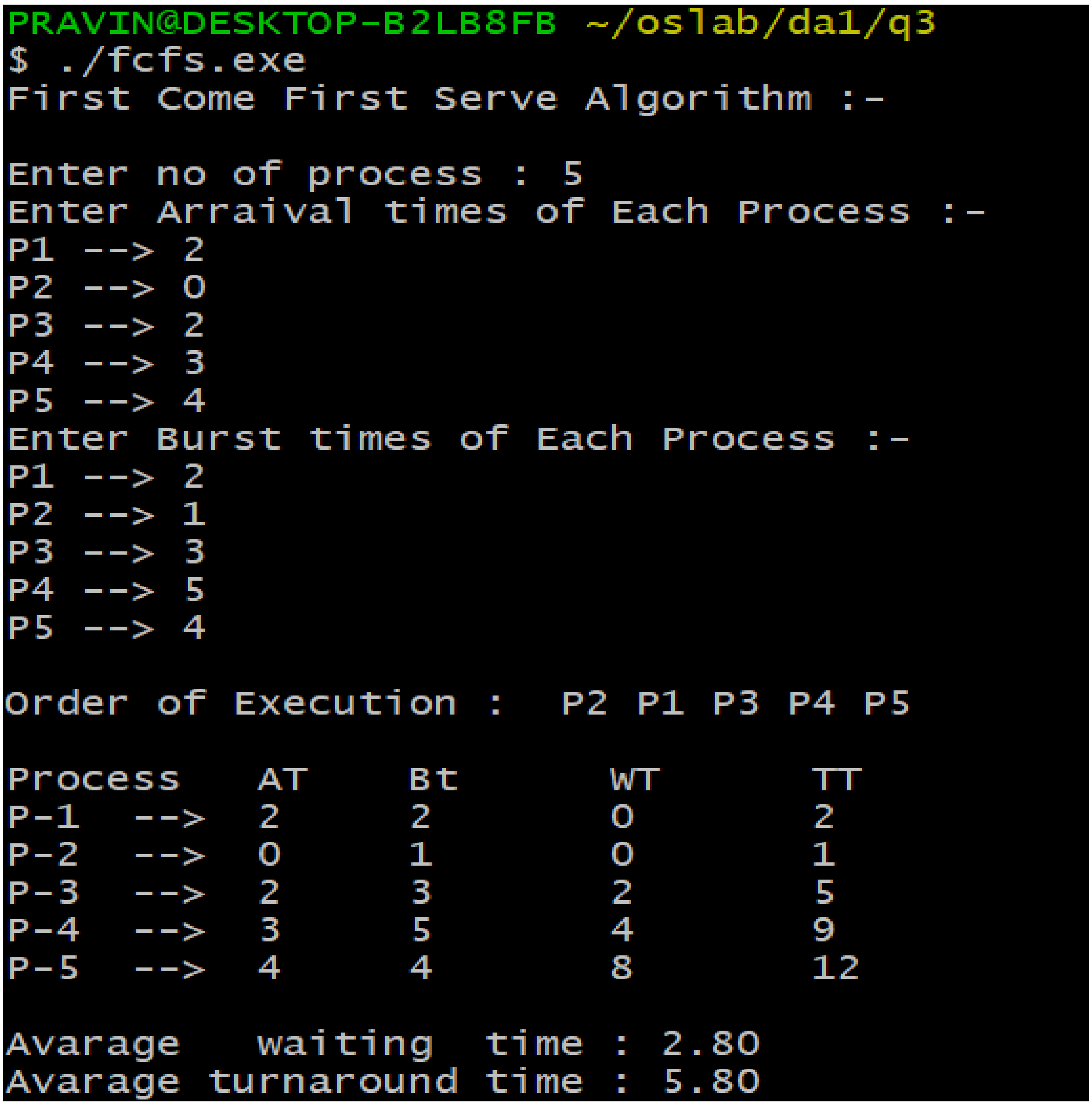
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q3
$ gcc processcreation.c -o processcreation.exe

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q3
$ ./processcreation.exe
Good Morning!
Good Morning!
```

4.First-Come,First-Served Scheduling

```c
#include<stdio.h>
struct process
{
    int at;
    int bt;
    int wt;
    int tt;
};
int main()
{
    printf("First Come First Serve Algorithm :-\n\n");
    struct process P[20];
    int fcfs[20];
    int n,i,j,cur=-1,curat;
    int curtime=0,totwt=0,tottt=0;
    printf("Enter no of process : ");
    scanf("%d",&n);
        printf("Enter Arraival times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].at);
        P[i].wt=0;
        P[i].tt=0;
        fcfs[i]=-1;
    }
    printf("Enter Burst times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].bt);
    }
    printf("\nOrder of Execution : ");
    for(i=0;i<n;i++)
    {
        curat=1000;
        for(j=0;j<n;j++)
        {
            if(fcfs[j]==-1 && P[j].at<curat)
            {
                cur=j;
                curat=P[j].at;
            }
        }
        fcfs[cur]=i;
        if(P[cur].at>curtime)
            curtime=P[cur].at;
        printf(" P%d",cur+1);
        curtime+=P[cur].bt;
        P[cur].tt=curtime-P[cur].at;
        P[cur].wt=P[cur].tt-P[cur].bt;
        totwt+=P[cur].wt;
        tottt+=P[cur].tt;
```



```c
    }
    printf("\n\nProcess   AT\tBt\tWT\tTT\n");
    for(i=0;i<n;i++)
    {
        printf("P-%d  -->  %d\t%d\t%d\t%d\n",
               i+1,P[i].at,P[i].bt,P[i].wt,P[i].tt);
    }
    printf("\nAvarage   waiting  time : %0.2f",
                                        (totwt/(float)n));
    printf("\nAvarage turnaround time : %0.2f",
                                        (tottt/(float)n));
    return 0;
}
```



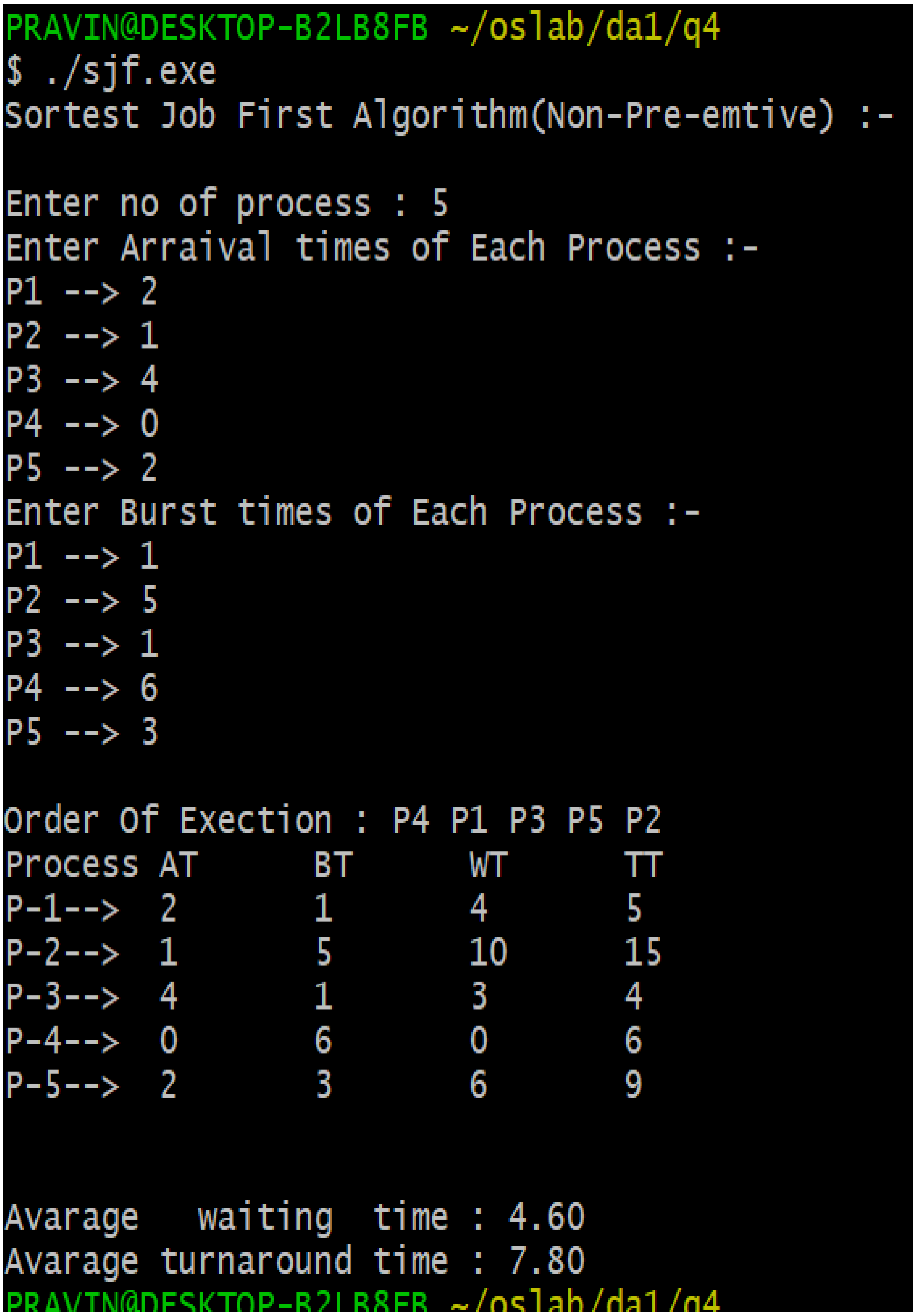
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q3
$ ./fcfs.exe
First Come First Serve Algorithm :-

Enter no of process : 5
Enter Arraival times of Each Process :-
P1 --> 2
P2 --> 0
P3 --> 2
P4 --> 3
P5 --> 4
Enter Burst times of Each Process :-
P1 --> 2
P2 --> 1
P3 --> 3
P4 --> 5
P5 --> 4

Order of Execution :  P2 P1 P3 P4 P5

Process   AT    Bt      WT      TT
P-1  -->  2     2       0       2
P-2  -->  0     1       0       1
P-3  -->  2     3       2       5
P-4  -->  3     5       4       9
P-5  -->  4     4       8       12

Avarage   waiting  time : 2.80
Avarage turnaround time : 5.80
```

5. Shortest-Job-First Scheduling

Non Pre-emtive:-

```
#include<stdio.h>
struct process
{
    int at;
    int bt;
    int wt;
    int tt;
}P[20];

int isNotDone(int* sjf,int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(sjf[i]!=1)
            return 1;
    }
    return 0;
}

int minpro(int* sjf,int n)
{
    int min=1000,minidx=-1,i;
    for(i=0;i<n;i++)
    {
        if(P[i].bt<=min && sjf[i]==0)
        {
            if(P[i].bt<min || P[i].at<P[minidx].at){
            minidx=i;
            min=P[i].bt;
            }
        }
    }
    return minidx;
}
```

```c
int main()
{
    printf("Sortest Job First Algorithm(Non-Pre-emtive) :-
\n\n");
    int sjf[20],readyqueue[20];
    int n,i,j,cur,curtime=0;
    int totwt=0,tottt=0;
    float avgwt,avgtt;
    printf("Enter no of process : ");
    scanf("%d",&n);


    printf("Enter Arraival times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].at);
        P[i].wt=0;
        P[i].tt=0;
        sjf[i]=-1;
    }


    printf("Enter Burst times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].bt);
    }


    printf("Order Of Exection : ");
    while(isNotDone(sjf,n))
    {
        for(i=0;i<n;i++)
        {
            if(sjf[i]==-1 && P[i].at<=curtime)
            {
                sjf[i]=0;
            }
        }
        cur=minpro(sjf,n);
        if(cur==-1)
        {
            curtime+=1;
            continue;
        }
        printf("P%d ",cur+1);
        curtime+=P[cur].bt;
        P[cur].tt=curtime-P[cur].at;
        tottt+=P[cur].tt;
        P[cur].wt=P[cur].tt-P[cur].bt;
        totwt+=P[cur].wt;
        sjf[cur]=1;
    }



    printf("\n\nProcess AT\tBT\tWT\tTT\n");
    for(i=0;i<n;i++)
    {
        printf("P-%d-->  %d\t%d\t%d\t%d\n",
                       i+1,P[i].at,P[i].bt,P[i].wt,P[i].tt);
    }
    printf("\n\nAvarage   waiting  time : %0.2f",
                                          (totwt/(float)n));
    printf("\nAvarage turnaround time : %0.2f",
                                          (tottt/(float)n));
    return 0;
}
```



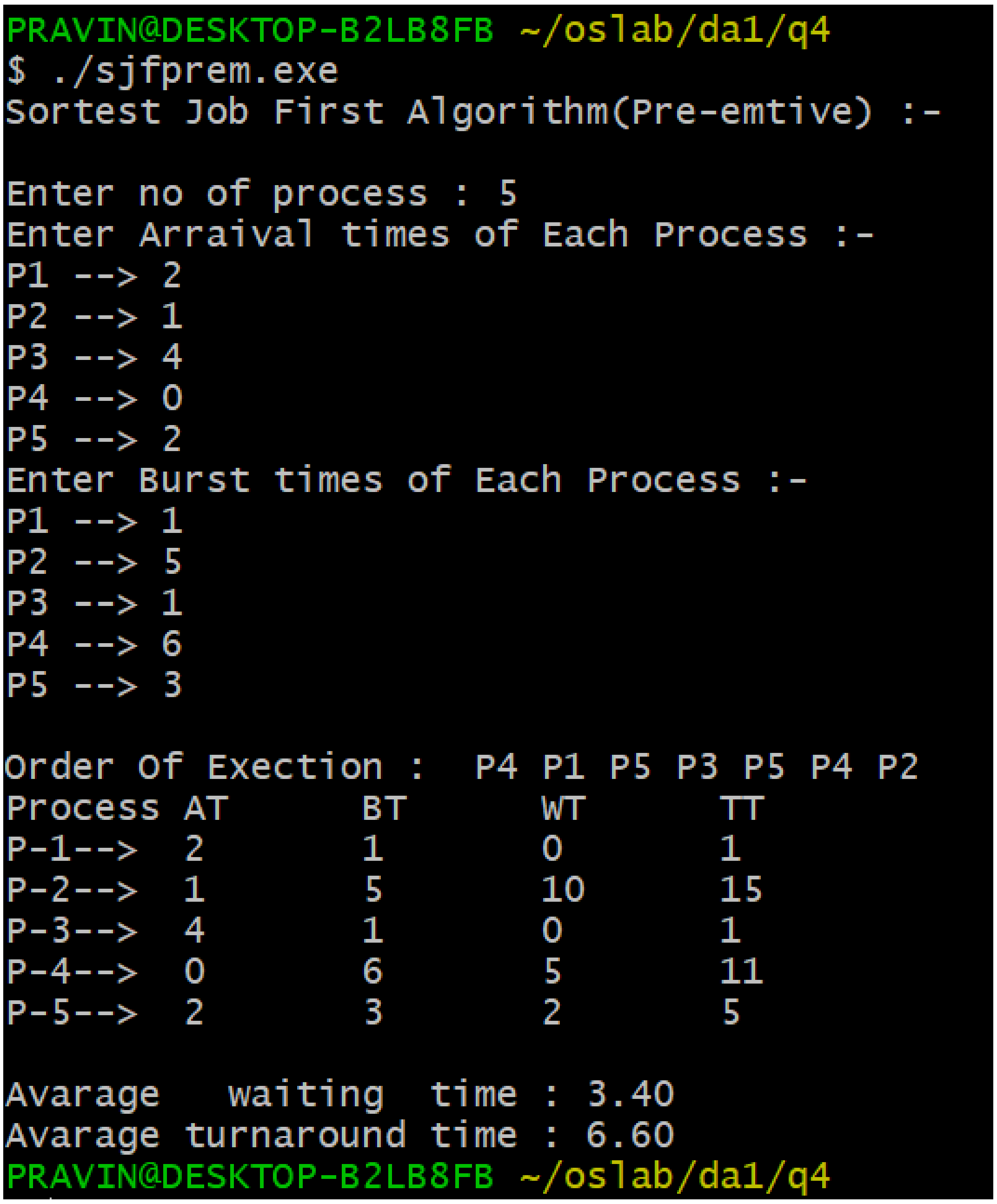
```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ ./sjf.exe
Sortest Job First Algorithm(Non-Pre-emtive) :-

Enter no of process : 5
Enter Arraival times of Each Process :-
P1 --> 2
P2 --> 1
P3 --> 4
P4 --> 0
P5 --> 2
Enter Burst times of Each Process :-
P1 --> 1
P2 --> 5
P3 --> 1
P4 --> 6
P5 --> 3

Order Of Exection : P4 P1 P3 P5 P2
Process AT      BT      WT      TT
P-1-->  2       1       4       5
P-2-->  1       5       10      15
P-3-->  4       1       3       4
P-4-->  0       6       0       6
P-5-->  2       3       6       9


Avarage   waiting  time : 4.60
Avarage turnaround time : 7.80
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
```

Pre-emtive:-

```c
#include<stdio.h>
struct process
{
    int at;
    int bt;
    int rbt;
    int wt;
    int tt;
}P[20];

int isNotDone(int* sjf,int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(sjf[i]!=1)
            return 1;
    }
    return 0;
}
int minpro(int* sjf,int n)
{
    int min=1000,minidx=-1,i;
    for(i=0;i<n;i++)
    {
        if(P[i].rbt<=min && sjf[i]==0)
        {
            if(P[i].rbt<min || P[i].at<P[minidx].at)
            {
            minidx=i;
            min=P[i].rbt;
            }
        }
    }
    return minidx;
}
```

```c
int main()
{
    printf("Sortest Job First Algorithm
                                  (Pre-emtive) :-\n\n");
    int sjf[20];
    int n,i,j,curpro,cur,pre,curtime=0;
    int totwt=0,tottt=0;
    float avgwt,avgtt;


    printf("Enter no of process : ");
    scanf("%d",&n);


    printf("Enter Arraival times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].at);
        P[i].wt=0;
        P[i].tt=0;
        sjf[i]=-1;
    }


    printf("Enter Burst times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].bt);
        P[i].rbt=P[i].bt;
    }
    printf("\nOrder Of Exection : ");



    while(isNotDone(sjf,n))
    {
        for(i=0;i<n;i++)
        {
            if(sjf[i]==-1 && P[i].at<=curtime)
            {
                sjf[i]=0;
            }
        }
        cur=minpro(sjf,n);
        if(cur==-1)
        {
            curtime+=1;
            continue;
        }
        if(cur!=pre)
            printf(" P%d",cur+1);
        P[cur].rbt-=1;
        curtime+=1;
        if(P[cur].rbt==0)
        {
            P[cur].tt=curtime-P[cur].at;
            P[cur].wt=P[cur].tt-P[cur].bt;
            tottt+=P[cur].tt;
            totwt+=P[cur].wt;
            sjf[cur]=1;
        }
        pre=cur;
    }
    printf("\nProcess AT\tBT\tWT\tTT\n");
    for(i=0;i<n;i++)
    {
        printf("P-%d-->  %d\t%d\t%d\t%d\n",
                       i+1,P[i].at,P[i].bt,P[i].wt,P[i].tt);
    }
    printf("\nAvarage   waiting  time : %0.2f",
                                          (totwt/(float)n));
    printf("\nAvarage turnaround time : %0.2f",
                                          (tottt/(float)n));
    return 0;
```

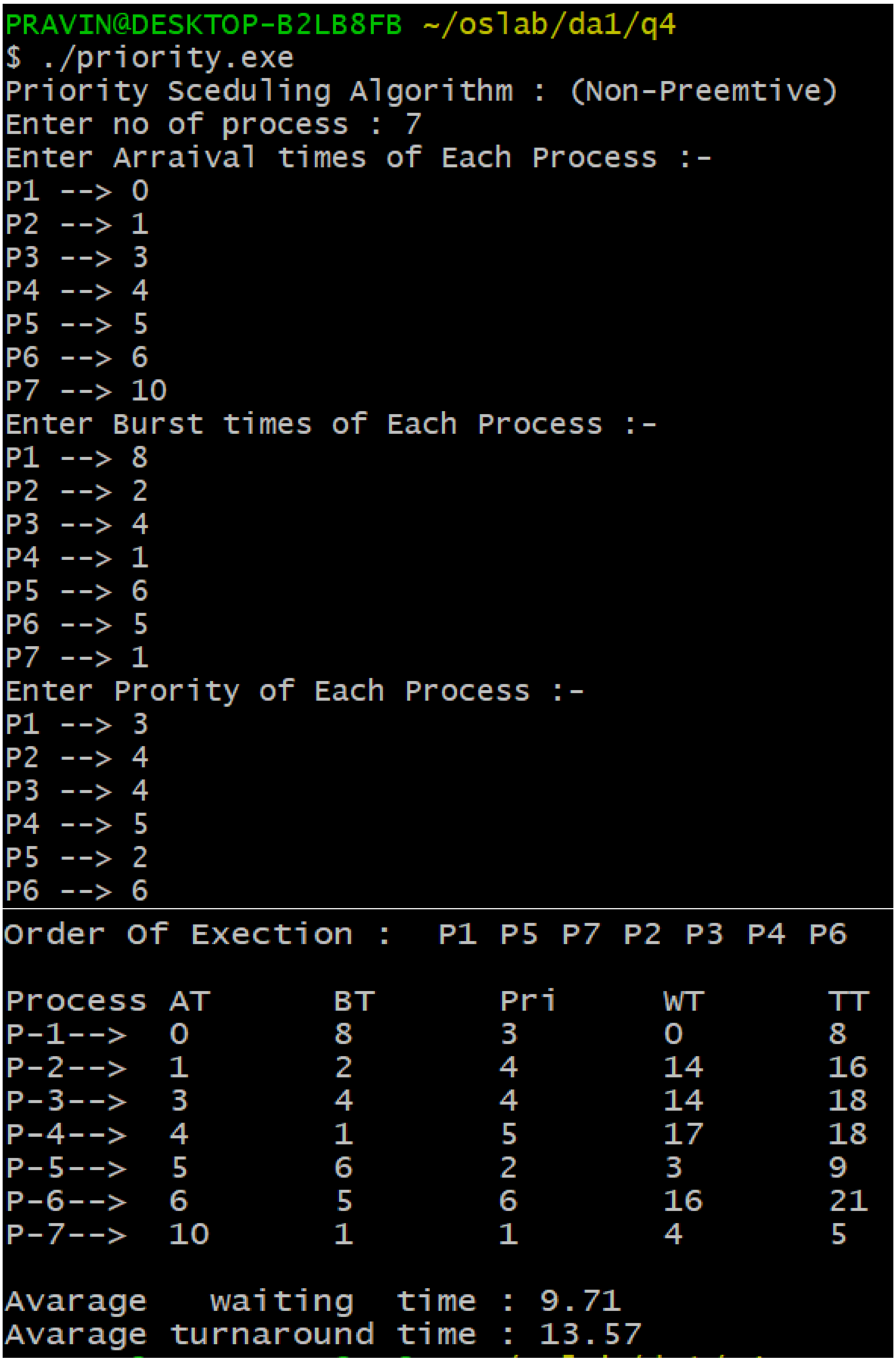
```
}
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ ./sjfprem.exe
Sortest Job First Algorithm(Pre-emtive) :-

Enter no of process : 5
Enter Arraival times of Each Process :-
P1 --> 2
P2 --> 1
P3 --> 4
P4 --> 0
P5 --> 2
Enter Burst times of Each Process :-
P1 --> 1
P2 --> 5
P3 --> 1
P4 --> 6
P5 --> 3

Order Of Exection :  P4 P1 P5 P3 P5 P4 P2
Process AT      BT      WT      TT
P-1-->  2       1       0       1
P-2-->  1       5       10      15
P-3-->  4       1       0       1
P-4-->  0       6       5       11
P-5-->  2       3       2       5

Avarage   waiting  time : 3.40
Avarage turnaround time : 6.60
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
```

6. Priority Scheduling
Non Preemtive:-

```c
#include<stdio.h>
struct process
{
    int priority;
    int at;
    int bt;
    int wt;
    int tt;
}P[20];
int isNotDone(int* psq,int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(psq[i]!=1)
            return 1;
    }
    return 0;
}
int nextpro(int* psq,int n)
{
    int maxpri=1000,idx=-1,i;
    for(i=0;i<n;i++)
    {
        if(P[i].priority<=maxpri && psq[i]==0)
        {
            if(P[i].priority<maxpri || P[i].at<P[idx].at)
            {
            idx=i;
            maxpri=P[i].priority;
            }
        }
    }
    return idx;
}

int main()
{
    printf("Priority Sceduling Algorithm :
                                        (Non-Preemtive)\n");
    int psq[20];
    int n,i,j,curpro,cur,curtime=0;
    int totwt=0,tottt=0;
    float avgwt,avgtt;
    printf("Enter no of process : ");
    scanf("%d",&n);


    printf("Enter Arraival times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].at);
        P[i].wt=0;
        P[i].tt=0;
        psq[i]=-1;
    }


    printf("Enter Burst times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].bt);
    }


    printf("Enter Prority of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].priority);
    }
    printf("\nOrder Of Exection : ");
    while(isNotDone(psq,n))
    {
        for(i=0;i<n;i++)
        {
            if(psq[i]==-1 && P[i].at<=curtime)
            {
                psq[i]=0;
            }
        }
        cur=nextpro(psq,n);
        if(cur==-1)
        {
            curtime+=1;
            continue;
        }
        printf(" P%d",cur+1);
        curtime+=P[cur].bt;
        P[cur].tt=curtime-P[cur].at;
        tottt+=P[cur].tt;
        P[cur].wt=P[cur].tt-P[cur].bt;
        totwt+=P[cur].wt;
        psq[cur]=1;
    }
    printf("\n\nProcess AT\tBT\tPri\tWT\tTT\n");
    for(i=0;i<n;i++)
    {
        printf("P-%d-->  %d\t%d\t%d\t%d\t%d\n",
         i+1,P[i].at,P[i].bt,P[i].priority,P[i].wt,P[i].tt);
    }
    printf("\nAvarage   waiting  time : %0.2f",
                                        (totwt/(float)n));
    printf("\nAvarage turnaround time : %0.2f",
                                        (tottt/(float)n));
    return 0;
}
```

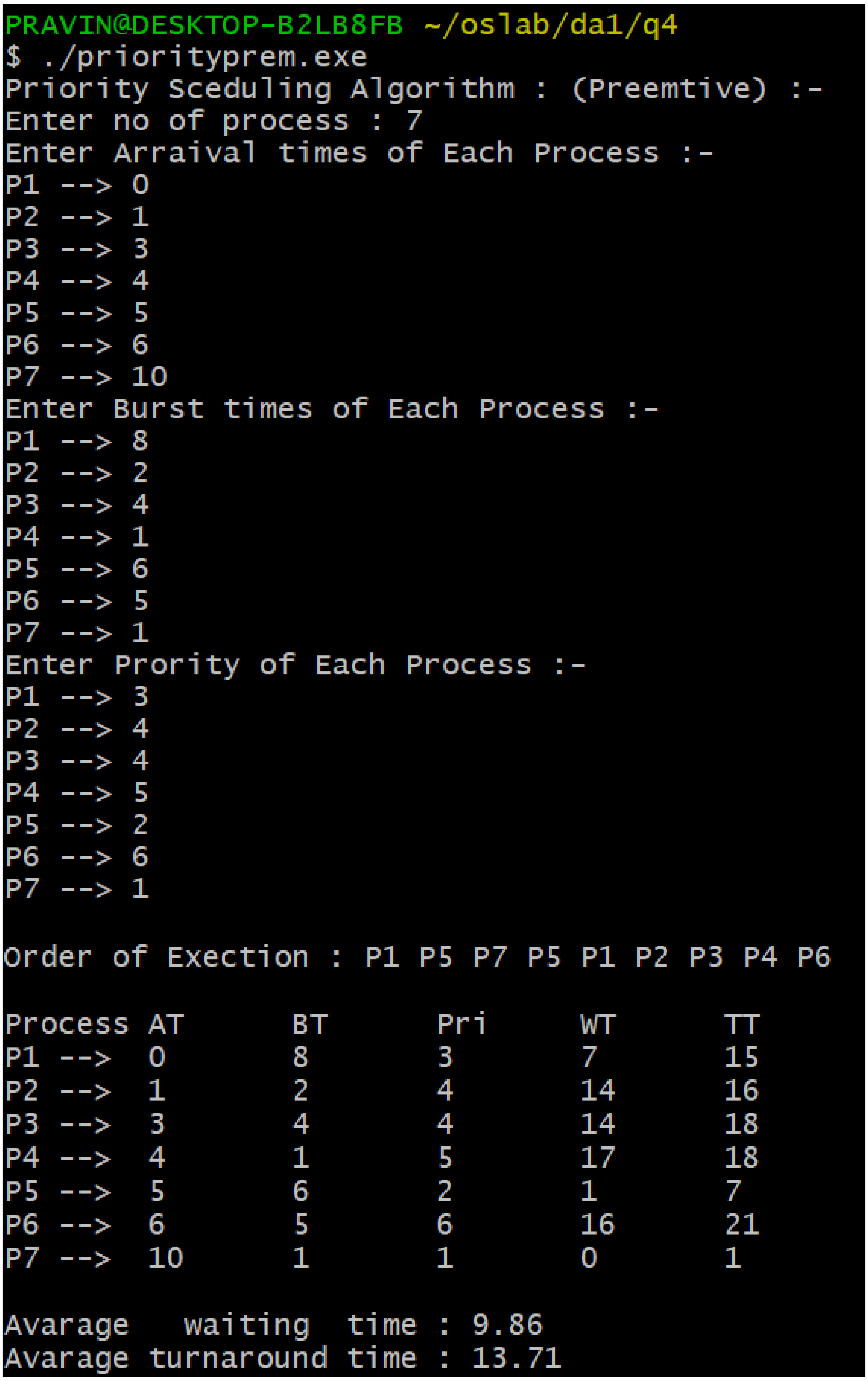


```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ ./priority.exe
Priority Sceduling Algorithm : (Non-Preemtive)
Enter no of process : 7
Enter Arraival times of Each Process :-
P1 --> 0
P2 --> 1
P3 --> 3
P4 --> 4
P5 --> 5
P6 --> 6
P7 --> 10
Enter Burst times of Each Process :-
P1 --> 8
P2 --> 2
P3 --> 4
P4 --> 1
P5 --> 6
P6 --> 5
P7 --> 1
Enter Prority of Each Process :-
P1 --> 3
P2 --> 4
P3 --> 4
P4 --> 5
P5 --> 2
P6 --> 6
Order Of Exection :   P1 P5 P7 P2 P3 P4 P6

Process AT       BT       Pri      WT       TT
P-1-->  0        8        3        0        8
P-2-->  1        2        4        14       16
P-3-->  3        4        4        14       18
P-4-->  4        1        5        17       18
P-5-->  5        6        2        3        9
P-6-->  6        5        6        16       21
P-7-->  10       1        1        4        5

Avarage   waiting  time : 9.71
Avarage turnaround time : 13.57
```

Preemtive:-

```c
#include<stdio.h>
struct process
{
    int priority;
    int at;
    int bt;
    int wt;
    int tt;
    int rbt;
}P[20];
int isNotDone(int* psq,int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        if(psq[i]!=1)
            return 1;
    }
    return 0;
}
int nextpro(int* psq,int n)
{
    int maxpri=1000,idx=-1,i;
    for(i=0;i<n;i++)
    {
        if(P[i].priority<=maxpri && psq[i]==0)
        {
            if(P[i].priority<maxpri || P[i].at<P[idx].at)
            {
            idx=i;
            maxpri=P[i].priority;
            }
        }
    }
    return idx;
```

```
}
int main()
{
    printf("Priority Sceduling Algorithm : (Preemtive) :-
\n");
    int psq[20];
    int n,i,j,curpro,cur,curtime=0,pre=-1;
    int totwt=0,tottt=0;
    float avgwt,avgtt;
    printf("Enter no of process : ");
    scanf("%d",&n);
    printf("Enter Arraival times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].at);
        P[i].wt=0;
        P[i].tt=0;
        psq[i]=-1;
    }
    printf("Enter Burst times of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].bt);
        P[i].rbt=P[i].bt;
    }
    printf("Enter Prority of Each Process :-\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --> ",i+1);
        scanf("%d",&P[i].priority);
    }
    printf("\nOrder of Exection : ");
    while(isNotDone(psq,n))
    {
        for(i=0;i<n;i++)
        {
            if(psq[i]==-1 && P[i].at<=curtime)
            {
                psq[i]=0;
            }
        }
        cur=nextpro(psq,n);
        if(cur==-1)
        {
            curtime+=1;
            continue;
        }
        P[cur].rbt-=1;
        if(pre != cur)
            printf("P%d ",cur+1);
        curtime+=1;
        if(P[cur].rbt==0)
        {
            P[cur].tt=curtime-P[cur].at;
            P[cur].wt=P[cur].tt-P[cur].bt;
            tottt+=P[cur].tt;
            totwt+=P[cur].wt;
            psq[cur]=1;
        }
        pre=cur;
    }
    printf("\n\nProcess AT\tBT\tPri\tWT\tTT\n");
    for(i=0;i<n;i++)
    {
        printf("P%d --
>   %d\t%d\t%d\t%d\t%d\n",i+1,P[i].at,P[i].bt,P[i].priority,P
[i].wt,P[i].tt);
    }
    printf("\nAvarage   waiting  time : %0.2f",(totwt/(float
)n));
    printf("\nAvarage turnaround time : %0.2f",(tottt/(float
)n));
    return 0;
}
```

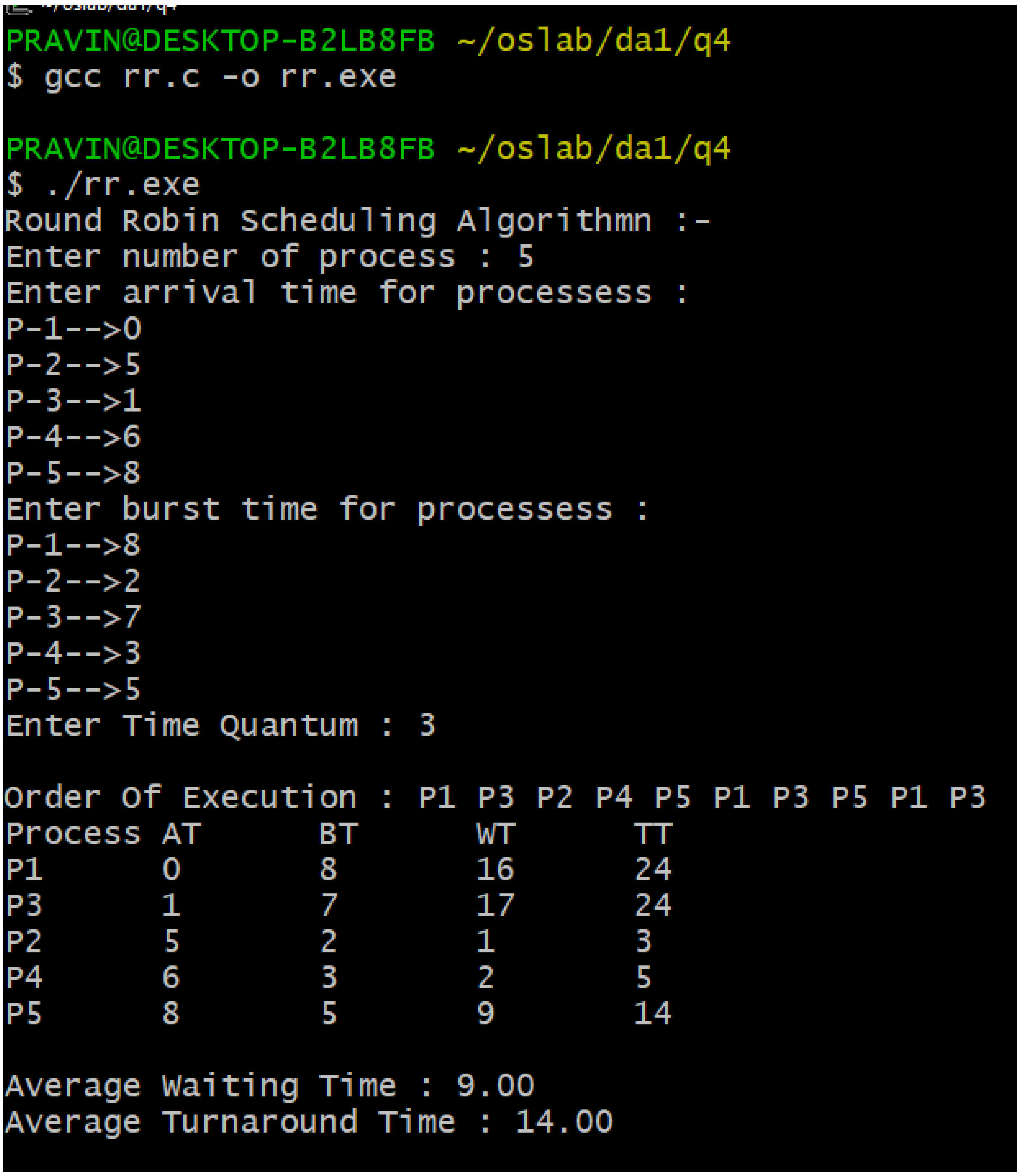


```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ ./priorityprem.exe
Priority Sceduling Algorithm : (Preemtive) :-
Enter no of process : 7
Enter Arraival times of Each Process :-
P1 --> 0
P2 --> 1
P3 --> 3
P4 --> 4
P5 --> 5
P6 --> 6
P7 --> 10
Enter Burst times of Each Process :-
P1 --> 8
P2 --> 2
P3 --> 4
P4 --> 1
P5 --> 6
P6 --> 5
P7 --> 1
Enter Prority of Each Process :-
P1 --> 3
P2 --> 4
P3 --> 4
P4 --> 5
P5 --> 2
P6 --> 6
P7 --> 1

Order of Exection : P1 P5 P7 P5 P1 P2 P3 P4 P6

Process AT       BT       Pri      WT       TT
P1 -->  0        8        3        7        15
P2 -->  1        2        4        14       16
P3 -->  3        4        4        14       18
P4 -->  4        1        5        17       18
P5 -->  5        6        2        1        7
P6 -->  6        5        6        16       21
P7 -->  10       1        1        0        1

Avarage   waiting  time : 9.86
Avarage turnaround time : 13.71
```

7. Round-Robin Scheduling

```c
#include<stdio.h>

struct process
{
	int p;
	int at;
	int bt;
	int wt;
	int tt;
	int rnt;
};



void sortart(struct process P[],int pro)
{
	int i,j;
	struct process temp;

	for(i=0;i<pro;i++)
	{
		for(j=i+1;j<pro;j++)
		{
			if(P[i].at > P[j].at)
			{
				temp = P[i];
				P[i] = P[j];
				P[j] = temp;
			}
		}
	}
	return;
}
```

```
int main()
{
       int i,j,n,time,remain,flag=0,ts;
       struct process P[100];
       int totwt=0,tottt=0;
       printf("Round Robin Scheduling Algorithmn :-\n");

       printf("Enter number of process : ");
       scanf("%d",&n);
       remain=n;
       printf("Enter arrival time for processess :\n");
       for(i=0;i<n;i++)
       {
                     printf("P-%d-->",i+1);
              scanf("%d",&P[i].at);
              P[i].p = i;
       }
       printf("Enter burst time for processess :\n");
       for(i=0;i<n;i++)
       {
                     printf("P-%d-->",i+1);
            scanf("%d",&P[i].bt);
            P[i].rnt = P[i].bt;
       }
       sortart(P,n);
       printf("Enter Time Quantum : ");
       scanf("%d",&ts);
       printf("\nOrder Of Execution :");
       for(time=0,i=0;remain!=0;)
       {
              if(P[i].rnt<=ts && P[i].rnt>0)
              {
                     time = time + P[i].rnt;
                     printf(" P%d",P[i].p+1);
                     P[i].rnt=0;
                     flag=1;
              }
              else if(P[i].rnt > 0)
              {
                     P[i].rnt = P[i].rnt - ts;
                     time = time + ts;
                     printf(" P%d",P[i].p+1);
              }
              if(P[i].rnt==0 && flag==1)
              {
                     remain--;
                     P[i].tt = time-P[i].at;
                     P[i].wt = time-P[i].at-P[i].bt;
                     totwt = totwt + time-P[i].at-P[i].bt;
                     tottt = tottt + time-P[i].at;
                     flag=0;
              }
              if(i==n-1)
                     i=0;
              else if(P[i+1].at <= time)
                     i++;
              else
                     i=0;
       }
       printf("\nProcess\tAT\tBT\tWT\tTT\n");
       for(i=0;i<n;i++)
       {
              printf("P%d\t%d\t%d\t%d\t%d\n",
                 P[i].p+1,P[i].at,P[i].bt,P[i].wt,P[i].tt);
       }

       printf("\nAverage Waiting Time : %.2f\n",
                                         totwt/(float)n);
       printf("Average Turnaround Time : %.2f\n\n",
                                        tottt/(float)n);
       return 0;
}
```



```
PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ gcc rr.c -o rr.exe

PRAVIN@DESKTOP-B2LB8FB ~/oslab/da1/q4
$ ./rr.exe
Round Robin Scheduling Algorithmn :-
Enter number of process : 5
Enter arrival time for processess :
P-1-->0
P-2-->5
P-3-->1
P-4-->6
P-5-->8
Enter burst time for processess :
P-1-->8
P-2-->2
P-3-->7
P-4-->3
P-5-->5
Enter Time Quantum : 3

Order Of Execution : P1 P3 P2 P4 P5 P1 P3 P5 P1 P3
Process AT      BT      WT      TT
P1      0       8       16      24
P3      1       7       17      24
P2      5       2       1       3
P4      6       3       2       5
P5      8       5       9       14

Average Waiting Time : 9.00
Average Turnaround Time : 14.00
```