LAB ASSIGNMENT -5
NAME - AVISEK MANDAL
ROLL-N0 - 102203700
GROUP - 2C035

Name: Avisek Mandal
Roll No.: 102203700
Subgroup: 2C035

LAB ASSIGNMENT 5

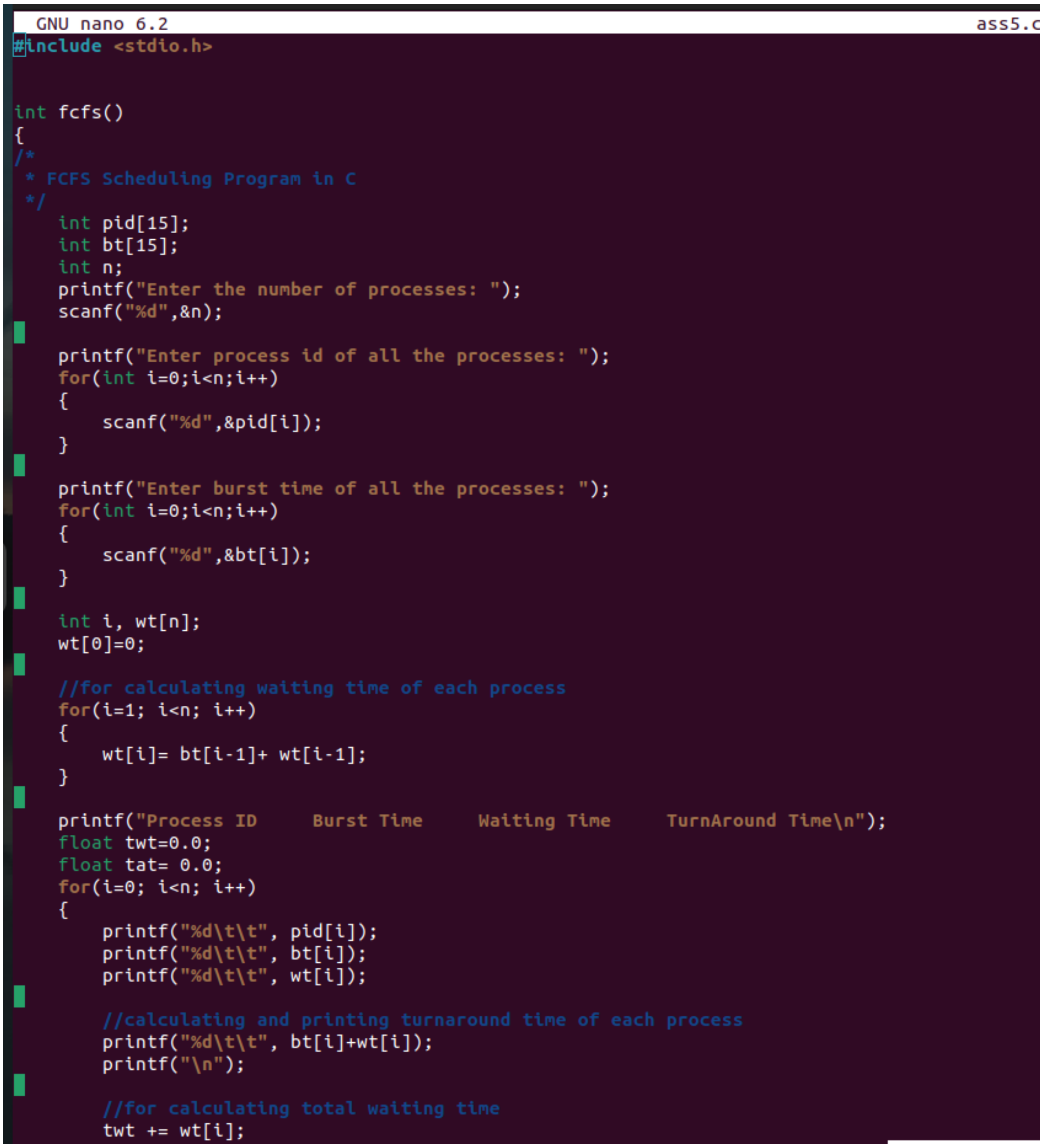
Ques. Write a program using C/C++/Java to simulate the FCFS, SJF (pre-emptive as well as non-preemptive approach). The scenario is: user may input n processes with respective CPU burst time and arrival time. System will ask the user to select the type of algorithm from the list mentioned above. System should display the waiting time for each process, average waiting time for the whole system, and final execution sequence.
Ans.

Code:
```
#include <stdio.h>
int fcfs()
{
/*
 * FCFS Scheduling Program in C
 */
   int pid[15];
   int bt[15];
   int n;
   printf("Enter the number of processes: ");
   scanf("%d",&n);

   printf("Enter process id of all the processes: ");
   for(int i=0;i<n;i++)
   {
      scanf("%d",&pid[i]);
   }

   printf("Enter burst time of all the processes: ");
   for(int i=0;i<n;i++)
   {
      scanf("%d",&bt[i]);
   }

   int i, wt[n];
   wt[0]=0;

   //for calculating waiting time of each process
   for(i=1; i<n; i++)
   {
      wt[i]= bt[i-1]+ wt[i-1];
```

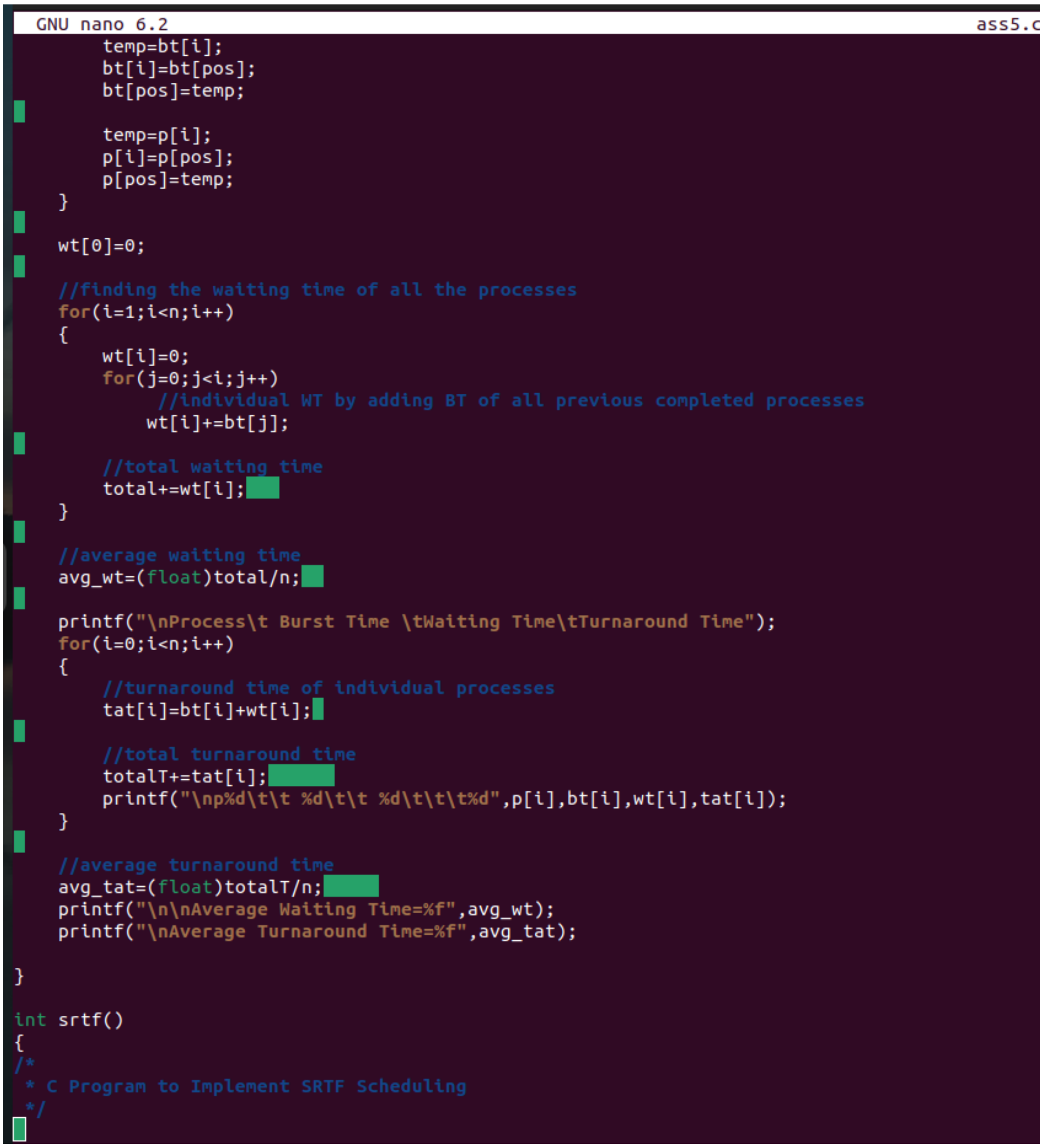
```c
    }

    printf("Process ID     Burst Time     Waiting Time     TurnAround Time\n");
    float twt=0.0;
    float tat= 0.0;
    for(i=0; i<n; i++)
    {
       printf("%d\t\t", pid[i]);
       printf("%d\t\t", bt[i]);
       printf("%d\t\t", wt[i]);

       //calculating and printing turnaround time of each process
       printf("%d\t\t", bt[i]+wt[i]);
       printf("\n");

       //for calculating total waiting time
       twt += wt[i];

       //for calculating total turnaround time
       tat += (wt[i]+bt[i]);
    }
    float att,awt;

    //for calculating average waiting time
    awt = twt/n;

    //for calculating average turnaround time
    att = tat/n;
    printf("Avg. waiting time= %f\n",awt);
    printf("Avg. turnaround time= %f",att);
}

int sjf()
{
/*
 * C Program to Implement SJF Scheduling
 */
    int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,totalT=0,pos,temp;
    float avg_wt,avg_tat;
    printf("Enter number of process:");
    scanf("%d",&n);

    printf("\nEnter Burst Time:\n");
    for(i=0;i<n;i++)
    {
```

```
    printf("p%d:",i+1);
    scanf("%d",&bt[i]);
    p[i]=i+1;
}

//sorting of burst times
for(i=0;i<n;i++)
{
    pos=i;
    for(j=i+1;j<n;j++)
    {
        if(bt[j]<bt[pos])
            pos=j;
    }

    temp=bt[i];
    bt[i]=bt[pos];
    bt[pos]=temp;

    temp=p[i];
    p[i]=p[pos];
    p[pos]=temp;
}

wt[0]=0;

//finding the waiting time of all the processes
for(i=1;i<n;i++)
{
    wt[i]=0;
    for(j=0;j<i;j++)
         //individual WT by adding BT of all previous completed processes
        wt[i]+=bt[j];

    //total waiting time
    total+=wt[i];
}

//average waiting time
avg_wt=(float)total/n;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");
for(i=0;i<n;i++)
{
    //turnaround time of individual processes
```

```
        tat[i]=bt[i]+wt[i];

        //total turnaround time
        totalT+=tat[i];
        printf("\np%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);
    }

    //average turnaround time
    avg_tat=(float)totalT/n;
    printf("\n\nAverage Waiting Time=%f",avg_wt);
    printf("\nAverage Turnaround Time=%f",avg_tat);

}

int srtf()
{
/*
 * C Program to Implement SRTF Scheduling
 */
 int at[10],bt[10],rt[10],endTime,i,smallest;
 int remain=0,n,time,sum_wait=0,sum_turnaround=0;
 printf("Enter no of Processes : ");
 scanf("%d",&n);

 for(i=0;i<n;i++)
{
 printf("Enter arrival time for Process P%d : ",i+1);
 scanf("%d",&at[i]);
 printf("Enter burst time for Process P%d : ",i+1);
 scanf("%d",&bt[i]);
 rt[i]=bt[i];
 }

 printf("\n\nProcess\t\tTurn around Time\t\tWaiting Time\n\n");
 rt[9]=9999;
 for(time=0;remain!=n;time++)
 {
 smallest=9;
 for(i=0;i<n;i++)
 {
 if(at[i]<=time && rt[i]<rt[smallest] && rt[i]>0)
 {
 smallest=i;
 }
 }
 rt[smallest]--;
 if(rt[smallest]==0)
 {
  remain++;
 endTime=time+1;
 printf("\nP%d\t\t\t%d\t\t\t%d",smallest+1,endTime-at[smallest],endTime-bt[smallest]-
at[smallest]);

 sum_wait+=endTime-bt[smallest]-at[smallest];
 sum_turnaround+=endTime-at[smallest];

 }
 }

 printf("\n\nAverage waiting time = %f\n",sum_wait*1.0/n);
 printf("Average Turnaround time = %f",sum_turnaround*1.0/5);
 }

  int main()
{
int val;
printf("Enter the number written against the task to be performed:\n");
printf("1. FCFS\n2. SJF\n3. SRTF\n");
scanf("%d",&val);

switch (val) {
case 1:
fcfs();
break;
case 2:
sjf();
break;
case 3:
srtf();
break;
default:
 printf("The number entered is not available");
}
 return 0;
 }
```

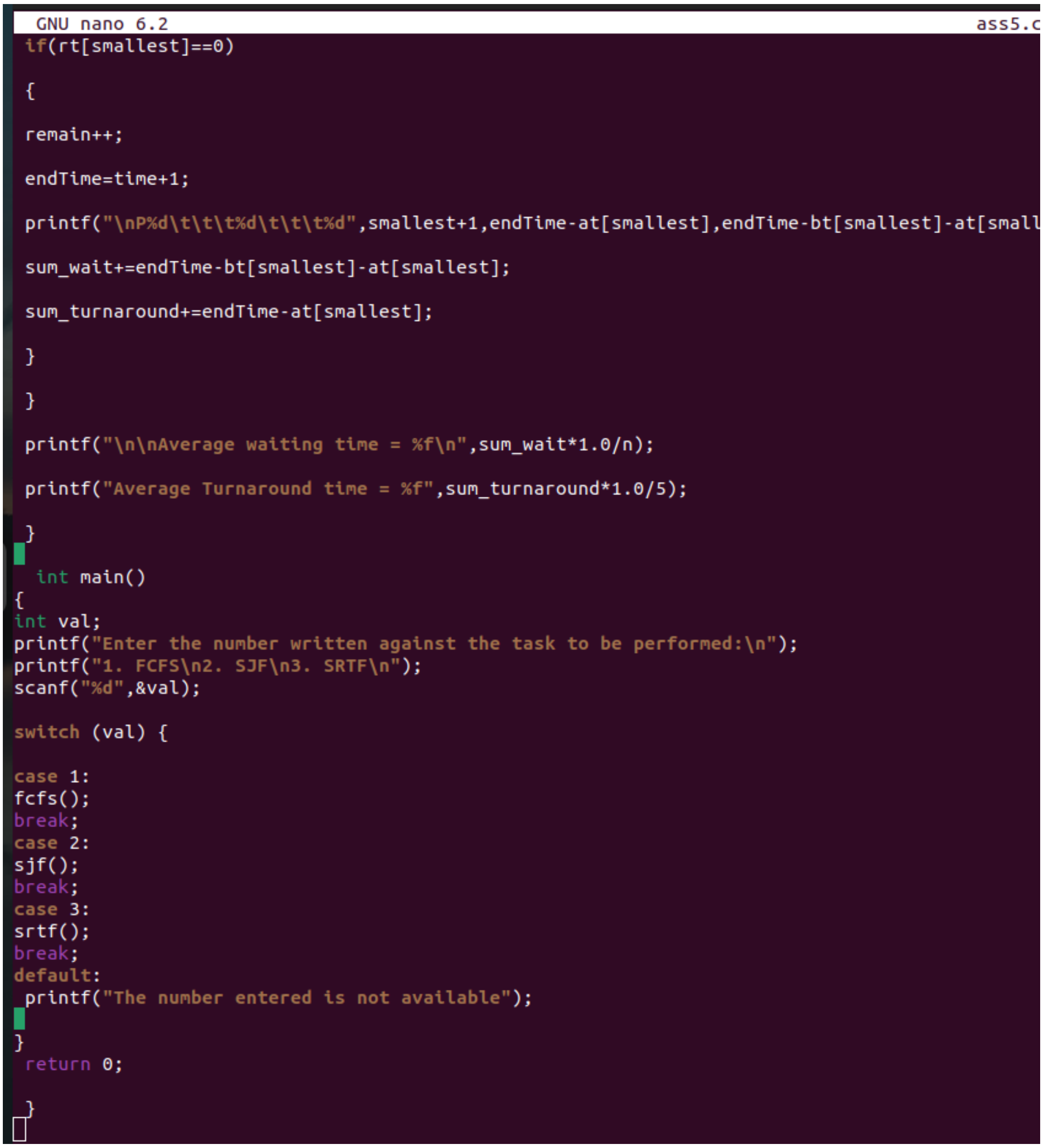


```
  GNU nano 6.2                                                                                     ass5.c
#include <stdio.h>


int fcfs()
{
/*
 * FCFS Scheduling Program in C
 */
    int pid[15];
    int bt[15];
    int n;
    printf("Enter the number of processes: ");
    scanf("%d",&n);

    printf("Enter process id of all the processes: ");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&pid[i]);
    }

    printf("Enter burst time of all the processes: ");
    for(int i=0;i<n;i++)
    {
        scanf("%d",&bt[i]);
    }

    int i, wt[n];
    wt[0]=0;

    //for calculating waiting time of each process
    for(i=1; i<n; i++)
    {
        wt[i]= bt[i-1]+ wt[i-1];
    }

    printf("Process ID     Burst Time     Waiting Time     TurnAround Time\n");
    float twt=0.0;
    float tat= 0.0;
    for(i=0; i<n; i++)
    {
        printf("%d\t\t", pid[i]);
        printf("%d\t\t", bt[i]);
        printf("%d\t\t", wt[i]);

        //calculating and printing turnaround time of each process
        printf("%d\t\t", bt[i]+wt[i]);
        printf("\n");

        //for calculating total waiting time
        twt += wt[i];
```



```
 GNU nano 6.2                                                                          ass5.c
        temp=bt[i];
        bt[i]=bt[pos];
        bt[pos]=temp;

        temp=p[i];
        p[i]=p[pos];
        p[pos]=temp;
    }

    wt[0]=0;

    //finding the waiting time of all the processes
    for(i=1;i<n;i++)
    {
        wt[i]=0;
        for(j=0;j<i;j++)
             //individual WT by adding BT of all previous completed processes
            wt[i]+=bt[j];

        //total waiting time
        total+=wt[i];
    }

    //average waiting time
    avg_wt=(float)total/n;

    printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");
    for(i=0;i<n;i++)
    {
        //turnaround time of individual processes
        tat[i]=bt[i]+wt[i];

        //total turnaround time
        totalT+=tat[i];
        printf("\np%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);
    }

    //average turnaround time
    avg_tat=(float)totalT/n;
    printf("\n\nAverage Waiting Time=%f",avg_wt);
    printf("\nAverage Turnaround Time=%f",avg_tat);

}

int srtf()
{
/*
 * C Program to Implement SRTF Scheduling
 */
```

Output:

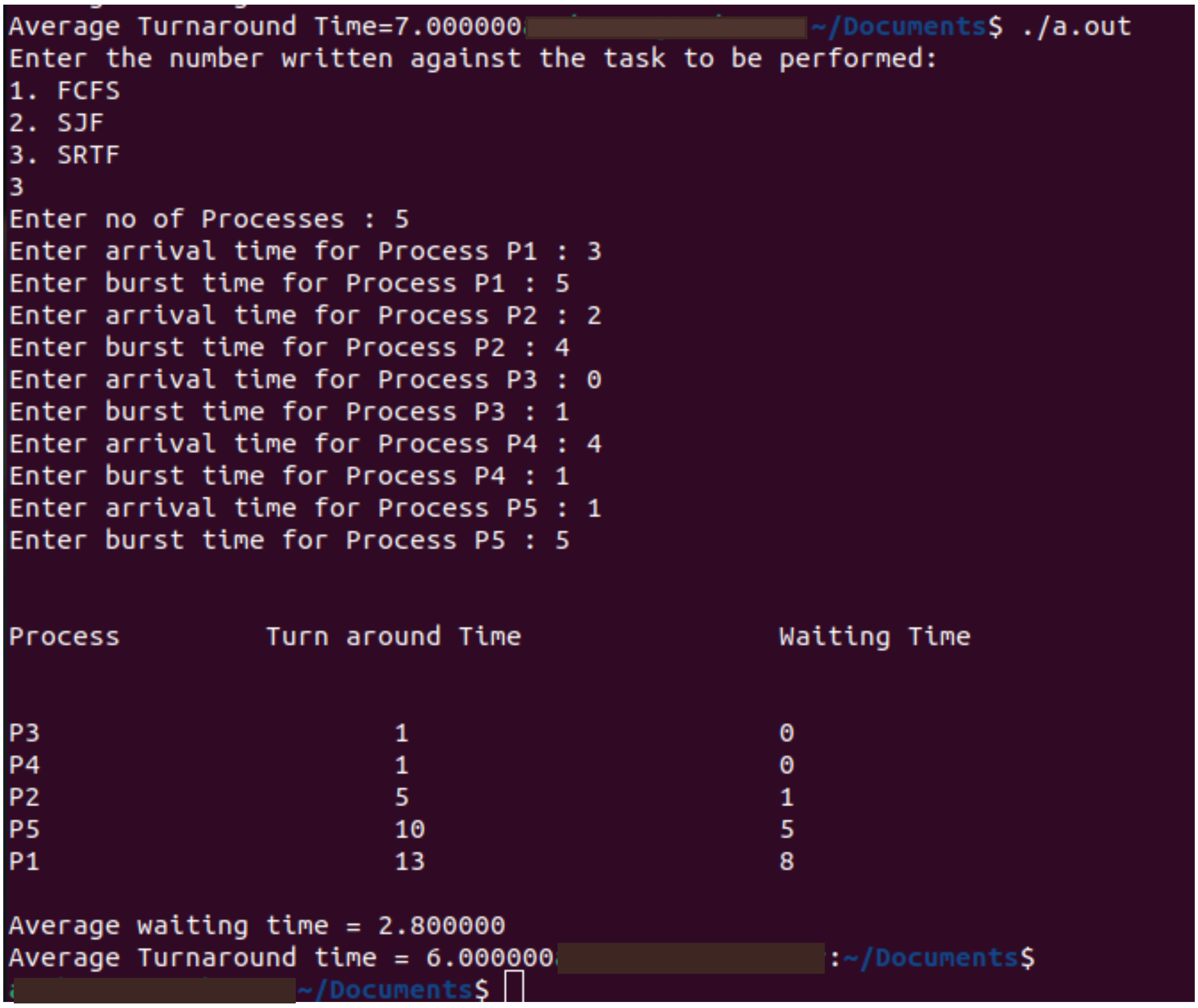


```
  GNU nano 6.2                                                                 ass5.c
 if(rt[smallest]==0)

 {

 remain++;

 endTime=time+1;

 printf("\nP%d\t\t\t%d\t\t\t%d",smallest+1,endTime-at[smallest],endTime-bt[smallest]-at[small

 sum_wait+=endTime-bt[smallest]-at[smallest];

 sum_turnaround+=endTime-at[smallest];

 }

 }

 printf("\n\nAverage waiting time = %f\n",sum_wait*1.0/n);

 printf("Average Turnaround time = %f",sum_turnaround*1.0/5);

 }

  int main()
{
int val;
printf("Enter the number written against the task to be performed:\n");
printf("1. FCFS\n2. SJF\n3. SRTF\n");
scanf("%d",&val);

switch (val) {

case 1:
fcfs();
break;
case 2:
sjf();
break;
case 3:
srtf();
break;
default:
 printf("The number entered is not available");

}
 return 0;

 }
```

FCFS:

SJF:

SRTF:



```
Average Turnaround Time=7.000000[redacted]~/Documents$ ./a.out
Enter the number written against the task to be performed:
1. FCFS
2. SJF
3. SRTF
3
Enter no of Processes : 5
Enter arrival time for Process P1 : 3
Enter burst time for Process P1 : 5
Enter arrival time for Process P2 : 2
Enter burst time for Process P2 : 4
Enter arrival time for Process P3 : 0
Enter burst time for Process P3 : 1
Enter arrival time for Process P4 : 4
Enter burst time for Process P4 : 1
Enter arrival time for Process P5 : 1
Enter burst time for Process P5 : 5


Process         Turn around Time                Waiting Time


P3                      1                       0
P4                      1                       0
P2                      5                       1
P5                      10                      5
P1                      13                      8

Average waiting time = 2.800000
Average Turnaround time = 6.000000[redacted]:~/Documents$
[redacted]~/Documents$
```