Name : Ashraf Ali
Reg No : 11810784
Email Address: khanashrafali.1997@gmail.com
GitHub Link: https://github.com/khanashrafali22/Operating-System-Assignment
Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>

int main()
{

  int p,i,j,m=0,n,y=0;
  int time,remain=0,min,flag=0;
  int wt=0,tat=0,at[10],bt[10],S[10],z[10];
  float k=0,x=0;
  printf("\nEnter number of Process: ");
  scanf("%d",&n);



  for(p=0;p<n;p++)
  {
    printf("\nEnter Arrival Time and Burst Time for Process  %d :",p+1);
    scanf("%d",&at[p]);
    scanf("%d",&bt[p]);
  }

for(i=0;i<n;i++)
{
          if(at[i]==0)
          {
```

```
                printf("\nIN THIS CASE ARRIVAL TIME CAN'T BE ZERO'\n");
                getch();
                exit(1);
        }
}
for(i=0;i<n;i++)
{
        if(at[i]<3)
        {
                printf("\nArrival should be greater than 3\n");
                getch();
                exit(1);
        }
}
  printf("\n\nProcess\t\tTurnaround Time\t\tWaiting Time\n");

for(i=0;i<n;i++)
{
        m=m+bt[i];
}
min=m;
time=m;
for(i=0;i<n;i++)
{
        if(at[i]<time)
        {
                time=at[i];
        }
}
for(i=time;i<=m;i=i+bt[j])
```

```
{
        min=m;
        remain=0;
        flag=0;

        for(p=0;p<n;p++)
        {

                if(at[p]<=i)
                {

                        if(bt[p]<min)
                        {

                                min = bt[p];
                                j=p;
                                flag=1;
                        }
                        remain=1;
                }
        }
        if(flag==1&&remain==1)
        {
                wt=i-at[j];
                tat=wt+bt[j];
                printf("P[%d]\t\t%d\t\t\t%d\n\n",j+1,tat,wt);
                k=k+wt;
                x=x+tat;

                at[j]=m+1;
```

```
                S[y]=j+1;
                z[y]=i;
                y++;
        }
}
printf("Average Waiting Time= %.2f\n\n",k/n);
printf("Avg Turnaround Time = %.2f\n\n",x/n);
printf("Queue  for order of execution:\n");
printf("Process                 ");

for(i=0;i<n;i++)
{
        printf(" P[%d]   ",S[i]);
        if(i==(n-1))
        {
                printf(".");
        }
}
  return 0;
}
```

Problem:

Design a scheduling program that is capable of scheduling many processes that comes in at some time interval and are allocated the CPU not more that 10-time units. CPU must schedule processes having short execution time first. CPU is idle for 3-time units and does not entertain any process prior this time. Scheduler must maintain a queue that keeps the order of execution of all the processes. Compute average waiting and turnaround time.

Description:

This program is based on the Shortest Remaining Time First (SRTF). It is the pre-emptive version of Shortest Job Next (SJN) algorithm, where the processor is allocated to the job

closest to completion. This algorithm requires advanced concept and knowledge of CPU time required to process the job in an interactive system, and hence can't be implemented there. But, in a batch system where it is desirable to give preference to short jobs, SRT algorithm is used.

However, SRT involves more overheads than SJN, as the OS is required to frequently monitor the CPU time of the jobs in the READY queue and perform context switching.

If a new process arrives with CPU burst length less than remaining time of current executing process, pre-empt. This scheme is known as the Shortest-Remaining-Time-First (SRTF). If a new process arrives with a shorter burst time than remaining of current process then schedule new process. Further reduces average waiting time and average response time.

Shortest Remaining Job First

1- Traverse until all process gets completely

executed.

a) Search the process which is having minimum remaining time

b) Decrement its time by 1.

c) Check if its remaining time becomes 0

d) Increment the counter of process completion.

e) Current Process's Completion time = current time +1;

e) Calculate waiting time for each completed process.

wt[i]= Completion time - arrival_time-burst_time

f) Increment time by one.

2- Find turnaround time (waiting_time+burst_time).

3- Find Average Waiting time (Waiting_time/n)

4- Find Average Turn Around Time (Turn Around Time/n)

Complexity:

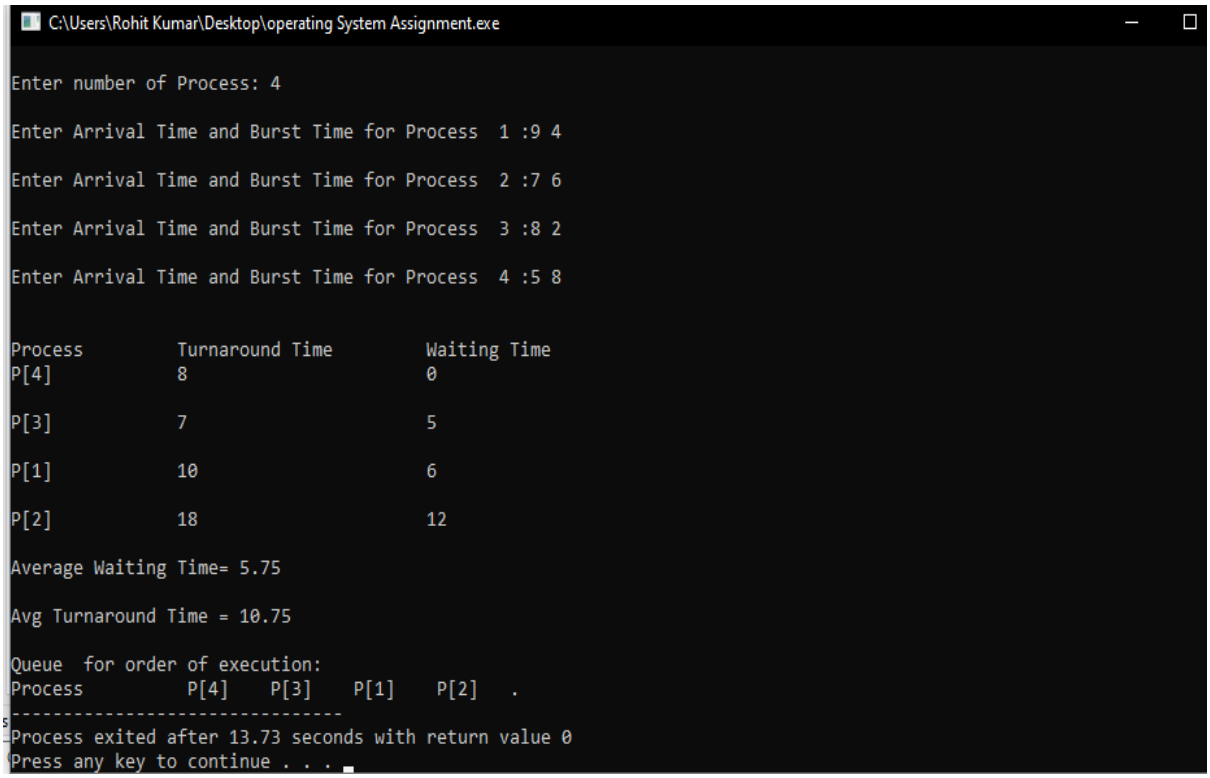
Select minimum burst time (use min-heap algorithm).

Adding an element takes log(n) time and deleting an element takes log(n) time.

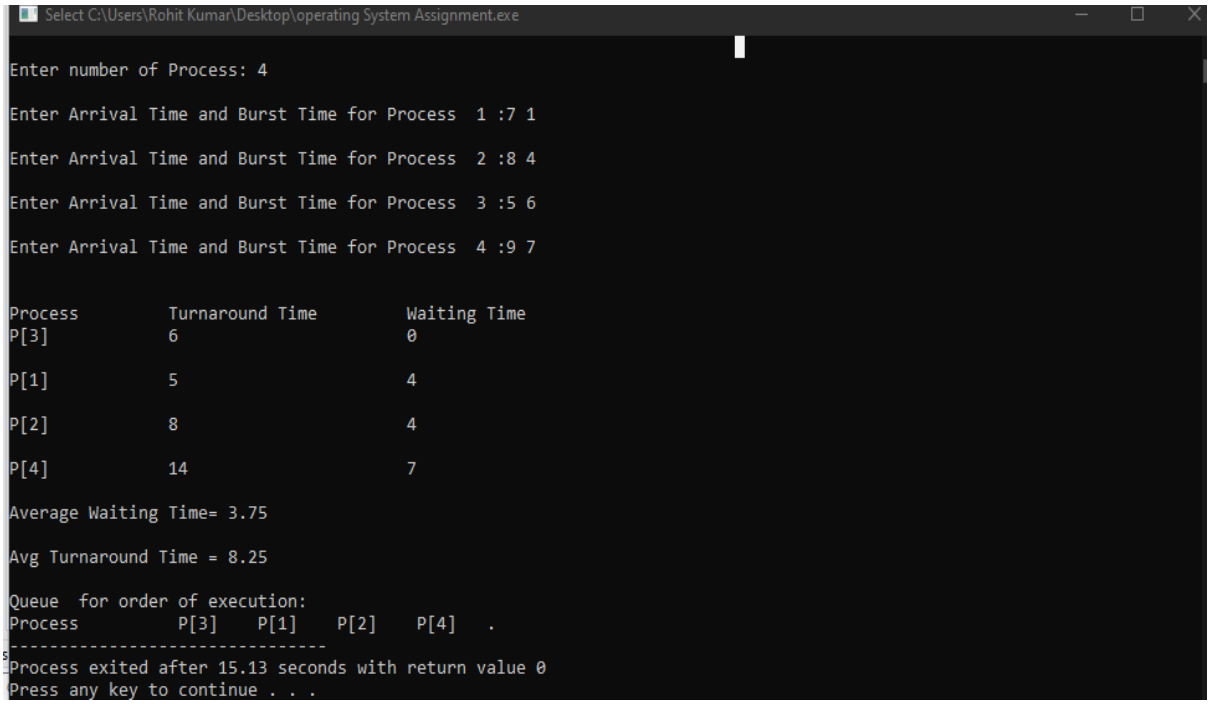
Thus, for n processes, **time complexity = n x 2log(n) = nlog(n)**

Test cases :

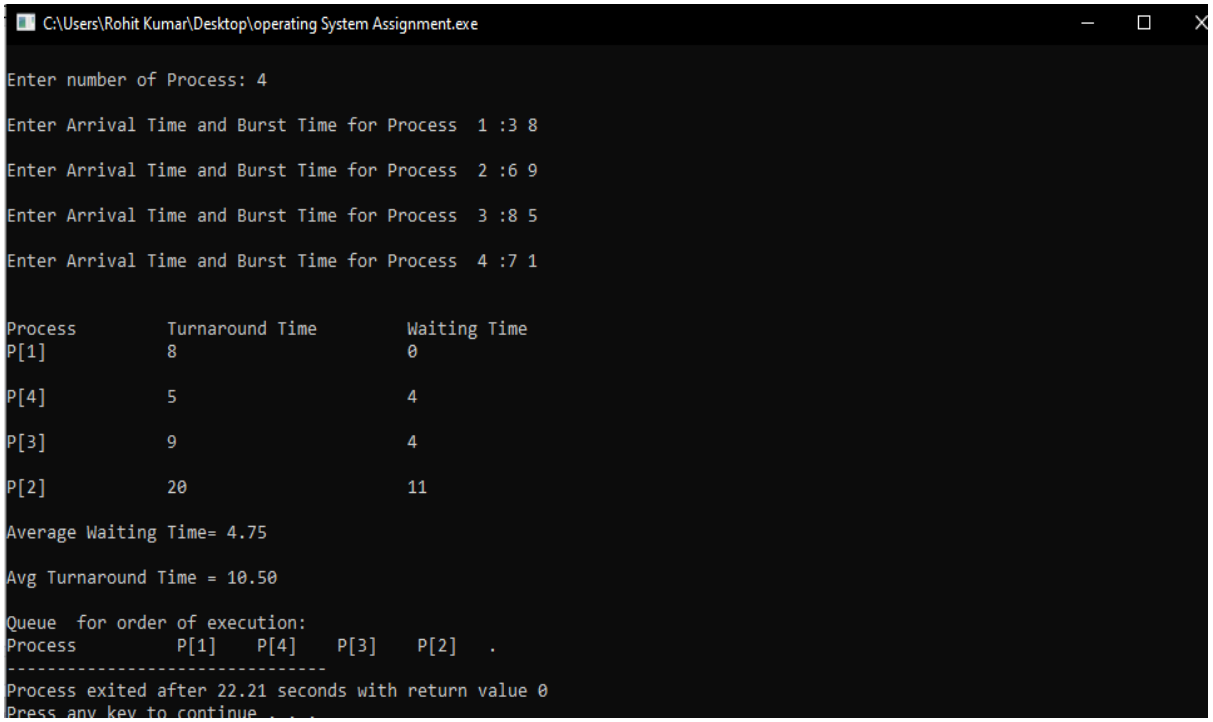
1.



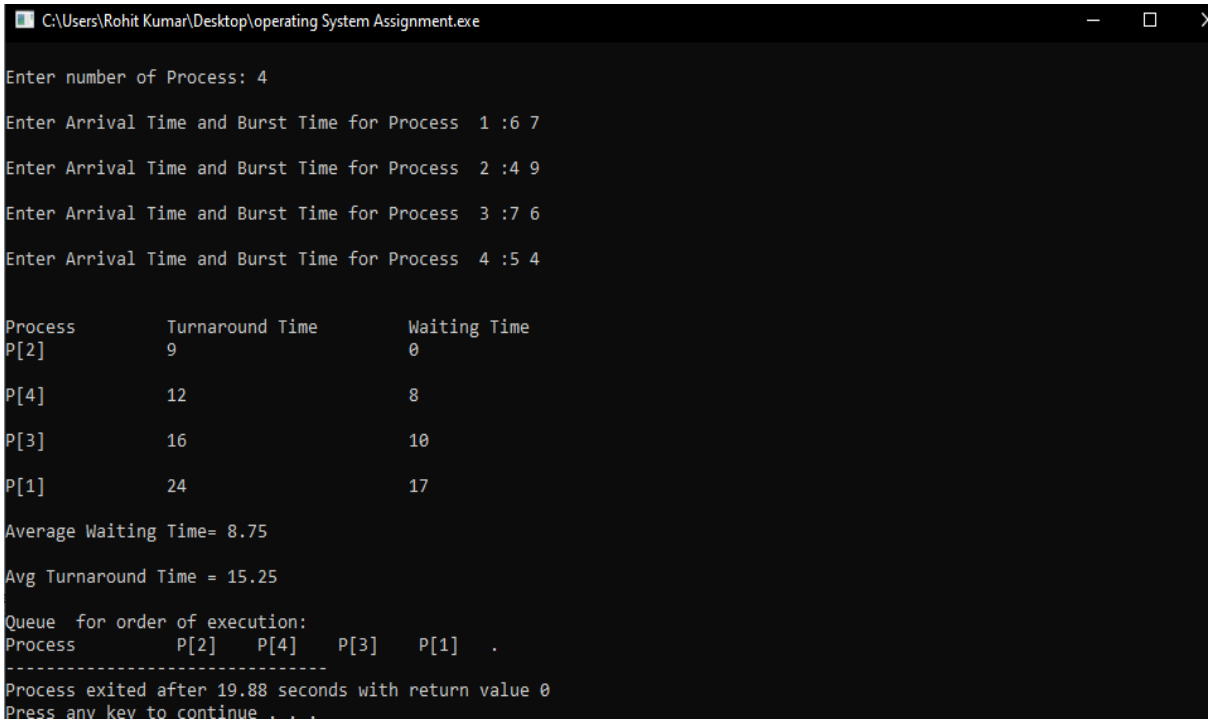
2.



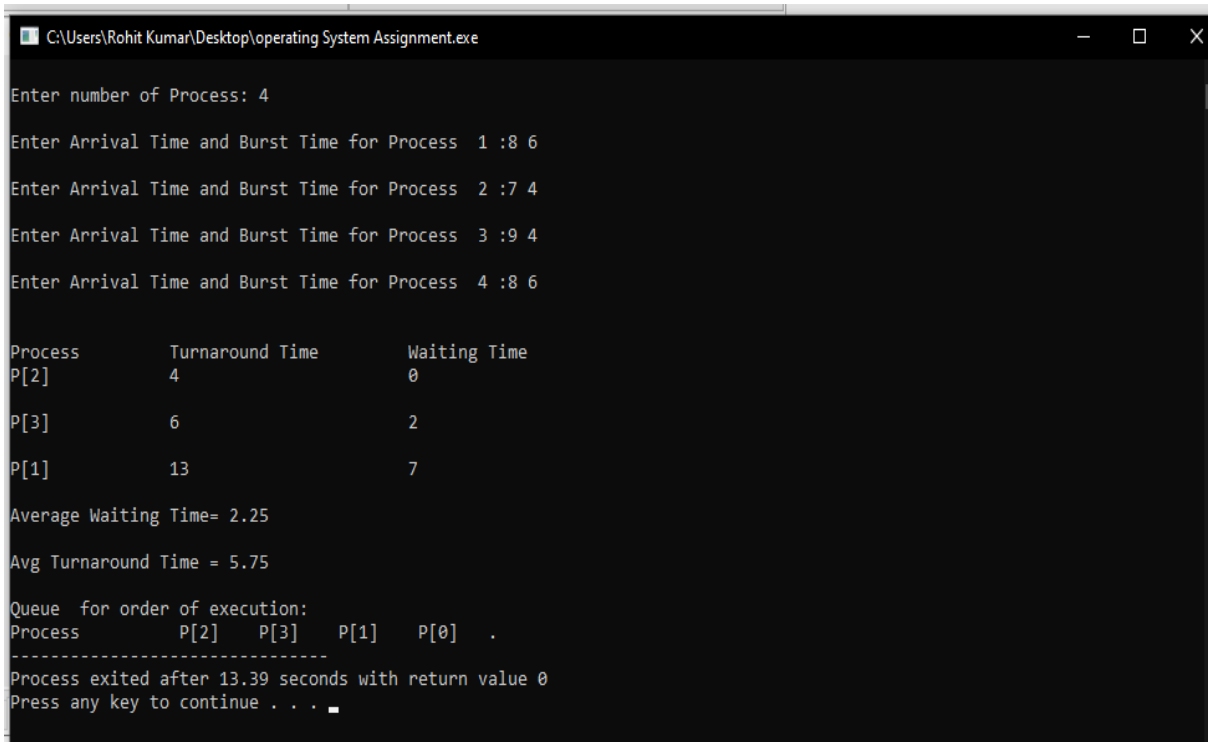
3.


C:\Users\Rohit Kumar\Desktop\operating System Assignment.exe
Enter number of Process: 4
Enter Arrival Time and Burst Time for Process 1 :3 8
Enter Arrival Time and Burst Time for Process 2 :6 9
Enter Arrival Time and Burst Time for Process 3 :8 5
Enter Arrival Time and Burst Time for Process 4 :7 1
Process Turnaround Time Waiting Time
P[1] 8 0
P[4] 5 4
P[3] 9 4
P[2] 20 11
Average Waiting Time= 4.75
Avg Turnaround Time = 10.50
Queue for order of execution:
Process P[1] P[4] P[3] P[2] .
Process exited after 22.21 seconds with return value 0
Press any key to continue . . .

4.


C:\Users\Rohit Kumar\Desktop\operating System Assignment.exe
Enter number of Process: 4
Enter Arrival Time and Burst Time for Process 1 :6 7
Enter Arrival Time and Burst Time for Process 2 :4 9
Enter Arrival Time and Burst Time for Process 3 :7 6
Enter Arrival Time and Burst Time for Process 4 :5 4
Process Turnaround Time Waiting Time
P[2] 9 0
P[4] 12 8
P[3] 16 10
P[1] 24 17
Average Waiting Time= 8.75
Avg Turnaround Time = 15.25
Queue for order of execution:
Process P[2] P[4] P[3] P[1] .
Process exited after 19.88 seconds with return value 0
Press any key to continue . . .

5.


C:\Users\Rohit Kumar\Desktop\operating System Assignment.exe
Enter number of Process: 4
Enter Arrival Time and Burst Time for Process 1 :8 6
Enter Arrival Time and Burst Time for Process 2 :7 4
Enter Arrival Time and Burst Time for Process 3 :9 4
Enter Arrival Time and Burst Time for Process 4 :8 6
Process Turnaround Time Waiting Time
P[2] 4 0
P[3] 6 2
P[1] 13 7
Average Waiting Time= 2.25
Avg Turnaround Time = 5.75
Queue for order of execution:
Process P[2] P[3] P[1] P[0] .
Process exited after 13.39 seconds with return value 0
Press any key to continue . . .