LOVELY PROFESSIONAL UNIVERSITY

Academic Task-3 (Simulation based assignment)

School of Computer Science and Engineering Faculty of Technology And Sciences

Name of the faculty member : Priya Virdi

Course Code: CSE316 Course Title: Operating System

Name -Lun singh Rajpurohit

Reg No 11802125 Roll No 25

Section :- K18EN email id 2000lunrajpurohit@gmail.com

Github id :-

Max. Marks:30 Date of Allotment: 01/02/2019 Date of Submission: 04/04/2019

1. Question Statement

Ques. 3. Consider a scheduler which schedules the job by considering the arrival time of the processes where arrival time if given as 0 is discarded or displayed as error. The scheduler implements the shortest job first scheduling policy, but checks the queue of the processes after the every process terminates and time taken for checking and arranging the process according to the shortest job is 2 time unit. Compute the waiting time, turnaround time and average waiting time and turnaround time of the processes. Also compute the total time taken by the processor to compute all the jobs.

The inputs for the number of requirements, arrival time and burst time should be provided by the user.

Consider the following units for reference.

Process	Arrival time	Burst Time
1	0	6
2	3	2
3	5	1
4	9	7
5	10	5
6	12	3

7	14	4
8	16	5
9	17	7
10	19	2

Develop a scheduler which submits the processes to the processor in the defined scenario, and compute the scheduler performance by providing the waiting time for process, turnaround time for process and average waiting time and turnaround time.

Description

Shortest job first (SJF) or shortest job next, is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN is a non-preemptive algorithm.

- Shortest Job first has the advantage of having a minimum average waiting time among all scheduling algorithms.
- It is a Greedy Algorithm.
- It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of ageing.
- It is practically infeasible as Operating System may not know burst time and therefore may not sort them. While it is not possible to predict execution time, several methods can be used to estimate the execution time for a job, such as a weighted average of previous execution times. SJF can be used in specialized environments where accurate estimates of running time are available.

CODE

```
#include<stdio.h>

#include<conio.h>

int main()

{

   int bt[10],p[10],n,temp,i,j,wt[10],sum=0;

   float avg;

   printf("Enter total no of proces:");

   scanf("%d",&n);
```

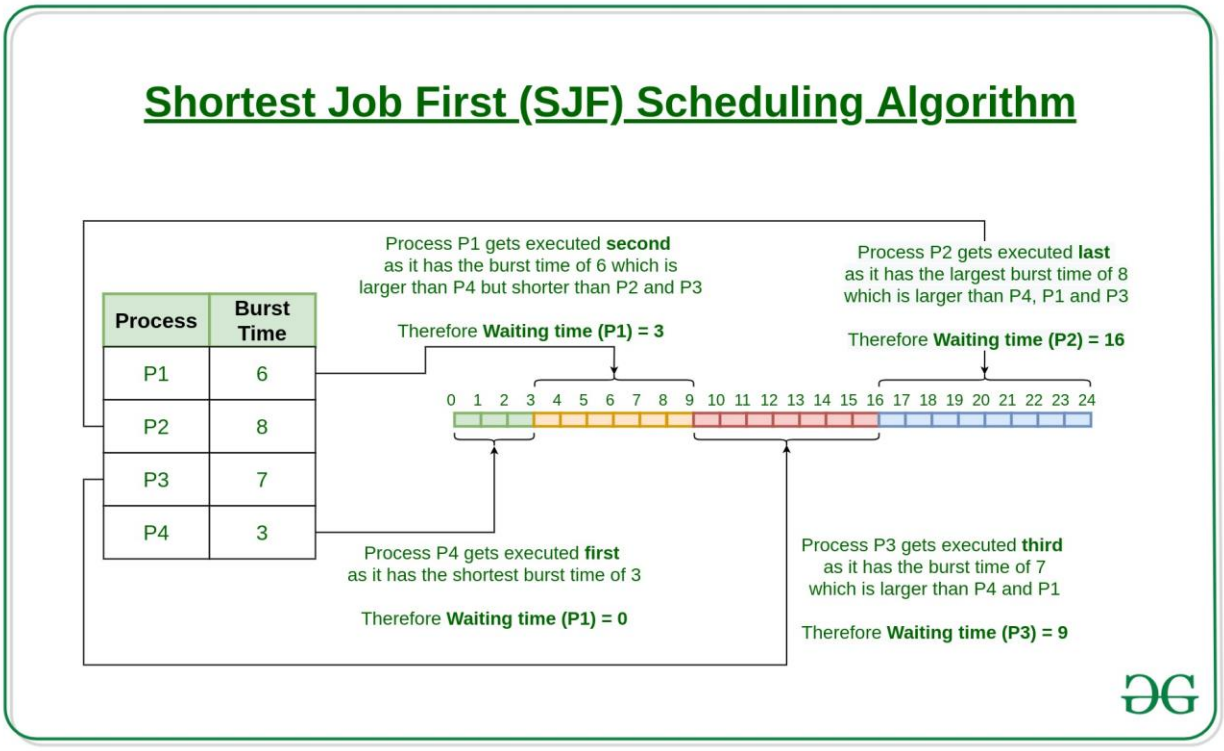
```
printf("\n Enter burst time for each process:-");
for(i=0;i<n;i++)
{
   printf("\nBurst time of process P%d:",i);
   scanf("%d",&bt[i]);
   p[i]=i;
}
for(i=0;i<n-1;i++)
{
   for(j=i+1;j<n;j++)
   {
      if(bt[i]>bt[j])
      {
         temp=bt[i];
         bt[i]=bt[j];
         bt[j]=temp;
         temp=p[i];
         p[i]=p[j];
         p[j]=temp;
      }
   }
}
wt[0]=0;
for(i=1;i<n;i++)
{
   wt[i]=wt[i-1]+bt[i-1];
```

```
    }
    for(i=0;i<n;i++)
    {
       sum+=wt[i];
    }
    avg=(float)sum/n;
    printf("\n Waiting time for each process:-");
    for(i=0;i<n;i++)
    {
       printf("\n Waiting time for process P%d is %d sec.",p[i],wt[i]);


    }
    printf("\n Average waiting time is %f sec.",avg);
    getch();
    return 0;
}
```

Algorithm:

1. Sort all the process according to the arrival time.
2. Then select that process which has minimum arrival time and minimum Burst time.
3. After completion of process make a pool of process which after till the completion of previous process and select that process among the pool which is having minimum Burst time.



COMPLEXITY

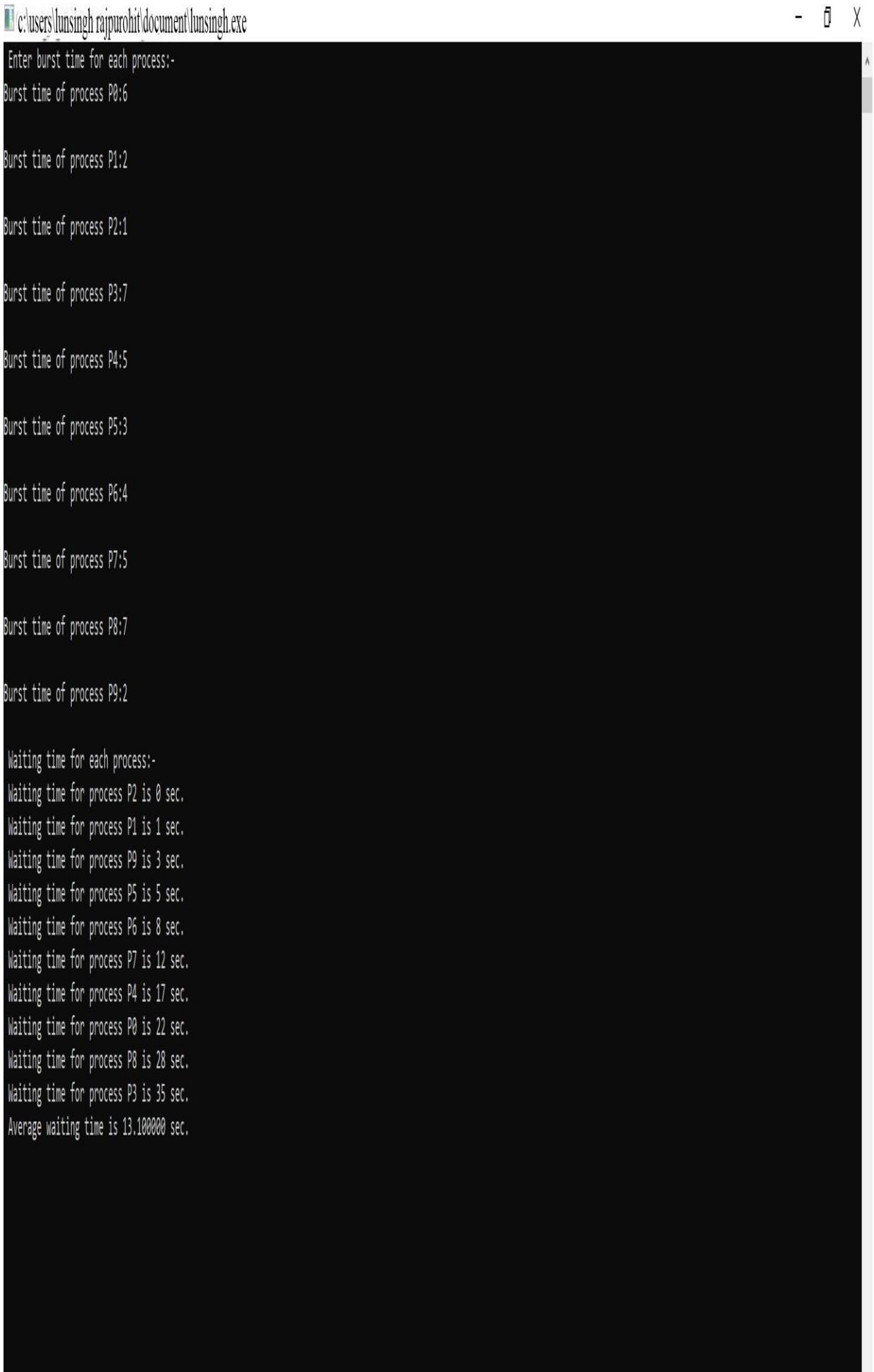
- The time **complexity** of **round-robin** scheduling algorithms is O(1). It is easy to realize and is suitable to use in high-speed.

Activity on github:

- I have uploaded the project on the GitHub. And I have also done the revisions of the project on the GitHub as my project is going on.

- I have made more than 5 revisions on that project and also uploaded the documented description file with it.

Screenshot:-


c:\users\lunsingh rajpurohit\document\lunsingh.exe
Enter burst time for each process:-
Burst time of process P0:6
Burst time of process P1:2
Burst time of process P2:1
Burst time of process P3:7
Burst time of process P4:5
Burst time of process P5:3
Burst time of process P6:4
Burst time of process P7:5
Burst time of process P8:7
Burst time of process P9:2
Waiting time for each process:-
Waiting time for process P2 is 0 sec.
Waiting time for process P1 is 1 sec.
Waiting time for process P9 is 3 sec.
Waiting time for process P5 is 5 sec.
Waiting time for process P6 is 8 sec.
Waiting time for process P7 is 12 sec.
Waiting time for process P4 is 17 sec.
Waiting time for process P0 is 22 sec.
Waiting time for process P8 is 28 sec.
Waiting time for process P3 is 35 sec.
Average waiting time is 13.100000 sec.