Mawlana Bhashani Science and Technology University

# Lab-Report

Report No: 09

Course code: ICT-3110

Course title: Operating Systems Lab

Date of Performance:

Date of Submission: 11/09/2020


**Submitted by**

Name: Golam Kibria Tuhin

ID:IT-18015

3th year 1ndsemester

Session: 2017-2018

Dept. of ICT

**Submitted To**

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

# Name of the lab report : Implementation of Priority Scheduling Algorithm.

**Objective**: Priority Scheduling algorithm Definition & executable code in c are followed.

**Q.1 What is priority Scheduling algorithm?**

**Ans:** Priorities can be either dynamic or static. Static priorities are allocated during creation, whereas dynamic priorities are assigned depending on the behavior of the processes while in the system. To illustrate, the scheduler could favor input/output (I/O) intensive tasks, which lets expensive requests to be issued as soon as possible.

**Q.2 How to implemented in C?**

**Ans:**

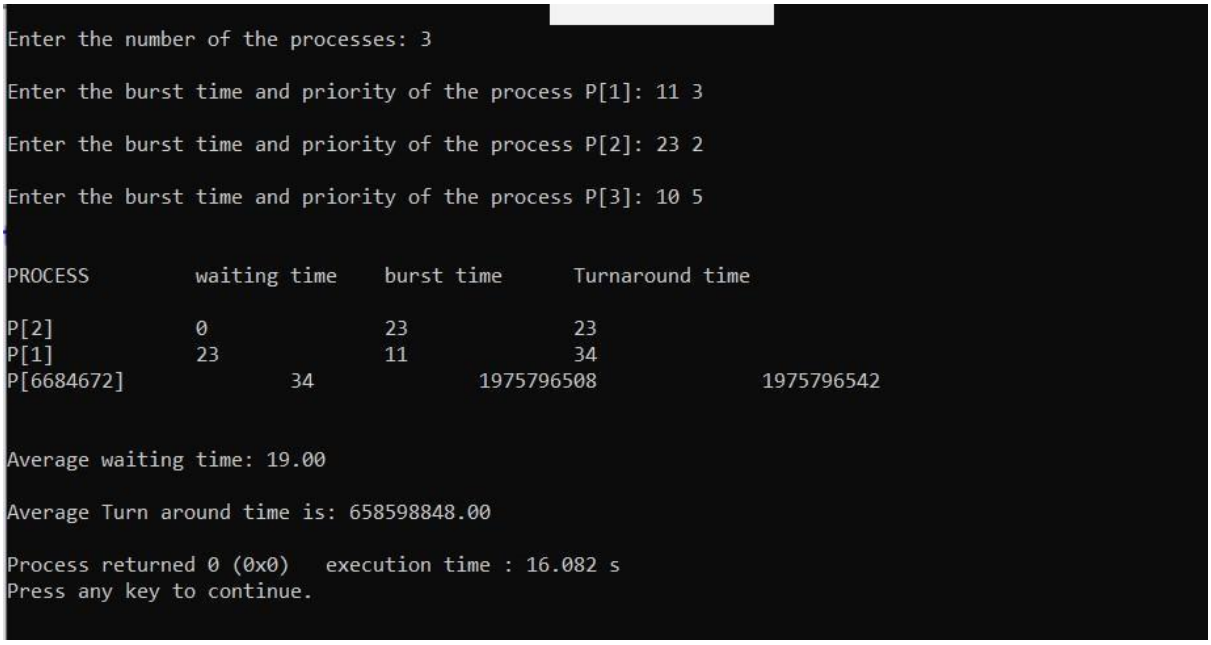
```
#include <stdio.h> int main()

{


   int b[30],w[30],p[30],ta[30],p[40];

float av=0,avt=0;


   int i,j,n,temp,key;


   printf("\nEnter the number of the processes: ");


   scanf("%d",&n);


   for(i=1; i<=n; i++)


   {


      printf("\nEnter the burst time and priority of the process P[%d]: ",i);


      scanf("%d",&b[i]);

  scanf("%d",&p[i]);

      p[i]=i;
```

```
    }

    for(i=0; i<n; i++)
{         key=i;
for(j=i+1; j<n; j++)
        {              if(p
[j]<p [key])
            {
key=j;
            }
}
temp=b[i];
b[i]=b[key];
b[key]=temp;

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

    w[0]=0;
ta[0]=b[0];
avt=ta[0];
```

```
    for(i=1; i<n; i++)
    {

        w[i]=w[i-1]+b[i-1];

        ta[i]=ta[i-1]+b[i];

        avt+=w[i];

        avt+=ta[i];

    }

    avwt=avwt/n;
avtat=avtat/n;

    printf("\n\nPROCESS\t\twaiting time\tburst time\tTurnaround time\n");
    printf("\n");

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

        printf("P[%d]\t\t%d\t\t%d\t\t%d\n",p[i],wt[i],bt[i],tat[i]);

    }

    printf("\n\nAverage waiting time: %.2f",avwt);
printf("\n\nAverage Turn around time is: %.2f",avtat);
    printf("\n");

    return 0;
```

}

**Output:**




## Conclusion:

It's similar to SRTF scheduling. In SRTF, burst time was the priority. Here, priority is explicitly provided. The process that has highest priority gets the CPU first. The processes gets serviced by the CPU in order of their priority in descending order. Higher number always doesn't represents higher priority. In some cases, 0 may be the highest priority and 100 the lowest. It depends on systems.