Lab Report No:	09
Lab Report Name:	Implementation of Priority Scheduling Algorithm
ID:	IT-17037

Objective:- To understand the Priority scheduling algorithm technique in detail.

To simulate Priority Scheduling Algorithm using C/C++.

To understand the advantage and disadvantage of priority scheduling algorithm technique.

Priority Scheduling:- Priority scheduling is a method of scheduling processes based on priority. In this method, the scheduler chooses the tasks to work as per the Priority, which is different from other types of scheduling, for example, a simple round robin.

Priority scheduling involves priority assignment to every process, and processes with higher priorities are carried out first, whereas tasks with equal priorities are carried out on a first-come-first-served (FCFS) or round robin basis. An example of a general-priority-scheduling algorithm is the shortest-job-first (SJF) algorithm.

process	Burst time	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

1. Completion Time: Time at which process completes its execution.

2. Turn Around Time: Time Difference between completion time and arrival time. Turn Around Time = Completion Time – Arrival Time
3. Waiting Time (W.T): Time Difference between turn-around time and burst time.
   Waiting Time = Turn Around Time – Burst Time

In this post, we have assumed arrival times as 0, so turn around and completion times are same.

Gantt chart:-

P2	P5	P1	P3	P4

0 1 6 16 18 19

process	Burst time	Priority	Completion time	Turn-around time	Waiting time
P1	10	3	16	16	6
P2	1	1	1	1	0
P3	2	4	18	18	16
P4	1	5	19	19	18
P5	5	2	6	6	1

Average waiting time = (6+0+16+18+1)/5

=8.2ms

Average turn-around time = (16+1+18+19+6)/5

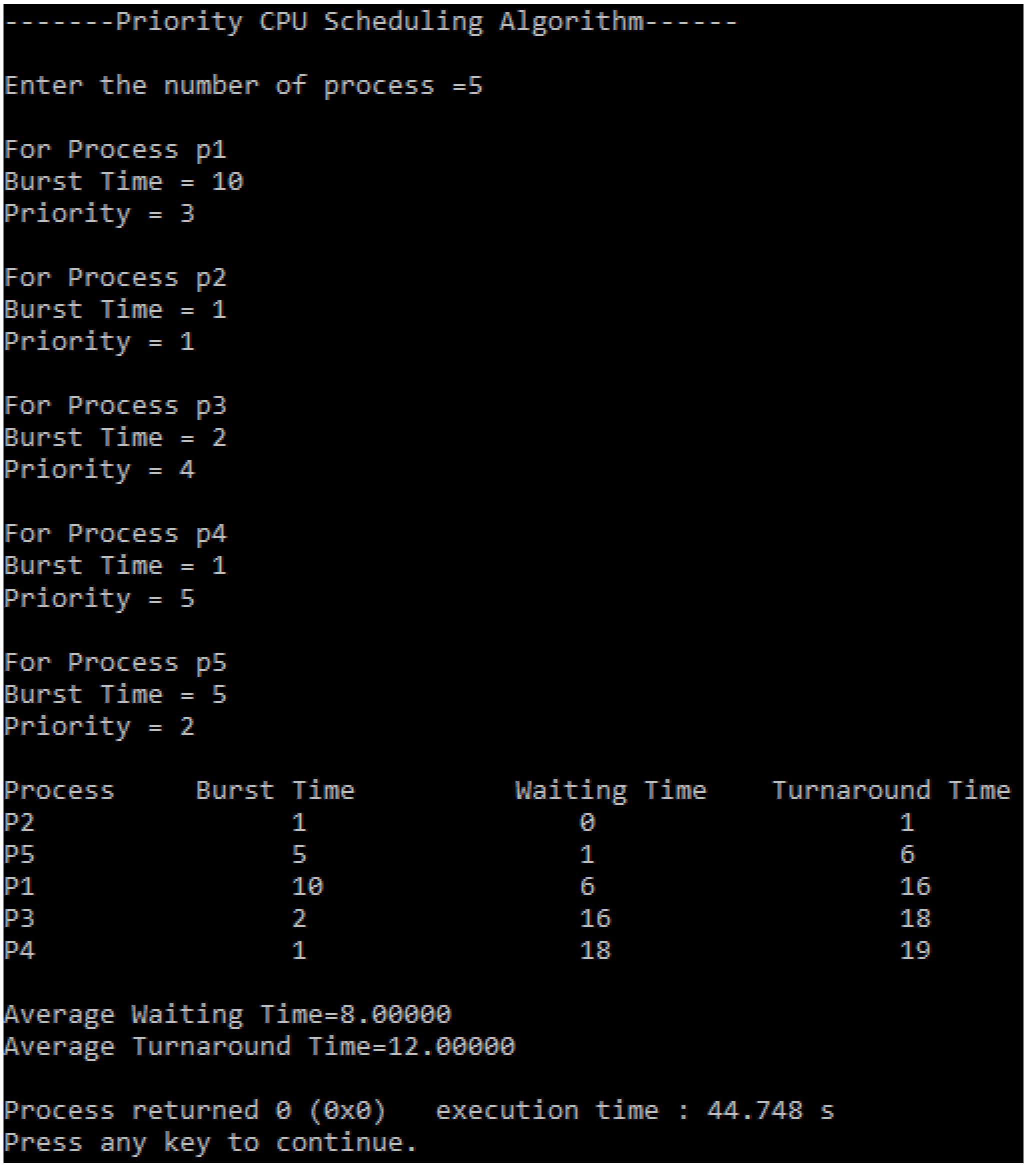
=12ms

Code:-

```
#include<stdio.h>
int main()
{
    int n;
    printf("-------Priority CPU Scheduling Algorithm------\n");
    printf("\nEnter the number of process =");
    scanf("%d",&n);
    int bt[n+2],p[n+2],pr[n+2],wt[n+2],tat[n+2],i,j,total=0,t,temp;
    float avg_wt,avg_tat;
    for(i=1;i<=n;i++)
    {
        printf("\nFor Process p%d\n",i);
        printf("Burst Time = ");
        scanf("%d",&bt[i]);
        printf("Priority = ");
        scanf("%d",&pr[i]);
        p[i]=i;

    }
    for(i=1;i<=n;i++)
    {    t=i;
        for(j=i+1;j<=n;j++)
        {
            if(pr[j]<pr[t])
                t=j;
        }
        temp=pr[i];
        pr[i]=pr[t];
        pr[t]=temp;

        temp=bt[i];
        bt[i]=bt[t];
        bt[t]=temp;

        temp=p[i];
        p[i]=p[t];
        p[t]=temp;
    }
    wt[0]=0;
    bt[0]=0;
    for(i=1;i<=n;i++)
    {   wt[i]=0;
        for(j=1;j<=i;j++)
            wt[i]+=bt[j-1];
        total=total+wt[i];
    }
    avg_wt =float(total/n);
    total=0;
```

```
    printf("\nProcess\t    Burst Time    \tWaiting Time\tTurnaround Time");

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

    {

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

        total= total+tat[i];

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

    }

     avg_tat=float(total/n);

    printf("\n\nAverage Waiting Time=%.5f",avg_wt);

    printf("\nAverage Turnaround Time=%.5f\n",avg_tat);

           return 0;

}
```

Output:-



```
-------Priority CPU Scheduling Algorithm------

Enter the number of process =5

For Process p1
Burst Time = 10
Priority = 3

For Process p2
Burst Time = 1
Priority = 1

For Process p3
Burst Time = 2
Priority = 4

For Process p4
Burst Time = 1
Priority = 5

For Process p5
Burst Time = 5
Priority = 2

Process     Burst Time          Waiting Time    Turnaround Time
P2                1                 0                   1
P5                5                 1                   6
P1                10                6                   16
P3                2                 16                  18
P4                1                 18                  19

Average Waiting Time=8.00000
Average Turnaround Time=12.00000

Process returned 0 (0x0)   execution time : 44.748 s
Press any key to continue.
```

Conclusion:- There are many scheduling algorithms having their own benefits and drawbacks. Scheduling can also be done on the basis of priority. Each process assigned a priority and the process which has highest priority will execute first. In case of similar priority generally FCFS is used to select the next process. If SJF based priority scheduling algorithm is used when two or more process having similar priority, instead of FCFS, then the average waiting time and average turnaround time is reduced. We proposed SJF based priority scheduling algorithm in which, the process that having lowest burst time will execute first. The existing FCFS based priority scheduling algorithm and proposed SJF based priority scheduling algorithm is analyzed and the result shows that the average waiting time and average turnaround time is reduced.