



Mawlawi Bhashani Science and Technology University

Lab-Report

Lab Report No: 09

Lab Report Name: Implementation of Priority Scheduling Algorithm.

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance: 7/9/2020

Date of Submission:

Submitted by

Name: Afra Ibnat Tethye
ID: IT-18055
3rd year 1st semester
Session: 2017-18
Dept. of ICT

Submitted To

Nazrul Islam
Assistant Professor
Dept. of ICT,
MBSTU.

Experiment Name: Implementation of Priority Scheduling Algorithm.

Question-01:

What is priority Scheduling algorithm?

Answer:

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.

Question-02:

How to implemented in C?

Answer:

The implemented code in C is given bellow:

```
#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%d\t%d\t%d\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:

```
Enter the number of the processes: 3

Enter the burst time and priority of the process P[1]: 11 3

Enter the burst time and priority of the process P[2]: 23 2

Enter the burst time and priority of the process P[3]: 10 5

      PROCESS      waiting time      burst time      Turnaround time
P[2]          0              23              23
P[1]          23             11              34
P[6684672]    34            1975796508        1975796542

Average waiting time: 19.00
Average Turn around time is: 658598848.00
Process returned 0 (0x0)  execution time : 16.082 s
Press any key to continue.
```

Conclusion:

Based on this Priority Number, the processes are executed. This scheduling algorithm is normally very useful in real-time systems. The process having the highest priority (1) is executed first and then priority 2, 3 and so on.