Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 08

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance:

Date of Submission:

Submitted by

Name: Ashikur Rahman Miran

ID:IT-18014

3rd year 1st semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Experiment No: 08

Experiment Name: Implementation of SJF scheduling algorithm.

Objectives:

    i)   What is SJF scheduling algorithm?
    ii)   How to implement SJF scheduling algorithm.

Theory:

SJF algorithm is also known as Shortest-Job-First algorithm. It is a non-preemptive discipline in which waiting job with small burst time is run next. In other words, when CPU is available, it is assigned to the process that has smallest next CPU burst. The SJF scheduling is especially appropriate for batch jobs for which the run times are known in advance.

Implementation:

1. Take input number of process, burst time and processes.
2. Sort the process according to burst time.
3. Calculate waiting time = starting time – arrival time.
4. Calculate turnaround time = burst time + waiting time.

Process	Arrival time	Burst time
P1	0	4
P2	0	3
P3	0	7
P4	0	1
P5	0	2

Grant chart:

P4	P5	P2	P1	P3
0	1	3	6	10	17

Process	Arrival time(At)	Burst time(Bt)	Waiting time Wt=st-at	Total turnaround time Tat=wt+bt
P4	0	1	0	1
P5	0	2	1	3
P2	0	3	3	6
P1	0	4	6	10
P3	0	7	10	17

Source code:

```c
#include<stdio.h>
int main()
{
    int bt[100],p[100],wt[100],tat[100],i,j,n,total=0,pos,temp;
    double avg_wt,avg_tat;
    printf("Enter number of process:");
    scanf("%d",&n);
    printf("Enter Burst Time:\n");
    for(i=0; i<n; i++)
    {
        printf("P%d: ",i+1);
        scanf("%d",&bt[i]);
        p[i]=i+1;
    }
    for(i=0; i<n; i++)
    {
        pos=i;
        for(j=i+1; j<n; j++)
        {
          if(bt[j]<bt[pos])
               pos=j;
        }

        temp=bt[i];
        bt[i]=bt[pos];
        bt[pos]=temp;
```

```c
            temp=p[i];
            p[i]=p[pos];
            p[pos]=temp;
        }
        wt[0]=0;
        for(i=1; i<n; i++)
        {
            wt[i]=0;
            for(j=0; j<i; j++)
                    wt[i]+=bt[j];
            total+=wt[i];
        }
        avg_wt=(double)total/n;
        total=0;
        printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time");
        for(i=0; i<n; i++)
        {
            tat[i]=bt[i]+wt[i];
            total+=tat[i];
            printf("\np%d\t%d\t\t%d\t\t%d",p[i],bt[i],wt[i],tat[i]);
        }

        avg_tat=(double)total/n;
        printf("\n\nAverage Waiting Time=%lf",avg_wt);
        printf("\nAverage Turnaround Time=%lf\n",avg_tat);
}
```

Output:

```
 "D:\programming\c _ c++ programming\algorithm\SJF scheduling algorithm in c.exe"
Enter number of process:5
Enter Burst Time:
P1: 4
P2: 3
P3: 7
P4: 1
P5: 2

Process Burst Time       Waiting Time      Turnaround Time
p4       1               0                 1
p5       2               1                 3
p2       3               3                 6
p1       4               6                 10
p3       7               10                17

Average Waiting Time=4.000000
Average Turnaround Time=7.400000

Process returned 0 (0x0)    execution time : 12.474 s
Press any key to continue.
```

Conclusion:

In this lab I learn how to implement SJF scheduling algorithm and also run the code and shows the output and output is expected.