# Mawlana Bhashani Science and Technology University

# Lab-Report

Report No:08

Course code:ICT-3110

Course title:Operating Systems Lab

Date of Performance:08-09-2020

Date of Submission:16-09-2020


**Submitted by**

Name:Meraz Ahmed

ID:IT-18005

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.

Objective:

Here we have learn about Shortest job First (Non preemptive) Algorithm

We also learn how to implement this algorithm in c programming language

Algorithm:.

- This is also known as **shortest job first**, or SJF
- This is a non-preemptive, pre-emptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where required CPU time is not known.
- The processer should know in advance how much time process will take.

**Example:**

| Process | P1 | P2 | P3 | P4 | P5 |
|---|---|---|---|---|---|
| Burst time | 5 | 13 | 8 | 4 | 10 |
| Arrival time | 2 | 3 | 0 | 5 | 1 |

**Code:**

```
#include<stdio.h>

void main()
{
   int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;
   float avg_wt,avg_tat;
   printf("Enter number of process:");
   scanf("%d",&n);

   printf("\nEnter Burst Time:\n");
```

```
for(i=0;i<n;i++)
{
   printf("p%d:",i+1);
   scanf("%d",&bt[i]);
   p[i]=i+1;          //contains process number
}

//sorting burst time in ascending order using selection sort
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;

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

wt[0]=0;           //waiting time for first process will be zero

//calculate waiting time
for(i=1;i<n;i++)
{
   wt[i]=0;
   for(j=0;j<i;j++)
      wt[i]+=bt[j];

   total+=wt[i];
}

avg_wt=(float)total/n;      //average waiting time
total=0;

printf("\nProcess\t    Burst Time    \tWaiting Time\tTurnaround Time");
for(i=0;i<n;i++)
{
   tat[i]=bt[i]+wt[i];     //calculate turnaround time
   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;     //average turnaround time
  printf("\n\nAverage Waiting Time=%f",avg_wt);
  printf("\nAverage Turnaround Time=%f\n",avg_tat);
}
```

## Output:

```
Enter number of process:4

Enter Burst Time:
p1:5 6 7 8
p2:p3:p4:
Process     Burst Time          Waiting Time    Turnaround Time
p1                5                 0                  5
p2                6                 5                  11
p3                7                 11                 18
p4                8                 18                 26

Average Waiting Time=8.500000
Average Turnaround Time=15.000000

Process returned 35 (0x23)   execution time : 28.043 s
Press any key to continue.
```