

Department of Computer Science and Engineering

**Course Code :** CSE -334

**Course Title :** Operating System Lab.

**Report :** 02.

**Report Name :** Implementation of SJF Algorithm.

**Performance Date :** 08-08-2015.

**Submission Date :** 17-08-2015.

**REMARKS**

**NAME : Afia Washima**

**ID :** **12-023-1-02-00396**

**DEPT :** **CSE**

**YEAR :** **2015**

**SEM. :** **6th**

**Introduction:** In this scheduling CPU is always assigned to the process with least CPU burst requirement. If the two processes having the same length, next CPU burst, FCFS is used i.e. One which arrives first, will be taken up first by the CPU. This algorithm can be preemptive or non-preemptive.

**Objective:** Implementation of SJF algorithm.

**Source Code:**

#include<stdio.h>

main()

{

int temp,temp1,i,j,x,bt[20],pt[20];

printf(“No of process:”);

scanf("%d",&x);

for(i=0;i<x;i++)

{

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

}

for(i=0;i<x;i++)

for(j=0;j<x;j++)

{

if(bt[i]>bt[i+1])

{

temp=bt[i];

bt[i]=bt[i+1];

bt[i+1]=temp;

temp1=pt[i];

pt[i]=pt[i+1];

pt[i+1]=temp1;

}

}

for(i=0;i<x;i++)

{

for(j=0;j<bt[i];j++)

{

printf("P%d",pt[i]);

}

}

}

**Input:**

No of process: 3

1 5

2 3

3 2

**Output:**

P2P2P2P3P3P1P1P1P1P1

**Discussion:**

1. It is easy to implement and
2. Its average wait time is high.