Contents

**USCSP301:USCS303-OperatingSystem(OS) Practical-02**

**Practial-02:Shortest Job First Scheduling Algorithm**

**Practical Date: 24th July,2021..................................................................................................1**

**Practical Aim:Implement SJF (with no preemption) scheduling algorithm in Java....................2**

**Algorithm 3**

**ExampleofSJF 4**

**GnattChart 5**

**Implementation 6**

**Input 7**

**Output 8**

**SampleOutput 9**

**Algorithm:**

CPU scheduling algorithm are used for scheduling different process present in the ready queue withavailableresourceinanoptimalwaysothateach andevery process getexecutebyCPU

Schedulingalgorithmarebroadlyclassifiedintotwomaintypenamelypreemptiveandnon-preemptive.

FIRSTCOMEFIRSTOUT(FCFS)isalsoknow asFIRST INFIRSTOUT(FIFO)SCHEDUALalgorithmistheandsimplest

CPU .

AprocessschedulingdifferentprocesstobeassignedtotheCPUbasedonparticularschedulingalgorithm.There

are six popular process scheduling algorithm which we are going to discuss in this chapter FIRST COME FIRST OUT(FCFS)scheduling.

**Example1:**Considerthefollowingexamplecontainfiveprocesses.

|  |  |
| --- | --- |
| **ProcessId** | **BurstTime** |
| P0 | 6 |
| P1 | 3 |
| P2 | 8 |
| P3 | 3 |
| P4 | 4 |

**Step1:**Processesgetexecuteaccordingtotheirlowestbursttimefirst.

|  |  |
| --- | --- |
| **ProcessId** | **BurstTime** |
| P0 | 6 |
| P1 | 3 |
| P2 | 8 |
| P3 | 3 |
| P4 | 4 |

**Step 2: Following shows the scheduling and execution of processesStep2.1:**AtstartP1shortestexecutiontimewhichis0-3second.

|  |  |
| --- | --- |
| Systemtime | 0 |
| Processesschedulingfinishtime | P1 |
| Finishtime | 0+3=3 |
| Watingtime | 3-3=0 |
| TurnAroundtime | 3-0=3 |

**Step2.2:**nextshortestexecutiontimeisforprocessP3forduration3-6second.

|  |  |
| --- | --- |
| Systemtime | 6 |
| Processesschedulingfinishtime | P1,p3 |
| Finishtime | 3+3=6 |
| Watingtime | 6-3=3 |
| TurnAroundtime | 6-0=6 |

**Step2.3:**NextjobwithshortestexecutiontimeisP4foraduration6-10second.

|  |  |
| --- | --- |
| Systemtime | 10 |
| Processesschedulingfinishtime | P1,p3’p4 |
| Finishtime | 6+4=10 |
| Watingtime | 10-4=6 |
| TurnAroundtime | 10-0=10 |

**Step2.4:**Nextjobwith shortestexecutiontimeisp0fordurationof10-16second.

|  |  |
| --- | --- |
| Systemtime | 10 |
| Processesschedulingfinishtime | P1,p3,p3,p4,p0 |
| Finishtime | 10+6=16 |
| Watingtime | 16-6=10 |
| TurnAroundtime | 16-0=16 |

**Step2.5:**SimilarlynextjobwithshortestexecutiontimeisP2fordurationof16-24second.

|  |  |
| --- | --- |
| Systemtime | 16 |
| Processesschedulingfinishtime | P1,p3,p3,p4,p0,p2 |

Finishtime 16+8=24

Watingtime 24-8=16

TurnAroundtime 24-0=24

**Step 3:** Calculate average wating time and average turn around time.Averagewatingtime=(0+3+6+10+16)/5

=35/5

=7

Averageturnaroundtime=(3+6+10+16+24)/5

= 59/5

=11.8

**GnattChart**

**Step4:**Afterschedulingof allprovidedprocesses.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Processid** | **Bursttime** | **Arrivaltime** | **Finishtime** | **TurnAroundtime** | **Watingtime** |
| P1 | 3 | 0 | 0+3=3 | 3-0=3 | 3-3=0 |
| P3 | 3 | 0 | 3+3=6 | 6-0=6 | 6-3=3 |
| P4 | 4 | 0 | 6+4=10 | 10-0=10 | 10-4=6 |
| P0 | 6 | 0 | 10+6=16 | 16-0=16 | 16-6=10 |
| P2 | 8 | 0 | 16+8=24 | 24-0=24 | 24-8=16 |
| Average |  |  |  | 11.8000000 | 7.000000 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P3 | P4 | P0 | P2 |

0 3 6 10 16 24

**Example2:**Considerthefollowingexamplecontainingfiveprocessesarrive atsametime.

|  |  |
| --- | --- |
| **ProcessesID** | **BurstTime** |
| P0 | 2 |
| P1 | 1 |
| P2 | 6 |

**GnattChart**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Processid** | **Bursttime** | **Arrivaltime** | **Finishtime** | **TurnAroundtime** | **Watingtime** |
| P1 | 1 | 0 | 1 | 1 | 1 |
| P0 | 2 | 0 | 3 | 3 | 3 |
| P2 | 6 | 0 | 9 | 9 | 9 |
| Average |  |  |  | 4.33333 | 1.33333 |
|  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| P1 | P0 | P2 |

0 1 3 9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Example3:**Considerthefollowingexamplecontainfiveprocesses arriveatsametime.  **Process ID Bursttime**  P0 25  P1 15  P2 10  P3 25  P4 10  P5 25 | | | | | |
| **Processid** | **Bursttime** | **Arrivaltime** | **Finishtime** | **Turn Aroundtime** | **Watingtime** |
| P2 | 10 | 0 | 10 | 10 | 0 |
| P4 | 10 | 0 | 20 | 20 | 10 |
| P1 | 15 | 0 | 35 | 35 | 20 |
| P0 | 25 | 0 | 60 | 60 | 35 |
| P3 | 25 | 0 | 85 | 85 | 60 |

**Gnattchart:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Processid** | **Bursttime** | **Arrivaltime** | **Finishtime** | **TurnAroundtime** | **Watingtime** |
| P2 | 10 | 0 | 10 | 10 | 0 |
| P4 | 10 | 0 | 20 | 20 | 10 |
| P1 | 15 | 0 | 35 | 35 | 20 |
| P0 | 25 | 0 | 60 | 60 | 35 |
| P3 | 25 | 0 | 85 | 85 | 60 |
| P5 | 25 | 0 | 110 | 110 | 85 |
| Average |  |  |  | 53.3333 | 35.000000 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P2 | P4 | P1 | P0 | P3 | P5 |

**0 10 20 35 60 85 110**

**Example4:**Considerthefollowingexamplecontainfiveprocesses arriveatsametime.

|  |  |
| --- | --- |
| **ProcessId** | **BurstTime** |
| P0 | 7 |
| P1 | 3 |
| P2 | 2 |
| P3 | 10 |
| P4 | 8 |

**Step4:**Afterschedulingof allprovidedprocesses.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Processid** | **Bursttime** | **Arrivaltime** | **Finishtime** | **TurnAroundtime** | **Watingtime** |
| P2 | 2 | 0 | 2 | 2 | 0 |
| P1 | 3 | 0 | 5 | 5 | 2 |
| P0 | 7 | 0 | 12 | 12 | 5 |
| P4 | 8 | 0 | 20 | 20 | 12 |
| P3 | 10 | 0 | 30 | 30 | 20 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Average |  |  |  | 13.80000 | 7.800000 |

//Name:Shraddha Sawant

//Batch:B1

//PRN:2020016400773862

//Date:24th July,2021

//Prac-02:SJF (with no preemption) Algorithm

import java.util.Scanner;

public class P2\_SJF\_SS {

int burstTime[];

int arrivalTime[] = {0};

String[] processId;

int numberOfProcess;

void getProcessData(Scanner input) {

System.out.print("Enter the number of Process for Scheduling: ");

int inputNumberOfProcess = input.nextInt();

numberOfProcess = inputNumberOfProcess;

burstTime = new int[numberOfProcess];

arrivalTime = new int[numberOfProcess];

processId = new String[numberOfProcess];

String st = "P";

for (int i = 0; i < numberOfProcess; i++) {

processId[i] = st.concat(Integer.toString(i));

System.out.print("Enter the burst time for Process - " +(i)+" : ");

burstTime[i] = input.nextInt(); }

}

void sortAccordingBurstTime(int[] at,int[] bt,String[] pid){

boolean swapped;

int temp;

String stemp;

for (int i = 0; i < numberOfProcess; i++) {

swapped = false;

for (int j = 0; j < numberOfProcess - i - 1; j++){

if (bt[j] > bt[j + 1]) {

temp = bt[j];

bt[j] = bt[j + 1];

bt[j + 1] = temp;

temp = at[j];

at[j] = at[j + 1];

at[j + 1] = temp;

stemp = pid[j];

pid[j] = pid[j + 1];

pid[j +1] = stemp;

swapped = true;

}

}

if(swapped == false){

break;

}

}

}

void shortestJobFirstNPAlgorithm() {

int finishTime[] = new int [numberOfProcess];

int bt[] = burstTime.clone();

int at[] = arrivalTime.clone();

String pid[] = processId.clone();

int waitingTime[] = new int[numberOfProcess];

int turnAroundTime[] = new int[numberOfProcess];

sortAccordingBurstTime(at,bt,pid);

finishTime[0] = at[0]+ bt[0];

turnAroundTime[0] = finishTime[0] - at[0];

waitingTime[0] = turnAroundTime[0] - bt[0];

for (int i = 1; i < numberOfProcess; i++) {

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

turnAroundTime[i] = finishTime[i] - at[i];

waitingTime[i] = turnAroundTime[i] - bt[i];

}

float sum = 0;

for (int n: waitingTime){

sum += n;

}

float averageWaitingTime = sum/ numberOfProcess;

sum = 0;

for (int n : turnAroundTime) {

sum += n;

}

float averageTurnAroundTime = sum/ numberOfProcess;

System.out.println("SJF (with no preemption) Scheduling Algorithm: ");

System.out.format("%20s%20s%20s%20s%20s%20s\n", "ProcessId", "BurstTime", "ArrivalTime", "FinishTime", "TurnAroundTime", "WaitingTime");

for (int i = 0; i < numberOfProcess; i++) {

System.out.format("%20s%20d%20d%20d%20d%20d\n", pid[i], bt[i], at[i], finishTime[i],turnAroundTime[i],waitingTime[i]);

}

System.out.format("%80s%20f%20f\n", "Average", averageTurnAroundTime,averageWaitingTime);

}

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

P2\_SJF\_SS obj = new P2\_SJF\_SS();

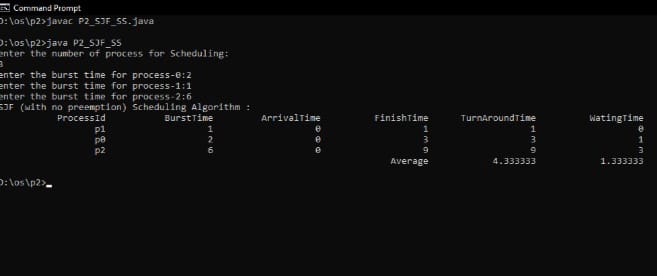
obj.getProcessData(input);

obj.shortestJobFirstNPAlgorithm();

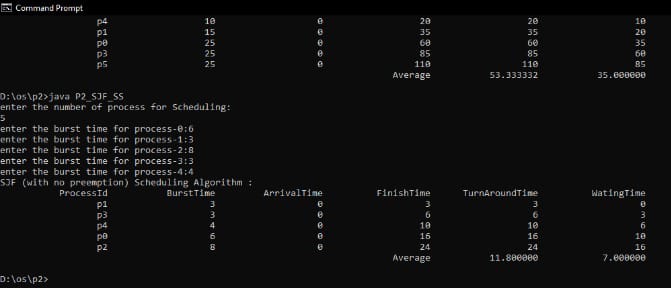
}

}

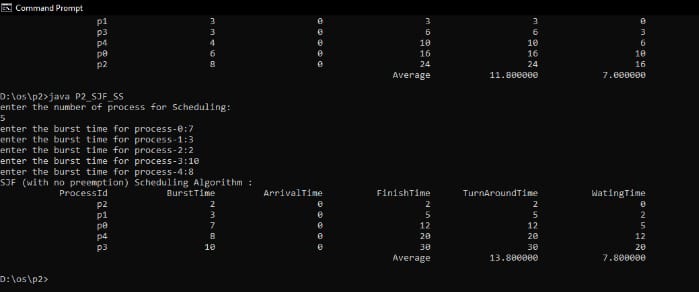
Sample Output1



Sample Output 2



Sample Output 3



Sample Output4

