Contents

[**USCSP301 – USCS303 : Operating system (OS) Practical – 01 2**](#_Toc81559352)

[**Practical – 01 First Come First Serve (FCFS) Algorithm 2**](#_Toc81559353)

[**Practical Date : 17 July 2021 2**](#_Toc81559354)

[**Practical Aim : Implement FCFS scheduling Algorithm using Java 2**](#_Toc81559355)

[**Algorithm : 2**](#_Toc81559356)

[**Solved Example 2**](#_Toc81559357)

[**Gnatt Chart 4**](#_Toc81559359)

[**Implementation 7**](#_Toc81559363)

[**Input 10**](#_Toc81559364)

[**Output :- 10**](#_Toc81559365)

[**Sample output 01 11**](#_Toc81559366)

[**Sample output 02 11**](#_Toc81559367)

[**Sample output 03 12**](#_Toc81559369)

[**Sample output 04 12**](#_Toc81559371)

**USCSP301 – USCS303 : Operating system (OS) Practical – 01**

**Practical – 01 First Come First Serve (FCFS) Algorithm**

# **Practical Date :** 17 July 2021

# **Practical Aim :** Implement FCFS scheduling Algorithm using Java

**Algorithm :**

Step 1: Input the number of processes required to be scheduled using FCFS, burst time for each process and its arrival time.

Step 2: Using enhanced bubble sort technique, sort the all given processes in ascending order according to arrival time in a ready queue.

Step 3: Calculate the Finish Time, Turn Around Time and Waiting Time for each process which in turn help to calculate Average Waiting Time and Average Turn Around Time required by CPU to schedule given set of process using FCFS.

Step 3.1: for i=0, Finish Time To Arrival Time To + Burst Time To

Step 3.2: for i >= 1, Finish Time Ti = Burst Time Ti + Finish Time T i-1

Step 3.3: for i=0, Turn Around Time To Finish Time To - Arrival Time To

Step 3.4: for i >= 1, Turn Around Time Ti = Finish Time To - Arrival Time To

Step 3.5: for i=0, Waiting Time To Turn Around Time To-Burst Time To

Step 3.6: for i >= 1, Waiting Time Ti- Turn Around Time T₁ - Burst Time Ti-1

Step 4: Process with less arrival time comes first and gets scheduled first by the CPU.

Step 5: Calculate the Average Waiting Time and Average Turn Around Time.

Step 6: Stop.

**Solved Example**

**Example 01**

Consider the following example containing five processes arrive at same time

|  |  |
| --- | --- |
| Process ID | Burst Time |
| P0 | 6 |
| P1 | 3 |
| P2 | 8 |
| P3 | 3 |
| P4 | 4 |

Step 1: Processes get executed according to their arrival time.

Step 2: Following shows the scheduling and execution of processes.

Step 2.1: At start PO arrives and get executed for 6 (i.e; 0-6) seconds.

System Time : 0

Process Scheduled : P0

Turn Around Time : 6 - 0 = 6

Waiting Time : 6 – 6 = 0

Step 2.2: P1 arrives after completion of P0, P1 is executed for 3 (i.e; 6-9) seconds.

System Time : 6

Process Scheduled : P0,P1

Turn Around Time : 9 - 0 = 9

Waiting Time : 9 – 3 = 6

Step 2.3: P2 arrives after complete execution of process P1, for 8 (i.e; 9-17) seconds

System Time : 9

Process Scheduled : P0,P1,P2

Turn Around Time : 17 - 0 = 17

Waiting Time : 17 – 8 = 9

Step 2.4: P3 arrives and gets executed for 3 (Le: 17-20) seconds.

System Time : 17

Process Scheduled : P0,P1,P2,P3

Turn Around Time : 20 - 0 = 20

Waiting Time : 20 – 3 = 17

Step 2.5: Similarly, P4 arrives and gets executed for 4 (i.e; 20-24) seconds.

System Time : 20

Process Scheduled : P0,P1,P2,P3,P4

Turn Around Time :24 – 0 = 24

Waiting Time : 24 – 4 = 20

Step 3: Calculate Average Waiting Time and Average Turn Around Time.

Average Waiting Time = (0 +6+9+17+20)/5

= 52/5

= 10.4

Average Turn Around Time = (0 +6+9+17+20)/5

= 76/5

= 15.2

Step 4 : After scheduling of all provided processes;

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process  ID | Burst Time | Arrival  Time | Finish Time (Prev.finish time+Burst time) | Turn Around Time  (Finish time – Arrival  Time) | Waiting Time  (Turn Around Time – Burst Time |
| P0 | 6 | 0 | (-+6=)6 | (6-0=)6 | (6-6=)0 |
| P1 | 3 | 0 | (6+3=)9 | (9-0=)9 | (9-3=)6 |
| P2 | 8 | 0 | (9+8=)17 | (17-0=)17 | (17-8=)9 |
| P3 | 3 | 0 | (17+3)20 | (20-0=)20 | (20-3=)17 |
| P4 | 4 | 0 | (20+4=)24 | (24-0=)24 | (24-4=)20 |
| Average |  |  |  | 15.200000 | 10.400000 |

Step 5 : Stop

**Gnatt Chart**

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

**Example 02**

|  |  |  |
| --- | --- | --- |
| Process ID | Burst Time | Arrival Time |
| P0 | 6 | 2 |
| P1 | 3 | 5 |
| P2 | 8 | 1 |
| P3 | 3 | 0 |
| P4 | 4 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process  ID | Burst Time | Arrival  Time | Finish Time (Prev.finish time+Burst time) | Turn Around Time  (Finish time – Arrival  Time) | Waiting Time  (Turn Around Time – Burst Time |
| P3 | 3 | 0 | (-+3=)3 | (3-0=)3 | (3-3=)0 |
| P2 | 8 | 1 | (3+8=)11 | (11-1=)10 | (10-8=)2 |
| P0 | 6 | 2 | (11+6=)17 | (17-2=)15 | (15-6=)9 |
| P4 | 4 | 4 | (17+4)21 | (20-4=)17 | (17-4=)13 |
| P1 | 3 | 5 | (21+3=)24 | (24-5=)19 | (19-3=)16 |
| Average |  |  |  | 12.800000 | 8.000000 |

**Gnatt Chart**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **P3** | **P2** | **P0** | **P1** | **P1** |

**Example 03.**

|  |  |
| --- | --- |
| Process ID | Burst Time |
| P0 | 2 |
| P1 | 1 |
| P2 | 6 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process  ID | Burst Time | Arrival  Time | Finish Time (Prev.finish time+Burst time) | Turn Around Time  (Finish time – Arrival  Time) | Waiting Time  (Turn Around Time – Burst Time |
| P0 | 2 | 0 | (-+2=)2 | (2-0=)2 | (2-2=)0 |
| P1 | 1 | 0 | (2+1=)3 | (3-0=)3 | (3-1=)2 |
| P2 | 6 | 0 | (3+6=)9 | (9-0=)9 | (9-6=)3 |
| Average |  |  |  | 4.66666667 | 1.66666667 |

**Gnatt chart**

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

**Example 04**

|  |  |  |
| --- | --- | --- |
| Process ID | Burst Time | Arrival Time |
| P0 | 4 | 3 |
| P1 | 3 | 5 |
| P2 | 2 | 0 |
| P3 | 1 | 5 |
| P4 | 3 | 4 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Process  ID | Burst Time | Arrival  Time | Finish Time (Prev.finish time+Burst time) | Turn Around Time  (Finish time – Arrival  Time) | Waiting Time  (Turn Around Time – Burst Time |
| P2 | 2 | 0 | (-+2=)2 | (2-0=)2 | (2-2=)0 |
| P0 | 4 | 3 | (2+4=)6 | (6-3=)3 | (3-4=)-1 |
| P4 | 3 | 4 | (6+3=)9 | (9-4=)5 | (5-3=)2 |
| P1 | 3 | 5 | (9+3)12 | (12-5=)7 | (7-3=)4 |
| P3 | 1 | 5 | (12+1=)13 | (13-5=)8 | (8-1=)7 |
| Average |  |  |  | 5.0000000 | 2.4000000 |

**Gnatt Chart**

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

**Implementation**

**Java Program:**

import java.util.Scanner;

public class P1\_FCFS\_YP{

int burstTime[]; int arrivalTime[]; 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();

System.out.print("Enter the arrival time for Process"+(i)+" :"); arrivalTime[i]=input.nextInt();

}

}

void sortAccordingArrivalTime(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(at[j]>at[j+1]){ temp = at[j]; at[j] =at[j+1]; at[j+1]=temp;

temp = bt[j];

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

stemp = pid[j];

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

swapped=true;

}

}

if(swapped==false){ break;

}

}

}

void firstComeFirstServeAlgorithm(){

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]; sortAccordingArrivalTime(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("FCFS Schedulling 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){

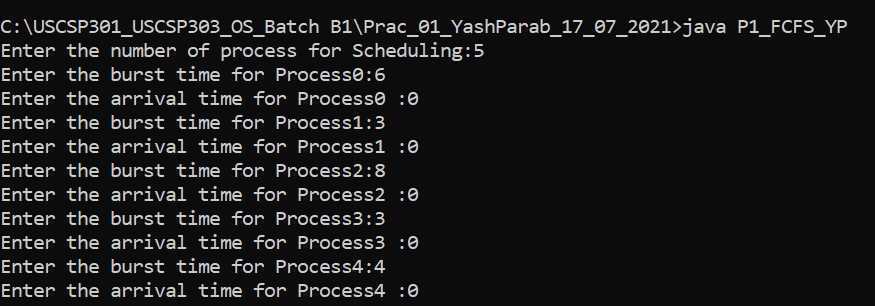
Scanner input= new Scanner(System.in);

P1\_FCFS\_YP obj = new P1\_FCFS\_YP(); obj.getProcessData(input); obj.firstComeFirstServeAlgorithm();

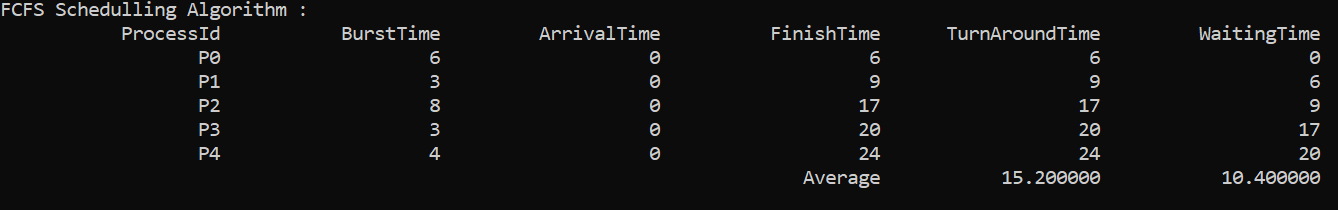
}

}

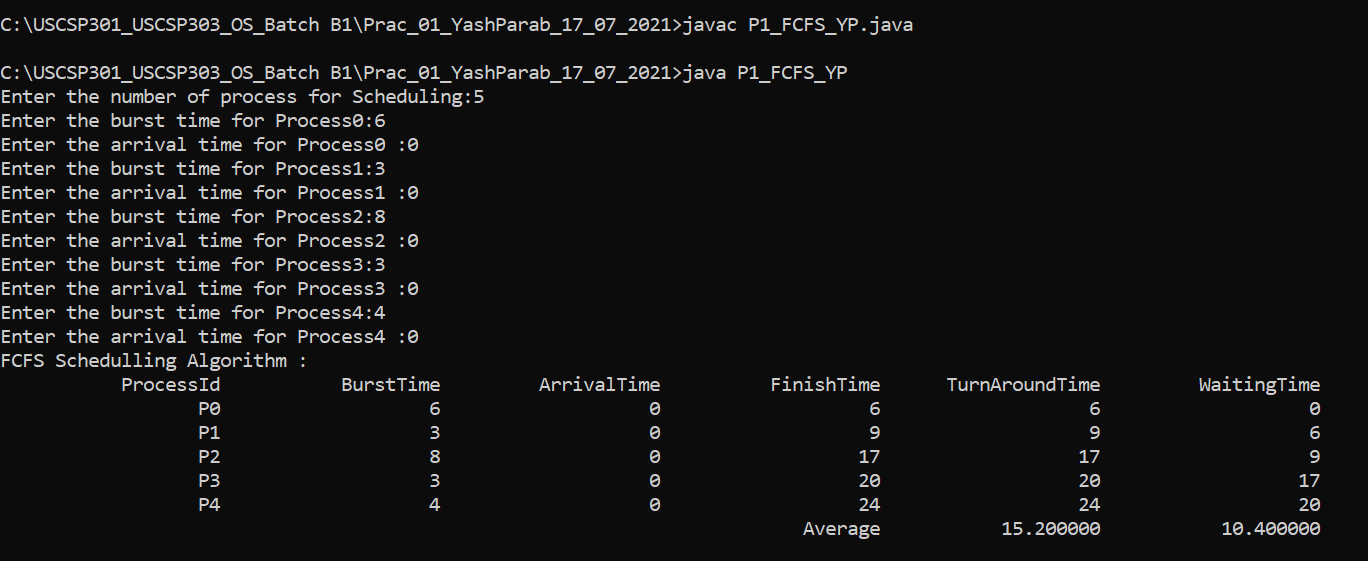
**Input**



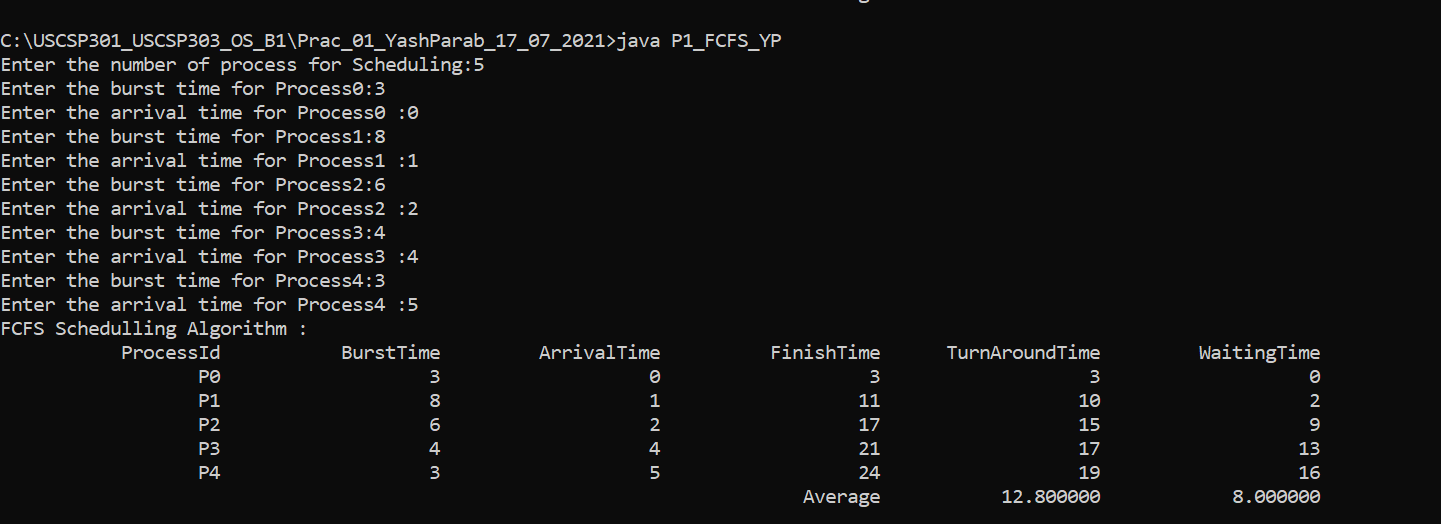
**Output :-**



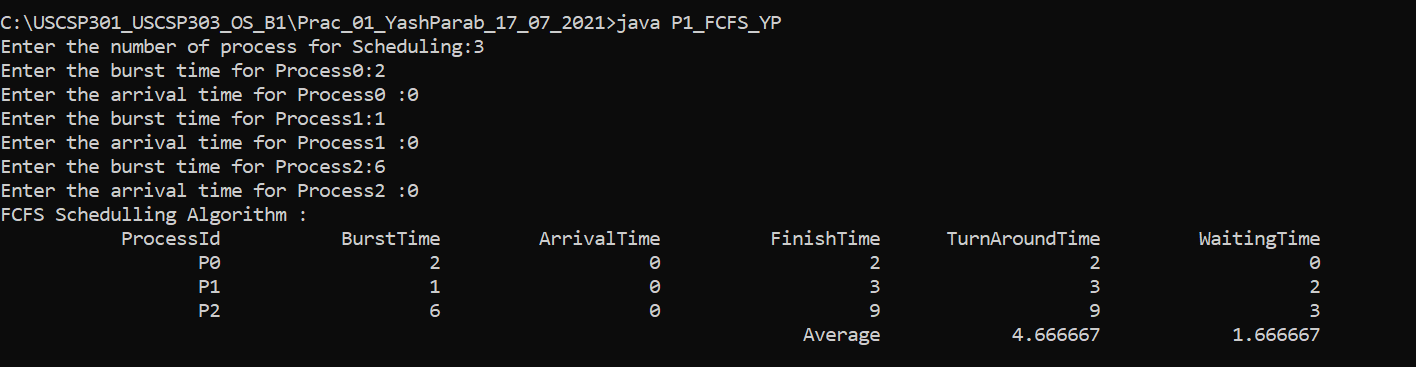
**Sample output 01**



**Sample output 02**



**Sample output 03**



**Sample output 04**

