

PART = I : System Programming & Operating System

(Group- B) Assignment No.04

Title : Statement: Write a program to simulate CPU Scheduling Algorithms: FCFS, SJF (Preemptive), Priority (Non-Preemptive) and Round Robin (Preemptive)

1. FCFS Code :

```
import java.io.*;

class Process
{
    String Pname;
    float ArrivalT;
    float BurstT;
    float WaitingT;
    float StartT;
    float FinishT;
    float TurnAroundT;
    public Process(String string, int i, int j) {
        Pname=string;
        ArrivalT=i;
        BurstT=j;
    }
}

public class FCFScode
{
    public static void main(String[] args)throws Exception {
        Process P[]=new Process[15];
        Process temp;
        int index=0;
        P[index]=new Process("T1",4,6);
        index++;
        P[index]=new Process("T2",0,2);
        index++;
    }
}
```

```

P[index]=new Process("T3",1,3);
index++;
P[index]=new Process("T4",3,4);
index++;
P[index]=new Process("T5",2,5);
index++;

System.out.println("Entered processes are ");
for(int a=0;a<index;a++)
{
    System.out.print(P[a].Pname+" ");
    System.out.print(P[a].ArrivalT+" ");
    System.out.println(P[a].BurstT);
}

for(int s=0;s<(index-1);s++)
{
    for(int t=0;t<(index-s-1);t++)
        if(P[t].ArrivalT>P[t+1].ArrivalT)
        {
            temp=P[t];
            P[t]=P[t+1];
            P[t+1]=temp;
        }
}

System.out.println("\nSorted Processes are: ");
for(int a=0;a<index;a++)
{
    System.out.print(P[a].Pname+" ");
    System.out.print(P[a].ArrivalT+" ");
    System.out.println(P[a].BurstT);
}

System.out.println("\nName"+" "+"AT"+" "+"BT"+" "+"WT"+" "+"ST"+"
"+"FT"+" "+"TAT");
P[0].StartT=P[0].ArrivalT;

```

```

        for(int k=0;k<index;k++)
        {
            P[k].WaitingT=P[k].StartT-P[k].ArrivalT;
            P[k].FinishT=P[k].StartT+P[k].BurstT;
            P[k].TurnAroundT=P[k].WaitingT+P[k].BurstT;
            if(k+1<index)
            {
                P[k+1].StartT=P[k].FinishT;
            }
        }
        System.out.println(P[k].Pname+"      "+P[k].ArrivalT+" "+P[k].BurstT+"
        "+P[k].WaitingT+" "+P[k].StartT+"      "+P[k].FinishT+"      "+P[k].TurnAroundT);
    }
}
}

```

FCFS Output :

```

PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

Entered processes are
T1      4.0      6.0
T2      0.0      2.0
T3      1.0      3.0
T4      3.0      4.0
T5      2.0      5.0

Sorted Processes are:
T2      0.0      2.0
T3      1.0      3.0
T5      2.0      5.0
T4      3.0      4.0
T1      4.0      6.0

Name    AT      BT      WT      ST      FT      TAT
T2      0.0      2.0      0.0      0.0      2.0      2.0
T3      1.0      3.0      1.0      2.0      5.0      4.0
T5      2.0      5.0      3.0      5.0      10.0     8.0
T4      3.0      4.0      7.0      10.0     14.0     11.0
T1      4.0      6.0      10.0     14.0     20.0     16.0

```

2.SJF Code:

```

import java.io.*;
import java.util.Scanner;

class Pro
{
    String pnm;

```

```

    int AT;
    int BT;
    int WT;
    int ST;
    int FT;
    int RT;
    int TAT;
    int flag;
    public Pro(String pnm1,int AT1,int BT1,int flag,int RT1)
    {
        this.pnm=pnm1;
        this.AT=AT1;
        this.BT=BT1;
        this.flag=flag;
        this.RT=RT1;
    }
}

public class SJFP
{
    public static void main(String args[])
    {
        Pro p[]=new Pro[15];
        int no;
        int att,btt,flag = 0,rtt;
        String pronm;
        Scanner sc=new Scanner(System.in);
        System.out.println("How many task to be entered??");
        no=sc.nextInt();
        for(int i=0;i<no;i++)
        {
            System.out.println("Enter Process_name,Arrival_Time ,Burst_Time");
            pronm=sc.next();
            att=sc.nextInt();
            btt=sc.nextInt();
            flag=sc.nextInt();

```

```

        rtt=sc.nextInt();
        p[i]=new Pro(prnm,att,btt,flag,rtt);
    }
    System.out.println("\nProcess\tArrival_time\tBurst_Time\tFlag\t Remaining
Time");
    for(int i=0;i<no;i++)
    {
        System.out.println("\n"+p[i].pnm+"\t\t"+p[i].AT+"\t"+p[i].BT);
    }
    //void sjf()
    //{
    int CT=0,min,index,i,flag1;
    int completeP=0;
    while(completeP<no)
    {
        min=9999;
        flag1=0;
        index=-1;
        for(i=0;i<no;i++)
        {
            if((p[i].AT<=CT)&&(p[i].flag==0))
            {
                if(p[i].RT<min)
                {
                    min=p[i].RT;
                    index=i;
                    flag1=1;
                }
            }
        }
        //if
        //for
        if(flag1==1)
        {
            if(p[index].RT==p[index].BT)
            {
                p[index].ST=CT;

```

```

        }//if
        p[index].RT--;
        CT++;
        if(p[index].RT==0)
        {
            p[index].FT=CT;
            p[index].WT=p[index].FT-p[index].AT-p[index].BT;
            p[index].TAT=p[index].WT+p[index].BT;
            completeP++;
            p[index].flag=1;
        }
    }//if
    else
    {
        CT++;
    }
}

}

System.out.println("\nProcess\tAT\tBT\tRT\tST\tFT\tWT\tTAT");

for(int j=0;j<no;j++)
{

    System.out.println("\n"+p[j].pnm+"\t\t"+p[j].AT+"\t"+p[j].BT+"\t"+p[j].RT+"\t"+p
[j].ST+"\t"+p[j].FT+"\t"+p[j].WT+"\t"+p[j].TAT);

}

}

```

SJF Output:

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\abhis\Desktop\SPOS Assignment CODES> javac SJFP.java
PS C:\Users\abhis\Desktop\SPOS Assignment CODES> java SJFP
How many task to be entered??
3
Enter Process_name,Arrival_Time ,Burst_Time
P1 0 10 0 10
Enter Process_name,Arrival_Time ,Burst_Time
P2 0 4 0 4
Enter Process_name,Arrival_Time ,Burst_Time
P3 1 2 0 2

Process	Arrival_time	Burst_Time	Flag	Remaining Time				
P1	0	10						
P2	0	4						
P3	1	2						
Process	AT	BT	RT	ST	FT	WT	TAT	
P1	0	10	0	0	6	16	6	16
P2	0	4	0	0	0	6	2	6
P3	1	2	0	1	1	3	0	2

3.Priority Code:

```
import java.util.Scanner;
```

```
class Process2{
    int pname;
    int pri;
    int bt;
    float wt;
    float st;
    float ft;
    float tat;
    public Process2(int a,int b){
        pri=a;
        bt=b;
    }
}

public class Priority
{
    int n;
    float awt=0,atat=0;
    void work()
    {
        int a,b;
        System.out.println("Enter no of processes.");
```

```

Scanner sc=new Scanner(System.in);
n=sc.nextInt();
Process2 P[]=new Process2[n];
for(int i=0;i<n;i++){
    System.out.println("enter the priority and burst time for process "
+(i+1));

    a=sc.nextInt();
    b=sc.nextInt();
    P[i]=new Process2(a,b);
    P[i].pname=(i+1);
}

for(int i=0;i<n-1;i++)
{
    for(int j=i+1;j<n;j++)
    {
        if(P[i].pri<P[j].pri)
        {
            int x=P[i].pri;
            P[i].pri=P[j].pri;
            P[j].pri=x;
            x=P[i].bt;
            P[i].bt=P[j].bt;
            P[j].bt=x;
            x=P[i].pname;
            P[i].pname=P[j].pname;
            P[j].pname=x;
        }
    }
}

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

    P[i].ft=P[i].st+P[i].bt;
    P[i].wt=P[i].st;
    P[i].tat=P[i].wt+P[i].bt;
}

```



```

        if(i<(n-1))
            P[i+1].st=P[i].ft;
            awt=awt+P[i].wt;
            atat=atat+P[i].tat;
    }
    awt=awt/n;
    atat=atat/n;

    System.out.println("Proc\tPrior\tburstT\tWaitT\tTAT\tStart\tFinish");
    for(int i=0;i<n;i++){

        System.out.println("P"+i+"\t"+P[i].pri+"\t"+P[i].bt+"\t"+P[i].wt+"\t"+P[i].tat+"
\t"+P[i].st+"\t"+P[i].ft);
    }
    System.out.println("Tha average waiting time is "+awt);
    System.out.println("Tha average turn around time is "+atat);
}

public static void main(String[] args) {
    Priority ob=new Priority();
    ob.work();
}
}

```

Priority Output:

```

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abhis\Desktop\SPOS Assignment CODES> javac Priority.java
PS C:\Users\abhis\Desktop\SPOS Assignment CODES> java Priority
Enter no of processes.
5
enter the priority and burst time for process 1
3 10
enter the priority and burst time for process 2
1 1
enter the priority and burst time for process 3
4 2
enter the priority and burst time for process 4
5 1
enter the priority and burst time for process 5
2 5
Proc   Prior  burstT  WaitT   TAT     Start  Finish
P0      5       1       0.0     1.0     0.0    1.0
P1      4       2       1.0     3.0     1.0    3.0
P2      3      10       3.0    13.0     3.0   13.0
P3      2       5      13.0    18.0    13.0   18.0
P4      1       1      18.0    19.0    18.0   19.0
Tha average waiting time is 7.0
Tha average turn around time is 10.8

```

4.Round Robin:

```
import java.util.Scanner;

public class RR
{
    public static void main(String args[])
    {
        int n,i,qt,count=0,temp,sq=0,bt[],wt[],tat[],rem_bt[];
        float awt=0,atat=0;

        bt = new int[10];
        wt = new int[10];
        tat = new int[10];
        rem_bt = new int[10];

        Scanner s=new Scanner(System.in);

        System.out.print("Enter the number of process (maximum 10) = ");
        n = s.nextInt();

        System.out.print("Enter the burst time of the process\n");
        for (i=0;i<n;i++)
        {
            System.out.print("P"+i+" = ");
            bt[i] = s.nextInt();
            rem_bt[i] = bt[i];
        }

        System.out.print("Enter the quantum time: ");
        qt = s.nextInt();

        while(true)
        {
            for (i=0,count=0;i<n;i++)
            {
                temp = qt;
                if(rem_bt[i] == 0)
                {
                    count++;
                    continue;
                }
                if(rem_bt[i]>qt)
```

```

rem_bt[i]= rem_bt[i] - qt;
else
if(rem_bt[i]>=0)
{
temp = rem_bt[i];
rem_bt[i] = 0;
}
sq = sq + temp;
tat[i] = sq;
}
if(n == count)
break;
}

System.out.print("-----
-----");

System.out.print("\nProcess\t      Burst Time\t      Turnaround Time\t
Waiting Time\n");

System.out.print("-----
-----");

for(i=0;i<n;i++)
{
wt[i]=tat[i]-bt[i];
awt=awt+wt[i];
atat=atat+tat[i];
System.out.print("\n "+(i+1)+"\t "+bt[i]+" \t\t "+tat[i]+" \t\t "+wt[i]+" \n");
}

awt=awt/n;
atat=atat/n;

System.out.println("\nAverage waiting Time = "+awt+"\n");
System.out.println("Average turnaround time = "+atat);
}
}

```

Round Robin Output:

```
PROBLEMS 10 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abhis\Desktop\SPOS Assignment CODES> javac RR.java
PS C:\Users\abhis\Desktop\SPOS Assignment CODES> java RR
Enter the number of process (maximum 10) = 3
Enter the burst time of the process
P0 = 10
P1 = 5
P2 = 8
Enter the quantum time: 2
-----
Process      Burst Time      Turnaround Time      Waiting Time
-----
1            10             23                   13
2             5             15                   10
3             8             21                   13

Average waiting Time = 12.0
Average turnaround time = 19.666666
```

All 4 CPU Scheduling Algorithms in one Programme:

```
import java.util.*;

import java.util.Arrays;

public class scheduling_Algorithms
{
    //int no_of_process;
    static int no_of_process;
    static int proc_no[]=new int[15];
    static int burst_time[]=new int[15];
    public static void fcfs()
    {
        int waiting_time=0;
        int t_time;
        burst_time[0]=0;

        System.out.println("Process"+"\\t"+"Burst time"+"\\t"+"waiting
time"+"\\t"+"turnaround time"+"\\n");

        for(int i=1;i<=no_of_process;i++)
        {
            waiting_time=waiting_time+burst_time[i-1];
            t_time=waiting_time+burst_time[i];
```

```

        System.out.println(proc_no[i]+"\\t"+burst_time[i]+"\\t"+"\\t"+waiting_time+"\\t"+"\\t"
"+t_time+"\\n");
    }
}
public static void SJF()
{
    int temp,temp1;
    int proc_no1[]=new int[15];
    int burst_time1[]=new int[15];
    for(int i=1;i<=no_of_process;i++)
    {
        proc_no1[i]=proc_no[i];
        burst_time1[i]=burst_time[i];
    }
    for(int i=1;i<=no_of_process-1;i++)
    {
        for(int j=1;j<=(no_of_process-1);j++)
        {
            if(burst_time1[j]>burst_time1[j+1])
            {
                temp=burst_time1[j+1];
                temp1=proc_no1[j+1];
                burst_time1[j+1]=burst_time1[j];
                proc_no1[j+1]=proc_no1[j];
                burst_time1[j]=temp;
                proc_no1[j]=temp1;
            }
        }
    }
    int waiting_time=0;
    int t_time;
    burst_time1[0]=0;

    System.out.println("Process"+"\\t"+"Burst time"+"\\t"+"waiting
time"+"\\t"+"turnaround time"+"\\n");
    for(int i=1;i<=no_of_process;i++)
    {
        waiting_time=waiting_time+burst_time1[i-1];

```

```

        t_time=waiting_time+burst_time1[i];

        System.out.println(proc_no1[i]+"\\t"+burst_time1[i]+"\\t"+"\\t"+waiting_time+"\\t"+"\\t"+t_time+"\\n");
    }

}

static void priority()
{
    Scanner sc=new Scanner(System.in);

    int pr1[]=new int[15];
    int temp,temp1,temp2;
    int proc_no1[]=new int[15];
    int burst_time1[]=new int[15];
    for(int i=1;i<=no_of_process;i++)
    {
        System.out.print("priority of "+proc_no[i]+": ");
        pr1[i]=sc.nextInt();
        System.out.println("\\n");
    }

    for(int i=1;i<=no_of_process;i++)
    {
        proc_no1[i]=proc_no[i];
        burst_time1[i]=burst_time[i];
    }

    for(int i=1;i<=no_of_process-1;i++)
    {
        for(int j=1;j<=(no_of_process-1);j++)
        {
            if(pr1[j]>pr1[j+1])
            {
                temp=burst_time1[j+1];
                temp1=proc_no1[j+1];

```

```

        temp2=pr1[j+1];
        burst_time1[j+1]=burst_time1[j];
        proc_no1[j+1]=proc_no1[j];
        pr1[j+1]=pr1[j];
        burst_time1[j]=temp;
        proc_no1[j]=temp1;
        pr1[j]=temp2;
    }
}

int waiting_time=0;
int t_time;
burst_time1[0]=0;

System.out.println("Process"+"\\t"+"Burst time"+"\\t"+"waiting
time"+"\\t"+"turnaround time"+"\\t"+"priority"+"\\n");

for(int i=1;i<=no_of_process;i++)
{
    waiting_time=waiting_time+burst_time1[i-1];
    t_time=waiting_time+burst_time1[i];

    System.out.println(proc_no1[i]+"\\t"+burst_time1[i]+"\\t"+"\\t"+waiting_time+"\\t"+"
\\t"+t_time+"\\t"+"\\t"+pr1[i]+"\\n");
}

}

static void RR()
{

Scanner sc=new Scanner(System.in);
System.out.println("enter Quantum: ");
int quantum=sc.nextInt();
// Make a copy of burst times bt[] to store remaining
// burst times.
int rem_bt[] = new int[no_of_process+1];
int wt[]=new int[no_of_process+1];
for (int i = 1 ; i <= no_of_process ; i++)

```

```

    rem_bt[i] = burst_time[i];

int t = 0; // Current time
int t_time;
// Keep traversing processes in round robin manner
// until all of them are not done.
while(true)
{
    boolean done = true;

    // Traverse all processes one by one repeatedly
    for (int i = 1 ; i <= no_of_process; i++)
    {
        // If burst time of a process is greater than 0
        // then only need to process further
        if (rem_bt[i] > 0)
        {
            done = false; // There is a pending process

            if (rem_bt[i] > quantum)
            {
                // Increase the value of t i.e. shows
                // how much time a process has been processed
                t += quantum;

                // Decrease the burst_time of current process
                // by quantum
                rem_bt[i] -= quantum;
            }

            // If burst time is smaller than or equal to
            // quantum. Last cycle for this process
            else
            {
                // Increase the value of t i.e. shows

```



```

        // how much time a process has been processed
        t = t + rem_bt[i];

        // Waiting time is current time minus time
        // used by this process
        wt[i] = t - burst_time[i];

        // As the process gets fully executed
        // make its remaining burst time = 0
        rem_bt[i] = 0;
    }
}

// If all processes are done
if (done == true)
    break;
}

for(int i=1;i<=no_of_process;i++)
{
    t_time=wt[i]+burst_time[i];

    System.out.println("Process"+"\\t"+"Burst time"+"\\t"+"waiting
time"+"\\t"+"turnaround time"+"\\t"+"priority"+"\\n");

    System.out.println(proc_no[i]+"\\t"+burst_time[i]+"\\t"+"\\t"+wt[i]+"\\t"+"\\t"+t_time+"\\n");
}
}

public static void main(String args[])
{
    int choice,choice1;
    Scanner sc=new Scanner(System.in);
    System.out.println("Enter no of processors"+"\\n");
    no_of_process=sc.nextInt();

```

```

        for (int i=1;i<=no_of_process;i++)
        {
            System.out.println("Enter Process no: ");
            proc_no[i]=sc.nextInt();
            System.out.println("Enter burst time: ");
            burst_time[i]=sc.nextInt();

        }
        do{
            System.out.println("    Sheduling Algorithms"+"\\n");

            System.out.println("1.FCFS"+"\\n"+"2.SJF"+"\\n"+"3.Priority"+"\\n"+"4.RR"+"\\n"+"5.Exit"+"\\n");

            System.out.println("enter your choice: ");
            choice=sc.nextInt();
            switch(choice)
            {
                case 1:fcfs();
                    break;

                case 2:SJF();
                    break;

                case 3:priority();
                    break;

                case 4:RR();
                    break;

                case 5:System.exit(0);
                    break;

            }
        }while(choice!=6);
    }
}

```

Output:

```
scheduling_Algorithms.java - SPOS Assignment CODES - Visual Stu...
PROBLEMS 10 OUTPUT TERMINAL ...
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\abhis\Desktop\SPOS Assignment CODES> java scheduling_Algorithms

Enter no of processors
5
Enter Process no:
1
Enter burst time:
10
Enter Process no:
2
Enter burst time:
1
Enter Process no:
3
Enter burst time:
2
Enter Process no:
4
Enter burst time:
1
Enter Process no:
5
Enter burst time:
5
```

```
scheduling_Algorithms.java - SPOS Assignment CODES - Visual Stu...
PROBLEMS 10 OUTPUT TERMINAL ...
Sheduling Algorithms

1.FCFS
2.SJF
3.Priority
4.RR
5.Exit

enter your choice:
1
Process Burst time      waiting time      turnaround time
1      10      0      10
2      1      10      11
3      2      11      13
4      1      13      14
5      5      14      19
```


Sheduling Algorithms

- 1.FCFS
- 2.SJF
- 3.Priority
- 4.RR
- 5.Exit

enter your choice:

4

enter Quantum:

3

Process	Burst time	waiting time	turnaround time	priority
---------	------------	--------------	-----------------	----------

1	10	9	19	
---	----	---	----	--

Process	Burst time	waiting time	turnaround time	priority
---------	------------	--------------	-----------------	----------

2	1	3	4	
---	---	---	---	--

Process	Burst time	waiting time	turnaround time	priority
---------	------------	--------------	-----------------	----------

3	2	4	6	
---	---	---	---	--

Process	Burst time	waiting time	turnaround time	priority
---------	------------	--------------	-----------------	----------

4	1	6	7	
---	---	---	---	--

Process	Burst time	waiting time	turnaround time	priority
---------	------------	--------------	-----------------	----------

5	5	10	15	
---	---	----	----	--

Sheduling Algorithms

- 1.FCFS
- 2.SJF
- 3.Priority
- 4.RR
- 5.Exit

enter your choice:

5