****

EXPERIMENT NO : 5

TITLE : Write a Java program (using OOP features) to implement following scheduling algorithms: FCFS , SJF (Preemptive), Priority (NonPreemptive) and Round Robin (Preemptive)

NAME : DESALE DARSHAN RAMDAS

CLASS : TE

ROLL NO : 23 A

DATE : 2-09-2022 BATCH : A

****

FCFS, SJF, PRIORITY SCHEDULING & ROUND ROBIN USING SWITCH CASE

```
//DESALE DARSHAN RAMDAS
//DESALE DARSHAN RAMDAS
import java.util.*;
public class Scheduling
{
    public static void main(String[] args)
    {
        Scheduling obj=new Scheduling();
        Scanner sc =new Scanner(System.in);
        while(true)
        {
            System.out.println("\n\t\tEnter the choice:");
            System.out.println("\n\t\t A:FCFS\n\t\t B:SJF\n\t\t C:Priority
Scheduling\n\t\t D:Round Robin Scheduling");
            char choice=sc.next().charAt(0);

            switch (choice) {
                case 'A':obj.FCFS();
                       break;
                case 'B':obj.SJF();
                       break;
                case 'C':obj.Priority();
                       break;
```

```
                case 'D':obj.round_robin();
                        break;
                default:
                        System.out.println("\n\t\tInvalid operations to be
performed.");
                        System.out.println("\n\t\t*******Thank you********");
                        System.exit(0);
                    break;
        }
        }

    }
    public void FCFS()
    {
        System.out.println("FCFS Scheduling");
        Scanner cin=new Scanner(System.in);
        System.out.println("Enter the no of processes:");
        int pr=cin.nextInt();
        int Processes[]=new int[pr];
        int Burst_T[]=new int[pr];
        int Waiting_T[]=new int[pr];
        int Turnaround_T[]=new int[pr];
        System.out.println("CPU time for"+(pr)+"Processes:\n");
        for(int i=0;i<pr;i++)
        {
            System.out.println("Burst time for Process ["+(i+1)+ "] :");
            Burst_T[i]=cin.nextInt();
            Processes[i]=i+1;
        }
        Waiting_T[0]=0;
        Turnaround_T[0]=Burst_T[0];
        int Total_WT=0,Total_TAT=0;
        for(int j=1;j<pr;j++)
        {
            Waiting_T[j]=Waiting_T[j-1]+Burst_T[j-1];
            Total_WT+=Waiting_T[j];
            Turnaround_T[j]=Waiting_T[j]+Burst_T[j];
            Total_TAT+=Turnaround_T[j];

        }
        float Avg_TurnaroundT=(float)Total_TAT/pr;
        float Avg_WaitingT=(float)Total_WT/pr;
        System.out.println("\t\t\t\t########### FCFS Scheduling
###########\n\n");
```

```
        System.out.println("\t\tProcess\t\tBurst_time\t\tWaiting_time\t\tTurnarou
nd_time");
        for(int k=0;k<pr;k++)
        {
            System.out.println("\t\t"+(Processes[k])+"\t\t\t"+(Burst_T[k])+"\t\t\
t"+(Waiting_T[k])+"\t\t\t"+(Turnaround_T[k]));
            System.out.println();
        }
        System.out.println("Average Waiting Time : " +Avg_WaitingT);
        System.out.println("Average Turnaround Time : " +Avg_TurnaroundT);


    }
    void SJF()
    {
        System.out.println("SJS Scheduling");
        Scanner cin=new Scanner(System.in);
        System.out.println("Enter the no of processes:");
        int pr=cin.nextInt();
        int Processes[]=new int[pr];
        int Burst_T[]=new int[pr];
        int Waiting_T[]=new int[pr];
        int Turnaround_T[]=new int[pr];
        System.out.println("CPU time for"+(pr)+"Processes:\n");
        for(int i=0;i<pr;i++)
        {
            System.out.println("Burst time for Process ["+(i+1)+ "] :");
            Burst_T[i]=cin.nextInt();
            Processes[i]=i+1;
        }
        Waiting_T[0]=0;
        Turnaround_T[0]=Burst_T[0];
        int Total_WT=0,Total_TAT=0;
        for (int s = 0; s < pr; s++)
        {
            for (int t = s + 1; t < pr; t++)
            {
                if (Burst_T[s] > Burst_T[t])
                {
                    int temp_Burst_T = Burst_T[s];
                    Burst_T[s] = Burst_T[t];
                    Burst_T[t] = temp_Burst_T;

                    int temp_Processes = Processes[s];
                    Processes[s] = Processes[t];
```

```
                    Processes[t] = temp_Processes;
                }
            }
        }
        for(int j=1;j<pr;j++)
        {
            Waiting_T[j]=Waiting_T[j-1]+Burst_T[j-1];
            Total_WT+=Waiting_T[j];
            Turnaround_T[j]=Waiting_T[j]+Burst_T[j];
            Total_TAT+=Turnaround_T[j];

        }
        float Avg_TurnaroundT=(float)Total_TAT/pr;
        float Avg_WaitingT=(float)Total_WT/pr;
        System.out.println("\t\t\t\t########### SJF Scheduling ###########\n\n");
        System.out.println("\t\tProcess\t\tBurst_time\t\tWaiting_time\t\tTurnarou
nd_time");
        for(int k=0;k<pr;k++)
        {
            System.out.println("\t\t"+(Processes[k])+"\t\t\t"+(Burst_T[k])+"\t\t\
t"+(Waiting_T[k])+"\t\t\t"+(Turnaround_T[k]));
            System.out.println();
        }
        System.out.println("Average Waiting Time : " +Avg_WaitingT);
        System.out.println("Average Turnaround Time : " +Avg_TurnaroundT);
    }


    void Priority()
    {

            System.out.println("Priority Scheduling");
            Scanner cin=new Scanner(System.in);
            System.out.println("Enter the no of processes:");
            int pr=cin.nextInt();
            int Processes[]=new int[pr];
            int Burst_T[]=new int[pr];
            int Waiting_T[]=new int[pr];
            int Turnaround_T[]=new int[pr];
            int Priority_P[]=new int[pr];
            System.out.println("CPU time for"+(pr)+"Processes:\n");
            for(int i=0;i<pr;i++)
            {
                System.out.println("Burst time for Process ["+(i+1)+ "] is :");
                Burst_T[i]=cin.nextInt();
```

```
                System.out.println("Priority for Process ["+(i+1)+ "] is :");
                Priority_P[i]=cin.nextInt();
                Processes[i]=i+1;
            }
            Waiting_T[0]=0;
            Turnaround_T[0]=Burst_T[0];
            int Total_WT=0,Total_TAT=0;
            for (int s = 0; s < pr; s++) {
                for (int t = s + 1; t < pr; t++) {
                    if (Priority_P[s] <Priority_P[t]) {             // sort on
basis of Priority

                        int temp_Burst_T =Burst_T[s];
                       Burst_T[s] = Burst_T[t];
                       Burst_T[t] = temp_Burst_T;

                        int temp_processes = Processes[s];
                        Processes[s] = Processes[t];
                        Processes[t] = temp_processes;

                        int temp_Priority=Priority_P[s];
                        Priority_P[s] = Priority_P[t];
                        Priority_P[t] = temp_Priority;
                    }
                }
            }
            for(int j=1;j<pr;j++)
            {
                Waiting_T[j]=Waiting_T[j-1]+Burst_T[j-1];
                Total_WT+=Waiting_T[j];
                Turnaround_T[j]=Waiting_T[j]+Burst_T[j];
                Total_TAT+=Turnaround_T[j];

            }
            float Avg_TurnaroundT=(float)Total_TAT/pr;
            float Avg_WaitingT=(float)Total_WT/pr;
            System.out.println("\t\t\t\t########### Priority Scheduling
###########\n\n");
            System.out.println("\t\tProcess\t\tBurst_time\t\tWaiting_time\t\tTurn
around_time");
            for(int k=0;k<pr;k++)
            {
                System.out.println("\t\t"+(Processes[k])+"\t\t\t"+(Burst_T[k])+"\
t\t\t"+(Waiting_T[k])+"\t\t\t"+(Turnaround_T[k]));
                System.out.println();
```

```
            }
            System.out.println("Average Waiting Time : " +Avg_WaitingT);
            System.out.println("Average Turnaround Time : " +Avg_TurnaroundT);
    }
    void round_robin()
    {

            Scanner cin = new Scanner(System.in);
            int pr, Quantum;
            System.out.print("\nEnter the Total number of Process :- \n");
            pr = cin.nextInt();
            int Waiting_T[] = new int[pr];
            int Burst_T[] = new int[pr];
            int R_Time[] = new int[pr];
            int Turnaround_T[] = new int[pr];
            int Processes[] = new int[pr];

            System.out.println("\nEnter the CPU Time :");
            for (int i = 0; i < pr; i++) {
                System.out.print("\tprocess P[" + (i + 1) + "] Burst Time :- ");
                Burst_T[i] = cin.nextInt();
                R_Time[i] = Burst_T[i];
                Processes[i]=i+1;
            }
            System.out.print("\n\nEnter Quantum Time : ");
            Quantum = cin.nextInt();
            int R_P = pr;
            int i = 0;
            int time = 0;
            int Total_WT=0,Total_TAT=0;
            System.out.println("\n\t************ The GANTT chart for Round Robin
Scheduling will be ************ \n");
            while (R_P != 0) {
                if (R_Time[i] > Quantum) {
                    R_Time[i] = R_Time[i] - Quantum;
                    System.out.print(" | P[" + (i + 1) + "] | ");
                    time += Quantum;
                    System.out.print(time);
                } else if (R_Time[i] <= Quantum && R_Time[i] > 0) {
                    time += R_Time[i];
                    R_Time[i] = R_Time[i] - R_Time[i];
                    System.out.print(" | P[" + (i + 1) + "] | ");
                    R_P--;
                    Waiting_T[i] = time - Burst_T[i];
                    Total_WT += Waiting_T[i];
```

```
                    Turnaround_T[i] = time;
                    Total_TAT += Turnaround_T[i];
                    System.out.print(time);
                }
                i++;
                if (i == pr) {
                    i = 0;
                }
            }
            float Avg_TurnaroundT=(float)Total_TAT/pr;
            float Avg_WaitingT=(float)Total_WT/pr;
            System.out.println("\t\t\t\t########### Round Robin Scheduling
###########\n\n");
            System.out.println("\t\tProcess\t\tBurst_time\t\tWaiting_time\t\tTurn
around_time");
            for(int k=0;k<pr;k++)
            {
                System.out.println("\t\t"+(Processes[k])+"\t\t\t"+(Burst_T[k])+"\
t\t\t"+(Waiting_T[k])+"\t\t\t"+(Turnaround_T[k]));
                System.out.println();
            }
            System.out.println("Average Waiting Time : " +Avg_WaitingT);
            System.out.println("Average Turnaround Time : " +Avg_TurnaroundT);
        }
}
```

Output :-

FCFS :

Enter the choice:

A:FCFS

B:SJF

C:Priority Scheduling

D:Round Robin Scheduling

A

FCFS Scheduling

Enter the no of processes:

4

CPU time for4Processes:

Burst time for Process [1] :

23

Burst time for Process [2] :

13

Burst time for Process [3] :

9

Burst time for Process [4] :

3

########## FCFS Scheduling ##########

Process	Burst_time	Waiting_time	Turnaround_time
1	23	0	23
2	13	23	36
3	9	36	45

Average Waiting Time : 26.0

Average Turnaround Time : 32.25

Enter the choice:

A:FCFS

B:SJF

C:Priority Scheduling

D:Round Robin Scheduling

B

SJS Scheduling

Enter the no of processes:

4

CPU time for4Processes:

Burst time for Process [1] :

26

Burst time for Process [2] :

13

Burst time for Process [3] :

9

Burst time for Process [4] :

6

############ SJF Scheduling ############

Process	Burst_time	Waiting_time	Turnaround_time
4	6	0	26
3	9	6	15
2	13	15	28
1	26	28	54

Average Waiting Time : 12.25

Average Turnaround Time : 24

Enter the choice:

A:FCFS

B:SJF

C:Priority Scheduling

D:Round Robin Scheduling

C

Priority Scheduling

Enter the no of processes:

4

CPU time for4Processes:

Burst time for Process [1] is :

26

Priority for Process [1] is :

4

Burst time for Process [2] is :

13

Priority for Process [2] is :

3

Burst time for Process [3] is :

9

Priority for Process [3] is :

2

Burst time for Process [4] is :

3

Priority for Process [4] is :

1

########## Priority Scheduling ##########

Process	Burst_time	Waiting_time	Turnaround_time
1	26	0	26
2	13	26	39
3	9	39	48

4 3 48 51

Average Waiting Time : 28.25

Average Turnaround Time : 34.5

A:FCFS

B:SJF

C:Priority Scheduling

D:Round Robin Scheduling

D

Enter the Total number of Process :-

4

Enter the CPU Time :

process P[1] Burst Time :- 26

process P[2] Burst Time :- 13

process P[3] Burst Time :- 9

process P[4] Burst Time :- 3

Enter Quantum Time : 5

********** The GANTT chart for Round Robin Scheduling will be **********

| P[1] | 5 | P[2] | 10 | P[3] | 15 | P[4] | 18 | P[1] | 23 | P[2] | 28 | P[3] | 32 | P[1] | 37 | P[2] | 40 | P[1] | 45 | P[1] | 50 | P[1]

| 51 ########## Round Robin Scheduling ##########

Process	Burst_time	Waiting_time	Turnaround_time
1	26	25	51
2	13	27	40
3	9	23	32
4	3	15	18

Average Waiting Time : 22.5

Average Turnaround Time : 35.25

Enter the choice:

A:FCFS

B:SJF

C:Priority Scheduling

D:Round Robin Scheduling

E

Invalid operations to be performed.

*******Thank you********