Experiment no :-5

FCFS

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

public class FCFS {

	private Scanner sc;

	public void  execute()
	{
		sc = new Scanner(System.in);

		//--------FCFS
		System.out.println("Enter Number of Processes:");
		int numProcess=sc.nextInt();
		Process []process=new Process[numProcess];

		//Accept Input
		for(int i=0;i<numProcess;i++)
		{
			System.out.println("P("+(i+1)+"):Enter Arrival time & Burst time");
			int at=sc.nextInt();
			int bt=sc.nextInt();
			//System.out.println("P("+(i+1)+"):Enter Arrival time");

			process[i]=new Process("P"+(i+1), bt, at);
		}

		//Sorting processes according to Arrival Time //No need if you take AT=0 or in ascending order
		Arrays.sort(process,new SortByArrival());

		int sum=0;
		double TotalWT=0,TotalTAT=0,avgWT=0,avgTAT=0;
		System.out.println("\n\nPRNo\tBT\tAT\tCT\tTAT\tWT");

	System.out.println("============================================================
==============================");
		for(int i=0;i<numProcess;i++)
		{
			sum=process[i].CT=sum+process[i].BT; //process 1 CT= sum=0+24=24+3=27
			process[i].TAT=process[i].CT-process[i].AT;//process 1TAT=27-0=27
			process[i].WT=process[i].TAT-process[i].BT;//Process 1 WT=27-3=24

			TotalWT=TotalWT+process[i].WT;
			TotalTAT=TotalTAT+process[i].TAT;

			process[i].display();
		}
		avgTAT=(double)TotalTAT/numProcess;
		avgWT=(double)TotalWT/numProcess;
		System.out.println("Average Waiting Time"+avgWT);
		System.out.println("Average TAT="+avgTAT);
```

```
	}

}
```

PriorityNonPreemptive

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

public class PriorityNonPreemptive {
	private Scanner sc;

	public void  execute()
	{
		sc = new Scanner(System.in);

		//--------FCFS
		System.out.println("Enter Number of Processes:");
		int numProcess=sc.nextInt();
		Process []process=new Process[numProcess];

		//Accept Input
		for(int i=0;i<numProcess;i++)
		{
			System.out.println("P("+(i+1)+"):Enter  Burst time  & priority"); //
			int at=0;//sc.nextInt();
			//Note: Arrival time is 0 for all processes;
			int bt=sc.nextInt();
			int priority=sc.nextInt();
			//System.out.println("P("+(i+1)+"):Enter Arrival time");

			process[i]=new Process("P"+(i+1), bt, at,priority);
		}
		//Sorting processes according to Arrival Time //No need if you take AT=0 or in ascending order
				Arrays.sort(process,new SortByPriority());

				int sum=0;
				double TotalWT=0, TotalTAT=0,avgWT=0,avgTAT=0;
				System.out.println("\n\nPRNo\tBT\tAT\tCT\tTAT\tWT\tPR");

	System.out.println("============================================================
==============================");
				for(int i=0;i<numProcess;i++)
				{
					sum=process[i].CT=sum+process[i].BT;
					process[i].TAT=process[i].CT-process[i].AT;
					process[i].WT=process[i].TAT-process[i].BT;

					TotalWT=TotalWT+process[i].WT;
					TotalTAT=TotalTAT+process[i].TAT;

					process[i].display();
				}
```

```
                avgTAT=(double)TotalTAT/numProcess;
                avgWT=(double)TotalWT/numProcess;
                System.out.println("Average Waiting Time"+avgWT);
                System.out.println("Average TAT="+avgTAT);
        }
}
```

Process

```
import java.util.Comparator;

public class Process {
String name;
int BT,WT,AT,CT,TAT,remBT,priority;
boolean flag;
public Process(String name,int burst,int AT)
{
        this.name=name;
        BT=burst;
        this.AT=AT;
        WT=CT=TAT=0;
        remBT=BT;
        priority=0;
}
public Process(String name,int burst,int AT,int PR)
{
        this.name=name;
        BT=burst;
        this.AT=AT;
        WT=CT=TAT=0;
        remBT=BT;
        priority=PR;
        flag=false;
}
public void display()
{
        System.out.println(name+"\t"+BT+"\t"+AT+"\t"+CT+"\t"+TAT+"\t"+WT+"\t"+priority);
}
}

//Class for sorting Processes
class SortByArrival implements Comparator<Process>
{

        @Override
        public int compare(Process p1, Process p2) {

                return p1.AT-p2.AT;
        }

}
```

```
class SortByPriority implements Comparator<Process>
{

	@Override
	public int compare(Process o1, Process o2) {

		return o1.priority-o2.priority;
	}

}
```

RoundRobin

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

public class RoundRobin {
	private Scanner sc;
	public void execute()
	{
		sc = new Scanner(System.in);

		//--------FCFS
		System.out.println("Enter Number of Processes:");
		int numProcess=sc.nextInt();
		Process []process=new Process[numProcess];

		//Accept Input
		for(int i=0;i<numProcess;i++)
		{
			System.out.println("P("+(i+1)+"):Enter Arrival time & Burst time");
			int at=sc.nextInt();
			int bt=sc.nextInt();
			//System.out.println("P("+(i+1)+"):Enter Arrival time");

			process[i]=new Process("P"+(i+1), bt, at);
		}
		Arrays.sort(process,new SortByArrival()); //sort according to arrival time

		System.out.println("Enter Quantum Time: ");
		int quantum=sc.nextInt();

		double TotalWT=0, TotalTAT=0,avgWT=0,avgTAT=0;
		int time=0;
		System.out.println("\n\nPRNo\tBT\tAT\tCT\tTAT\tWT\tPR");

	System.out.println("============================================================
============================");
		while(true) //upto all process completion
		{
			boolean done=true;
			for(int i=0;i<numProcess;i++)
			{
```

```
                        if(process[i].remBT>0 && process[i].AT<=time)
                        {
                                done=false;

                                if(process[i].remBT>quantum) // time remaining :v 2>4
                                {
                                    time=time+quantum;  //0+4=4
                                process[i].remBT=process[i].remBT-quantum; //procee[i].rembt=6-4=2
                                        System.out.println(i+" TIME "+time);

                                }
                                else //process will finish execution
                                {

                                        time+=process[i].remBT; //4+2=6
                                        System.out.println(i+" TIME "+time);

                                        process[i].remBT=0;
                                        process[i].CT=time;//6
                                        process[i].TAT=process[i].CT-process[i].AT;
                                        process[i].WT=process[i].TAT-process[i].BT;
                                        TotalWT=TotalWT+process[i].WT;
                                        TotalTAT=TotalTAT+process[i].TAT;
                                        process[i].display();
                                }
                        }
                        /*else //no process is arrived currently
                        {
                                time++;

                        }*/

                    }
                    if(done==true) //done executing all processes
                    {
                            break;
                    }

                }


        }

}
```

Scheduling

```
import java.util.Scanner;

public class Scheduling {

        public static void main(String[] args) {

        int ch;
```

```
        Scanner s=new Scanner (System.in);

        do {
                System.out.println("Enter Your Choice: ");
                ch=s.nextInt();
        switch(ch) {

        case 1:
                System.out.println("First Come First Serve Algorithm: ");
                FCFS fcfs=new FCFS();
                fcfs.execute();
                break;

        case 2:
                System.out.println("Shortest Job Algorithm: ");
                SJF sjf=new SJF();
           sjf.execute();
           break;

        case 3:
                System.out.println("Priority based Non Preemptive Algorithm: ");
                PriorityNonPreemptive pr=new PriorityNonPreemptive();
                pr.execute();
                break;

        case 4:
                System.out.println("Round Robin Algorithm: ");
                RoundRobin rr=new RoundRobin();
                rr.execute();
                break;

        default:
                System.out.println("Exit");


        }


        }while(ch!=5);

        }
}
```

SJF

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

import javax.swing.text.html.MinimalHTMLWriter;

public class SJF {
        private Scanner sc;
```

```
public void  execute()
{
        sc = new Scanner(System.in);

        //--------FCFS
        System.out.println("Enter Number of Processes:");
        int numProcess=sc.nextInt();
        Process []process=new Process[numProcess];

        //Accept Input
        for(int i=0;i<numProcess;i++)
        {
           System.out.println("P("+(i+1)+"):Enter Arrival time & Burst time");
           int at=sc.nextInt();
           int bt=sc.nextInt();
           //System.out.println("P("+(i+1)+"):Enter Arrival time");

                process[i]=new Process("P"+(i+1), bt, at);
        }

        int min=Integer.MAX_VALUE;
        int count=0,shortest=0;
        int time=0;
        int sum=0;
        double TotalWT=0, TotalTAT=0,avgWT=0,avgTAT=0;
        boolean check=false;
        System.out.println("\n\nPRNo\tBT\tAT\tCT\tTAT\tWT");

    System.out.println("================================================================
==============================");
        while(count<numProcess)
        {
           //      check=false;//remove this if given wrong i=output
                //find shortest process till time
                for(int i=0;i<numProcess;i++)
                {
                    if(process[i].AT<=time &&(process[i].remBT<min && process[i].remBT>0))
                        {
                                shortest=i; //3
                                min=process[i].remBT;
                                check=true;
                        }
                }
                        if(check==false) //No process is present currently
                        {
                                time++;
                                continue;
                        }
                        process[shortest].remBT--;
                       //1=1-1=0scheduled shortest process for one unit time
                        min=process[shortest].remBT; //0

                        if(min==0) //process completes its execution
                        {
                          min=Integer.MAX_VALUE;
                          count++;
```

```
                                    sum=time+1;
                                    process[shortest].CT=sum;
                                    process[shortest].TAT=process[shortest].CT-process[shortest].AT;
                                            process[shortest].WT=process[shortest].TAT-process[shortest].BT;
                                            //if(process[shortest].WT<0)
                                            //          process[shortest].WT=0;
                                            TotalWT=TotalWT+process[shortest].WT;
                                            TotalTAT=TotalTAT+process[shortest].TAT;

                                            process[shortest].display();
                                    }
                                    time++;


                        }

                        avgTAT=(double)TotalTAT/numProcess;
                        avgWT=(double)TotalWT/numProcess;
                        System.out.println("Average Waiting Time"+avgWT);
                        System.out.println("Average TAT="+avgTAT);
                }
}
```

Output

```
Enter Your Choice:
1
First Come First Serve Algorithm:
Enter Number of Processes:
3
P(1):Enter Arrival time & Burst time
1
24
P(2):Enter Arrival time & Burst time
1
34
P(3):Enter Arrival time & Burst time
1
45


PRNo   BT     AT     CT     TAT    WT
================================================================================
=======
P1     24     1      24     23     -1     0
P2     34     1      58     57     23     0
P3     45     1      103    102    57     0
Average Waiting Time26.333333333333332
Average TAT=60.666666666666664
Enter Your Choice:
2
Shortest Job Algorithm:
Enter Number of Processes:
2
```

```
P(1):Enter Arrival time & Burst time
1
23
P(2):Enter Arrival time & Burst time
1
34


PRNo   BT     AT     CT     TAT    WT
================================================================================
=======
P1     23     1      24     23     0      0
P2     34     1      58     57     23     0
Average Waiting Time11.5
Average TAT=40.0
Enter Your Choice:
3
Priority based Non Preemptive Algorithm:
Enter Number of Processes:
4
P(1):Enter  Burst time  & priority
1
2
P(2):Enter  Burst time  & priority
1
3
P(3):Enter  Burst time  & priority
1
4
P(4):Enter  Burst time  & priority
1
5


PRNo   BT     AT     CT     TAT    WT     PR
================================================================================
=======
P1     1      0      1      1      0      2
P2     1      0      2      2      1      3
P3     1      0      3      3      2      4
P4     1      0      4      4      3      5
Average Waiting Time1.5
Average TAT=2.5
```