import java.util.Scanner;  
class fcfs{  
public static void main(String args[]){  
int  
burst\_time[],process[],waiting\_time[],tat[],i,j,n,total=0,pos,temp;  
float wait\_avg, TAT\_avg;  
Scanner s = new Scanner(System.in);  
System.out.print("Enter number of process: ");  
n = s.nextInt();  
process = new int[n];  
burst\_time = new int[n];  
waiting\_time = new int[n];  
tat = new int[n];  
System.out.println("\nEnter Burst time:");  
for(i=0;i<n;i++)  
{  
System.out.print("\nProcess["+(i+1)+"]: ");  
burst\_time[i] = s.nextInt();;  
process[i]=i+1; //Process Number  
}  
//First process has 0 waiting time  
waiting\_time[0]=0;  
//calculate waiting time  
for(i=1;i<n;i++)  
{  
waiting\_time[i]=0;  
for(j=0;j<i;j++)  
waiting\_time[i]+=burst\_time[j];  
total+=waiting\_time[i];  
}  
//Calculating Average waiting time  
wait\_avg=(float)total/n;  
total=0;  
System.out.println("\nProcess\t Burst Time \tWaiting Time\tTurnaround  
Time");  
for(i=0;i<n;i++)  
{  
tat[i]=burst\_time[i]+waiting\_time[i];  
total+=tat[i];//Calculating TurnaroundTimetotal+=tat[i];  
System.out.println("\n  
p"+process[i]+"\t\t"+burst\_time[i]+"\t\t"+waiting\_time[i]+"\t\t  
"+tat[i]);  
}

//Calculation of Average Turnaround Time  
TAT\_avg=(float)total/n;  
System.out.println("\n\nAverage Waiting Time: "+wait\_avg);  
System.out.println("\nAverage Turnaround Time: "+TAT\_avg);  
}  
}  
/\* OUTPUT  
D:\SPOS>java fcfs  
Enter number of process: 4  
Enter Burst time:  
Process[1]: 3  
Process[2]: 5  
Process[3]: 2  
Process[4]: 10  
Process Burst Time Waiting Time Turnaround Time  
p1 3 0 3  
p2 5 3 8  
p3 2 8 10  
p4 10 10 20  
Average Waiting Time: 5.25  
Average Turnaround Time: 10.25  
\*/

1.FCFS

\*/

import java.io.\*;

import java.util.Scanner;

public class FCFS

{

public static void main(String args[])

{

int i,no\_p,burst\_time[],TT[],WT[];

float avg\_wait=0,avg\_TT=0;

burst\_time=new int[50];

TT=new int[50];

WT=new int[50];

WT[0]=0;

Scanner s=new Scanner(System.in);

System.out.println("Enter the number of process: ");

no\_p=s.nextInt();

System.out.println("\nEnter Burst Time for processes:");

for(i=0;i<no\_p;i++)

{

System.out.print("\tP"+(i+1)+": ");

burst\_time[i]=s.nextInt();

}

for(i=1;i<no\_p;i++)

{

WT[i]=WT[i-1]+burst\_time[i-1];

avg\_wait+=WT[i];

}

avg\_wait/=no\_p;

for(i=0;i<no\_p;i++)

{

TT[i]=WT[i]+burst\_time[i];

avg\_TT+=TT[i];

}

avg\_TT/=no\_p;

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\tProcesses:");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println(" Process\tBurst Time\tWaiting Time\tTurn Around Time");

for(i=0;i<no\_p;i++)

{

System.out.println("\tP"+(i+1)+"\t "+burst\_time[i]+"\t\t "+WT[i]+"\t\t "+TT[i]);

}

System.out.println("\n----------------------------------------------------------------");

System.out.println("\nAverage waiting time : "+avg\_wait);

System.out.println("\nAverage Turn Around time : "+avg\_TT+"\n");

}

}

/\*Output:

Enter the number of process:

3

Enter Burst Time for processes:

P1: 24

P2: 3

P3: 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Processes:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Process Burst Time Waiting Time Turn Around Time

P1 24 0 24

P2 3 24 27

P3 3 27 30

----------------------------------------------------------------

Average waiting time : 17.0

Average Turn Around time : 27.0 \*/

**/\*Round Robin(Preemptive)\*/**

import java.util.\*;

import java.io.\*;

class RoundR

{

public static void main(String args[])

{

int Process[]=new int[10];

int a[]=new int[10];

int Arrival\_time[]=new int[10];

int Burst\_time[]=new int[10];

int WT[]=new int[10];

int TAT[]=new int[10];

int Pno,sum=0;;

int TimeQuantum;

System.out.println("\nEnter the no. of Process::");

Scanner sc=new Scanner(System.in);

Pno=sc.nextInt();

System.out.println("\nEnter each process::");

for(int i=0;i<Pno;i++)

{

Process[i]=sc.nextInt();

}

System.out.println("\nEnter the Burst Time of each process::");

for(int i=0;i<Pno;i++)

{

Burst\_time[i]=sc.nextInt();

}

System.out.println("\nEnter the Time Quantum::");

TimeQuantum=sc.nextInt();

do{

for(int i=0;i<Pno;i++)

{

if(Burst\_time[i]>TimeQuantum)

{

Burst\_time[i]-=TimeQuantum;

for(int j=0;j<Pno;j++)

{

if((j!=i)&&(Burst\_time[j]!=0))

WT[j]+=TimeQuantum;

}

}

else

{

for(int j=0;j<Pno;j++)

{

if((j!=i)&&(Burst\_time[j]!=0))

WT[j]+=Burst\_time[i];

}

Burst\_time[i]=0;

}

}

sum=0;

for(int k=0;k<Pno;k++)

sum=sum+Burst\_time[k];

} while(sum!=0);

for(int i=0;i<Pno;i++)

TAT[i]=WT[i]+a[i];

System.out.println("process\t\tBT\tWT\tTAT");

for(int i=0;i<Pno;i++)

{

System.out.println("process"+(i+1)+"\t"+a[i]+"\t"+WT[i]+"\t"+TAT[i]);

}

float avg\_wt=0;

float avg\_tat=0;

for(int j=0;j<Pno;j++)

{

avg\_wt+=WT[j];

}

for(int j=0;j<Pno;j++)

{

avg\_tat+=TAT[j];

}

System.out.println("average waiting time "+(avg\_wt/Pno)+"\n Average turn around time"+(avg\_tat/Pno));

}

}

/\*OUTPUT::

unix@unix-HP-280-G1-

MT:~/TEA33$ java RoundR

Enter the no. of Process::

5

Enter each process::

1

2

3

4

5

Enter the Burst Time of each process::

2

1

8

4

5

Enter the Time Quantum::

2

process BT WT TAT

process1 0 0 0

process2 0 2 2

process3 0 12 12

process4 0 9 9

process5 0 13 13

average waiting time 7.2

Average turn around time7.2 \*/

**Round Robin**

import java.util.Scanner;  
public class Roundfinal1 {  
public static void main(String args[]) {  
Scanner s = new Scanner(System.in);  
int wtime[],btime[],rtime[],num,quantum,total;  
wtime = new int[10];  
btime = new int[10];  
rtime = new int[10];  
System.out.print("Enter number of processes(MAX 10): ");  
num = s.nextInt();  
System.out.print("Enter burst time");  
for(int i=0;i<num;i++) { System.out.print("\nP["+(i+1)+"]: ");  
btime[i] = s.nextInt(); rtime[i] = btime[i]; wtime[i]=0; }  
System.out.print("\n\nEnter quantum: "); quantum = s.nextInt();  
int rp = num; int i=0; int time=0; System.out.print("0");  
wtime[0]=0; while(rp!=0) { if(rtime[i]>quantum)  
{  
rtime[i]=rtime[i]-quantum;  
System.out.print(" | P["+(i+1)+"] | ");  
time+=quantum;  
System.out.print(time);  
}  
else if(rtime[i]<=quantum && rtime[i]>0)  
{time+=rtime[i];  
rtime[i]=rtime[i]-rtime[i];  
System.out.print(" | P["+(i+1)+"] | ");  
rp--;  
System.out.print(time);  
}

i++;  
if(i==num)  
{  
i=0;  
}  
}  
}  
}

**3. Priority**

import java.util.Scanner;  
public class Priority {  
public static void main(String args[]) {  
Scanner s = new Scanner(System.in);  
int x,n,p[],pp[],bt[],w[],t[],awt,atat,i;  
p = new int[10];  
pp = new int[10];  
bt = new int[10];  
w = new int[10];  
t = new int[10];  
//n is number of process  
//p is process  
//pp is process priority  
//bt is process burst time  
//w is wait time  
// t is turnaround time  
//awt is average waiting time  
//atat is average turnaround time  
System.out.print("Enter the number of process : ");  
n = s.nextInt();  
System.out.print("\n\t Enter burst time : time priorities \n");  
for(i=0;i<n;i++)  
{  
System.out.print("\nProcess["+(i+1)+"]:");  
bt[i] = s.nextInt();  
pp[i] = s.nextInt();

p[i]=i+1;  
}  
//sorting on the basis of priority  
for(i=0;i<n-1;i++)  
{  
for(int j=i+1;j<n;j++)  
{  
if(pp[i]<pp[j])  
{  
x=pp[i];  
pp[i]=pp[j];  
pp[j]=x;  
x=bt[i];  
bt[i]=bt[j];  
bt[j]=x;  
x=p[i];  
p[i]=p[j];  
p[j]=x;  
}  
}  
}  
w[0]=0;  
awt=0;  
t[0]=bt[0];  
atat=t[0];  
for(i=1;i<n;i++)  
{  
w[i]=t[i-1];  
awt+=w[i];  
t[i]=w[i]+bt[i];  
atat+=t[i];  
}

/\* 2. SJF(Non-Preemptive) \*/

import java.util.Scanner;

class SJF1{

public static void main(String args[]){

int burst\_time[],process[],waiting\_time[],tat[],i,j,n,total=0,pos,temp;

float wait\_avg,TAT\_avg;

Scanner s = new Scanner(System.in);

System.out.print("Enter number of process: ");

n = s.nextInt();

process = new int[n];

burst\_time = new int[n];

waiting\_time = new int[n];

tat = new int[n];

System.out.println("\nEnter Burst time:");

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

{

System.out.print("\nProcess["+(i+1)+"]: ");

burst\_time[i] = s.nextInt();;

process[i]=i+1; //Process Number

}

//Sorting

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

{

pos=i;

for(j=i+1;j<n;j++)

{

if(burst\_time[j]<burst\_time[pos])

pos=j;

}

temp=burst\_time[i];

burst\_time[i]=burst\_time[pos];

burst\_time[pos]=temp;

temp=process[i];

process[i]=process[pos];

process[pos]=temp;

}

//First process has 0 waiting time

waiting\_time[0]=0;

//calculate waiting time

for(i=1;i<n;i++)

{

waiting\_time[i]=0;

for(j=0;j<i;j++)

waiting\_time[i]+=burst\_time[j];

total+=waiting\_time[i];

}

//Calculating Average waiting time

wait\_avg=(float)total/n;

total=0;

System.out.println("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");

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

{

tat[i]=burst\_time[i]+waiting\_time[i]; //Calculating Turnaround Time

total+=tat[i];

System.out.println("\n p"+process[i]+"\t\t "+burst\_time[i]+"\t\t "+waiting\_time[i]+"\t\t "+tat[i]);

}

//Calculation of Average Turnaround Time

TAT\_avg=(float)total/n;

System.out.println("\n\nAverage Waiting Time: "+wait\_avg);

System.out.println("\nAverage Turnaround Time: "+TAT\_avg);

}

}

/\* 2. SJF(Preemptive)\*/

import java.util.Scanner;

class sjf\_swap1{

public static void main(String args[])

{

int burst\_time[],process[],waiting\_time[],tat[],arr\_time[],completion\_time[],i,j,n,total=0,total\_comp=0,pos,temp;

float wait\_avg,TAT\_avg;

Scanner s = new Scanner(System.in);

System.out.print("Enter number of process: ");

n = s.nextInt();

process = new int[n];

burst\_time = new int[n];

waiting\_time = new int[n];

arr\_time=new int[n];

tat = new int[n];

completion\_time=new int[n];

//burst time

System.out.println("\nEnter Burst time:");

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

{

System.out.print("\nProcess["+(i+1)+"]: ");

burst\_time[i] = s.nextInt();;

process[i]=i+1; //Process Number

}

//arrival time

System.out.println("\nEnter arrival time:");

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

{

System.out.print("\nProcess["+(i+1)+"]: ");

arr\_time[i] = s.nextInt();;

process[i]=i+1; //Process Number

}

//Sorting

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

{

pos=i;

for(j=i+1;j<n;j++)

{

if(burst\_time[j]<burst\_time[pos])

pos=j;

}

temp=burst\_time[i];

burst\_time[i]=burst\_time[pos];

burst\_time[pos]=temp;

temp=process[i];

process[i]=process[pos];

process[pos]=temp;

System.out.println("process"+process[i]);

}

//completion time new

for(i=1;i<n;i++)

{

completion\_time[i]=0;

for(j=0;j<i;j++)

completion\_time[i]+=burst\_time[j];

total\_comp+=completion\_time[i];

}

//First process has 0 waiting time

waiting\_time[0]=0;

//calculate

waiting time

for(i=1;i<n;i++)

{

waiting\_time[i]=0;

for(j=0;j<i;j++)

waiting\_time[i]+=burst\_time[j];

total+=waiting\_time[i];

}