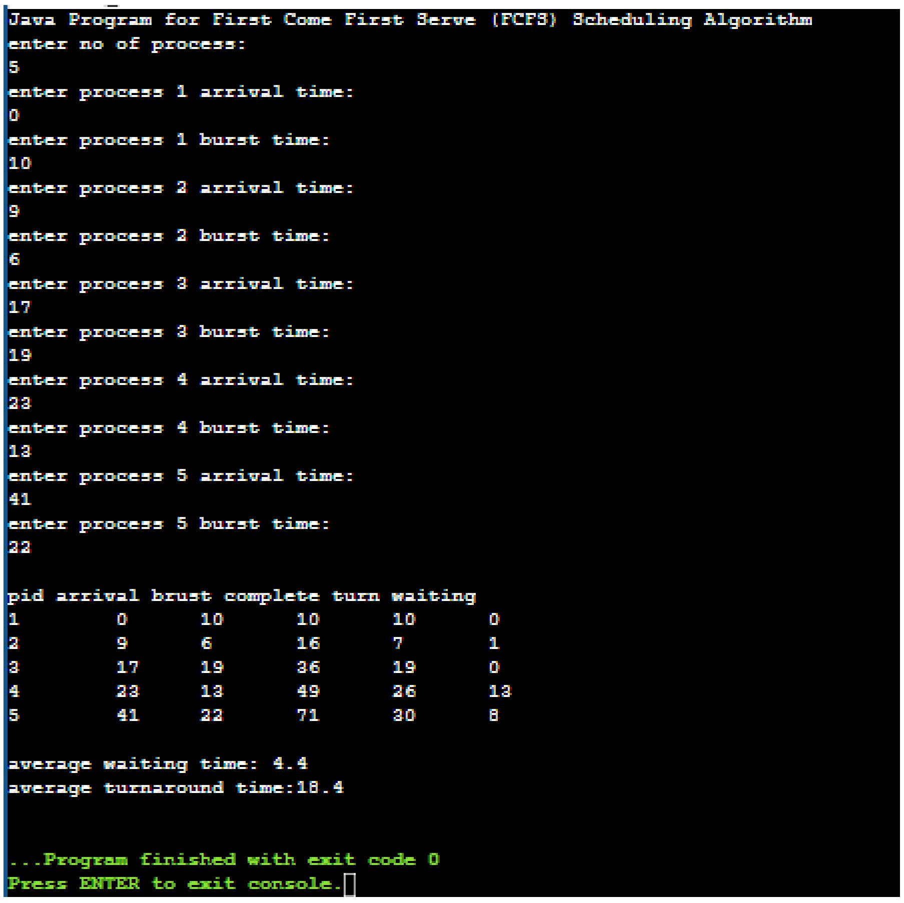
Assignment-4

Name:Majjiga jaswanth
Regno:20BCD7171

First Come First Serve (FCFS) Scheduling

```
import java.util.*;
class Main {
public static void main(String args[])
{
System.out.println("Java Program for First Come First Serve (FCFS) Scheduling
Algorithm");
Scanner sc = new Scanner(System.in);
System.out.println("enter no of process: ");
int n = sc.nextInt();
int pid[] = new int[n]; // process ids
int ar[] = new int[n]; // arrival times
int bt[] = new int[n]; // burst or execution times
int ct[] = new int[n]; // completion times
int ta[] = new int[n]; // turn around times
int wt[] = new int[n]; // waiting times
int temp;
float avgwt=0,avgta=0;
for(int i = 0; i < n; i++)
{
System.out.println("enter process " + (i+1) + " arrival time: ");
ar[i] = sc.nextInt();
System.out.println("enter process " + (i+1) + " burst time: ");
bt[i] = sc.nextInt();
pid[i] = i+1;
}
//sorting according to arrival times
for(int i = 0 ; i <n; i++)
{
for(int j=0; j < n-(i+1) ; j++)
{
if( ar[j] > ar[j+1] )
{
temp = ar[j];
ar[j] = ar[j+1];
ar[j+1] = temp;
temp = bt[j];
bt[j] = bt[j+1];
bt[j+1] = temp;
temp = pid[j];
pid[j] = pid[j+1];
pid[j+1] = temp;
}
}
}
// finding completion times
for(int i = 0 ; i < n; i++)
{
if( i == 0)
{
ct[i] = ar[i] + bt[i];
}
else
{
if( ar[i] > ct[i-1])
{
ct[i] = ar[i] + bt[i];
}
else
ct[i] = ct[i-1] + bt[i];
}
ta[i] = ct[i] - ar[i] ; // turnaround time= completion time- arrival time
wt[i] = ta[i] - bt[i] ; // waiting time= turnaround time- burst time
avgwt += wt[i] ; // total waiting time
avgta += ta[i] ; // total turnaround time
}
System.out.println("\npid arrival brust complete turn waiting");
for(int i = 0 ; i< n; i++)
{
System.out.println(pid[i] + " \t " + ar[i] + "\t" + bt[i] + "\t" + ct[i] + "\t" + ta[i] + "\t" + wt[i] )
;
}
sc.close();
System.out.println("\naverage waiting time: "+ (avgwt/n)); // printing average waiting
time.
System.out.println("average turnaround time:"+(avgta/n)); // printing average
turnaround time.
}
}
```

OUTPUT:

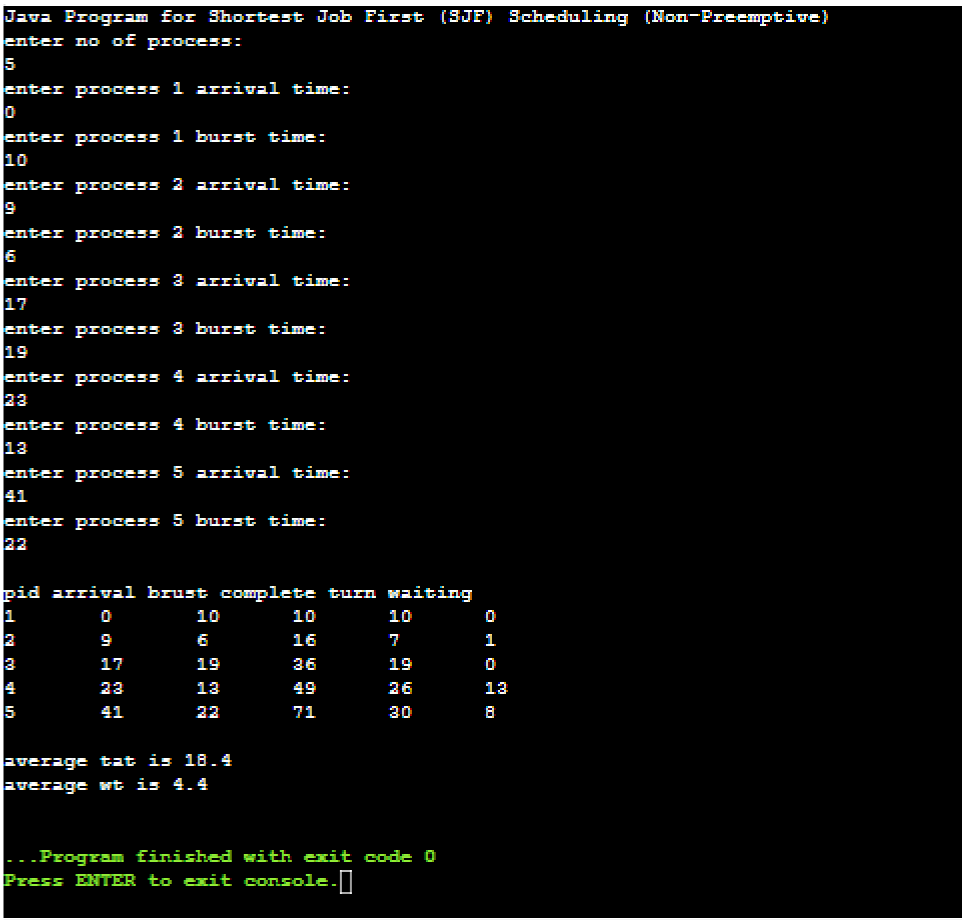


```
Java Program for First Come First Serve (FCFS) Scheduling Algorithm
enter no of process:
5
enter process 1 arrival time:
0
enter process 1 burst time:
10
enter process 2 arrival time:
9
enter process 2 burst time:
6
enter process 3 arrival time:
17
enter process 3 burst time:
19
enter process 4 arrival time:
23
enter process 4 burst time:
13
enter process 5 arrival time:
41
enter process 5 burst time:
22

pid arrival brust complete turn waiting
1        0       10       10       10       0
2        9       6        16       7        1
3        17      19       36       19       0
4        23      13       49       26       13
5        41      22       71       30       8

average waiting time: 4.4
average turnaround time:18.4


...Program finished with exit code 0
Press ENTER to exit console.
```

Shortest Job First (SJF) Scheduling (Non-Preemptive):

```
import java.util.*;
class Main {
public static void main(String args[])
{
System.out.println("Java Program for Shortest Job First (SJF) Scheduling
(Non-Preemptive)");
Scanner sc = new Scanner(System.in);
System.out.println ("enter no of process:");
int n = sc.nextInt();
int pid[] = new int[n];
int at[] = new int[n]; // at means arrival time
int bt[] = new int[n]; // bt means burst time
int ct[] = new int[n]; // ct means complete time
int ta[] = new int[n]; // ta means turn around time
int wt[] = new int[n]; //wt means waiting time
int f[] = new int[n]; // f means it is flag it checks process is completed or not
int st=0, tot=0;
float avgwt=0, avgta=0;
for(int i=0;i<n;i++)
{
System.out.println ("enter process " + (i+1) + " arrival time:");
at[i] = sc.nextInt();
System.out.println ("enter process " + (i+1) + " brust time:");
bt[i] = sc.nextInt();
pid[i] = i+1;
f[i] = 0;
}
boolean a = true;
while(true)
{
int c=n, min=999;
if (tot == n) // total no of process = completed process loop will be terminated
break;
for (int i=0; i<n; i++)
{
/*
* If i'th process arrival time <= system time and its flag=0 and burst<min
* That process will be executed first
*/
if ((at[i] <= st) && (f[i] == 0) && (bt[i]<min))
{
min=bt[i];
c=i;
}
}
/* If c==n means c value can not updated because no process arrival time< system
time so we
increase the system time */
if (c==n)
st++;
else
{
ct[c]=st+bt[c];
st+=bt[c];
ta[c]=ct[c]-at[c];
wt[c]=ta[c]-bt[c];
f[c]=1;
tot++;
}
}
System.out.println("\npid arrival brust complete turn waiting");
for(int i=0;i<n;i++)
{
avgwt+= wt[i];
avgta+= ta[i];
System.out.println(pid[i]+"\t"+at[i]+"\t"+bt[i]+"\t"+ct[i]+"\t"+ta[i]+"\t"+wt[i]);
}
System.out.println ("\naverage tat is "+ (float)(avgta/n));
System.out.println ("average wt is "+ (float)(avgwt/n));
sc.close();
}
}
```

OUTPUT:

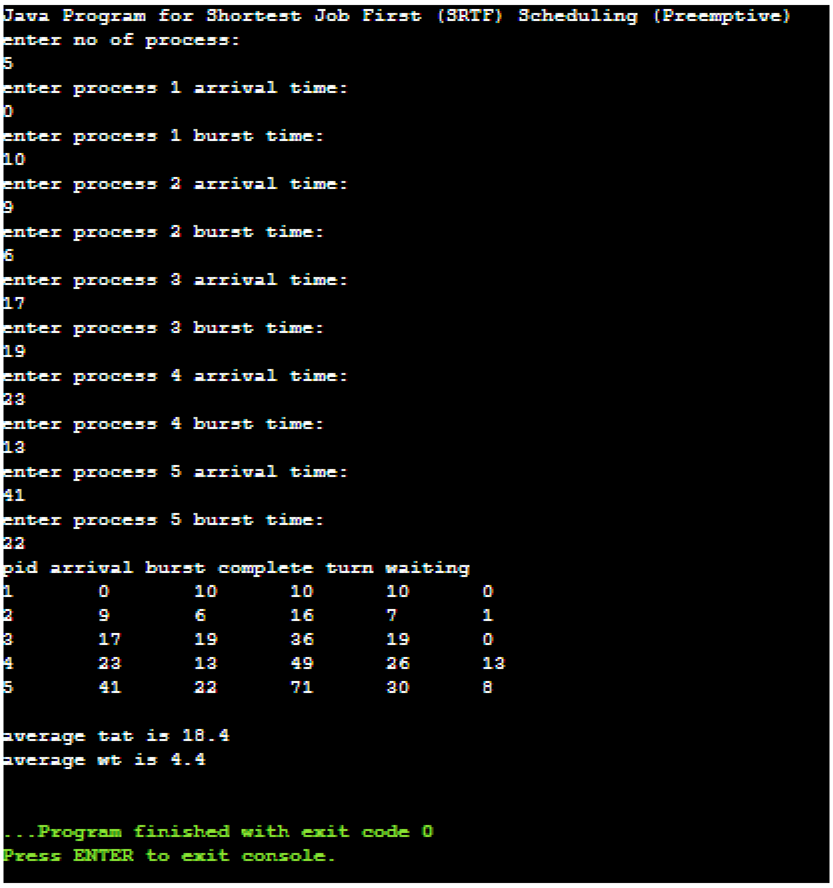


```
Java Program for Shortest Job First (SJF) Scheduling (Non-Preemptive)
enter no of process:
5
enter process 1 arrival time:
0
enter process 1 burst time:
10
enter process 2 arrival time:
9
enter process 2 burst time:
6
enter process 3 arrival time:
17
enter process 3 burst time:
19
enter process 4 arrival time:
23
enter process 4 burst time:
13
enter process 5 arrival time:
41
enter process 5 burst time:
22

pid arrival brust complete turn waiting
1       0       10      10      10      0
2       9       6       16      7       1
3       17      19      36      19      0
4       23      13      49      26      13
5       41      22      71      30      8

average tat is 18.4
average wt is 4.4


...Program finished with exit code 0
Press ENTER to exit console.
```

Shortest Job First (SRTF) Scheduling (Preemptive):

```
import java.util.*;
```

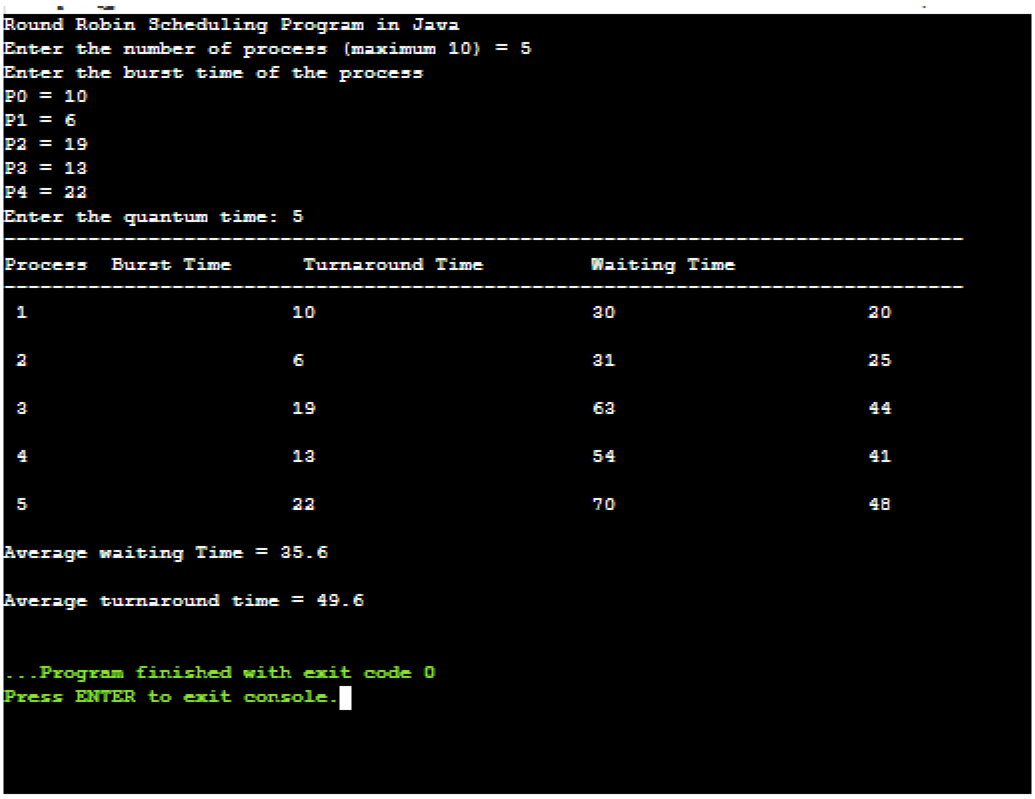
```
class Main {
public static void main (String args[])
{
System.out.println("Java Program for Shortest Job First (SRTF) Scheduling (Preemptive)");
Scanner sc=new Scanner(System.in);
System.out.println ("enter no of process:");
int n= sc.nextInt();
int pid[] = new int[n]; // it takes pid of process
int at[] = new int[n]; // at means arrival time
int bt[] = new int[n]; // bt means burst time
int ct[] = new int[n]; // ct means complete time
int ta[] = new int[n];// ta means turn around time
int wt[] = new int[n]; // wt means waiting time
int f[] = new int[n]; // f means it is flag it checks process is completed or not
int k[]= new int[n]; // it is also stores brust time
int i, st=0, tot=0;
float avgwt=0, avgta=0;
for (i=0;i<n;i++)
{
pid[i]= i+1;
System.out.println ("enter process " +(i+1)+ " arrival time:");
at[i]= sc.nextInt();
System.out.println("enter process " +(i+1)+ " burst time:");
bt[i]= sc.nextInt();
k[i]= bt[i];
f[i]= 0;
}
while(true){
int min=99,c=n;
if (tot==n)
break;
for ( i=0;i<n;i++)
{
if ((at[i]<=st) && (f[i]==0) && (bt[i]<min))
{
min=bt[i];
c=i;
}
}
if (c==n)
st++;
else
{
bt[c]--;
st++;
if (bt[c]==0)
{
ct[c]= st;
f[c]=1;
tot++;
}
}
}
for(i=0;i<n;i++)
{
ta[i] = ct[i] - at[i];
wt[i] = ta[i] - k[i];
avgwt+= wt[i];
avgta+= ta[i];
}
System.out.println("pid arrival burst complete turn waiting");
for(i=0;i<n;i++)
{
System.out.println(pid[i] +"\t"+ at[i]+"\t"+ k[i] +"\t"+ ct[i] +"\t"+ ta[i] +"\t"+ wt[i]);
}
System.out.println("\naverage tat is "+ (float)(avgta/n));
System.out.println("average wt is "+ (float)(avgwt/n));
sc.close();
}
}
```
OUTPUT:



```
Java Program for Shortest Job First (SRTF) Scheduling (Preemptive)
enter no of process:
5
enter process 1 arrival time:
0
enter process 1 burst time:
10
enter process 2 arrival time:
9
enter process 2 burst time:
6
enter process 3 arrival time:
17
enter process 3 burst time:
19
enter process 4 arrival time:
23
enter process 4 burst time:
13
enter process 5 arrival time:
41
enter process 5 burst time:
22
pid arrival burst complete turn waiting
1       0       10      10      10      0
2       9       6       16      7       1
3       17      19      36      19      0
4       23      13      49      26      13
5       41      22      71      30      8

average tat is 18.4
average wt is 4.4


...Program finished with exit code 0
Press ENTER to exit console.
```

Round Robin Scheduling Java Program:

```
import java.util.Scanner;
class Main
{
public static void main(String args[])
{
System.out.println("Round Robin Scheduling Program in Java");
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\t\t"+bt[i]+"\t\t\t "+tat[i]+"\t\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 Scheduling Program in Java
Enter the number of process (maximum 10) = 5
Enter the burst time of the process
P0 = 10
P1 = 6
P2 = 19
P3 = 13
P4 = 22
Enter the quantum time: 5
--------------------------------------------------------------------------------
Process  Burst Time      Turnaround Time         Waiting Time
--------------------------------------------------------------------------------
 1                      10                        30                      20

 2                      6                         31                      25

 3                      19                        63                      44

 4                      13                        54                      41

 5                      22                        70                      48

Average waiting Time = 35.6

Average turnaround time = 49.6


...Program finished with exit code 0
Press ENTER to exit console.
```

Lets differentiate turn around time (TAT) for FCFS, SJF, and Round Robin:
Average TAT(First Come First Serve) = 18.4
Average TAT(Shortest Job First(Non Preemptive)) = 18.4
Average TAT(Round Robin Scheduling) = 49.6

Average Tat Fcfs	Average Tat(Sjf)(Non preemptive)	Average Tat (Round Robin Scheduling)
18.4	18.4	49.6

Average waiting time(First Come First Serve) = 4.4
Average waiting time(Shortest Job First(Non Preemptive)) = 4.4
Average waiting time(Round Robin Scheduling) = 35.6
 Therefore the average turnaround time for FCFS and SJF is smaller compared to TAT of Round Robin Scheduling Algorithms. So, according to average turnaround time FCFS and SJF are the best algorithm

Average Waiting Time for FCFS	Average waiting time for SJF(Non preemptive)	Average waiting time for Round robin scheduling
4.4	4.4	35.6

Therefore the average waiting time for FCFS and SJF is smaller compared to TAT of the Round Robin Scheduling Algorithm. So, according to average waiting time FCFS and SJF are the best algorithms