

Naming the file: 20BCE1798_EX-4_CPU Scheduling Algorithms.

Reg No: **20BCE1798**

Name: **ANSH GOEL**

Course Code: CSE2005

Course Name: Operating Systems (Embedded Lab)

Slot: L27+L28

Ex No 4: CPU Scheduling Algorithms

Date: 04-02-2022

1. FCFS (Non -Pre-emptive)

CODE:

```
#include<stdio.h>
int main()
{
int n,bt[20],wt[20],tat[20],avwt=0,avtat=0,i,j;
printf("Enter total number of processes(maximum 20):");
scanf("%d",&n);
printf("\nEnter Process Burst Timen");
for(i=0;i<n;i++)
{
printf("\nP[%d]:",i+1);
scanf("%d",&bt[i]);
}
wt[0]=0;
for(i=1;i<n;i++)
{
wt[i]=0;
for(j=0;j<i;j++)
wt[i]+=bt[j];
}
printf("\nProcessttBurst TimetWaiting TimetTurnaround Time");
for(i=0;i<n;i++)
{
tat[i]=bt[i]+wt[i];
avwt+=wt[i];
avtat+=tat[i];
printf("\nP[%d] tt%d tt%d tt%d",i+1,bt[i],wt[i],tat[i]);
}
avwt/=i;
avtat/=i;
printf("\nAverage Waiting Time:%d",avwt);
printf("\nAverage Turnaround Time:%d",avtat);
return 0;
}
```

```

1  #include<stdio.h>
2  int main()
3  {
4  int n,bt[20],wt[20],tat[20],avwt=0,avtat=0,i,j;
5  printf("Enter total number of processes(maximum 20):");
6  scanf("%d",&n);
7  printf("\nEnter Process Burst Timen");
8  for(i=0;i<n;i++)
9  {
10 printf("\nP[%d]:",i+1);
11 scanf("%d",&bt[i]);
12 }
13 wt[0]=0;
14 for(i=1;i<n;i++)
15 {
16 wt[i]=0;
17 for(j=0;j<i;j++)
18 wt[i]+=bt[j];
19 }
20 printf("\nProcessttBurst TimetWaiting TimetTurnaround Time");
21 for(i=0;i<n;i++)
22 {
23 tat[i]=bt[i]+wt[i];
24 avwt+=wt[i];
25 avtat+=tat[i];
26 printf("\nP[%d] tt%d tt%d tt%d",i+1,bt[i],wt[i],tat[i]);
27 }
28 avwt/=i;
29 avtat/=i;
30 printf("\nAverage Waiting Time:%d",avwt);
31 printf("\nAverage Turnaround Time:%d",avtat);
32 return 0;
33 }

```

OUTPUT:

```

Enter total number of processes(maximum 20):4
Enter Process Burst Timen
P[1]:2
P[2]:5
P[3]:1
P[4]:7
ProcessttBurst TimetWaiting TimetTurnaround Time
P[1] tt2 tt0 tt2
P[2] tt5 tt2 tt7
P[3] tt1 tt7 tt8
P[4] tt7 tt8 tt15
Average Waiting Time:4
Average Turnaround Time:8
...Program finished with exit code 0
Press ENTER to exit console.

```

2. SJF (Shortest Job First) falls under non-pre-emptive category

CODE:

```
#include<stdio.h>
int main()
{
int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;
float avg_wt,avg_tat;
printf("Enter number of process:");
scanf("%d",&n);
printf("\nProcessId : Enter Burst Time");
for(i=0;i<n;i++)
{
printf("\nP%d : ",i+1);
scanf("%d",&bt[i]);
p[i]=i+1;
}
//sorting of burst times
for(i=0;i<n;i++)
{
pos=i;
for(j=i+1;j<n;j++)
{
if(bt[j]<bt[pos])
pos=j;
}
temp=bt[i];
bt[i]=bt[pos];
bt[pos]=temp;
temp=p[i];
p[i]=p[pos];
p[pos]=temp;
}
wt[0]=0;
for(i=1;i<n;i++)
{
wt[i]=0;
for(j=0;j<i;j++)
wt[i]+=bt[j];
total+=wt[i];
}
avg_wt=(float)total/n;
total=0;
printf("\nProcess Burst Time tWaiting TimeTurnaround Time");
for(i=0;i<n;i++)
{
tat[i]=bt[i]+wt[i];
total+=tat[i];
printf("\np%d tt%d tt%d tt%d",p[i],bt[i],wt[i],tat[i]);
}
avg_tat=(float)total/n;
printf("\nAverage Waiting Time=%f",avg_wt);
printf("\nAverage Turnaround Time=%f",avg_tat);
}
```

```

1  #include<stdio.h>
2  int main()
3  {
4  int bt[20],p[20],wt[20],tat[20],i,j,n,total=0,pos,temp;
5  float avg_wt,avg_tat;
6  printf("Enter number of process:");
7  scanf("%d",&n);
8  printf("\nProcessId : Enter Burst Time");
9  for(i=0;i<n;i++)
10 {
11 printf("\nP%d : ",i+1);
12 scanf("%d",&bt[i]);
13 p[i]=i+1;
14 }
15 //sorting of burst times
16 for(i=0;i<n;i++)
17 {
18 pos=i;
19 for(j=i+1;j<n;j++)
20 {
21 if(bt[j]<bt[pos])
22 pos=j;
23 }
24 temp=bt[i];
25 bt[i]=bt[pos];
26 bt[pos]=temp;
27 temp=p[i];
28 p[i]=p[pos];
29 p[pos]=temp;
30 }
31 wt[0]=0;
32 for(i=1;i<n;i++)
33 {
34 wt[i]=0;
35 for(j=0;j<i;j++)
36 wt[i]+=bt[j];
37 total+=wt[i];
38 }
39 avg_wt=(float)total/n;
40 total=0;
41 printf("\nProcesst Burst Time tWaiting TimetTurnaround Time");
42 for(i=0;i<n;i++)
43 {
44 tat[i]=bt[i]+wt[i];
45 total+=tat[i];
46 printf("\np%d tt%d tt1 tt2 tt3",p[i],bt[i],wt[i],tat[i]);
47 }
48 avg_tat=(float)total/n;
49 printf("\nAverage Waiting Time=%f",avg_wt);
50 printf("\nAverage Turnaround Time=%f",avg_tat);
51 }

```

OUTPUT:

```

Enter number of process:4

ProcessId : Enter Burst Time
P1 : 2

P2 : 1

P3 : 5

P4 : 7

Processt Burst Time tWaiting TimetTurnaround Time
p2 tt1 tt0 tt1
p1 tt2 tt1 tt3
p3 tt5 tt3 tt8
p4 tt7 tt8 tt15
Average Waiting Time=3.000000
Average Turnaround Time=6.750000

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

```

3. **SRTF (Shortest Remaining Time First)**, another name of SJF & falls under Pre-emptive Category.

CODE:

```
#include <stdio.h>
int main()
{
    int arrival_time[10], burst_time[10], temp[10];
    int i, smallest, count = 0, time, limit;
    double wait_time = 0, turnaround_time = 0, end;
    float average_waiting_time, average_turnaround_time;
    printf("Enter the Total Number of Processes:");
    scanf("%d", &limit);
    printf("\nEnter Details of %d Processes", limit);
    for(i = 0; i < limit; i++)
    {
        printf("\nEnter Arrival Time:");
        scanf("%d", &arrival_time[i]);
        printf("Enter Burst Time:");
        scanf("%d", &burst_time[i]);
        temp[i] = burst_time[i];
    }
    burst_time[9] = 9999;
    for(time = 0; count != limit; time++)
    {
        smallest = 9;
        for(i = 0; i < limit; i++)
        {
            if(arrival_time[i] <= time && burst_time[i] < burst_time[smallest] && burst_time[i] > 0)
            {
                smallest = i;
            }
        }
        burst_time[smallest]--;
        if(burst_time[smallest] == 0)
        {
            count++;
            end = time + 1;
            wait_time = wait_time + end - arrival_time[smallest] - temp[smallest];
            turnaround_time = turnaround_time + end - arrival_time[smallest];
        }
    }
    average_waiting_time = wait_time / limit;
    average_turnaround_time = turnaround_time / limit;
    printf("\nAverage Waiting Time: %lf", average_waiting_time);
    printf("\nAverage Turnaround Time: %lf", average_turnaround_time);
    return 0;
}
```

```

1  #include <stdio.h>
2  int main()
3  {
4  int arrival_time[10], burst_time[10], temp[10];
5  int i, smallest, count = 0, time, limit;
6  double wait_time = 0, turnaround_time = 0, end;
7  float average_waiting_time, average_turnaround_time;
8  printf("Enter the Total Number of Processes:");
9  scanf("%d", &limit);
10 printf("\nEnter Details of %d Processes", limit);
11 for(i = 0; i < limit; i++)
12 {
13 printf("\nEnter Arrival Time:");
14 scanf("%d", &arrival_time[i]);
15 printf("Enter Burst Time:");
16 scanf("%d", &burst_time[i]);
17 temp[i] = burst_time[i];
18 }
19 burst_time[9] = 9999;
20 for(time = 0; count != limit; time++)
21 {
22 smallest = 9;
23 for(i = 0; i < limit; i++)
24 {
25 if(arrival_time[i] <= time && burst_time[i] < burst_time[smallest] && burst_time[i] > 0)
26 {
27 smallest = i;
28 }
29 }
30 burst_time[smallest]--;
31 if(burst_time[smallest] == 0)
32 {
33 count++;
34 end = time + 1;
35 wait_time = wait_time + end - arrival_time[smallest] - temp[smallest];
36 turnaround_time = turnaround_time + end - arrival_time[smallest];
37 }
38 }
39 average_waiting_time = wait_time / limit;
40 average_turnaround_time = turnaround_time / limit;
41 printf("\nAverage Waiting Time: %lf", average_waiting_time);
42 printf("\nAverage Turnaround Time: %lf", average_turnaround_time);
43 return 0;
44 }

```

OUTPUT:

```

Enter the Total Number of Processes:4

Enter Details of 4 Processes
Enter Arrival Time:5
Enter Burst Time:2

Enter Arrival Time:9
Enter Burst Time:5

Enter Arrival Time:0
Enter Burst Time:1

Enter Arrival Time:1
Enter Burst Time:7

Average Waiting Time: 0.750000
Average Turnaround Time: 4.500000

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

```