

# EXPERIMENT - 2

## CPU SCHEDULING

- i. To write a C program to implement the CPU scheduling algorithm for **FIRST COME FIRST SERVE**.

```
1  #include<stdio.h>
2  struct process
3  {
4      int id,WT,AT,BT,TAT;
5  };
6  struct process a[10];
7  void swap(int *b,int *c)
8  {
9      int tem;
10     tem=*c;
11     *c=*b;
12     *b=tem;
13 }
14 int main()
15 {
16     int n,check_ar=0;
17     int Cmp_time=0;
18     float Total_WT=0,Total_TAT=0,Avg_WT,Avg_TAT;
19     printf("Enter the number of process \n");
20     scanf("%d",&n);
21     printf("Enter the Arrival time and Burst time of the process\n");
22     printf("AT BT\n");
23     for(int i=0;i<n;i++)
24     {
25         scanf("%d%d",&a[i].AT,&a[i].BT);
26         a[i].id=i+1;
27         if(i==0)
28             check_ar=a[i].AT;
29
30         if(check_ar!=a[i].AT )
31             check_ar=1;
32     }
33
34     if(check_ar!=0)
35     {
36         for(int i=0;i<n;i++)
37         {
38             for(int j=0;j<n-i-1;j++)
39             {
40                 if(a[j].AT>a[j+1].AT)
41                 {
42                     swap(&a[j].id,&a[j+1].id);
43                     swap(&a[j].AT,&a[j+1].AT);
44                     swap(&a[j].BT,&a[j+1].BT);
45                 }
```

```

42         swap(&a[j].id,&a[j+1].id);
43         swap(&a[j].AT,&a[j+1].AT);
44         swap(&a[j].BT,&a[j+1].BT);
45     }
46 }
47 }
48 }
49 if(check_ar!=0)
50 {
51     a[0].WT=a[0].AT;
52     a[0].TAT=a[0].BT-a[0].AT;
53     Cmp_time=a[0].TAT;
54     Total_WT=Total_WT+a[0].WT;
55     Total_TAT=Total_TAT+a[0].TAT;
56     for(int i=1;i<n;i++)
57     {
58         int min=a[i].BT;
59         for(int j=i+1;j<n;j++)
60         {
61             if(min>a[j].BT && a[j].AT<=Cmp_time)
62             {
63                 min=a[j].BT;
64                 swap(&a[i].id,&a[j].id);
65                 swap(&a[i].AT,&a[j].AT);
66                 swap(&a[i].BT,&a[j].BT);
67             }
68         }
69         a[i].WT=Cmp_time-a[i].AT;
70         Total_WT=Total_WT+a[i].WT;
71         Cmp_time=Cmp_time+a[i].BT;
72         a[i].TAT=Cmp_time-a[i].AT;
73         Total_TAT=Total_TAT+a[i].TAT;
74     }
75 }
76 else
77 {
78     for(int i=0;i<n;i++)
79     {
80         int min=a[i].BT;
81         for(int j=i+1;j<n;j++)
82         {
83             if(min>a[j].BT && a[j].AT<=Cmp_time)
84             {
85                 min=a[j].BT;
86                 swap(&a[i].id,&a[j].id);

```

```

80     int min=a[i].BT;
81     for(int j=i+1;j<n;j++)
82     {
83         if(min>a[j].BT && a[j].AT<=Cmp_time)
84         {
85             min=a[j].BT;
86             swap(&a[i].id,&a[j].id);
87             swap(&a[i].AT,&a[j].AT);
88             swap(&a[i].BT,&a[j].BT);
89         }
90     }
91     a[i].WT=Cmp_time-a[i].AT;
92     Cmp_time=Cmp_time+a[i].BT;
93     a[i].TAT=Cmp_time-a[i].AT;
94     Total_WT=Total_WT+a[i].WT;
95     Total_TAT=Total_TAT+a[i].TAT;
96 }
97 }
98 Avg_WT=Total_WT/n;
99 Avg_TAT=Total_TAT/n;
100 printf("The process are\n");
101 printf("ID WT TAT\n");
102 for(int i=0;i<n;i++)
103 {
104     printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
105 }
106 printf("Avg waiting time is:- %f\n",Avg_WT);
107 printf("Avg turn around time is:- %f",Avg_TAT);
108 return 0;
109 }
110

```

## OUTPUT

```

Enter number of the process
5
Enter Arrival time and Burst time of the process
AT    BT
2      2
2      0
1      2
3      3
3      5
4      4
4      4
Process ,Waiting_time ,TurnA_time
1          2          0
2          2          3
3          1          4
4          3          8
5          7          11
Average waiting time is : 3.000000
Average turn around time is : 5.200000

-----
Process exited after 90.05 seconds with return value 0
Press any key to continue . . .

```

- ii. To write a C program to implement the CPU scheduling algorithm for **Shortest Job First**

```
1  #include<stdio.h>
2  struct process
3  {
4      int id,WT,AT,BT,TAT;
5  };
6  struct process a[10];
7  void swap(int *b,int *c)
8  {
9      int tem;
10     tem=*c;
11     *c=*b;
12     *b=tem;
13 }
14 int main()
15 {
16     int n,check_ar=0;
17     int Cmp_time=0;
18     float Total_WT=0,Total_TAT=0,Avg_WT,Avg_TAT;
19     printf("Enter the number of process \n");
20     scanf("%d",&n);
21     printf("Enter the Arrival time and Burst time of the process\n");
22     printf("AT BT\n");
23     for(int i=0;i<n;i++)
24     {
25         scanf("%d%d",&a[i].AT,&a[i].BT);
26         a[i].id=i+1;
27         if(i==0)
28             check_ar=a[i].AT;
29
30         if(check_ar!=a[i].AT )
31             check_ar=1;
32     }
33
34     if(check_ar!=0)
35     {
36         for(int i=0;i<n;i++)
37         {
38             for(int j=0;j<n-i-1;j++)
39             {
40                 if(a[j].AT>a[j+1].AT)
41                 {
42                     swap(&a[j].id,&a[j+1].id);
43                     swap(&a[j].AT,&a[j+1].AT);
44                     swap(&a[j].BT,&a[j+1].BT);
45                 }
46             }
47         }
48     }
49 }
```

```

45     }
46 }
47 }
48 }
49 if(check_ar!=0)
50 {
51     a[0].WT=a[0].AT;
52     a[0].TAT=a[0].BT-a[0].AT;
53     Cmp_time=a[0].TAT;
54     Total_WT=Total_WT+a[0].WT;
55     Total_TAT=Total_TAT+a[0].TAT;
56     for(int i=1;i<n;i++)
57     {
58         int min=a[i].BT;
59         for(int j=i+1;j<n;j++)
60         {
61             if(min>a[j].BT && a[j].AT<=Cmp_time)
62             {
63                 min=a[j].BT;
64                 swap(&a[i].id,&a[j].id);
65                 swap(&a[i].AT,&a[j].AT);
66                 swap(&a[i].BT,&a[j].BT);
67             }
68         }
69         a[i].WT=Cmp_time-a[i].AT;
70         Total_WT=Total_WT+a[i].WT;
71         Cmp_time=Cmp_time+a[i].BT;
72         a[i].TAT=Cmp_time-a[i].AT;
73         Total_TAT=Total_TAT+a[i].TAT;
74     }
75 }
76 else
77 {
78     for(int i=0;i<n;i++)
79     {
80         int min=a[i].BT;
81         for(int j=i+1;j<n;j++)
82         {
83             if(min>a[j].BT && a[j].AT<=Cmp_time)
84             {
85                 min=a[j].BT;
86                 swap(&a[i].id,&a[j].id);
87                 swap(&a[i].AT,&a[j].AT);
88                 swap(&a[i].BT,&a[j].BT);

```

```

84         {
85             min=a[j].BT;
86             swap(&a[i].id,&a[j].id);
87             swap(&a[i].AT,&a[j].AT);
88             swap(&a[i].BT,&a[j].BT);
89         }
90     }
91     a[i].WT=Cmp_time-a[i].AT;
92     Cmp_time=Cmp_time+a[i].BT;
93     a[i].TAT=Cmp_time-a[i].AT;
94     Total_WT=Total_WT+a[i].WT;
95     Total_TAT=Total_TAT+a[i].TAT;
96 }
97 }
98 Avg_WT=Total_WT/n;
99 Avg_TAT=Total_TAT/n;
100 printf("The process are\n");
101 printf("ID WT TAT\n");
102 for(int i=0;i<n;i++)
103 {
104     printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
105 }
106 printf("Avg waiting time is:- %f\n",Avg_WT);
107 printf("Avg turn around time is:- %f",Avg_TAT);
108 return 0;
109 }

```

## OUTPUT

```

Enter the number of process
5
Enter the Arrival time and Burst time of the process
AT BT
2
1
1
5
4
1
0
6
2
3
The process are
ID WT TAT
4      0      6
1      4      5
3      3      4
5      6      9
2     10     15
Avg waiting time is:- 4.600000
Avg turn around time is:- 7.800000
-----
Process exited after 29.99 seconds with return value 0
Press any key to continue . . .

```

- i) To write a C program to implement the CPU scheduling algorithm for Round Robin.

```
1  #include<stdio.h>
2  struct process
3  {
4      int id,AT,BT,WT,TAT;
5  };
6
7  struct process a[10];
8  int queue[100];
9  int front=-1;
10 int rear=-1;
11 void insert(int n)
12 {
13     if(front==-1)
14         front=0;
15     rear=rear+1;
16     queue[rear]=n;
17 }
18 int Delete()
19 {
20     int n;
21     n=queue[front];
22     front=front+1;
23     return n;
24 }
25 int main()
26 {
27     int n,TQ,p,TIME=0;
28     int temp[10],exist[10]={0};
29     float total_wt=0,total_tat=0,Avg_WT,Avg_TAT;
30     printf("Enter the number of the process\n");
31     scanf("%d",&n);
32     printf("Enter the arrival time and burst time of the process\n");
33     printf("AT BT\n");
34     for(int i=0;i<n;i++)
35     {
36         scanf("%d%d",&a[i].AT,&a[i].BT);
37         a[i].id=i;
38         temp[i]=a[i].BT;
39     }
40     printf("Enter the time quantum\n");
41     scanf("%d",&TQ);
42     insert(0);
43     exist[0]=1;
44
```

```

45 while(front<=rear)
46 {
47     p=Delete();
48     if(a[p].BT>=TQ)
49     {
50         a[p].BT=a[p].BT-TQ;
51         TIME=TIME+TQ;
52     }
53     else
54     {
55         TIME=TIME+a[p].BT;
56         a[p].BT=0;
57     }
58     for(int i=0;i<n;i++)
59     {
60         if(exist[i]==0 && a[i].AT<=TIME)
61         {
62             insert(i);
63             exist[i]=1;
64         }
65     }
66     if(a[p].BT==0)
67     {
68         a[p].TAT=TIME-a[p].AT;
69         a[p].WT=a[p].TAT-temp[p];
70         total_tat=total_tat+a[p].TAT;
71         total_wt=total_wt+a[p].WT;
72     }
73     else
74     {
75         insert(p);
76     }
77 }
78 Avg_TAT=total_tat/n;
79 Avg_WT=total_wt/n;
80 printf("ID WT TAT\n");
81 for(int i=0;i<n;i++)
82 {
83     printf("%d %d %d\n",a[i].id,a[i].WT,a[i].TAT);
84 }
85 printf("Average waiting time of the processes is : %f\n",Avg_WT);
86 printf("Average turn around time of the processes is : %f\n",Avg_TAT);
87 return 0;
88 }

```



# OUTPUT

```
Enter the number of the process
5
Enter the arrival time and burst time of the process
AT BT
3
0
4
1
4
3
5
4
2
5
Enter the time quantum
2
ID WT TAT
0 -3 -3
1 0 0
2 0 0
3 0 0
4 0 0
Average waiting time of the processes is : -0.600000
Average turn around time of the processes is : -0.600000

-----
Process exited after 27.38 seconds with return value 0
Press any key to continue . . .
```



```

42         swap(&a[j].id,&a[j+1].id);
43         swap(&a[j].AT,&a[j+1].AT);
44         swap(&a[j].BT,&a[j+1].BT);
45         swap(&a[j].PR,&a[j+1].PR);
46     }
47 }
48 }
49 }
50 if(check_ar!=0)
51 {
52     a[0].WT=a[0].AT;
53     a[0].TAT=a[0].BT-a[0].AT;
54
55     Cmp_time=a[0].TAT;
56     Total_WT=Total_WT+a[0].WT;
57     Total_TAT=Total_TAT+a[0].TAT;
58     for(int i=1;i<n;i++)
59     {
60         int min=a[i].PR;
61         for(int j=i+1;j<n;j++)
62         {
63             if(min>a[j].PR && a[j].AT<=Cmp_time)
64             {
65                 min=a[j].PR;
66                 swap(&a[i].id,&a[j].id);
67                 swap(&a[i].AT,&a[j].AT);
68                 swap(&a[i].BT,&a[j].BT);
69                 swap(&a[i].PR,&a[j].PR);
70             }
71         }
72         a[i].WT=Cmp_time-a[i].AT;
73         Total_WT=Total_WT+a[i].WT;
74         Cmp_time=Cmp_time+a[i].BT;
75         a[i].TAT=Cmp_time-a[i].AT;
76         Total_TAT=Total_TAT+a[i].TAT;
77     }
78 }
79 else
80 {
81     for(int i=0;i<n;i++)
82     {
83         int min=a[i].PR;
84         for(int j=i+1;j<n;j++)
85         {

```

```

74         Cmp_time=Cmp_time+a[i].BT;
75         a[i].TAT=Cmp_time-a[i].AT;
76         Total_TAT=Total_TAT+a[i].TAT;
77     }
78 }
79 else
80 {
81     for(int i=0;i<n;i++)
82     {
83         int min=a[i].PR;
84         for(int j=i+1;j<n;j++)
85         {
86             if(min>a[j].PR && a[j].AT<=Cmp_time)
87             {
88                 min=a[j].PR;
89                 swap(&a[i].id,&a[j].id);
90                 swap(&a[i].AT,&a[j].AT);
91                 swap(&a[i].BT,&a[j].BT);
92                 swap(&a[i].PR,&a[j].PR);
93             }
94         }
95         a[i].WT=Cmp_time-a[i].AT;
96         Cmp_time=Cmp_time+a[i].BT;
97         a[i].TAT=Cmp_time-a[i].AT;
98         Total_WT=Total_WT+a[i].WT;
99         Total_TAT=Total_TAT+a[i].TAT;
100     }
101 }
102 Avg_WT=Total_WT/n;
103 Avg_TAT=Total_TAT/n;
104 printf("The process are\n");
105 printf("ID WT TAT\n");
106 for(int i=0;i<n;i++)
107 {
108     printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
109 }
110 printf("Avg waiting time is: %f\n",Avg_WT);
111 printf("Avg turn around time is: %f",Avg_TAT);
112 return 0;
113 }

```

# OUTPUT

```
Enter the number of process
5
Enter the Arrival time , Burst time and priority of the process
AT BT PR
3
3
8
0
4
2
1
4
4
3
5
1
4
2
6
5
5
The process are
ID WT TAT
5      2      4
1      1      9
2      8     10
3     10     14
4     13     14
Avg waiting time is: 6.800000
Avg turn around time is: 10.200000
-----
Process exited after 38.01 seconds with return value 0
```