

SIDDHI SINGH

17BIT0028

OPERATING SYSTEMS ASSESSMENT-4

NON-PREEMPTIVE

FCFS

```
#include<stdio.h>

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 Time\n");

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

{

    printf("P[%d]:",i+1);

    scanf("%d",&bt[i]);

}

wt[0]=0; //waiting time for first process is 0

//calculating waiting time

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

{

    wt[i]=0;

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

        wt[i]+=bt[j];

}

printf("\nProcess\t\tBurst Time\tWaiting Time\tTurnaround Time");

//calculating turnaround time

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

{

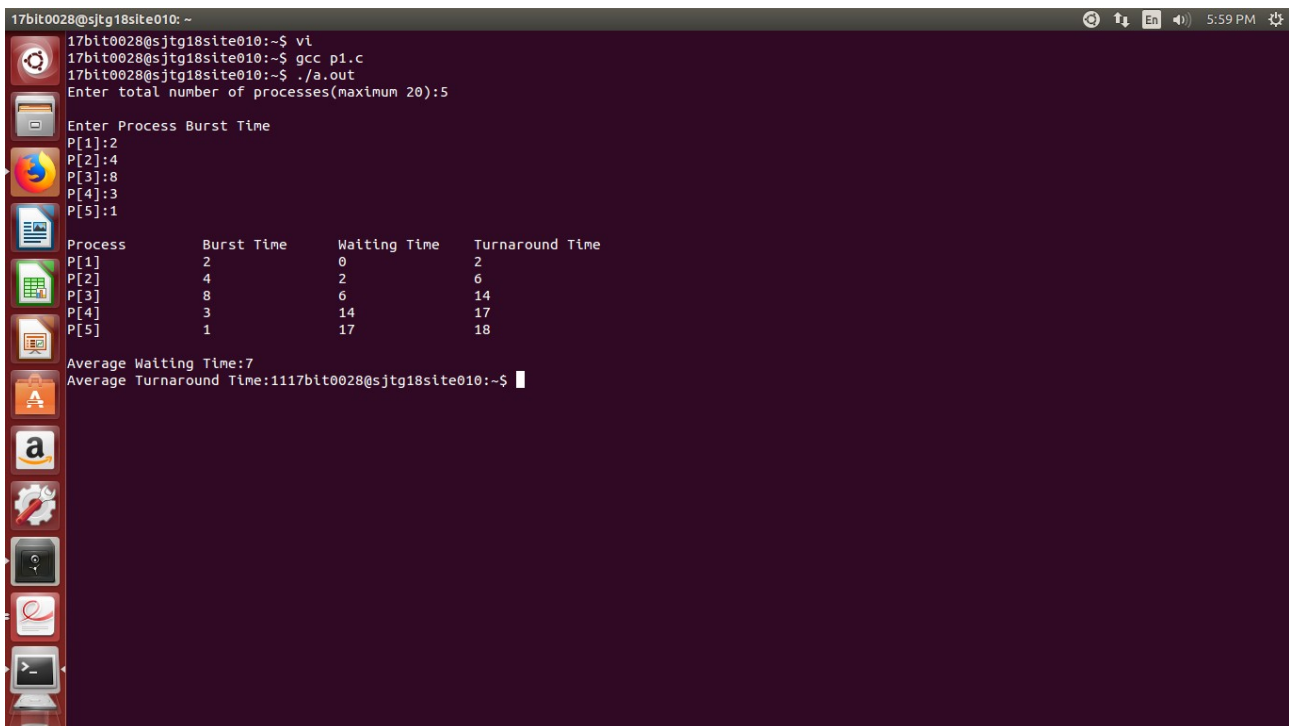
    tat[i]=bt[i]+wt[i];

    avwt+=wt[i];

    avtat+=tat[i];

    printf("\nP[%d]\t\t%d\t\t%d\t\t%d",i+1,bt[i],wt[i],tat[i]);
```

```
}  
  
avwt/=i;  
  
avtat/=i;  
  
printf("\n\nAverage Waiting Time:%d",avwt);  
printf("\n\nAverage Turnaround Time:%d",avtat);  
  
return 0;  
  
}
```



```
17bit0028@sjtg18site010: ~  
17bit0028@sjtg18site010:~$ vi  
17bit0028@sjtg18site010:~$ gcc p1.c  
17bit0028@sjtg18site010:~$ ./a.out  
Enter total number of processes(maximum 20):5  
Enter Process Burst Time  
P[1]:2  
P[2]:4  
P[3]:8  
P[4]:3  
P[5]:1  


| Process | Burst Time | Waiting Time | Turnaround Time |
|---------|------------|--------------|-----------------|
| P[1]    | 2          | 0            | 2               |
| P[2]    | 4          | 2            | 6               |
| P[3]    | 8          | 6            | 14              |
| P[4]    | 3          | 14           | 17              |
| P[5]    | 1          | 17           | 18              |

  
Average Waiting Time:7  
Average Turnaround Time:1117bit0028@sjtg18site010:~$
```

SJF

```
#include<stdio.h>

void 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("\nEnter Burst Time:\n");
    for(i=0;i<n;i++)
    {
        printf("p%d:",i+1);
        scanf("%d",&bt[i]);
        p[i]=i+1;        //contains process number
    }

    //sorting burst time in ascending order using selection sort
    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;        //waiting time for first process will be zero

    //calculate waiting time
    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;    //average waiting time
```

```

total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");
for(i=0;i<n;i++)
{
    tat[i]=bt[i]+wt[i];    //calculate turnaround time
    total+=tat[i];
    printf("\np%d\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);
}

avg_tat=(float)total/n;    //average turnaround time
printf("\n\nAverage Waiting Time=%f",avg_wt);
printf("\nAverage Turnaround Time=%f\n",avg_tat);
}

```

26,1 Top

```

17bit0028@sjtg18site010: ~
17bit0028@sjtg18site010:~$ VI
VI: command not found
17bit0028@sjtg18site010:~$ vi
17bit0028@sjtg18site010:~$ clear
17bit0028@sjtg18site010:~$ gcc sjf.c
17bit0028@sjtg18site010:~$ ./a.out
Enter number of process:5
Enter Burst Time:
p1:6
p2:3
p3:8
p4:9
p5:1

```

Process	Burst Time	Waiting Time	Turnaround Time
p5	1	0	1
p2	3	1	4
p1	6	4	10
p3	8	10	18
p4	9	18	27

```

Average Waiting Time=6.600000
Average Turnaround Time=12.000000
17bit0028@sjtg18site010:~$

```

PRIORITY SCHEDULING

```
#include<stdio.h>
int main()
{
    int bt[20],p[20],wt[20],tat[20],pr[20],i,j,n,total=0,pos,temp,avg_wt,avg_tat;
    printf("Enter Total Number of Process:");
    scanf("%d",&n);

    printf("\nEnter Burst Time and Priority\n");
    for(i=0;i<n;i++)
    {
        printf("\nP[%d]\n",i+1);
        printf("Burst Time:");
        scanf("%d",&bt[i]);
        printf("Priority:");
        scanf("%d",&pr[i]);
        p[i]=i+1;          //contains process number
    }

    //sorting burst time, priority and process number in ascending order using selection sort
    for(i=0;i<n;i++)
    {
        pos=i;
        for(j=i+1;j<n;j++)
        {
            if(pr[j]<pr[pos])
                pos=j;
        }

        temp=pr[i];
        pr[i]=pr[pos];
        pr[pos]=temp;

        temp=bt[i];
        bt[i]=bt[pos];
        bt[pos]=temp;

        temp=p[i];
        p[i]=p[pos];
        p[pos]=temp;
    }

    wt[0]=0;  //waiting time for first process is zero

    //calculate waiting time
    for(i=1;i<n;i++)
    {
        wt[i]=0;
        for(j=0;j<i;j++)
            wt[i]+=bt[j];
    }
```

```

    total+=wt[i];
}

avg_wt=total/n;    //average waiting time
total=0;

printf("\nProcess\t Burst Time \tWaiting Time\tTurnaround Time");
for(i=0;i<n;i++)
{
    tat[i]=bt[i]+wt[i];    //calculate turnaround time
    total+=tat[i];
    printf("\nP[%d]\t\t %d\t\t %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);
}

avg_tat=total/n;    //average turnaround time
printf("\n\nAverage Waiting Time=%d",avg_wt);
printf("\nAverage Turnaround Time=%d\n",avg_tat);

return 0;
}

```

```

17bit0028@sjtg18site010: ~
17bit0028@sjtg18site010:~$ vi
17bit0028@sjtg18site010:~$ gcc priority.c
17bit0028@sjtg18site010:~$ ./a.out
Enter Total Number of Process:5

Enter Burst Time and Priority

P[1]
Burst Time:2
Priority:3

P[2]
Burst Time:4
Priority:5

P[3]
Burst Time:6
Priority:7

P[4]
Burst Time:8
Priority:9

P[5]
Burst Time:1
Priority:5

Process    Burst Time    Waiting Time    Turnaround Time
P[1]       2              0               2
P[2]       4              2               6
P[5]       1              6               7
P[3]       6              7              13
P[4]       8              13             21

Average Waiting Time=5
Average Turnaround Time=9
17bit0028@sjtg18site010:~$ 

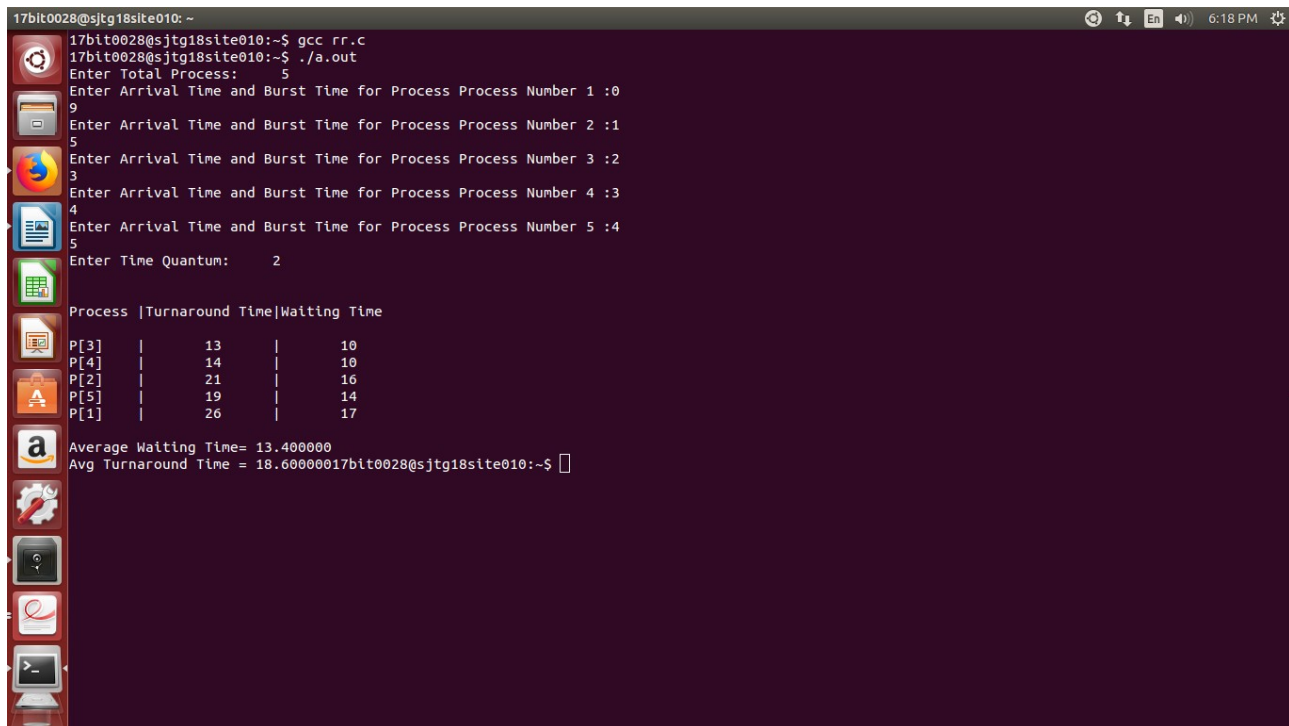
```

PREEMPTIVE ROUND ROBIN SCHEDULING

```
#include<stdio.h>
int main()
{

    int count,j,n,time,remain,flag=0,time_quantum;
    int wait_time=0,turnaround_time=0,at[10],bt[10],rt[10];
    printf("Enter Total Process:\t ");
    scanf("%d",&n);
    remain=n;
    for(count=0;count<n;count++)
    {
        printf("Enter Arrival Time and Burst Time for Process Process Number %d :",count+1);
        scanf("%d",&at[count]);
        scanf("%d",&bt[count]);
        rt[count]=bt[count];
    }
    printf("Enter Time Quantum:\t");
    scanf("%d",&time_quantum);
    printf("\n\nProcess\t|Turnaround Time|Waiting Time\n\n");
    for(time=0,count=0;remain!=0;)
    {
        if(rt[count]<=time_quantum && rt[count]>0)
        {
            time+=rt[count];
            rt[count]=0;
            flag=1;
        }
        else if(rt[count]>0)
        {
            rt[count]-=time_quantum;
            time+=time_quantum;
        }
        if(rt[count]==0 && flag==1)
        {
            remain--;
            printf("P[%d]\t|\t%d\t|\t%d\n",count+1,time-at[count],time-at[count]-bt[count]);
            wait_time+=time-at[count]-bt[count];
            turnaround_time+=time-at[count];
            flag=0;
        }
    }
    if(count==n-1)
        count=0;
    else if(at[count+1]<=time)
        count++;
    else
        count=0;
}
printf("\nAverage Waiting Time= %f\n",wait_time*1.0/n);
```

```
printf("Avg Turnaround Time = %f",turnaround_time*1.0/n);  
return 0;  
}
```



```
17bit0028@sjtg18site010: ~  
17bit0028@sjtg18site010:~$ gcc rr.c  
17bit0028@sjtg18site010:~$ ./a.out  
Enter Total Process: 5  
Enter Arrival Time and Burst Time for Process Process Number 1 :0  
9  
Enter Arrival Time and Burst Time for Process Process Number 2 :1  
5  
Enter Arrival Time and Burst Time for Process Process Number 3 :2  
3  
Enter Arrival Time and Burst Time for Process Process Number 4 :3  
4  
Enter Arrival Time and Burst Time for Process Process Number 5 :4  
5  
Enter Time Quantum: 2  
  
Process |Turnaround Time|Waiting Time  
P[3] | 13 | 10  
P[4] | 14 | 10  
P[2] | 21 | 16  
P[5] | 19 | 14  
P[1] | 26 | 17  
  
Average Waiting Time= 13.400000  
Avg Turnaround Time = 18.60000017bit0028@sjtg18site010:~$
```


SJF PREEMPTIVE SCHEDULING

```
#include<stdio.h>
struct process
{
    char process_name;
    int arrival_time, burst_time, ct, waiting_time, turnaround_time, priority;
    int status;
}process_queue[10];

int limit;

void Arrival_Time_Sorting()
{
    struct process temp;
    int i, j;
    for(i = 0; i < limit - 1; i++)
    {
        for(j = i + 1; j < limit; j++)
        {
            if(process_queue[i].arrival_time > process_queue[j].arrival_time)
            {
                temp = process_queue[i];
                process_queue[i] = process_queue[j];
                process_queue[j] = temp;
            }
        }
    }
}

void main()
{
    int i, time = 0, burst_time = 0, largest;
    char c;
    float wait_time = 0, turnaround_time = 0, average_waiting_time, average_turnaround_time;
    printf("\nEnter Total Number of Processes:\t");
    scanf("%d", &limit);
    for(i = 0, c = 'A'; i < limit; i++, c++)
    {
        process_queue[i].process_name = c;
        printf("\nEnter Details For Process[%C]:\n", process_queue[i].process_name);
        printf("Enter Arrival Time:\t");
        scanf("%d", &process_queue[i].arrival_time );
        printf("Enter Burst Time:\t");
        scanf("%d", &process_queue[i].burst_time);
        printf("Enter Priority:\t");
        scanf("%d", &process_queue[i].priority);
        process_queue[i].status = 0;
        burst_time = burst_time + process_queue[i].burst_time;
    }
    Arrival_Time_Sorting();
    process_queue[9].priority = -9999;
```

```

printf("\nProcess Name\tArrival Time\tBurst Time\tPriority\tWaiting Time");
for(time = process_queue[0].arrival_time; time < burst_time;)
{
    largest = 9;
    for(i = 0; i < limit; i++)
    {
        if(process_queue[i].arrival_time <= time && process_queue[i].status != 1 &&
process_queue[i].priority > process_queue[largest].priority)
        {
            largest = i;
        }
    }
    time = time + process_queue[largest].burst_time;
    process_queue[largest].ct = time;
    process_queue[largest].waiting_time = process_queue[largest].ct -
process_queue[largest].arrival_time - process_queue[largest].burst_time;
    process_queue[largest].turnaround_time = process_queue[largest].ct -
process_queue[largest].arrival_time;
    process_queue[largest].status = 1;
    wait_time = wait_time + process_queue[largest].waiting_time;
    turnaround_time = turnaround_time + process_queue[largest].turnaround_time;
    printf("\n%c\t\t%d\t\t%d\t\t%d\t\t%d", process_queue[largest].process_name,
process_queue[largest].arrival_time, process_queue[largest].burst_time,
process_queue[largest].priority, process_queue[largest].waiting_time);
}
average_waiting_time = wait_time / limit;
average_turnaround_time = turnaround_time / limit;
printf("\n\nAverage waiting time:\t%f\n", average_waiting_time);
printf("Average Turnaround Time:\t%f\n", average_turnaround_time);
}

```

```

17bit0028@sjtg18site010: ~
17bit0028@sjtg18site010:~$ vi
17bit0028@sjtg18site010:~$ gcc sjfp.c
17bit0028@sjtg18site010:~$ ./a.out
Enter Total Number of Processes:      5
Enter Details For Process[A]:
Enter Arrival Time:      2
Enter Burst Time:      1
Enter Priority:      4
Enter Details For Process[B]:
Enter Arrival Time:      2
Enter Burst Time:      4
Enter Priority:      1
Enter Details For Process[C]:
Enter Arrival Time:      3
Enter Burst Time:      6
Enter Priority:      3
Enter Details For Process[D]:
Enter Arrival Time:      6
Enter Burst Time:      6
Enter Priority:      2
Enter Details For Process[E]:
Enter Arrival Time:      7
Enter Burst Time:      3
Enter Priority:      5
Process Name  Arrival Time  Burst Time  Priority  Waiting Time
A            2            1            4            0
C            3            6            3            0
E            7            3            5            2
D            6            6            2            6
B            2            4            1           16
Average waiting time:  4.800000
Average Turnaround Time:  8.800000
17bit0028@sjtg18site010:~$ 

```