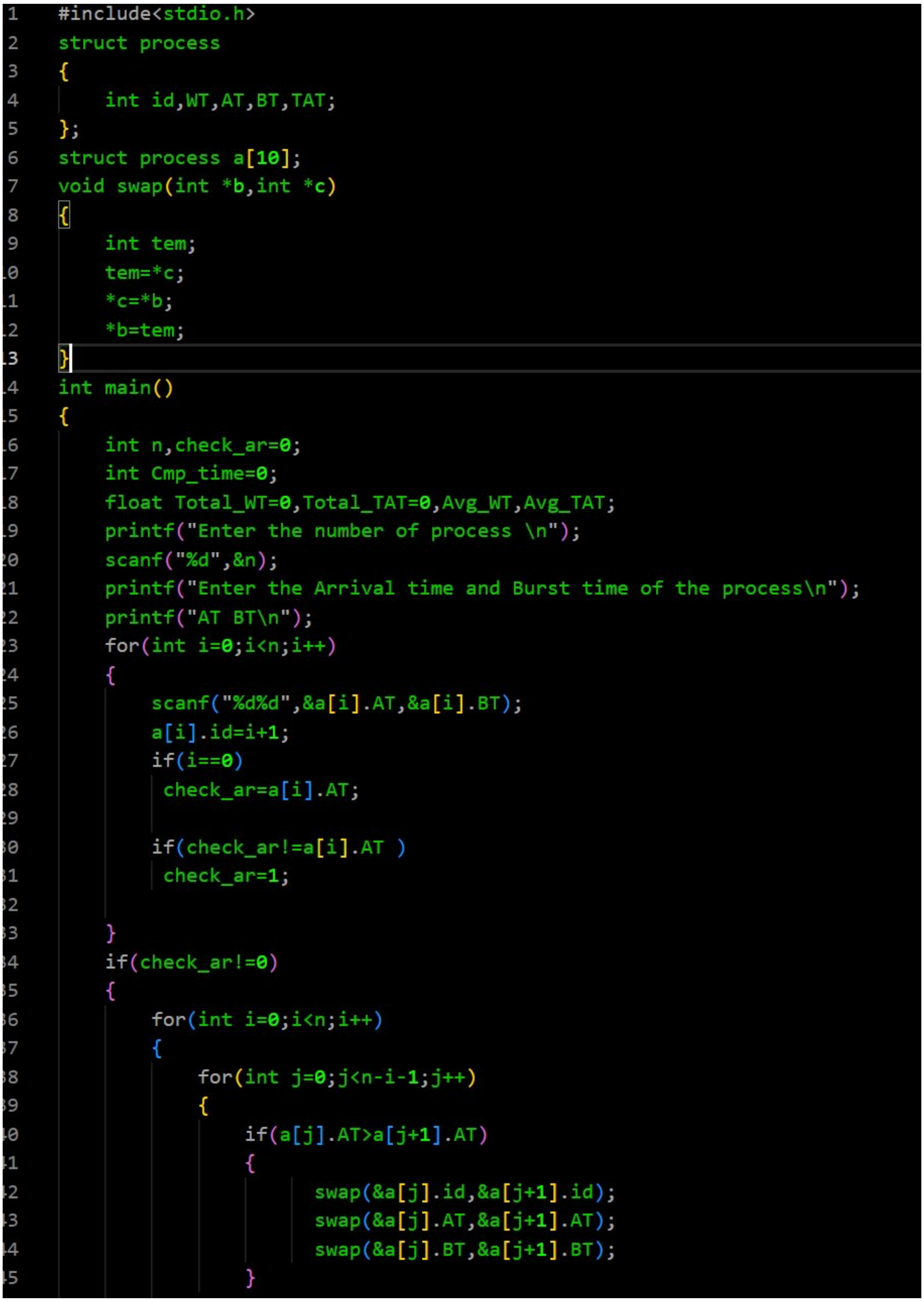
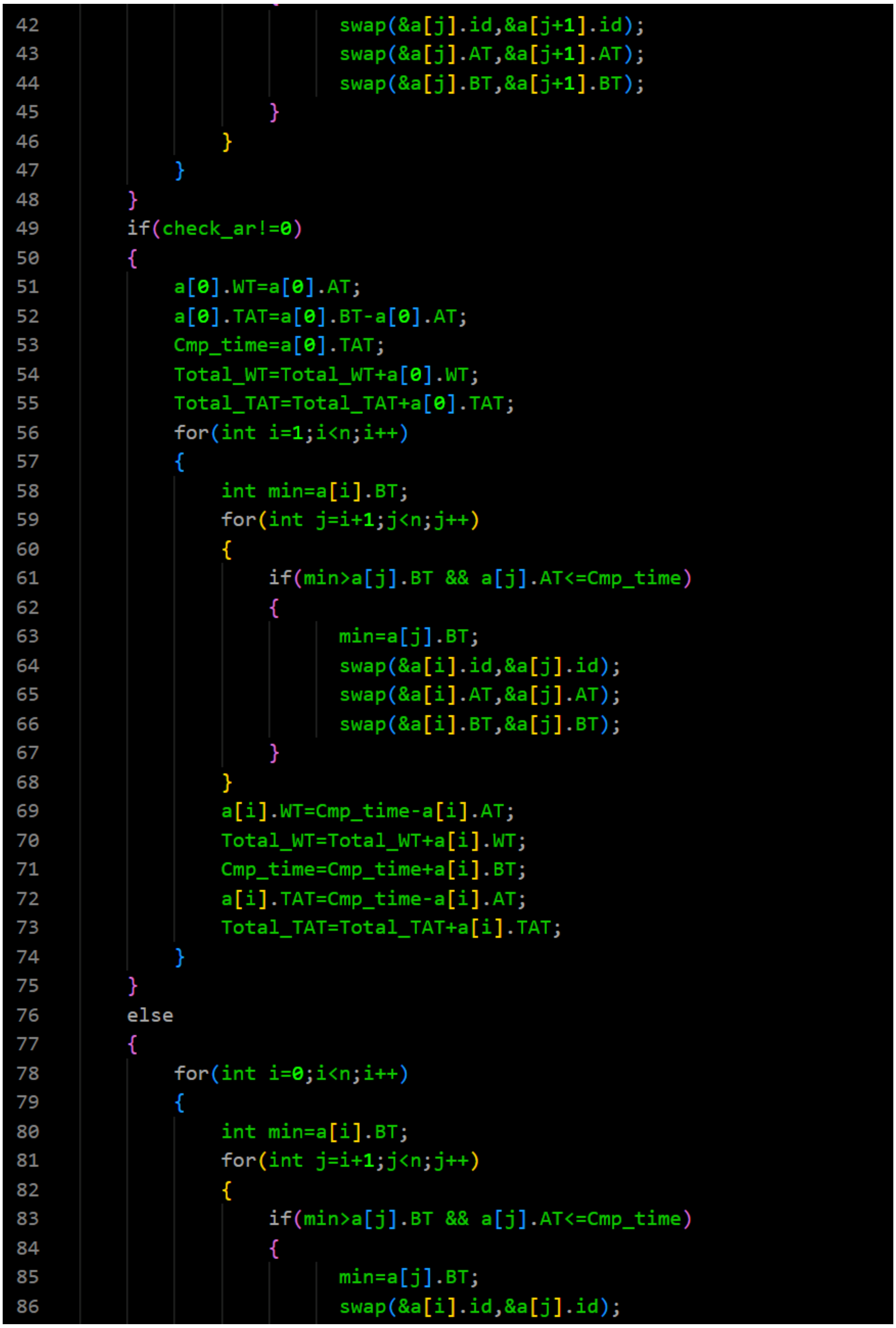
EXPERIMENT - 2

CPU SHEDULING

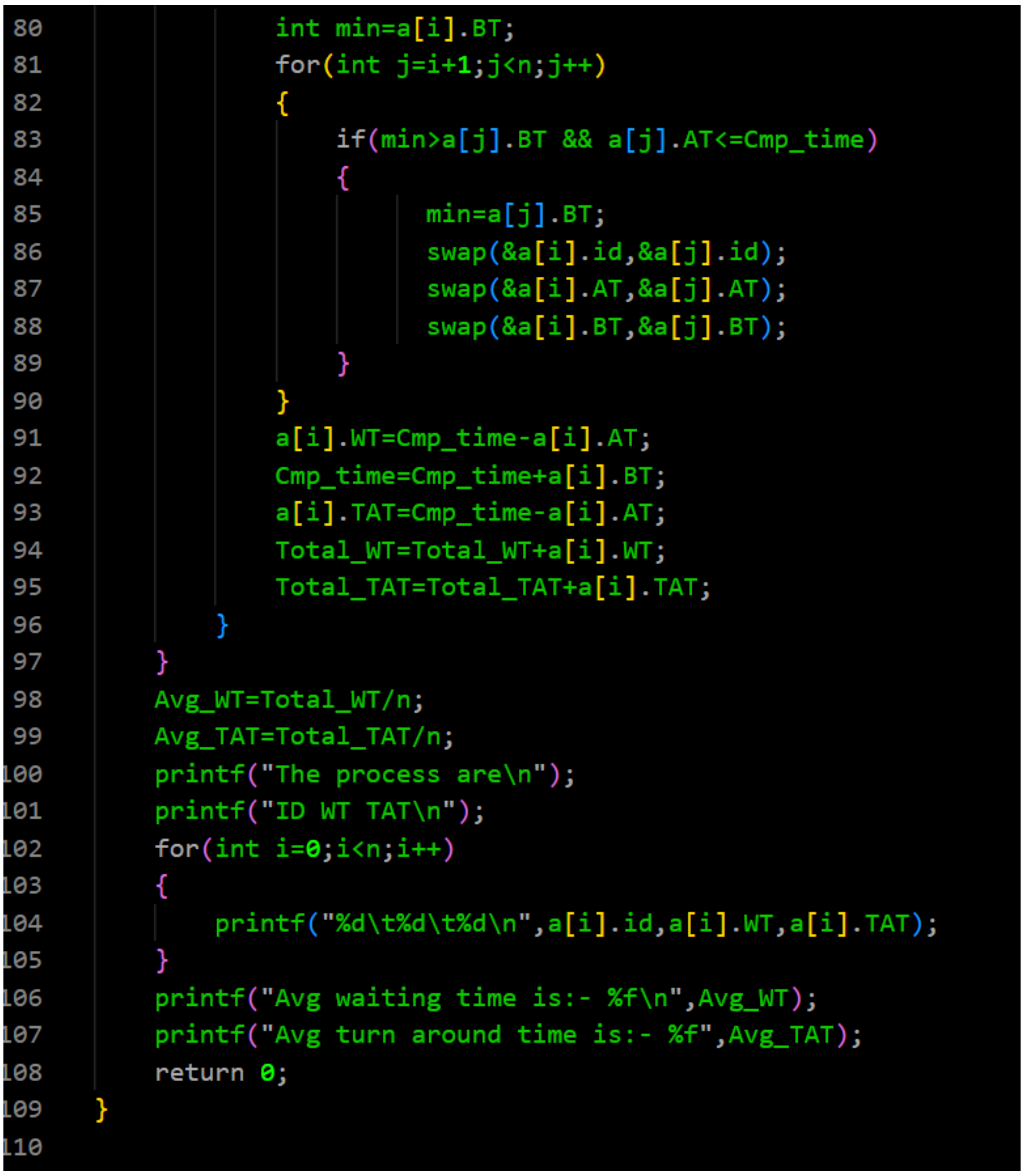
i. To write a C program to implement the CPU scheduling algorithm for FIRST COME FIRST SERVE.



```c
#include<stdio.h>
struct process
{
    int id,WT,AT,BT,TAT;
};
struct process a[10];
void swap(int *b,int *c)
{
    int tem;
    tem=*c;
    *c=*b;
    *b=tem;
}
int main()
{
    int n,check_ar=0;
    int Cmp_time=0;
    float Total_WT=0,Total_TAT=0,Avg_WT,Avg_TAT;
    printf("Enter the number of process \n");
    scanf("%d",&n);
    printf("Enter the Arrival time and Burst time of the process\n");
    printf("AT BT\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d%d",&a[i].AT,&a[i].BT);
        a[i].id=i+1;
        if(i==0)
         check_ar=a[i].AT;

        if(check_ar!=a[i].AT )
         check_ar=1;

    }
    if(check_ar!=0)
    {
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n-i-1;j++)
            {
                if(a[j].AT>a[j+1].AT)
                {
                      swap(&a[j].id,&a[j+1].id);
                      swap(&a[j].AT,&a[j+1].AT);
                      swap(&a[j].BT,&a[j+1].BT);
                }
```

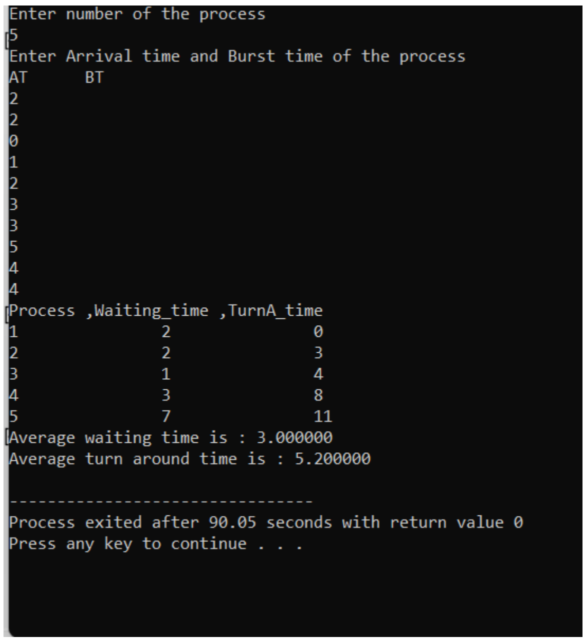


```
                    swap(&a[j].id,&a[j+1].id);
                    swap(&a[j].AT,&a[j+1].AT);
                    swap(&a[j].BT,&a[j+1].BT);
                }
            }
        }
    }
    if(check_ar!=0)
    {
        a[0].WT=a[0].AT;
        a[0].TAT=a[0].BT-a[0].AT;
        Cmp_time=a[0].TAT;
        Total_WT=Total_WT+a[0].WT;
        Total_TAT=Total_TAT+a[0].TAT;
        for(int i=1;i<n;i++)
        {
            int min=a[i].BT;
            for(int j=i+1;j<n;j++)
            {
                if(min>a[j].BT && a[j].AT<=Cmp_time)
                {
                    min=a[j].BT;
                    swap(&a[i].id,&a[j].id);
                    swap(&a[i].AT,&a[j].AT);
                    swap(&a[i].BT,&a[j].BT);
                }
            }
            a[i].WT=Cmp_time-a[i].AT;
            Total_WT=Total_WT+a[i].WT;
            Cmp_time=Cmp_time+a[i].BT;
            a[i].TAT=Cmp_time-a[i].AT;
            Total_TAT=Total_TAT+a[i].TAT;
        }
    }
    else
    {
        for(int i=0;i<n;i++)
        {
            int min=a[i].BT;
            for(int j=i+1;j<n;j++)
            {
                if(min>a[j].BT && a[j].AT<=Cmp_time)
                {
                    min=a[j].BT;
                    swap(&a[i].id,&a[j].id);
```



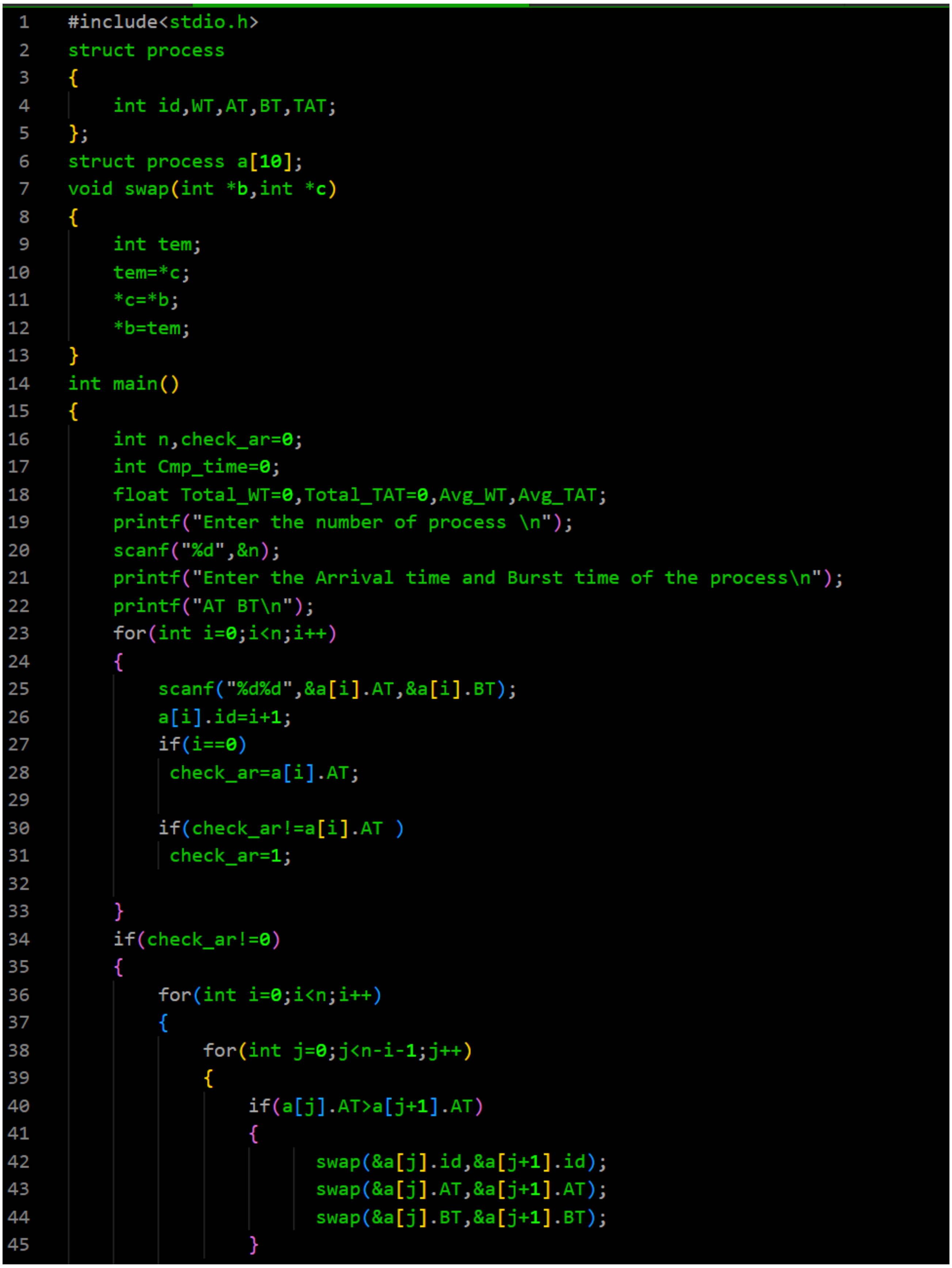
```c
            int min=a[i].BT;
            for(int j=i+1;j<n;j++)
            {
                if(min>a[j].BT && a[j].AT<=Cmp_time)
                {
                    min=a[j].BT;
                    swap(&a[i].id,&a[j].id);
                    swap(&a[i].AT,&a[j].AT);
                    swap(&a[i].BT,&a[j].BT);
                }
            }
            a[i].WT=Cmp_time-a[i].AT;
            Cmp_time=Cmp_time+a[i].BT;
            a[i].TAT=Cmp_time-a[i].AT;
            Total_WT=Total_WT+a[i].WT;
            Total_TAT=Total_TAT+a[i].TAT;
        }
    }
    Avg_WT=Total_WT/n;
    Avg_TAT=Total_TAT/n;
    printf("The process are\n");
    printf("ID WT TAT\n");
    for(int i=0;i<n;i++)
    {
        printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
    }
    printf("Avg waiting time is:- %f\n",Avg_WT);
    printf("Avg turn around time is:- %f",Avg_TAT);
    return 0;
}
```

OUTPUT

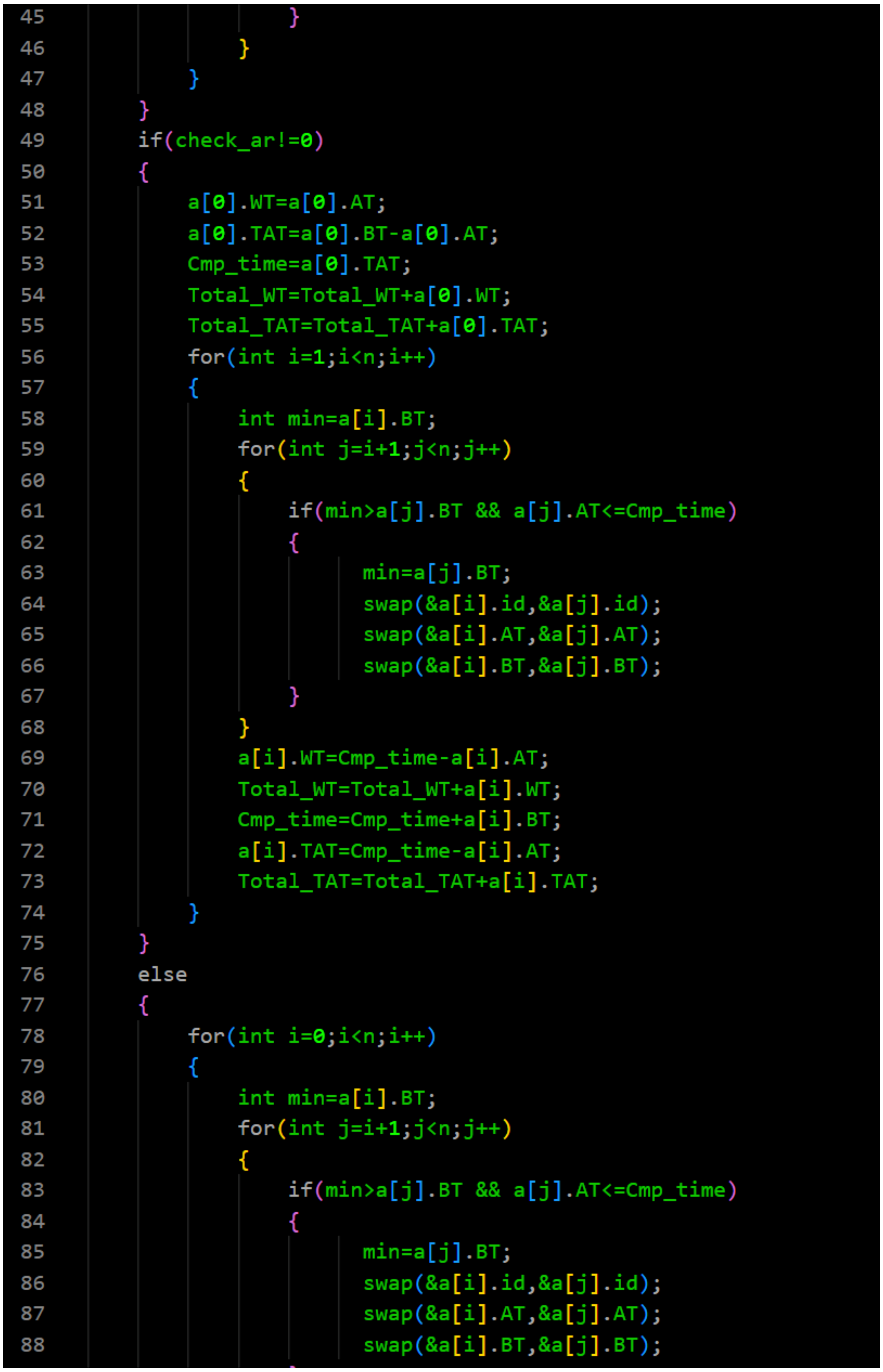


```
Enter number of the process
5
Enter Arrival time and Burst time of the process
AT      BT
2
2
0
1
2
3
3
5
4
4
Process ,Waiting_time ,TurnA_time
1               2               0
2               2               3
3               1               4
4               3               8
5               7               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 . . .
```

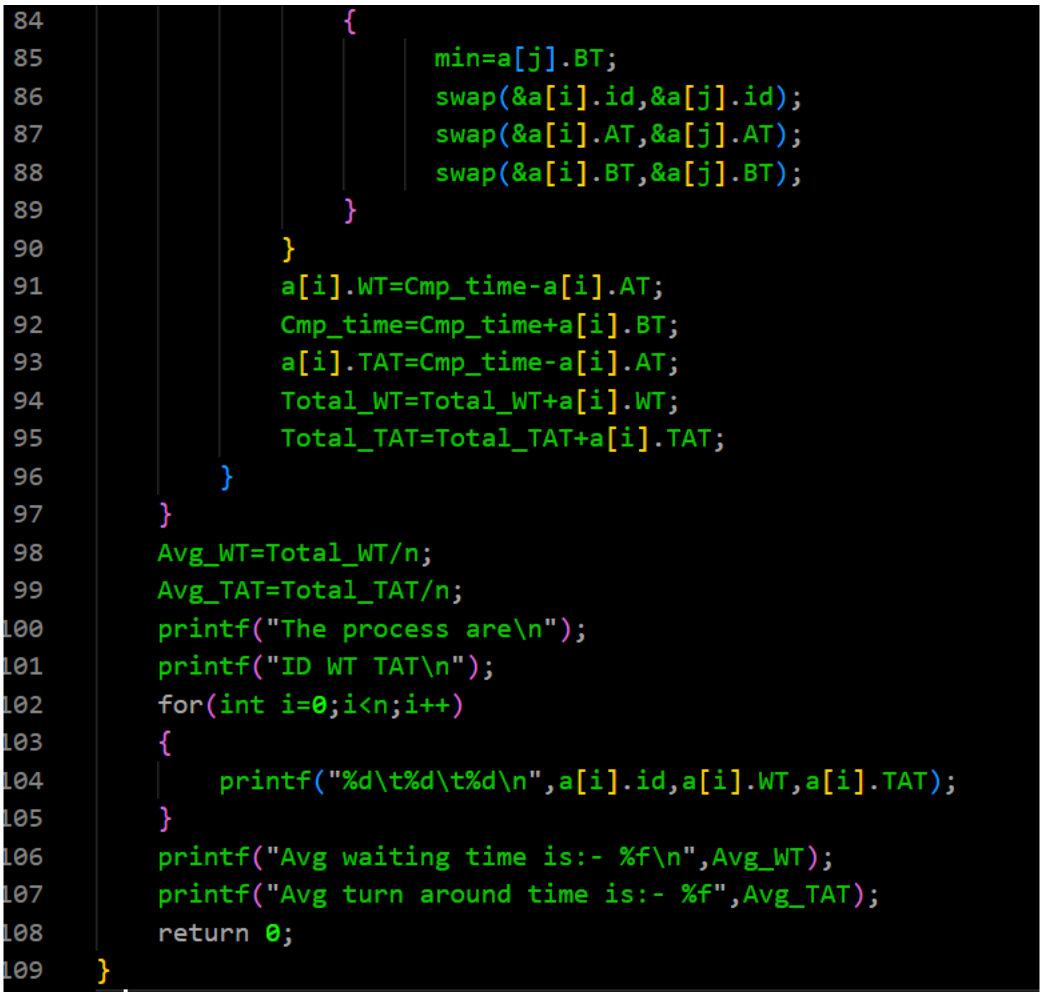
ii. To write a C program to implement the CPU scheduling algorithm for **Shortest Job First**



```c
#include<stdio.h>
struct process
{
    int id,WT,AT,BT,TAT;
};
struct process a[10];
void swap(int *b,int *c)
{
    int tem;
    tem=*c;
    *c=*b;
    *b=tem;
}
int main()
{
    int n,check_ar=0;
    int Cmp_time=0;
    float Total_WT=0,Total_TAT=0,Avg_WT,Avg_TAT;
    printf("Enter the number of process \n");
    scanf("%d",&n);
    printf("Enter the Arrival time and Burst time of the process\n");
    printf("AT BT\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d%d",&a[i].AT,&a[i].BT);
        a[i].id=i+1;
        if(i==0)
         check_ar=a[i].AT;

        if(check_ar!=a[i].AT )
         check_ar=1;

    }
    if(check_ar!=0)
    {
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n-i-1;j++)
            {
                if(a[j].AT>a[j+1].AT)
                {
                      swap(&a[j].id,&a[j+1].id);
                      swap(&a[j].AT,&a[j+1].AT);
                      swap(&a[j].BT,&a[j+1].BT);
                }
```

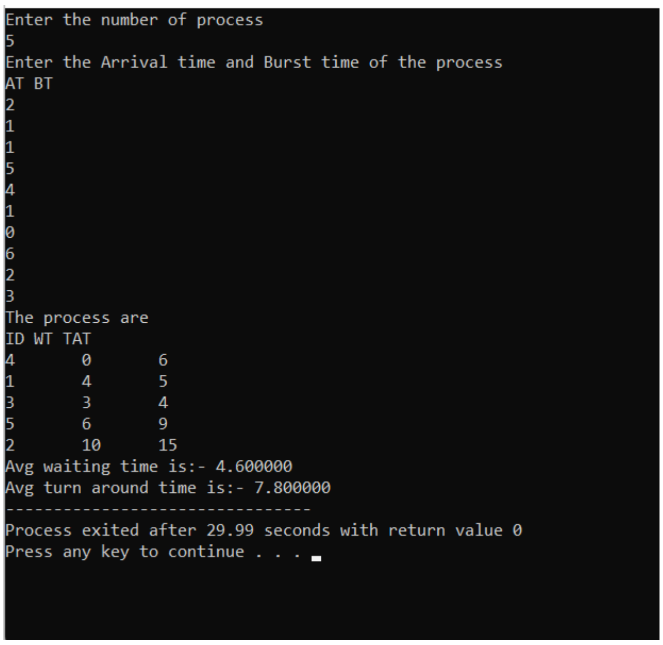


```
                    }
                }
            }
        }
        if(check_ar!=0)
        {
            a[0].WT=a[0].AT;
            a[0].TAT=a[0].BT-a[0].AT;
            Cmp_time=a[0].TAT;
            Total_WT=Total_WT+a[0].WT;
            Total_TAT=Total_TAT+a[0].TAT;
            for(int i=1;i<n;i++)
            {
                int min=a[i].BT;
                for(int j=i+1;j<n;j++)
                {
                    if(min>a[j].BT && a[j].AT<=Cmp_time)
                    {
                          min=a[j].BT;
                          swap(&a[i].id,&a[j].id);
                          swap(&a[i].AT,&a[j].AT);
                          swap(&a[i].BT,&a[j].BT);
                    }
                }
                a[i].WT=Cmp_time-a[i].AT;
                Total_WT=Total_WT+a[i].WT;
                Cmp_time=Cmp_time+a[i].BT;
                a[i].TAT=Cmp_time-a[i].AT;
                Total_TAT=Total_TAT+a[i].TAT;
            }
        }
        else
        {
            for(int i=0;i<n;i++)
            {
                int min=a[i].BT;
                for(int j=i+1;j<n;j++)
                {
                    if(min>a[j].BT && a[j].AT<=Cmp_time)
                    {
                          min=a[j].BT;
                          swap(&a[i].id,&a[j].id);
                          swap(&a[i].AT,&a[j].AT);
                          swap(&a[i].BT,&a[j].BT);
```



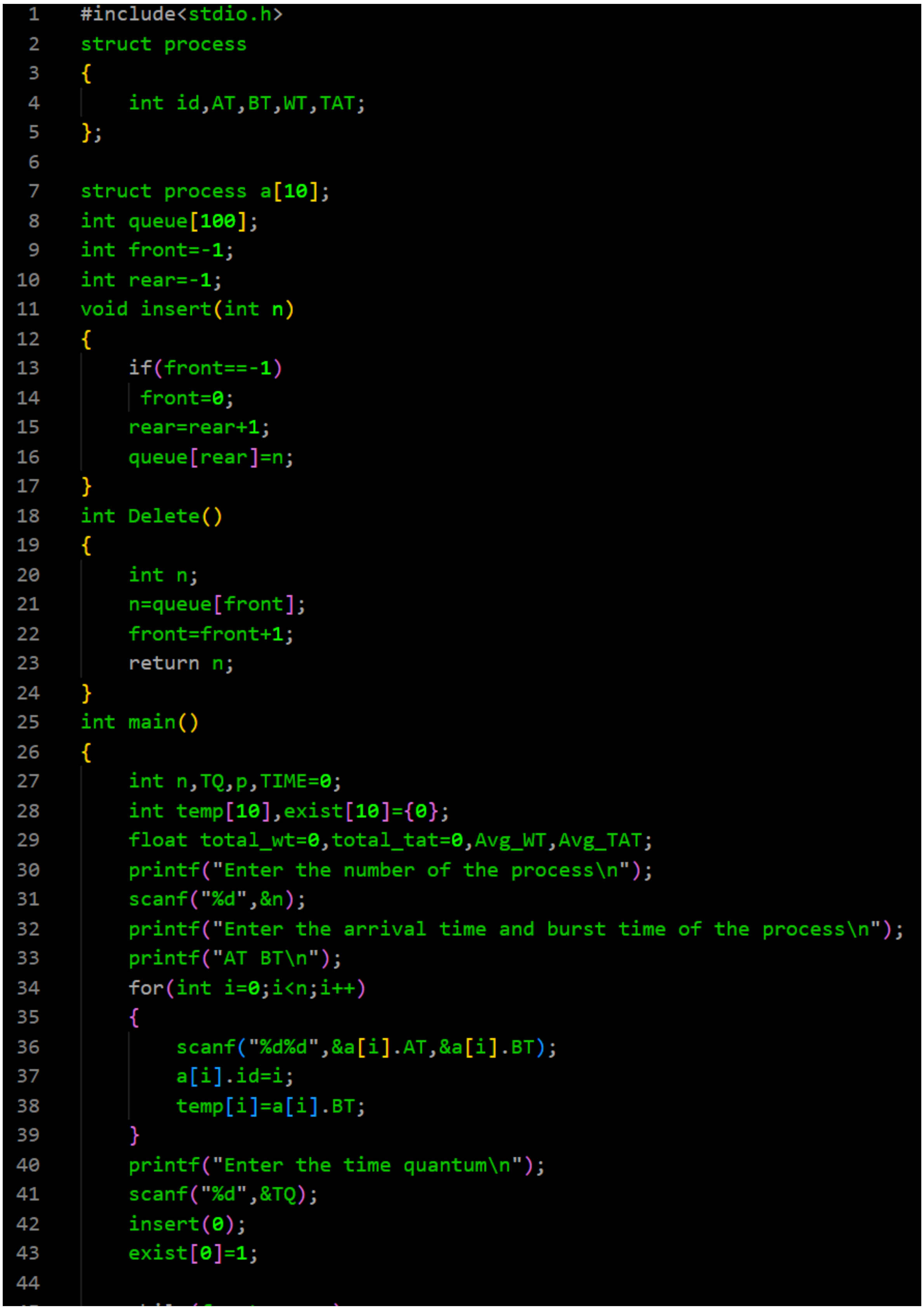
```
                {
                    min=a[j].BT;
                    swap(&a[i].id,&a[j].id);
                    swap(&a[i].AT,&a[j].AT);
                    swap(&a[i].BT,&a[j].BT);
                }
            }
            a[i].WT=Cmp_time-a[i].AT;
            Cmp_time=Cmp_time+a[i].BT;
            a[i].TAT=Cmp_time-a[i].AT;
            Total_WT=Total_WT+a[i].WT;
            Total_TAT=Total_TAT+a[i].TAT;
        }
    }
    Avg_WT=Total_WT/n;
    Avg_TAT=Total_TAT/n;
    printf("The process are\n");
    printf("ID WT TAT\n");
    for(int i=0;i<n;i++)
    {
        printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
    }
    printf("Avg waiting time is:- %f\n",Avg_WT);
    printf("Avg turn around time is:- %f",Avg_TAT);
    return 0;
}
```

OUTPUT

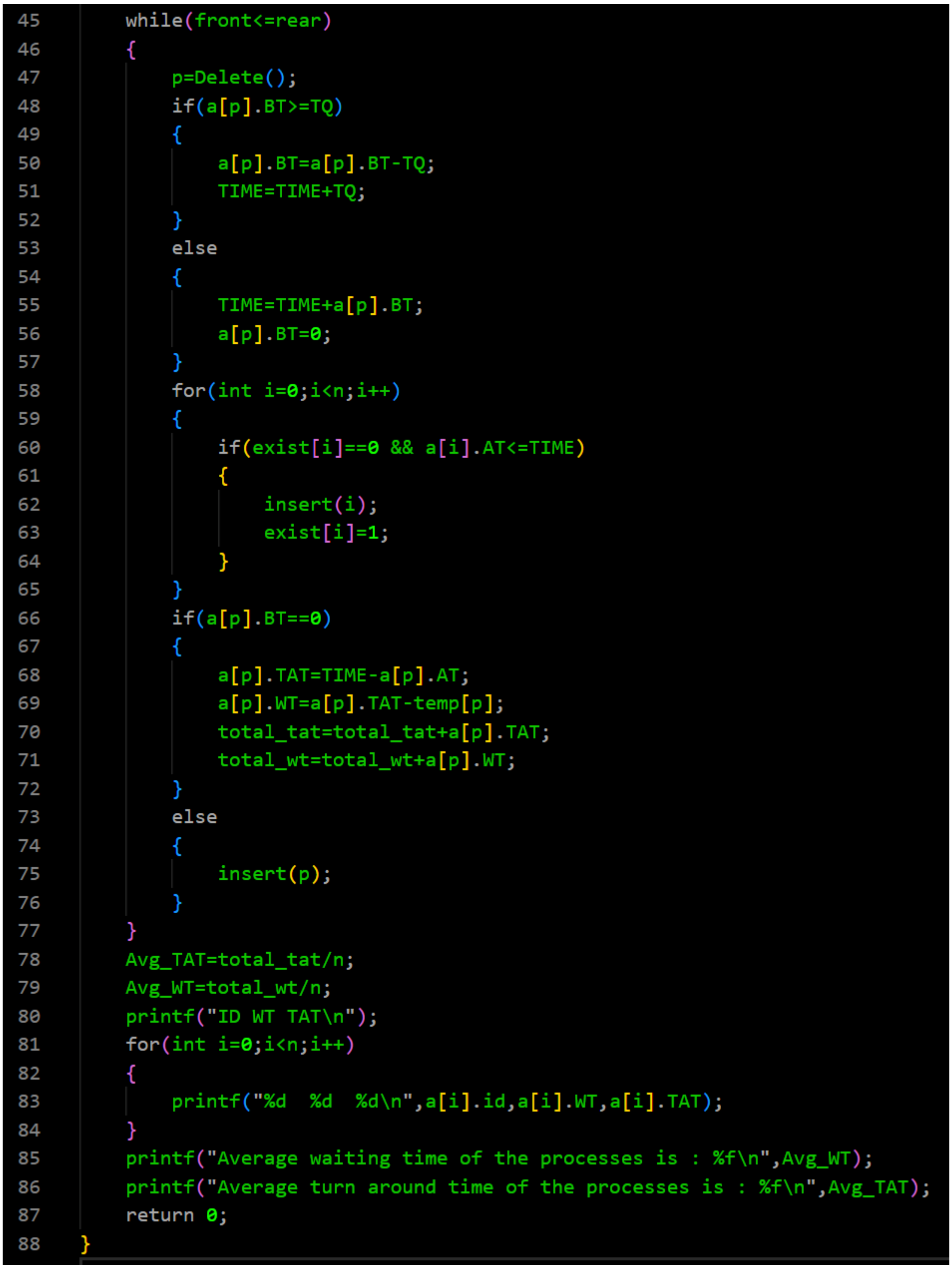


```
Enter the number of process
5
Enter the Arrival time and Burst time of the process
AT BT
2
1
1
5
4
1
0
6
2
3
The process are
ID WT TAT
4       0       6
1       4       5
3       3       4
5       6       9
2       10      15
Avg waiting time is:- 4.600000
Avg turn around time is:- 7.800000
--------------------------------
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.**

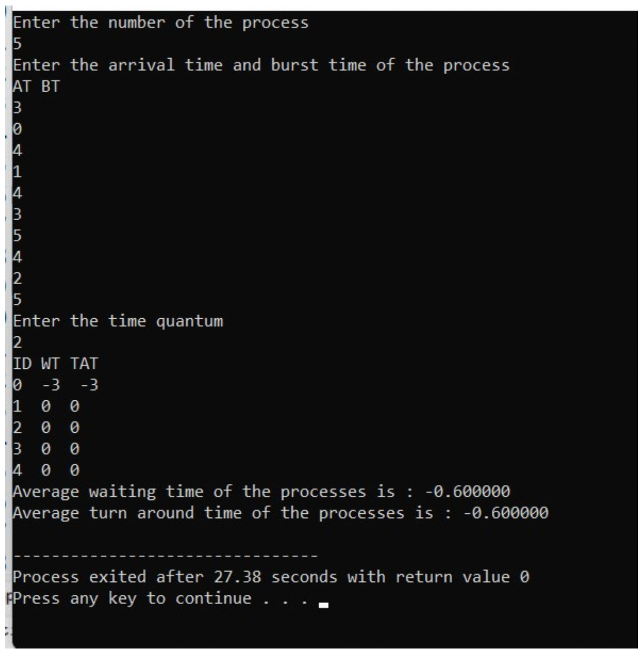


```c
#include<stdio.h>
struct process
{
    int id,AT,BT,WT,TAT;
};

struct process a[10];
int queue[100];
int front=-1;
int rear=-1;
void insert(int n)
{
    if(front==-1)
     front=0;
    rear=rear+1;
    queue[rear]=n;
}
int Delete()
{
    int n;
    n=queue[front];
    front=front+1;
    return n;
}
int main()
{
    int n,TQ,p,TIME=0;
    int temp[10],exist[10]={0};
    float total_wt=0,total_tat=0,Avg_WT,Avg_TAT;
    printf("Enter the number of the process\n");
    scanf("%d",&n);
    printf("Enter the arrival time and burst time of the process\n");
    printf("AT BT\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d%d",&a[i].AT,&a[i].BT);
        a[i].id=i;
        temp[i]=a[i].BT;
    }
    printf("Enter the time quantum\n");
    scanf("%d",&TQ);
    insert(0);
    exist[0]=1;

```



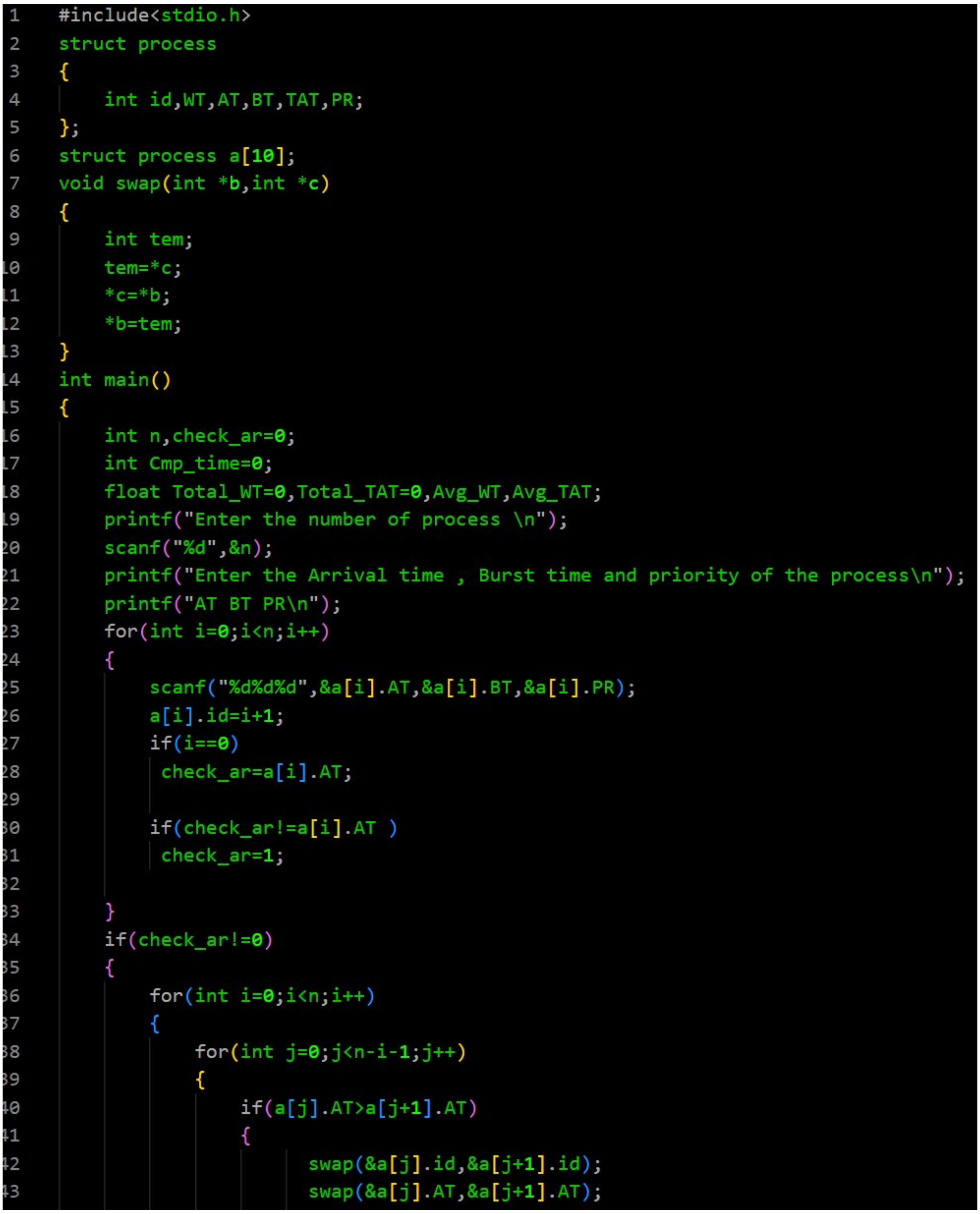
```c
    while(front<=rear)
    {
        p=Delete();
        if(a[p].BT>=TQ)
        {
            a[p].BT=a[p].BT-TQ;
            TIME=TIME+TQ;
        }
        else
        {
            TIME=TIME+a[p].BT;
            a[p].BT=0;
        }
        for(int i=0;i<n;i++)
        {
            if(exist[i]==0 && a[i].AT<=TIME)
            {
                insert(i);
                exist[i]=1;
            }
        }
        if(a[p].BT==0)
        {
            a[p].TAT=TIME-a[p].AT;
            a[p].WT=a[p].TAT-temp[p];
            total_tat=total_tat+a[p].TAT;
            total_wt=total_wt+a[p].WT;
        }
        else
        {
            insert(p);
        }
    }
    Avg_TAT=total_tat/n;
    Avg_WT=total_wt/n;
    printf("ID WT TAT\n");
    for(int i=0;i<n;i++)
    {
        printf("%d  %d  %d\n",a[i].id,a[i].WT,a[i].TAT);
    }
    printf("Average waiting time of the processes is : %f\n",Avg_WT);
    printf("Average turn around time of the processes is : %f\n",Avg_TAT);
    return 0;
}
```

OUTPUT

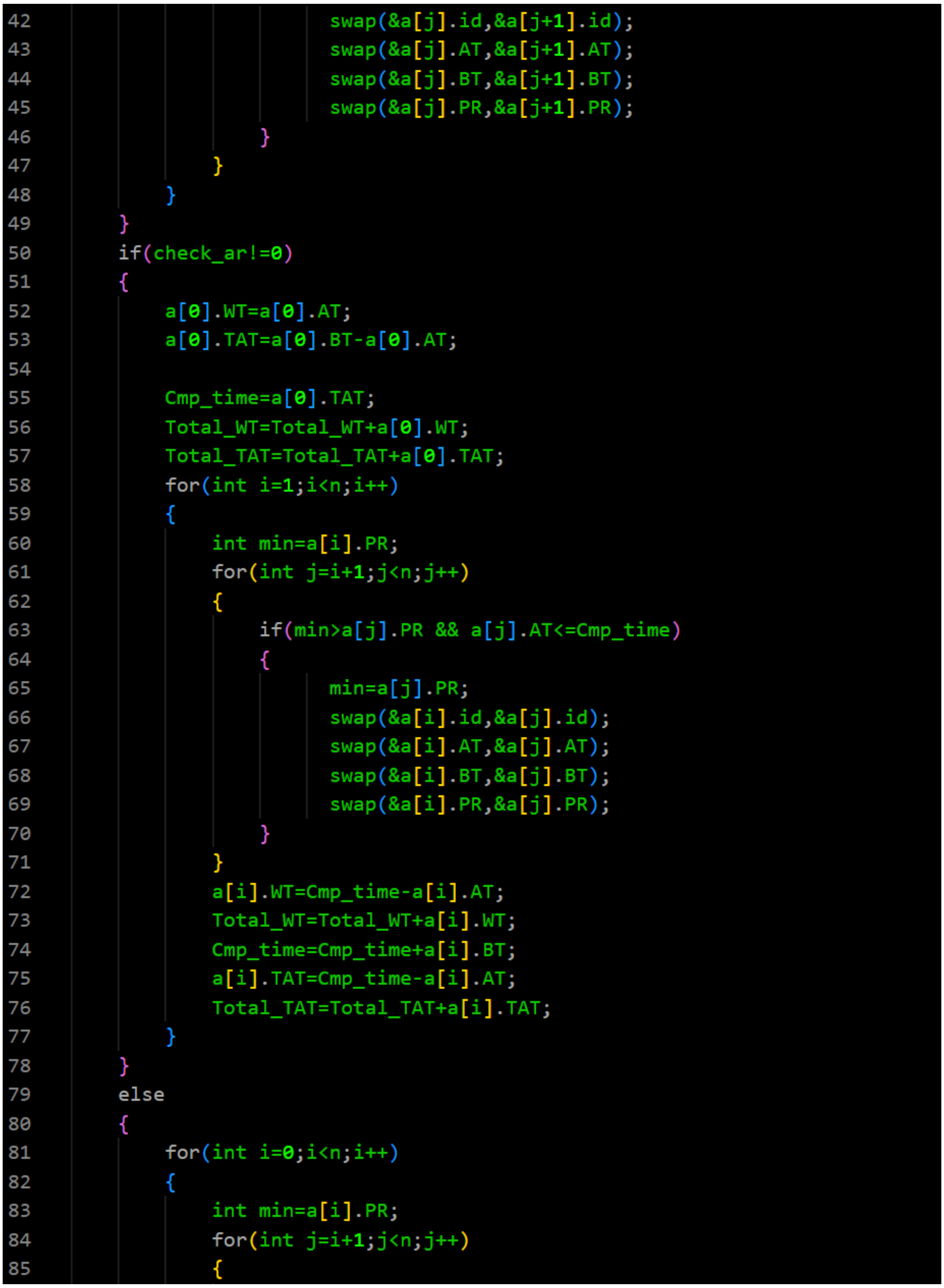


```
Enter the number of the process
5
Enter the arrival time and burst time of the process
AT BT
3
0
4
1
4
3
5
4
2
5
Enter the time quantum
2
ID WT TAT
0  -3  -3
1  0  0
2  0  0
3  0  0
4  0  0
Average waiting time of the processes is : -0.600000
Average turn around time of the processes is : -0.600000

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

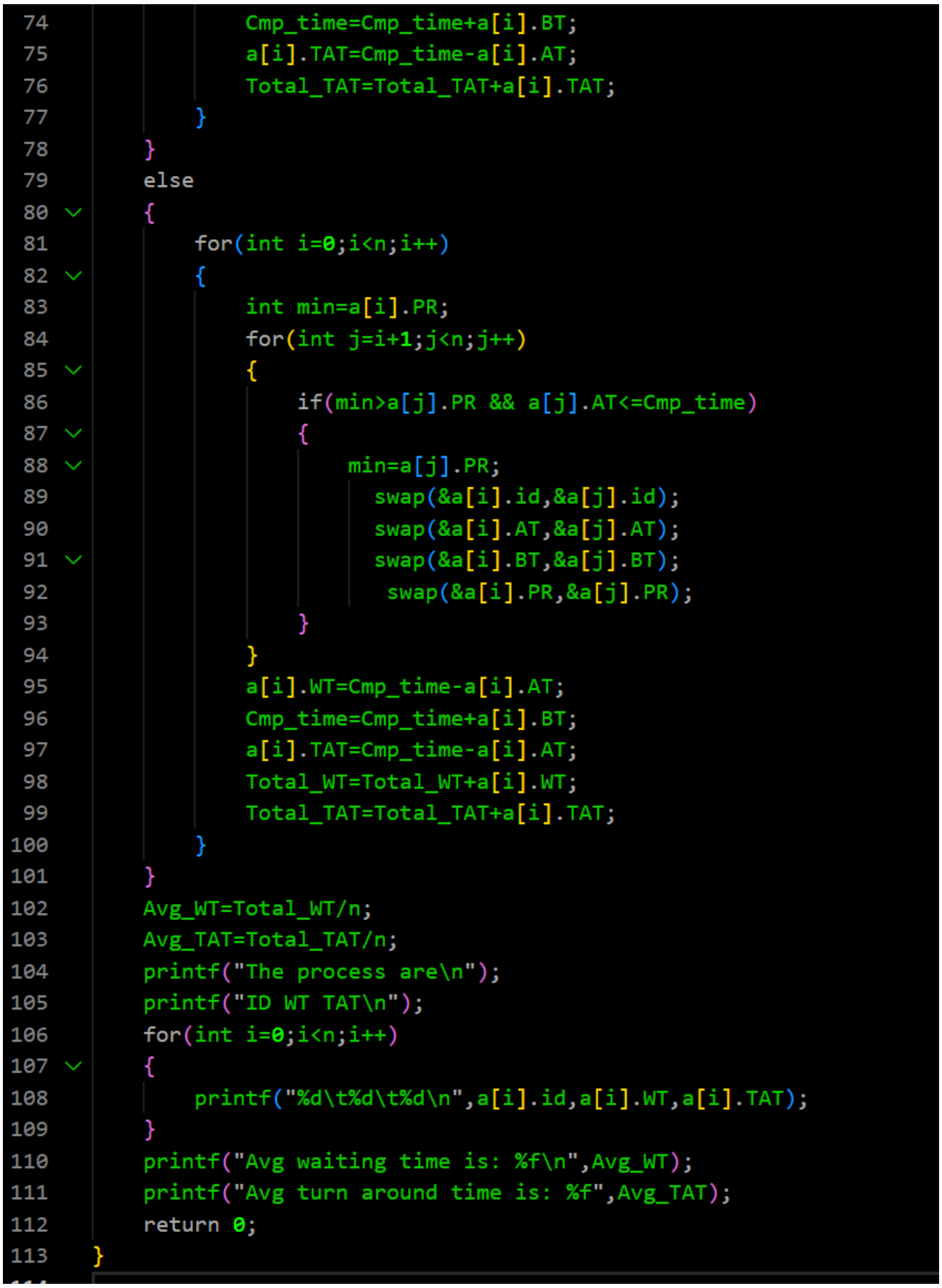
i) To write a C program to implement the CPU scheduling algorithm for **Priority Scheduling.**



```c
#include<stdio.h>
struct process
{
    int id,WT,AT,BT,TAT,PR;
};
struct process a[10];
void swap(int *b,int *c)
{
    int tem;
    tem=*c;
    *c=*b;
    *b=tem;
}
int main()
{
    int n,check_ar=0;
    int Cmp_time=0;
    float Total_WT=0,Total_TAT=0,Avg_WT,Avg_TAT;
    printf("Enter the number of process \n");
    scanf("%d",&n);
    printf("Enter the Arrival time , Burst time and priority of the process\n");
    printf("AT BT PR\n");
    for(int i=0;i<n;i++)
    {
        scanf("%d%d%d",&a[i].AT,&a[i].BT,&a[i].PR);
        a[i].id=i+1;
        if(i==0)
         check_ar=a[i].AT;

        if(check_ar!=a[i].AT )
         check_ar=1;

    }
    if(check_ar!=0)
    {
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n-i-1;j++)
            {
                if(a[j].AT>a[j+1].AT)
                {
                      swap(&a[j].id,&a[j+1].id);
                      swap(&a[j].AT,&a[j+1].AT);
```

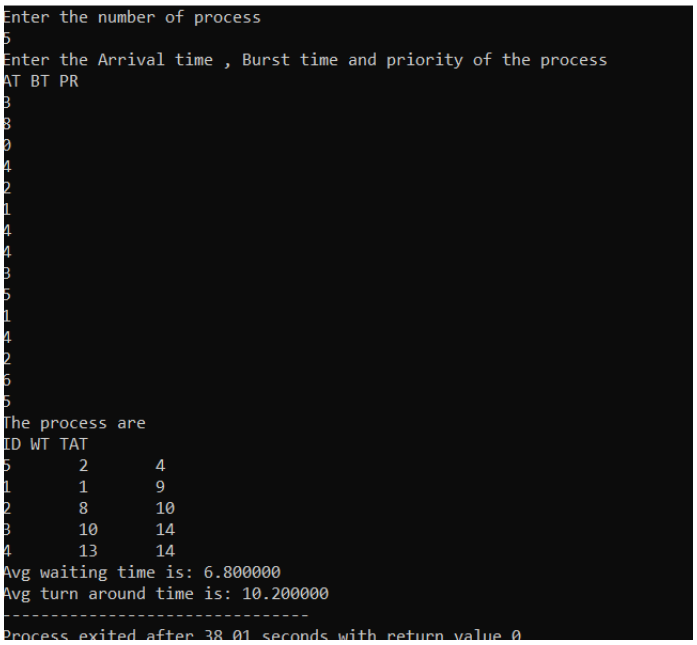


```
                        swap(&a[j].id,&a[j+1].id);
                        swap(&a[j].AT,&a[j+1].AT);
                        swap(&a[j].BT,&a[j+1].BT);
                        swap(&a[j].PR,&a[j+1].PR);
                    }
                }
            }
        }
        if(check_ar!=0)
        {
            a[0].WT=a[0].AT;
            a[0].TAT=a[0].BT-a[0].AT;

            Cmp_time=a[0].TAT;
            Total_WT=Total_WT+a[0].WT;
            Total_TAT=Total_TAT+a[0].TAT;
            for(int i=1;i<n;i++)
            {
                int min=a[i].PR;
                for(int j=i+1;j<n;j++)
                {
                    if(min>a[j].PR && a[j].AT<=Cmp_time)
                    {
                        min=a[j].PR;
                        swap(&a[i].id,&a[j].id);
                        swap(&a[i].AT,&a[j].AT);
                        swap(&a[i].BT,&a[j].BT);
                        swap(&a[i].PR,&a[j].PR);
                    }
                }
                a[i].WT=Cmp_time-a[i].AT;
                Total_WT=Total_WT+a[i].WT;
                Cmp_time=Cmp_time+a[i].BT;
                a[i].TAT=Cmp_time-a[i].AT;
                Total_TAT=Total_TAT+a[i].TAT;
            }
        }
        else
        {
            for(int i=0;i<n;i++)
            {
                int min=a[i].PR;
                for(int j=i+1;j<n;j++)
                {
```



```
                Cmp_time=Cmp_time+a[i].BT;
                a[i].TAT=Cmp_time-a[i].AT;
                Total_TAT=Total_TAT+a[i].TAT;
            }
        }
        else
        {
            for(int i=0;i<n;i++)
            {
                int min=a[i].PR;
                for(int j=i+1;j<n;j++)
                {
                    if(min>a[j].PR && a[j].AT<=Cmp_time)
                    {
                        min=a[j].PR;
                          swap(&a[i].id,&a[j].id);
                          swap(&a[i].AT,&a[j].AT);
                          swap(&a[i].BT,&a[j].BT);
                           swap(&a[i].PR,&a[j].PR);
                    }
                }
                a[i].WT=Cmp_time-a[i].AT;
                Cmp_time=Cmp_time+a[i].BT;
                a[i].TAT=Cmp_time-a[i].AT;
                Total_WT=Total_WT+a[i].WT;
                Total_TAT=Total_TAT+a[i].TAT;
            }
        }
        Avg_WT=Total_WT/n;
        Avg_TAT=Total_TAT/n;
        printf("The process are\n");
        printf("ID WT TAT\n");
        for(int i=0;i<n;i++)
        {
            printf("%d\t%d\t%d\n",a[i].id,a[i].WT,a[i].TAT);
        }
        printf("Avg waiting time is: %f\n",Avg_WT);
        printf("Avg turn around time is: %f",Avg_TAT);
        return 0;
    }
```

OUTPUT



```
Enter the number of process
5
Enter the Arrival time , Burst time and priority of the process
AT BT PR
3
8
0
4
2
1
4
4
3
5
1
4
2
6
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
```