```
/*
 * Round Robin Scheduling Program in C
 */

#include<stdio.h>

int main()
{
    //Input no of processed
    int  n;
    printf("Enter Total Number of Processes:");
    scanf("%d", &n);
    int wait_time = 0, ta_time = 0, arr_time[n], burst_time[n], temp_burst_time[n];
    int x = n;

    //Input details of processes
    for(int i = 0; i < n; i++)
    {
        printf("Enter Details of Process %d \n", i + 1);
        printf("Arrival Time:  ");
        scanf("%d", &arr_time[i]);
        printf("Burst Time:   ");
        scanf("%d", &burst_time[i]);
        temp_burst_time[i] = burst_time[i];
    }
```

```
//Input time slot
int time_slot;
printf("Enter Time Slot:");
scanf("%d", &time_slot);

//Total indicates total time
//counter indicates which process is executed
int total = 0,  counter = 0,i;
printf("Process ID       Burst Time       Turnaround Time      Waiting Time\n");
for(total=0, i = 0; x!=0; )
{
   // define the conditions
   if(temp_burst_time[i] <= time_slot && temp_burst_time[i] > 0)
   {
      total = total + temp_burst_time[i];
      temp_burst_time[i] = 0;
      counter=1;
   }
   else if(temp_burst_time[i] > 0)
   {
      temp_burst_time[i] = temp_burst_time[i] - time_slot;
      total  += time_slot;
   }
   if(temp_burst_time[i]==0 && counter==1)
   {
      x--; //decrement the process no.
```

```
            printf("\nProcess No %d  \t\t %d\t\t\t\t %d\t\t\t %d", i+1, burst_time[i],
                  total-arr_time[i], total-arr_time[i]-burst_time[i]);
            wait_time = wait_time+total-arr_time[i]-burst_time[i];
            ta_time += total -arr_time[i];
            counter =0;
        }
        if(i==n-1)
        {
            i=0;
        }
        else if(arr_time[i+1]<=total)
        {
            i++;
        }
        else
        {
            i=0;
        }
    }
    float average_wait_time = wait_time * 1.0 / n;
    float average_turnaround_time = ta_time * 1.0 / n;
    printf("\nAverage Waiting Time:%f", average_wait_time);
    printf("\nAvg Turnaround Time:%f", average_turnaround_time);
    return 0;
}
```