**ROUND ROBIN SCHEDULER**

****

**SUBMITTED TO:-**

**Ms.Shaina Gupta**

**34-204-CH13**

**SUBMITTED BY:-**

**V.Pavan Kumar Reddy**

**K17UG**

**Roll no:46**

**RoundRobin:-**

This type of scheduling algorithm is basically designed for a time-sharing system. It is similar to FCFS with preemption added. Round-Robin Scheduling is also called as time-slicing scheduling and it is a preemptive version based on a clock. That is a clock interrupt is generated at periodic intervals usually 10-100ms. When the interrupt occurs, the currently running process is placed in the ready queue and the next ready job is selected on a First-come, First-serve basis. This process is known as time-slicing because each process is given a slice of time before being preempted.

One of the following happens:

* The process may have a CPU burst of less than the time quantum or
* CPU burst of currently executing process be longer than the time quantum. In this case, the context switch occurs the process is put at the tail of the ready queue.

In round-robin scheduling, the principal design issue is the length of the time quantum or time-slice to be used. If the quantum is very short, then short processes will move quickly.  
  
[​IMG]

**Advantages:-**

* Round-robin is effective in a general-purpose, time-sharing system or transaction-processing system.
* Fair treatment for all the processes.
* Overhead on the processor is low.

**Disadvantages:-**

* Care must be taken in choosing a quantum value.
* Processing overhead is there in handling clock interrupt.
* Throughput is low if time quantum is too small.

**Application:-**

* 1. In sports tournaments and other games, round-robin scheduling arranges to have all teams or players take turns playing each other, with the winner emerging from the succession of events.

**Round Robin Algorithm:-**

**Step 1**:- Start. //starts the program

**Step 2**:- Declare Variables Int i, j, n, total, qt, count,temp1[10],p[10]={1,2,3,4,5,6,7,8,9,10},

at[10],bt[10],wt[10],tat[10],temp,tat\_time=0,wait\_time;float wt\_avg=0,tat\_avg=0;

**Step 4**:- Read value n ,n=limit //reads value of n

if (n>0)

go to step 5;

else

go to step 10;

**Step 5**:- Read values at[10] and bt[10] for all process && temp1[i]=bt[i].

**Step 6**:- Read value qt (quantum time). //reads quantum time value

**Step 7**:- Repeat upto given condition get false

for(total=0,i=0;limit!=0;) //continues upto condition gets false

if(temp1[i]<=qt && temp1[i]>0)

total=total+temp1[i];

temp1[i]=0;

count=1;

else

temp1[i]=temp1[i]-qt;

total+=qt;

if(temp1[i]==0 && count==1)

limit--;

wt[i]=total-at[i]-bt[i];

tat[i]=total-at[i];

wait\_time=wait\_time+total-(at[i]+bt[i]);

tat\_time=tat\_time+total-at[i];

if(i==n-1)

i=0;

else if(at[i+1] <=total)

i++;

else

i=0;

printf(“ p[%d] %d %d %d %d”,i+1,at[i],bt[i],wt[i],tat[i]); //prints values

**Step 8**:- calculates the average of waiting time and turn around time

wt\_avg=(float)wait\_time/n;

tat\_avg=(float)tat\_time/n;

printf(“average waiting time is : %f”,wt\_avg); //prints average of waiting time

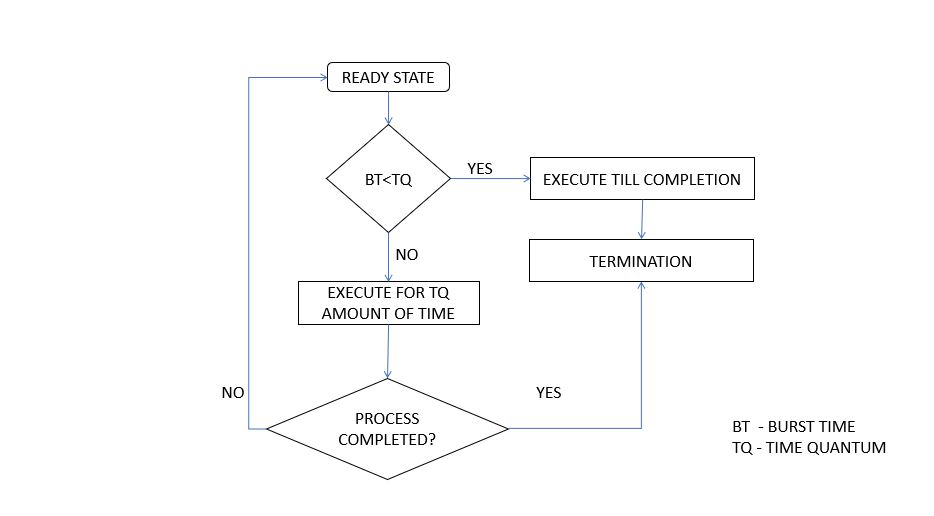
printf(“average turn around time is :%f”,tat\_avg); //prints average of turn around

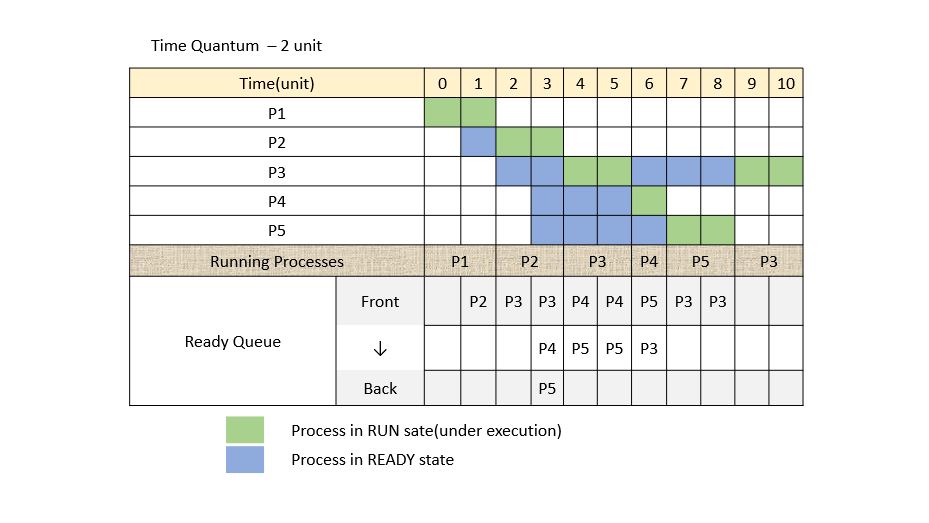
**Step 9**:- Stop

**Step 10**:- print below command

(a)printf(“error !! Scheduling can’t be done with 0 processes”);

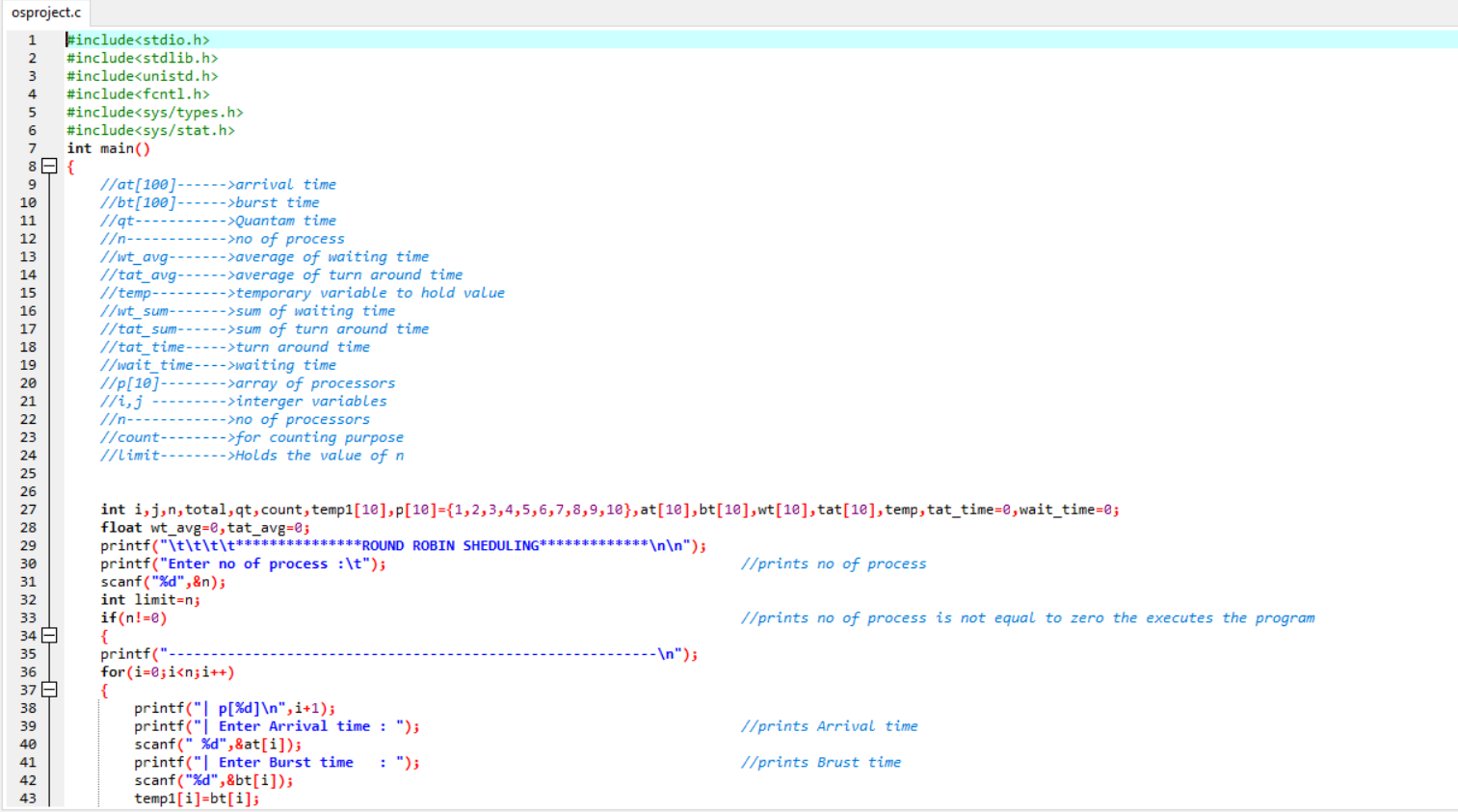
**Flow Chart:-**

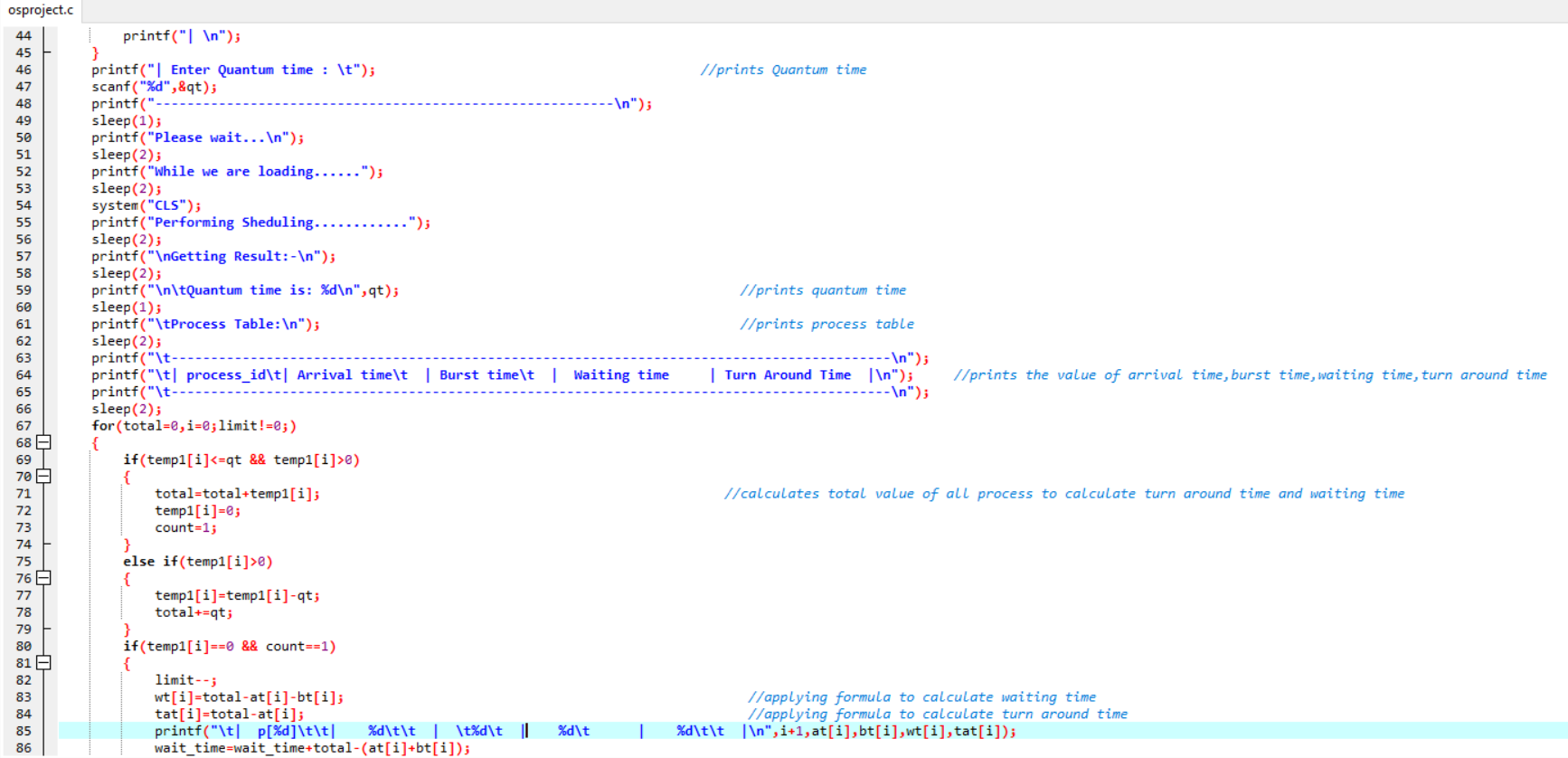
****

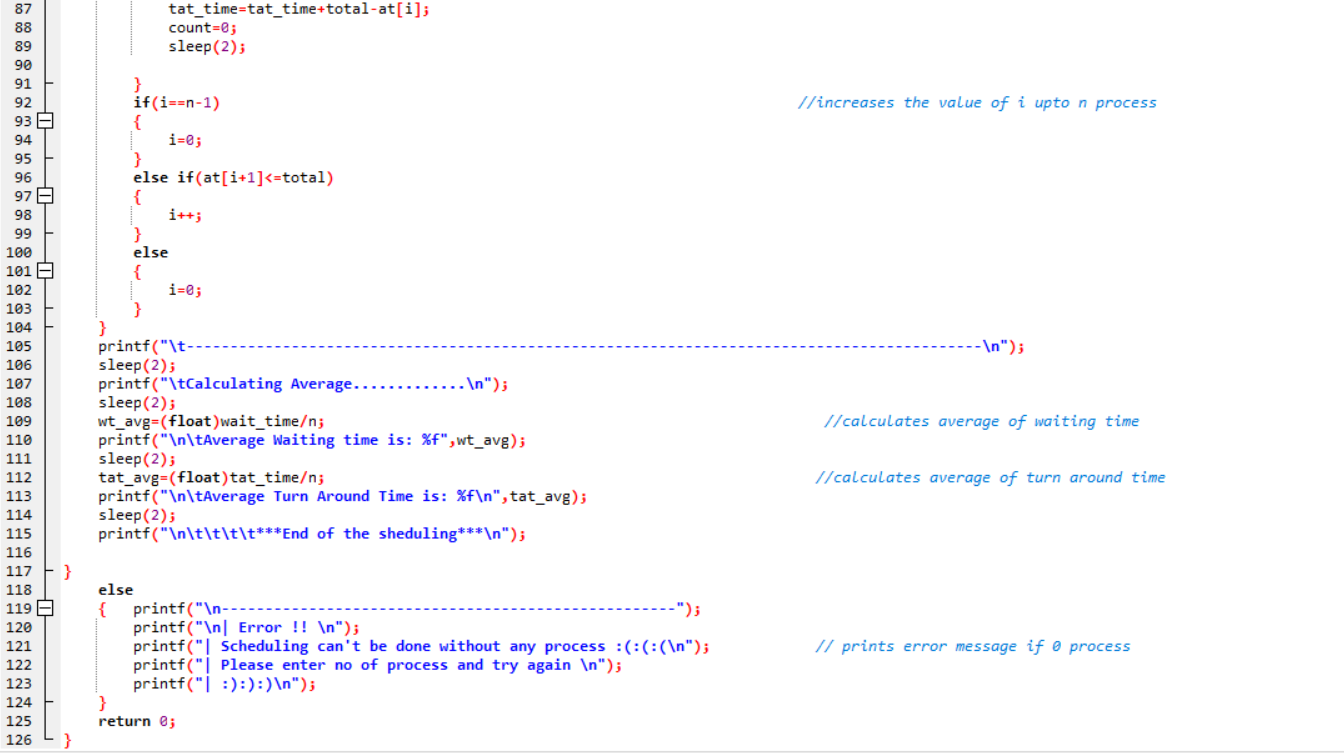


Q.Design a Scheduler that can schedule the processes arriving system at periodical intervals. Every Process is assigned with a fixed time slice t milliseconds. If it is not able to complete its execution within the assigned time quantum, then an automated timer generates an interrupt. The Scheduler will select the next process in the queue and dispatcher dispatches the process to the processor for execution. Compute the total time for which process were in the queue waiting for the processor. Take the input CPU burst, arrival time and quantum time from the user.

Sol:-



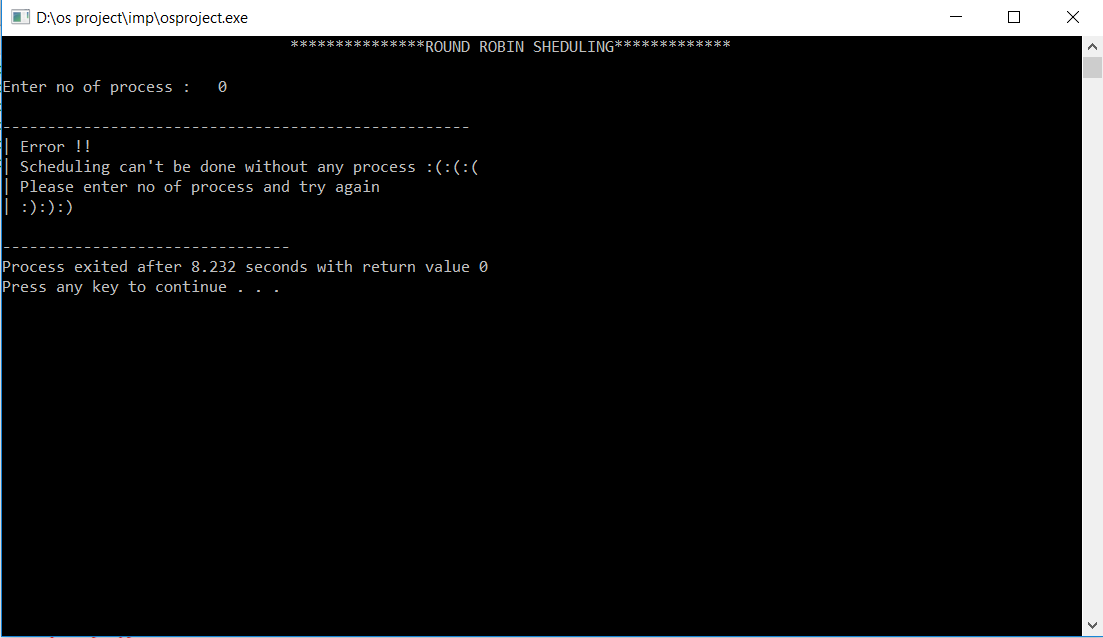




Output:-

Case 1:

If we enter 0(zero) process then an error will be raised by saying scheduling can’t be done without process….



Case 2:-

If we enter n process then it will ask arrival time and burst time for all n process. At last, it will ask quantum time for scheduling and after giving quantum it will process your scheduling. Then it will show you the sequence of process processed in process table and it will give an average of waiting time and turn around time.

