SAVEETHA UNIVERSITY

Declared U/S 3 of UGC Act 1956

A MICRO PROJECT REPORT ON

"Program In 'C' For Round Robin Scheduling Algorithm."

SUBMITTED TO

SAVEETHA SCHOOL OF ENFINEERING

BY STUDENTS OF SECOND YEAR FROM COMPUTER SCIENCE ENGINEERING

(name of students)

UNDER THE GUIDENCE OF

(LECTURER IN COMPUTER DEPARTMENT)

SAVEETHA UNIVERSITY

Declared U/S 3 of UGC Act 1956

CERTIFICATE

This is to certify that, the project report entitled

"Program In 'C' For Round Robin Scheduling Algorithm"

Roll. No	Name

ACKNOWLEDGEMENT

I take this opportunity to express deep sense of gratitude and sincere thanks for the invaluable guidance that I have received at the worthy hands of my guide **Guide Name**

I express my sincere thanks to our H.O.D. **Name** for permitting to do this project and also to the entire staff member who have helped me directly or indirectly.

(PART B)

MICRO-PROJECT REPORT

0.1 Rationale :

A round robin is an arrangement of choosing all elements in a group equally in some rational order, usually from the top to the bottom of a list and then starting again at the top of the list and so on. Round Robin (RR) scheduling algorithm is widely used scheduling algorithm in multitasking. It ensures fairness and starvation free execution of processes. Choosing the time quantum in RR is very crucial as small time slice results in large number of context switches and large time quantum increases the response time. Experiment analysis reveals that the proposed algorithm produces better average turnaround time, average waiting time and fewer number of context switches than existing algorithms.

0.2 Course Outcomes:

- Install operating system and configure it.
- Use operating system tools to perform various functions.
- Execute process commands for performing process management operations.
- Apply scheduling algorithms to calculate turnaround time and average waiting time.

0.3 Litreture Review:

Round Robin scheduling algorithm is one of the most popular scheduling algorithm which can actually be implemented in most of the operating systems. This is the preemptive version of first come first serve scheduling. The Algorithm focuses on Time Sharing. In this algorithm, every process gets executed in a cyclic way. Round Robin is a CPU scheduling algorithm where each process is assigned a fixed timeslot in a cyclic way. It is simple, easy to implement. And starvationfree as all processes get fair share of CPU. One of the most commonly used technique in CPU scheduling as a core. Itis pre-emptive as processes are assigned CPU only for a fixed slice of time at the most. The disadvantage of it is more overhead of context switching. Each process gets equal priority and fair allocation of CPU. Round Robin scheduling algorithm enables the Context switching method to save the states of preempted processes. It is easily implementable on the system because round robin scheduling in os doesn't depend upon burst time.

0.4 Actual Procedure Followed:

Description –

- The Round Robin (RR) scheduling algorithm is specially used for time sharing systems.It is quite same like quite same like FCFS,only difference is that in Round Robin algorithm preamption is added to switch from one process to another process.

- A small unit of time is called as time quantum.It is also known as time slice.A time quantum is generally from 10 to 100 miliseconds.In this method a ready queue is considered and treated as a circular queue.

- The CPU scheduler takes round along the ready queue and allocates CPU to each an every process for a time interval of up to 1 time quantum.

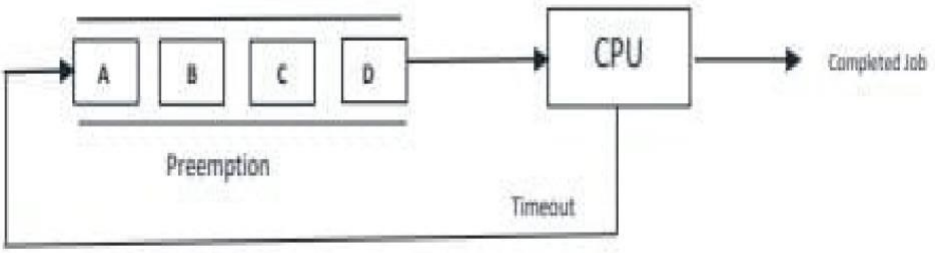
- For implementing round robin scheduling,the processes are kept in FIFO (First In First Out) order.The new processes are added to the tail i.e the last position of ready queue.

- The scheduler picks one job and assigns it to CPU;CPU executes it and bursts for a time quantum.In this case there are possibilities.

- If time quantum of processer is greater than process burst time then process itself releases CPU,otherwise interrupt interrupts the CPU.then CPU stops the execution and the process is shifted to tail of the ready process queue.after this,CPU scheduler selects next job for execution.

- The average waiting time in case of RR algorithm is genrally longer.

- To implement RR scheduling we keep ready queue as FIFO queue of processes. New processes are added to the tail of the ready queue. The average waiting time under the RR policy however is often quite long.



Example of round robin scheduling algorithm:

Assume there are 5 processes with process ID and burst time given below

PID	Burst Time
P1	6
P2	5
P3	2
P4	3
P5	7

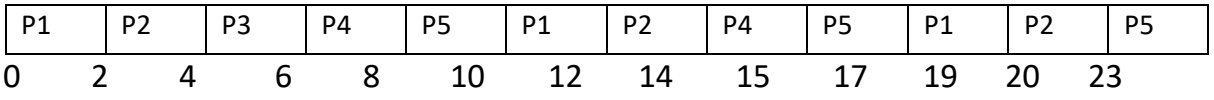
- Time quantum=2.
- Assume that all process arrives at 0.

Now, We will calculate average waiting time for this process to complete.

Solution :

We can represent execution of above processes using GANTT chart as shown below –

Gantt Chart :



Round Robin Example Gantt Chart -1

Explanation :

- First p1 process is picked from the ready queue and executes for 2 per unit time (timeslice = 2). If arrival time is not available, it behaves like FCFS with time slice.
- After P2 is executed for 2 per unit time, P3 is picked up from the ready queue. SinceP3 burst time is 2 so it will finish the process execution at once.
- Like P1 & P2 process execution, P4 and p5 will execute 2 time slices and then again itwill start from P1 same as above.

Waiting time = Turn Around Time - Burst Time

P1 = 19 – 6 = 13

P2 = 20 – 5 = 15

P3 = 6 – 2 = 4

P4 = 15 –3 = 12

P5 = 23 –7 = 16

Average waiting time = (13+15+4+12+16) / 5 = 12.

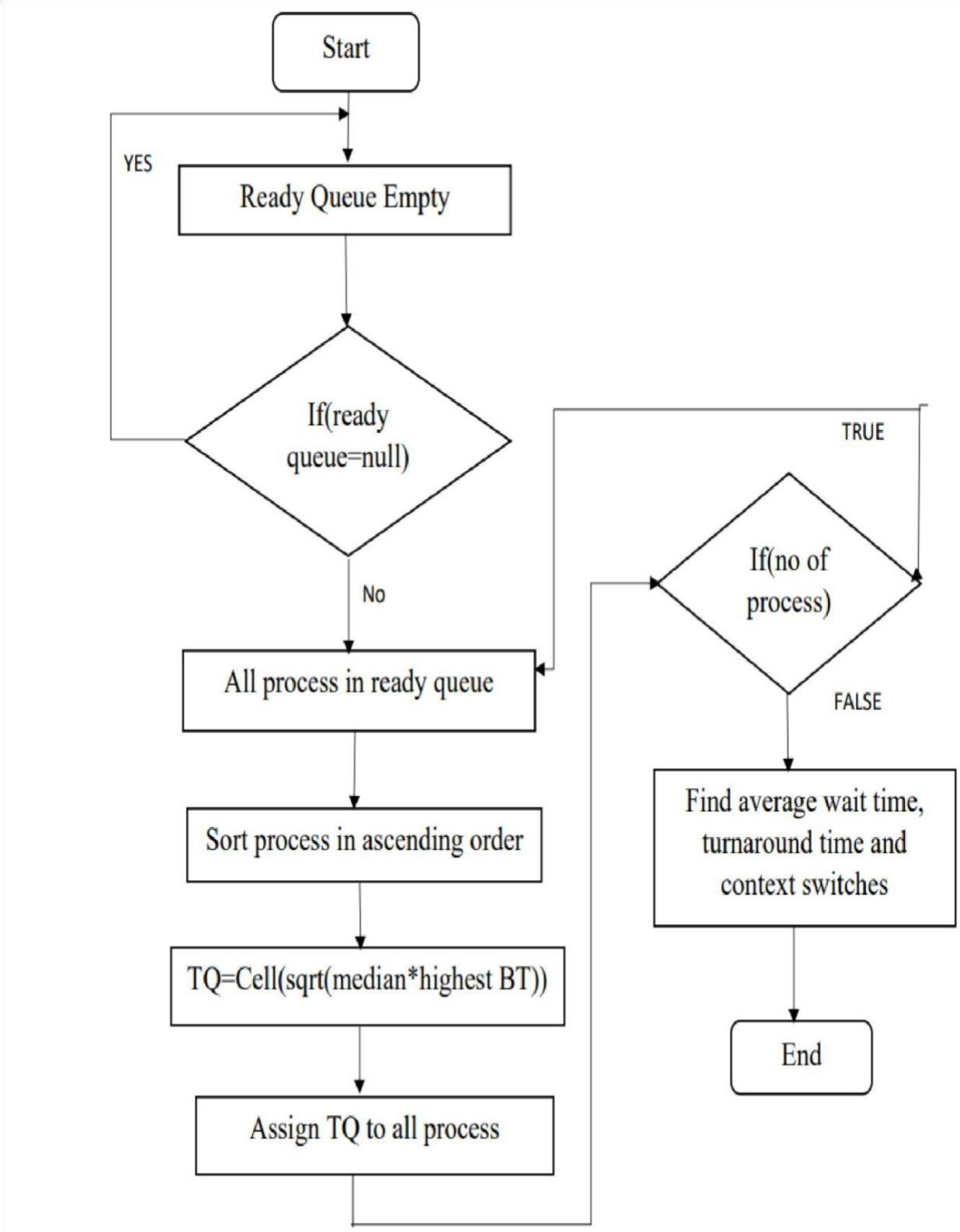
Advantages of Round Robin Algorithm

- No issues of starvation or convoy effect.
- Every job gets a fair allocation of CPU.
- No priority scheduling is involved.
- Total number of processes on the run queue helps assume the worst-case response time for a process.
- Doesn't depend on burst time and is easily implementable.

Disadvantages of Round Robin Algorithm

- Low slicing time reduces processor output.
- Spends more time on context switching.
- Performance depends on time quantum.
- Decreases comprehension.
- Higher context switching overhead due to lower time quantum.

Flowchart:


Start
YES
Ready Queue Empty
If(ready queue=null)
No
All process in ready queue
Sort process in ascending order
TQ=Cell(sqrt(median*highest BT))
Assign TQ to all process
TRUE
If(no of process)
FALSE
Find average wait time, turnaround time and context switches
End

C Program:

```
#include<stdio.h>
#include<conio.h>

void main()
{
   // initlialize the variable name
   int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10], bt[10], temp[10];
   float avg_wt, avg_tat;
   printf(" Total number of process in the system: ");
   scanf("%d", &NOP);
   y = NOP; // Assign the number of process to variable y

// Use for loop to enter the details of the process like Arrival time and the Burst Time
for(i=0; i<NOP; i++)
{
printf("\n Enter the Arrival and Burst time of the Process[%d]\n", i+1);
printf(" Arrival time is: \t");  // Accept arrival time
scanf("%d", &at[i]);
printf(" \nBurst time is: \t"); // Accept the Burst time
scanf("%d", &bt[i]);
temp[i] = bt[i]; // store the burst time in temp array
}
// Accept the Time qunat
printf("Enter the Time Quantum for the process: \t");
scanf("%d", &quant);
// Display the process No, burst time, Turn Around Time and the waiting time
printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time ");
for(sum=0, i = 0; y!=0; )
{
if(temp[i] <= quant && temp[i] > 0) // define the conditions
{
   sum = sum + temp[i];
   temp[i] = 0;
   count=1;
```

```
 }
    else if(temp[i] > 0)
    {
        temp[i] = temp[i] - quant;
        sum = sum + quant;
    }
    if(temp[i]==0 && count==1)
    {
        y--; //decrement the process no.
        printf("\nProcess No[%d] \t\t %d\t\t\t\t %d\t\t\t %d", i+1, bt[i], sum-at[i], sum-at[i]-
bt[i]);
        wt = wt+sum-at[i]-bt[i];
        tat = tat+sum-at[i];
        count =0;
    }
    if(i==NOP-1)
    {
        i=0;
    }
    else if(at[i+1]<=sum)
    {
        i++;
    }
    else
    {
        i=0;
   }
}
// represents the average waiting time and Turn Around time
```

5.0 Output of Micro-project:

```
Total number of process in the system: 5

Enter the Arrival and Burst time of the Process[1]
Arrival time is:       0

Burst time is:  6

Enter the Arrival and Burst time of the Process[2]
Arrival time is:       0

Burst time is:  5

Enter the Arrival and Burst time of the Process[3]
Arrival time is:       0

Burst time is:  2

Enter the Arrival and Burst time of the Process[4]
Arrival time is:       0

Burst time is:  3

Enter the Arrival and Burst time of the Process[5]
Arrival time is:       0

Burst time is:  7
Enter the Time Quantum for the process:         2

 Process No             Burst Time                    TAT                   Waiting Time
Process No[3]           2                             6                     4
Process No[4]           3                             15                    12
Process No[1]           6                             19                    13
Process No[2]           5                             20                    15
Process No[5]           7                             23                    16
 Average Turn Around Time:      12.000000
 Average Waiting Time:  16.600000

...Program finished with exit code 0
Press ENTER to exit console.
```

5.0 Actual Resources Used:

Sr.no	Resource	Specifications
1.	Computer System	Processor Intel Core I3, RAM 8GB, 1TB.HDD
2.	Operating System	Windows 10
3.	Software	Turbo C++, Ubantu
4.	Microsoft Word	Microsoft office 2010

7.0 Skill Developed :

- ✓ From this we learnt about CPU scheduling in operating system.
- ✓ Also, we got to know about finding average waiting time and average turnaround time each process executing in operating system.
- ✓ Round Robin scheduling is an algorithm mainly used by operating systems and applications that serve multiple clients that request to use resources.

8.0 Applications of this Micro - Project :

Round-Robin is a useful tool for having your team generate ideas, without being influenced unduly by others in the group. This method also used to ensures that everyone on your team gets an equal say in the ideas that you generate. We can use either the written and verbal variations of this technique.

9.0 Area of Future Improvement :

We can improve performance for round robin by arranging processes in an increasing order according to the burst time, then calculating the mean of the burst time as quantum time finally calculating turnaround time and waiting time, thus to obtain better results comparing with the standard RR.