Mawlana Bhashani Science and Technology University

Lab-Report

Report No:10

Course code:ICT-3110

Course title:Operating Systems Lab

Date of Performance:10-09-2020

Date of Submission:16-09-2020

Submitted by

Name:Meraz Ahmed

ID:IT-18005

3rd year 1st semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Experiment No: 10

Experiment Name: Implementation of Round Robin Scheduling Algorithm.

Round Robin Scheduling Algorithm:

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. A simple way to think of round robin is that it is about "taking turns." Used as an adjective, round robin becomes "round-robin."

Algorithm:

1. The queue structure in ready queue is of First In First Out (FIFO) type.

2. A fixed time is allotted to every process that arrives in the queue. This fixed time is known as time slice or time quantum.

3. The first process that arrives is selected and sent to the processor for execution. If it is not able to complete its execution within the time quantum provided, then an interrupt is generated using an automated timer.

4. The process is then stopped and is sent back at the end of the queue. However, the state is saved and context is thereby stored in memory. This helps the process to resume from the point where it was interrupted.

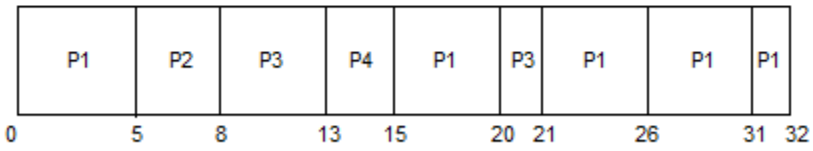
5. The scheduler selects another process from the ready queue and dispatches it to the processor for its execution. It is executed until the time Quantum does not exceed.

6. The same steps are repeated until all the process are finished.

The round robin algorithm is simple and the overhead in decision making is very low. It is the best scheduling algorithm for achieving better and evenly distributed response time.

PROCESS	BURST TIME
P1	21
P2	3
P3	6
P4	2

The GANTT chart for round robin scheduling will be,



The average waiting time will be, 11 ms.

PsudoCode:

```
#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;
}
```

Output:

```
Enter Total Process:     4
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
4
Enter Arrival Time and Burst Time for Process Process Number 3 :2
2
Enter Arrival Time and Burst Time for Process Process Number 4 :3
6
Enter Time Quantum:     4


Process |Turnaround Time|Waiting Time

P[2]    |       7       |       3
P[3]    |       8       |       6
P[4]    |       17      |       11
P[1]    |       21      |       12

Average Waiting Time= 8.000000
Avg Turnaround Time = 13.250000
Process returned 0 (0x0)   execution time : 90.581 s
Press any key to continue.
```