Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 10

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance:

Date of Submission:

Submitted by

Name: Ashikur Rahman Miran

ID:IT-18014

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.

Objectives:

i) What is Round Robin scheduling algorithm?
ii) How to implement Round Robin scheduling algorithm.

Theory:

Round robin is the most widely used process scheduling algorithm .The basic strategy for round robin scheduling is that if there are n process ,each of the process will receive 1/n CPU Execution Time. Each process is allotted a time quantum, for which it is executed. The incoming processes are kept in a ready list while another one is executing. If the time quanta allotted for a process is over, then that process is moved to ready and the next process in the ready list is executed for the allotted time quantum.

Implementation:

Time Quantum: 1

Process	Arrival time	Burst time
P1	0	4
P2	0	1
P3	0	8
P4	0	1

Grant chart:

P1	P2	P3	P4	P1	P3	P1	P3	P1	P3	P3	P3	P3	P3

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

Process	Arrival time(At)	Burst time(Bt)	Waiting time	Total turnaround time
P1	0	4	8	9
P2	0	1	9	2
P3	0	8	10	14
P4	0	1	11	4

Source code:

```
#include<stdio.h>
int main()
{
	int n,i,k,x=0,s=0,r=0,q=0,bt[100],rem_bt[100],t[100];
	float m,p=0;
	printf("Enter the number of process: ");
	scanf("%d",&n);
	printf("Enter the Burst time: \n");
	for(i=0; i<n; i++)
	{
		printf("P%d: ",i+1);
		scanf("%d",&bt[i]);
		rem_bt[i]=bt[i];
	}
	printf("Enter Time quantum: ");
	scanf("%d",&q);
	printf("After Round Robin sheduling: ");
	for(i=0; i<n; i++)
	{
		if(x<bt[i])
		{
			x=bt[i];
		}
	}
	k=x/q;
	while(s<=k)
	{
		for(i=0; i<n; i++)
		{
			if(bt[i]>0)
```

```
                {
                    if(bt[i]>q)
                    {
                        r=r+q;
                        bt[i]=bt[i]-q;
                        printf("P%d ",i+1);
                    }
                    else
                    {
                        r=r+bt[i];
                        bt[i]=bt[i]-q;
                        printf("P%d ",i+1);
                        t[i]=r;
                    }
                }
            }
            s++;
        }
        printf("\n\nProcess\tBurstTime\tWaitingTime\tTurnAroundTime\n");
        for(i=0; i<n; i++)
        {
            printf("%d\t%d\t\t%d\t\t%d\t\t\n",i+1,rem_bt[i],x,t[i]);
            x=x+q;
        }
        m=x/n;
        printf("\nAverage waiting time=%f: ",m);
        printf("\nAverage turn around time: ");
        for(i=0; i<n; i++)
        p=p+t[i];
        p=p/n;
        printf("%f: ",p);
        printf("\n");
        return 0;
}
```

Output:

```
"D:\programming\c _ c++ programming\algorithm\Round robin scheduling algo.exe"
Enter the number of process: 4
Enter the Burst time:
P1: 4
P2: 1
P3: 8
P4: 1
Enter Time quantum: 1
After Round Robin sheduling: P1 P2 P3 P4 P1 P3 P1 P3 P1 P3 P3 P3 P3 P3

Process BurstTime       WaitingTime     TurnAroundTime
P1      4               8               9
P2      1               9               2
P3      8               10              14
P4      1               11              4

Average waiting time=3.000000:
Average turn around time: 7.250000:
```

Conclusion:

In this lab I learn how to implement Round robin scheduling algorithm and also run the code and shows the output and output is expected.