



Mawlana Bhashani Science and Technology University

Lab-Report

Lab Report No: 10

Lab Report Name: Implementation of Round Robin Scheduling algorithm.

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance: 18-09-2020

Date of Submission:

Submitted by

Name: Md Masud Rana

ID: IT-18002

3rd year 1st semester

Session: 2017-18

Dept. of ICT.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT,

MBSTU.

Experiment No: 10

Experiment Name: Implementation of 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 quanta, for which its 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 quanta.

Need to calculate-

Completion Time is the time required by the process to complete its execution

Turnaround Time is the time interval between the submission of a process and its completion.

Turnaround Time = completion of a process – submission of a process

Waiting Time is the difference between turnaround time and burst time

Waiting Time = turnaround time – burst time.

Code-

```

#include<stdio.h>
int main()
{
    int i, n, total = 0, x, counter = 0, qt;
    int wt = 0, tt = 0, at[10], bt[10], temp[10];
    float average_wait_time, average_turnaround_time;
    printf("\nEnter Total Number of Processes : ");
    scanf("%d", &n);
    x = n;
    for(i = 0; i < n; i++)
    {
        printf("\nEnter Process of-[%d]\n", i + 1);
        printf("Arrival Time:\t");
        scanf("%d", &at[i]);
        printf("Burst Time:\t");
        scanf("%d", &bt[i]);
        temp[i] = bt[i];
    }
    printf("\nEnter Time Quantum:\t");
    scanf("%d", &qt);
    printf("\nProcess ID\t\tBurst Time\t\tarrival time\t\tTurnaround Time\t\tWait Time\n");
    for(total = 0, i = 0; x != 0;)
    {
        if(temp[i] <= qt && temp[i] > 0)
        {
            total = total + temp[i];
            temp[i] = 0;
            counter = 1;
        }
        else if(temp[i] > 0)
        {
            temp[i] = temp[i] - qt;
            total = total + qt;
        }
        if(temp[i] == 0 && counter == 1)
        {
            x--;
            printf("\nProcess%d\t\t\t%d\t\t\t%d\t\t\t%d\t\t\t%d", i + 1, bt[i], at[i], total - at[i], total - at[i] - bt[i]);
            wt = wt + total - at[i] - bt[i];
            tt = tt + total - at[i];
            counter = 0;
        }
    }
}

```

```

        counter = 0;
    }
    if(i == n - 1)
    {
        i = 0;
    }
    else if(at[i + 1] <= total)
    {
        i++;
    }
    else
    {
        i = 0;
    }
}
average_wait_time = wt * 1.0 / n;
average_turnaround_time = tt * 1.0 / n;
printf("\n\nAverage Wait Time:\t%f", average_wait_time);
printf("\n\nAvg Turnaround Time:\t%f\n", average_turnaround_time);
return 0;
}

```

Output:

```

masud@masud-VirtualBox: ~/program
File Edit View Search Terminal Help
masud@masud-VirtualBox:~/program$ ./robin.out

Enter Total Number of Processes : 4

Enter Process of-[1]
Arrival Time: 0
Burst Time: 16

Enter Process of-[2]
Arrival Time: 3
Burst Time: 9

Enter Process of-[3]
Arrival Time: 4
Burst Time: 12

Enter Process of-[4]
Arrival Time: 8
Burst Time: 6

Enter Time Quantum: 2

Process ID      Burst Time      arrival time      Turnaround Time      Wait Time
Process4        6              8                18                   12
Process2        9              3                32                   23
Process3        12             4                37                   25
Process1        16             0                43                   27

Average Wait Time: 21.750000
Avg Turnaround Time: 32.500000
masud@masud-VirtualBox:~/program$

```

Discussion: In this lab we have implemented round robin scheduling algorithm with c programming language. This algorithm mainly depends on the time quantum and preemptive in nature. This lab helps to realize about round robin scheduling algorithm.