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:
Date of Submission: 12/09/2020

Submitted By

Name: Ali Ashadullah Arif
ID:IT-18031
3rd Year 1st Semester
Session: 2017-2018
Dept. of ICT
MBSTU.

Submitted To

Nazrul Islam
Assistant Professor
Dept. of ICT
MBSTU.

Lab Report No: 10

Name of the Lab Report: Implementation of Round Robin Scheduling Algorithm.

Objective: Round Robin Scheduling algorithm Definition & executable code in c are followed.

1. What is Round Robin Scheduling algorithm?

Answer: Round robin scheduling is the preemptive scheduling in which every process get executed in a cyclic way, i.e. in this a particular time slice is allotted to each process which is known as time quantum. Every process, which is present in the queue for processing, CPU is assigned to that process for that time quantum. Now, if the execution of the process gets completed in that time quantum, then the process will get terminate otherwise the process will again go to the ready queue, and the previous process will wait for the turn to complete its execution.

The scheduling drives its name from the principle which is known as a round robin in which every person takes an equal share of anything they have in turn. We make use of round robin scheduling algorithm in a time-sharing system. It is generally used by those os which has multiple clients to make use of resources.

2. How to implemented in C?

Answer:

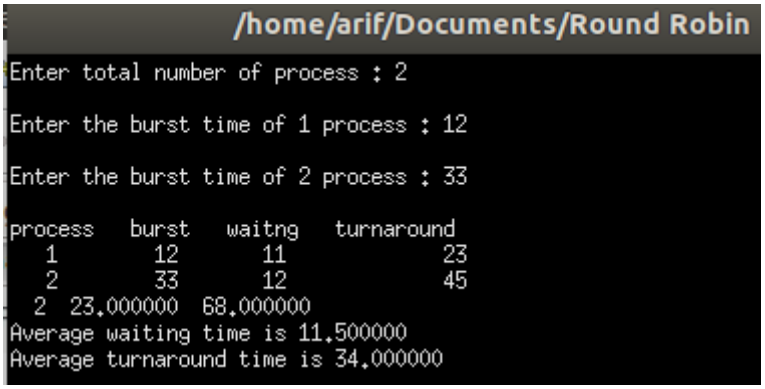
Source Code:

```
#include<stdio.h>
int main()
{
        int BuT[100],WaT[100],TaT[100],b[100];
        int i,n,time,count=0;
        float totalwt=0,totalTT=0,avgwt,avgtt;

        printf("Enter total number of process : ");
        scanf("%d",&n);
        for(i=0; i<n; i++) {
```

```
        printf("\nEnter the burst time of %d process : ",i+1);
        scanf("%d",&BuT[i]);
        b[i] = BuT[i];
    }
    i=0;
    for(time=0; count!=n; time++) {
        while(BuT[i] == 0) {
            i=(i+1)%n;
        }
        BuT[i]--;
        if(BuT[i]==0) {
            TaT[i] = time+1;
            count++;
        }
        i = (i+1)%n;
    }
    printf("\nprocess burst waitng turnaround ");
    for(i=0; i<n; i++) {
        WaT[i] = TaT[i] - b[i];
        printf("\n %d \t %d \t %d \t %d",i+1,b[i],WaT[i],TaT[i]);
        totalwt = totalwt + WaT[i];
        totalTT = totalTT + TaT[i];
    }
    printf("\n %d %f %f",n,totalwt,totalTT);
    avgwt = totalwt / n;
    avgtt = totalTT / n;
    printf("\nAverage waiting time is %f",avgwt);
    printf("\nAverage turnaround time is %f \n\n",avgtt);
    return 0;
}
```

Output:


/home/arif/Documents/Round Robin
Enter total number of process : 2
Enter the burst time of 1 process : 12
Enter the burst time of 2 process : 33
process burst waitng turnaround
1 12 11 23
2 33 12 45
2 23.000000 68.000000
Average waiting time is 11.500000
Average turnaround time is 34.000000