

**Mawlana Bhashani Science and Technology University**

**Lab-Report**

Report No:10

Course code: ICT-3110

Course title: Operating System Lab

Date of Performance:01/09/2020

Date of Submission:10/09/2020

**Submitted by Submitted To**

Name:Md Milon Ahmed

ID:IT-18034

3rd  year 1st semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

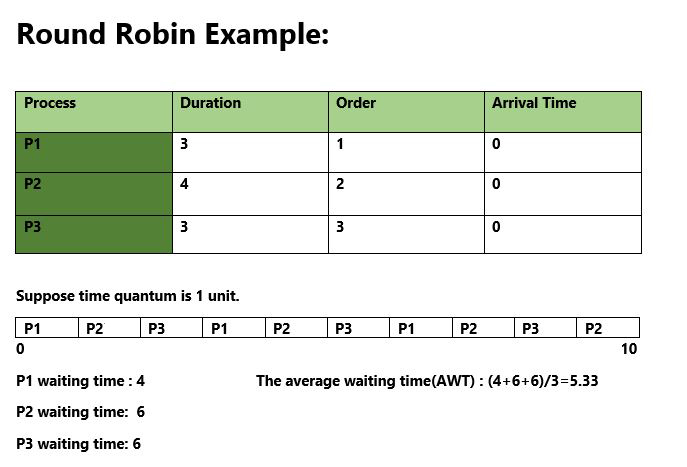
# Experiment No : 10

# Experiment Name : Implementation of Round Robin Scheduling Algorithm

# Theory:

The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turns. It is the oldest, simplest scheduling algorithm, which is mostly used for multitasking.

In Round-robin scheduling, each ready task runs turn by turn only in a cyclic queue for a limited time slice. This algorithm also offers starvation free execution of processes.



# Working Procedure:

Coding implementation with c ….

#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;

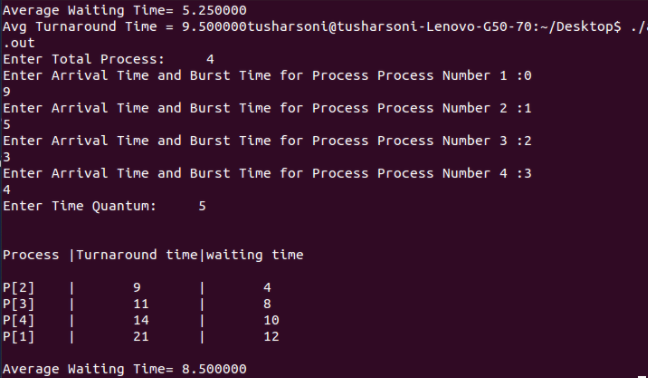
  }

  printf("\nAverage Waiting Time= %f\n",wait\_time\*1.0/n);

  printf("Avg Turnaround Time = %f",turnaround\_time\*1.0/n);

  return 0; }

Output:



# Discussion:

We learn characteristic and advantages of round robin algorithm

1.Round robin is a pre-emptive algorithm

2.The CPU is shifted to the next process after fixed interval time, which is called time quantum/time slice.

3.The process that is preempted is added to the end of the queue.

4.Round robin is a hybrid model which is clock-driven

5.Time slice should be minimum, which is assigned for a specific task that needs to be processed. However, it may differ OS to OS.

6.It is a real time algorithm which responds to the event within a specific time limit.

7.Round robin is one of the oldest, fairest, and easiest algorithm.

8.Widely used scheduling method in traditional OS.

9.It doesn't face the issues of starvation or convoy effect.

10.All the jobs get a fair allocation of CPU.

11.It deals with all process without any priority