Smt. Chandibai Himathmal Mansukhani College

Table of Contents

USCSP301: USCS303 – Operating System (OS)

Contents

USCSP301 – USCS303 : Operating system (OS) Practical – 03 .... **2**

Practical – 03 Round-robin (RR) scheduling algorithm 2

Practical Date : 28 July 2021 2

Practical Aim Implement RR scheduling algorithm in Java **2**

Algorithm 2

Solved Example 1 3

Gnatt Chart 4

Solved Example 2 5

Solved Example 3 6

Solved Example 4 7

Implementation 8

Input 10

Output 10

Sample Output – 01 11

Sample Output – 02 11

Sample Output – 03 12

Sample Output – 04 12

Smt. Chandibai Himathmal Mansukhani College

USCS303 – OS: Practical – 03: RR Scheduling Algorithm

• Round-robin (RR) scheduling algorithm is mainly designed for time-sharing systems.

• This algorithm is similar to FCFS scheduling, but in Round robin (RR) scheduling, preemption is added which enables the system to switch between processes.

• Round-robin (RR) scheduling algorithm is mainly designed for time-sharing systems.

• This algorithm is similar to FCFS scheduling, but in Round robin (RR) scheduling, preemption is added which enables the system to switch between processes.

• Round-robin scheduling algorithm is used to schedule process fairly each job a time slot or quantum and the interrupting the job if it is not completed by then the job come after the other job which is arrived in the quantum time that makes these scheduling fairly.

Aim
Implement RR scheduling algorithm in Java.

Algorithm
Step 1: Input the number of processes and time quanta or time slice required to be scheduled using RR, burst time for the process.

Step 2: Choose the first process in the ready queue, set a interrupt it after time quantum and dispatches it. Check if any other process request has arrived. If a process request arrives during the quantum time in which another process is executing, then add the new process to the Ready queue.

Step 3: After the quantum time has passed, check for any processes in the Ready queue. If the ready queue is empty then continue the current process. If the queue not empty and the current process is not complete, then add the current process to the end of the ready queue.

Step 4: Take the first process from the Ready queue ar executing it. Calculate the Turn Around Time and Waiting T each process using RR.

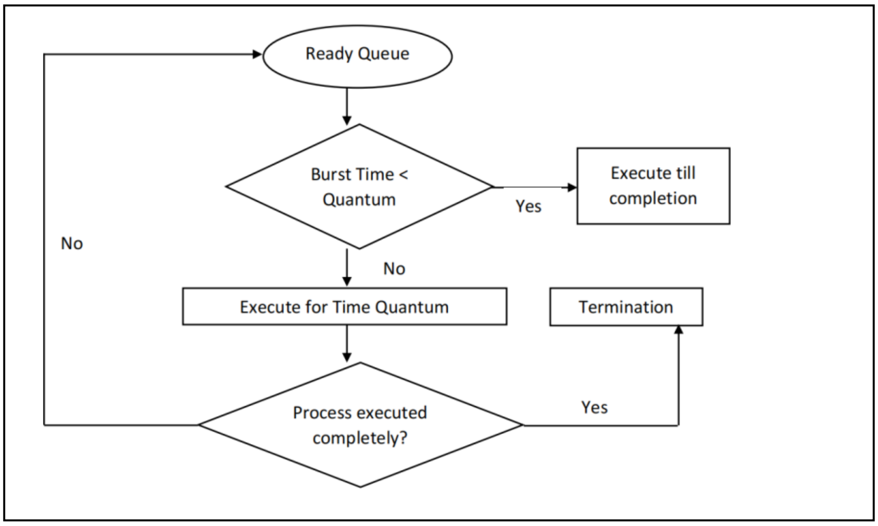
Step 5: Repeat all steps above from Step 2 to Step 4.

Step 6: If the process is complete and the ready queue is empty then the task is complete.

Step 7: Calculate the Average Waiting Time and Average Turn Around Time.

Step 8: Stop.

Flowchart:



Consider the following example containing three processes arriving at time t = 0 ms.

Process ID	Burst Time
P0	24
P1	3
P2	3

Assume time quanta = 4ms

Step 1: Consider the time quanta / time slice = 4 ms.

Step 2: Following shows the scheduling and execution of processes.

Step 2.1: P0 process arrives at 0 with 24 ms as the burst time which is greater than time quanta 4 ms. So P0 executes for 4 ms and goes in waiting queue.
System Time : 0
Process Scheduled : P0
Remaining Time: 24-4 = 20
Waiting Time : 0-0 =0
Turn Around Time : 0 + 4 = 4

Step 2.2: Next P1 process executes for 3 ms which is greater than quanta time. So P1 executes and gets terminated.

System Time:4
Process Scheduled: P0, P1

Remaining Time: 3 - 4 = -1 = 0

Waiting Time : 4 - 0 =4
Turn Around Time : 4 + 3 = 7

Step 2.3: Next P2 process executes for 3 ms which is greater than quanta time. So P2 executes and gets terminated.

System Time: 7

Process Scheduled: P0, P1, P2

Remaining Time: 3 - 4 = -1 = 0

Waiting Time : 7 - 0 = 7

Turn Around Time : 7 + 3 = 10

Step 2.4: Now PO turns comes again and it's the only process for execution so for 4 ms of quanta it gets executed.

System Time: 10

Process Scheduled: P0, P1, P2, PO

Finish Time: 20-4 = 16

Waiting Time : 0

Turn Around Time: 14

Step 2.5: Again, P0 continues to execute for next 4 ms. Waiting Time for P0 will be zero.

System Time : 10+ 4 = 14

Process Scheduled: PO, P1, P2, PO

Finish Time: 16-4-12

Waiting Time: 0

Turn Around Time 14+4=18

Step 2.6: PO continues to execute for next 4 ms.

System Time: 18
Process Scheduled: PO, P1, P2, PO, PO, PO
Finish Time: 12-4 = 8
Turn Around Time : 18 + 4 = 22

Step 2.7: PO continues to execute for next 4 ms.

System Time: 22

Process Scheduled: PO, P1, P2, PO, PO, PO, PO

Finish Time: 8-4 = 4

Turn Around Time: 22 + 4 = 26

Step 2.8: PO continues to execute for next 4 ms.

System Time: 26

Process Scheduled: PO, P1, P2, PO, PO, PO, PO, PO

Finish Time: 4-4 = 0

Turn Around Time: 26 + 4 = 30

Step 3: Calculate Average Waiting Time and Average Turn Around Time.
Average Waiting Time = (6+4+7)/3
= 17/3

= 5.666667

Average Turn Around Time = (30+7+10)/3

= 47/3

= 15.666667

Step 4 : After scheduling of all provided processes.

Process ID	Burst time	Turn around time (Finish time – Arrival time)	Waiting time (Turn Around time – Burst Time)
P0	24	30-0=30	30-24=6
P1	3	4+3=7	7-3=4
P2	3	7+3=10	10-3=7
Average		15.666667	5.666667

Step 5 : Stop

Gnatt Chart

P0	P1	P2	P0	P0	P0	P0	P0	
0	4	7	10	14	18	22	26	30

Example 2

Sample Output -2

Process ID	Burst Time
P0	2
P1	1
P2	6

Assume Time Slice = 1ms

Process ID	**Burst time**	**Turn around time**	**Waiting time**
		(Finish time – Arrival time)	**(Turn Around time – Burst Time)**
P0	2	(4 – 0) = 4	(4 – 2) = 2
P1	1	(2 – 0) = 2	(2 – 1) = 1
P2	6	(9 – 0) = 9	(9 – 6) = 3
Average		2.000000	5.000000

Gnatt Chart :-

P0	P1	P2	P0	P2	P2	P2	P2	P2	
0	1	2	3	4	5	6	7	8	9

Sample Output – 3

Process ID	Burst time
P0	7
P1	3
P2	2
P3	10
P4	8

Assume Time Slice = 3ms.

Process ID	**Burst time**	**Turn around time**	**Waiting time**
		(Finish time – Arrival time)	**(Turn Around time – Burst Time)**
P0	7	(24 – 0) = 24	(24 – 7) = 17
P1	3	(6 – 0) = 6	(6 – 3) = 3
P2	2	(8 – 0) = 8	(8 – 2) = 6
P3	10	(30 – 0) = 30	(30 – 10) = 20
P4	8	(29 – 0) = 29	(29 – 8) = 21
Average		13.400000	5.000000

Gnatt Chart

P0	P1	P2	P3	P4	P0	P3	P4	P0	P3	P4	P3

0 3 6 8 11 14 17 20 23 24 27 29 30

Smt. Chandibai Himathmal Mansukhani College

Implementation

Java Program :-

```
// Name: Vrushabh Gade

// Batch :  B2

// PRN :  2020016400926511

//Date :  28 July 2021

// Practical:- 3 (RR)

import java.util.Scanner;
class P3_RR_VG
{

            public static void main(String args[])

            {
                Scanner input = new Scanner(System.in);
                int i, j, k, q, sum = 0;
                System.out.print("Enter number of process: ");
                int n = input.nextInt();
                int burstTime[] = new int[n];
                int waitingTime[] = new int[n];
                int turnAroundTime[] = new int[n];
                int a[] = new int[n];
                System.out.println("Enter Burst Time of each process: ");
                for (i = 0; i < n; i++)

                {
                    System.out.print("Enter Burst Time for Process - P" + (i) +" : ");
                    burstTime[i] = input.nextInt();
                   a[i] = burstTime[i];
                }

                System.out.print("Enter Time quantum: ");
                q=input.nextInt();
                for (i = 0; i<n; i++)
                        waitingTime[i] = 0;
                int timer = 0; // Current time
// Keep traversing processes in round robin manner until all of them are not done.
do

                {
                      for (i = 0; i<n; i++)
```

```
                {
// If burst time of a process is greater than 0 then only need to process further
                 if (burstTime[i] > q)
                {
// Increase the value of ti.e. shows how much time a process has been processed
                timer += q;
// Decrease the burst time of current process by quantum
                burstTime[i] -= q;
                for (j = 0; j<n; j++)
                {
                if ((j!= i) && (burstTime[j] != 0))
                waitingTime[j] += q;
                }
                } // if ends
// If burst time is smaller than or equal to quantum. Last cycle for this process
                else
                {
// Increase the value of t i.e. shows how much time a process has been processed
                  timer += burstTime[i];
                  for (j = 0; j<n; j++)
                 {
                   if ((j != i) && (burstTime[j] != 0))
                    waitingTime[j] += burstTime[i];
                  }
// As the process gets fully executed make its remaining burst time = 0
                           burstTime[i] = 0;
                           } // else ends
                      }
                sum = 0;
                for (k = 0; k<n; k++)
                sum += burstTime[k];
                } while (sum != 0);
// calculating turnaround time by adding waiting Time + burst Time
                for (i = 0; i<n; i++)
                turnAroundTime[i] = waitingTime[i] + a[i];
                float total = 0;
                for (int x : waitingTime)
                {
                total += x;
                }
                float averageWaitingTime = total / n;
                total = 0;
                for (int y : turnAroundTime)
                {
                    total += y;
```

```
                }
                float averageTurnAroundTime = total / n;
// print on console the order of processes scheduled using Round-robin Algorithm
                System.out.println("RR Algorithm: ");
                System.out.format("%20s %20s %20s
%20s\n","ProcessId","BurstTime","Waiting Time","TurnAroundTime");
                for (i = 0; i<n; i++)
                {
                  System.out.format("%20s %20d %20d %20d\n","P"+(i),a[i],
                  waitingTime[i],turnAroundTime[i]);
                }
                System.out.format("%40s %20f
%20f\n","Average",averageWaitingTime,averageTurnAroundTime);
                }
}
```

Input

Enter number of process: 3

Enter Burst Time of each process:

Enter Burst Time for Process - P0 : 24

Enter Burst Time for Process - P1 : 3

Enter Burst Time for Process - P2 : 3

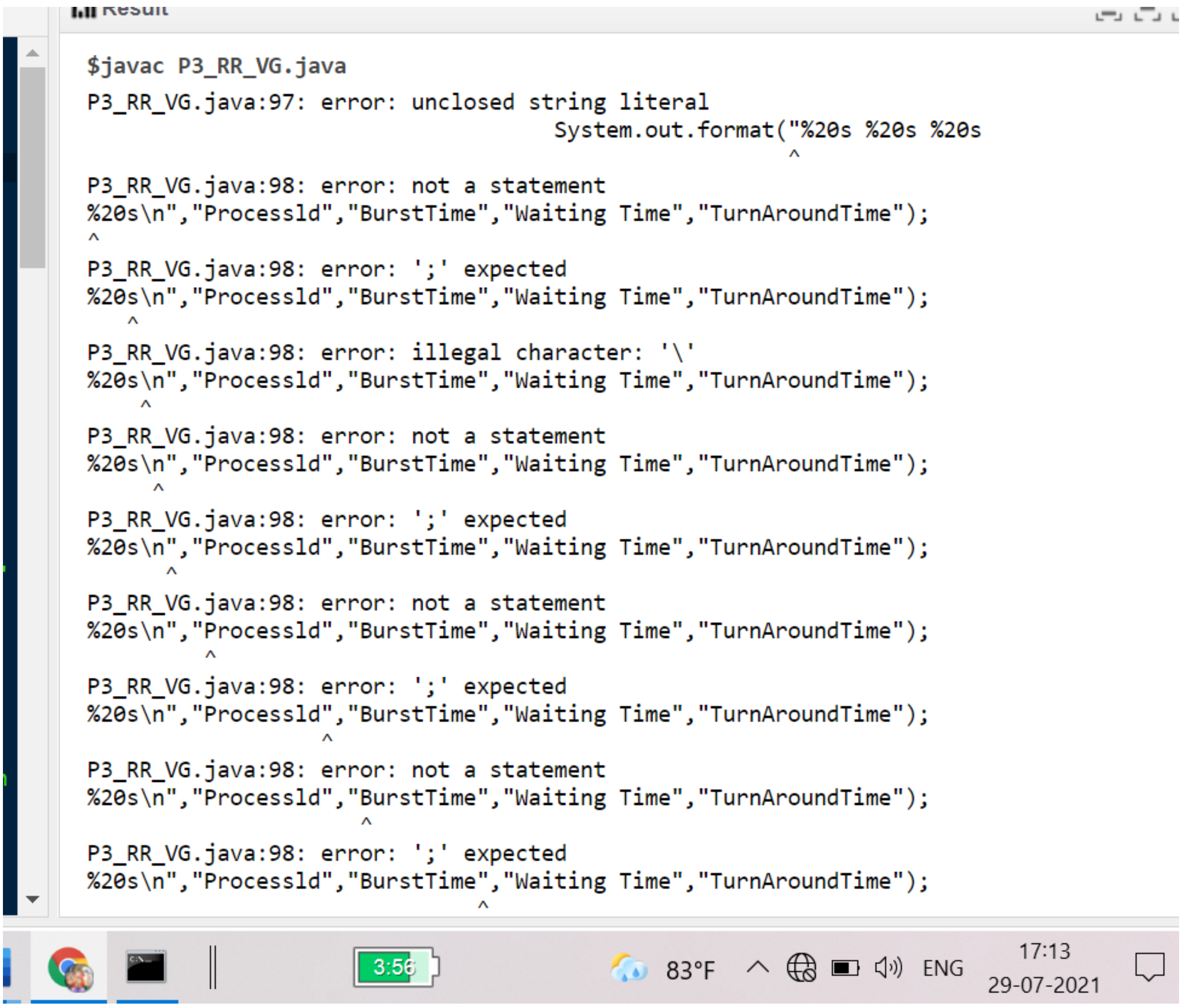
Enter Time quantum: 4

Output :-

RR Algorithm:

ProcessId	BurstTime	Waiting Time	TurnAroundTime
P0	24	6	30
P1	3	4	7
P2	3	7	10
	Average	5.666667	15.666667

Smt. Chandibai Himathmal Mansukhani College



```
$javac P3_RR_VG.java
P3_RR_VG.java:97: error: unclosed string literal
                                    System.out.format("%20s %20s %20s
                                                      ^
P3_RR_VG.java:98: error: not a statement
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
^
P3_RR_VG.java:98: error: ';' expected
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
   ^
P3_RR_VG.java:98: error: illegal character: '\'
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
    ^
P3_RR_VG.java:98: error: not a statement
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
     ^
P3_RR_VG.java:98: error: ';' expected
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
      ^
P3_RR_VG.java:98: error: not a statement
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
         ^
P3_RR_VG.java:98: error: ';' expected
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
                  ^
P3_RR_VG.java:98: error: not a statement
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
                     ^
P3_RR_VG.java:98: error: ';' expected
%20s\n","Processld","BurstTime","Waiting Time","TurnAroundTime");
                              ^
```

Smt. Chandibai Himathmal Mansukhani College

NAME: VRUSHABH GADE BATCH: B1 DATE: 28 JULY 2021