**GANDAKI COLLEGE OF ENGINEERING AND SCIENCE**

LAB 3: IMPLEMENT ROUND ROBIN SCHEDULING ALGORITHM

**SUBMITTED BY: SUBMITTED TO: -**

**Shivaji Chalise PRATIKSHYA SHRESTHA**

**BE2019SE685 Teacher**

**OBJECTIVE:** TO IMPLEMENT ROUND ROBIN SCHEDULING WITH/WITHOUT ARRIVAL TIME

**THEORY:** The Round-robin scheduling algorithm is a kind of preemptive First come First Serve CPU Scheduling algorithm where each process in the ready state gets the CPU for a fixed time in a cyclic way (turn by turn). It is the oldest scheduling algorithm, which is mainly used for multitasking. A round-robin scheduling algorithm is used to schedule the process fairly for 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.

**SOURCE CODE:**

#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);

printf("\n");

return 0;

}

Output:

Enter Total Process: 3

Enter Arrival Time and Burst Time for Process Process Number 1 :0 2

Enter Arrival Time and Burst Time for Process Process Number 2 :2 2

Enter Arrival Time and Burst Time for Process Process Number 3 :1 3

Enter Time Quantum: 2

Process Turnaround Time Waiting Time

P[1] 2 0

P[2] 2 0

P[3] 6 3

Average Waiting Time= 1.000000

Avg Turnaround Time = 3.333333

**CONCLUSION:** Finally we understood and learned to implement Round Robin Scheduling Algorithm.