Uday Krishna N

231901057

Ex. No.: 6D

Date:22.03.2025

ROUND ROBIN SCHEDULING

Aim:

To implement the Round Robin (RR) scheduling technique

Algorithm:

1. Declare the structure and its elements.
2. Get number of processes and Time quantum as input from the user.
3. Read the process name, arrival time and burst time
4. Create an array rem_bt[] to keep track of remaining burst time of processes which is initially copy of bt[] (burst times array)
5. Create another array wt[] to store waiting times of processes. Initialize this array as 0. 6. Initialize time : t = 0
7. Keep traversing the all processes while all processes are not done. Do following for i'th process if it is not done yet.

a- If rem_bt[i] > quantum (i) t = t + quantum (ii) bt_rem[i] -= quantum; b- Else // Last cycle for this process

(i) t = t + bt_rem[i];

(ii) wt[i] = t - bt[i]

(iii) bt_rem[i] = 0; // This process is over

8. Calculate the waiting time and turnaround time for each process.
9. Calculate the average waiting time and average turnaround time.
10. Display the results.

Program Code:

```
#include <stdio.h>
int main() {
    int n, quantum, i, t = 0, x, done;
    printf("Enter number of processes and time quantum: ");
    scanf("%d %d", &n, &quantum);
    int at[n], bt[n], rem_bt[n], wt[n], tat[n];
    x = n;
    printf("Enter arrival time and burst time for each process:\n");
    for (i = 0; i < n; i++) {        scanf("%d %d", &at[i], &bt[i]);
```

```
        rem_bt[i] = bt[i];
    wt[i] = 0;
        }
         printf("\nProcess\tAT\tBT\tWT\tTAT\n");
        int total_wt = 0, total_tat = 0;
    for (t = 0, i = 0; x != 0;) {           if
       (rem_bt[i] > 0 && at[i] <= t) {
        if (rem_bt[i] > quantum) {
        t += quantum;
        rem_bt[i] -= quantum;
        } else {
        t += rem_bt[i];
        wt[i] = t - at[i] - bt[i];
        tat[i] = t - at[i];
        total_wt += wt[i];
        total_tat += tat[i];
        rem_bt[i] = 0;
        x--;
                printf("P%d\t%d\t%d\t%d\t%d\n", i + 1, at[i], bt[i], wt[i], tat[i]);
                }
        }
        i = (i + 1) % n;
        }
        printf("\nAverage WT: %.2f\nAverage TAT: %.2f\n", (float)total_wt / n, (float)total_tat /
    n);
        return 0;
    }
```

OUTPUT:

```
Enter number of processes and time quantum: 4 3
Enter arrival time and burst time for each process:
0 4
1 7
2 5
3 6

Process AT      BT      WT      TAT
P1      0       4       9       13
P3      2       5       11      16
P4      3       6       12      18
P2      1       7       14      21

Average WT: 11.50
Average TAT: 17.00
```

RESULT:

Hence, RoundRobin CPU Scheduling has been executed successfully.