

Ex No: 6(d)**ROUND ROBIN SCHEDULING****Date: 26.03.2025****Aim:**

To implement the Round Robin (RR) scheduling technique using C programming.

Algorithm:

1. Start.
 2. Get the number of processes and the time quantum from the user.
 3. Read the process burst time (arrival time is assumed 0 for simplicity).
 4. Initialize an array `rem_bt[]` (remaining burst time) as a copy of burst time.
 5. Initialize an array `wt[]` (waiting time) as 0 for all processes.
 6. Set current time `t = 0`.
 7. Repeat while all processes are not completed:
 - For each process `i`:
 - If `rem_bt[i] > 0`:
 - If `rem_bt[i] > quantum`:
 - `t += quantum`
 - `rem_bt[i] -= quantum`
 - Else:
 - `t += rem_bt[i]`
 - `wt[i] = t - bt[i]`
 - `rem_bt[i] = 0`
 8. Calculate Turnaround Time for each process as: `tat[i] = bt[i] + wt[i]`
 9. Compute Average Waiting Time and Average Turnaround Time.
 10. Display the process-wise result.
 11. End.
-

Program Code (C):

```
#include <stdio.h>
int main() { int i, n, time = 0, quantum;

    int bt[20], rem_bt[20], wt[20],
    tat[20]; float avg_wt = 0, avg_tat = 0;

    printf("Enter total number of processes: ");
    scanf("%d", &n);

    printf("Enter burst time for each process:\n");
    for (i = 0; i < n; i++) {
        printf("P[%d]: ", i +
            1); scanf("%d",
            &bt[i]); rem_bt[i] =
            bt[i]; wt[i] = 0;
    }

    printf("Enter Time Quantum: ");
    scanf("%d", &quantum);

    int done;
    do { done =
        1;

        for (i = 0; i < n; i++) { if
            (rem_bt[i] > 0) { done = 0;
            if (rem_bt[i] > quantum) {
                time += quantum;
                rem_bt[i] -= quantum;
            } else {
```

```
        time += rem_bt[i];
        wt[i] = time - bt[i];
        rem_bt[i] = 0;
    }

    }

    }
} while (!done);

printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");
for (i = 0; i < n; i++) {
    tat[i] = bt[i] + wt[i];
    avg_wt += wt[i]; avg_tat += tat[i];
    printf("P[%d]\t%d\t%d\t%d\n", i + 1, bt[i], wt[i],
    tat[i]);
}

avg_wt /= n;
avg_tat /= n;

printf("\nAverage Waiting Time = %.2f", avg_wt);
printf("\nAverage Turnaround Time = %.2f\n", avg_tat);

return 0;

}
```

Sample Output:

Enter total number of processes: 4 Enter

burst time for each process:

P[1]: 5

P[2]: 15

P[3]: 4

P[4]: 3

Enter Time Quantum: 5

Process	Burst Time	Waiting Time	Turnaround Time
P[1]	5	0	5
P[2]	15	12	27
P[3]	4	5	9
P[4]	3	9	12

Average Waiting Time = 6.50

Average Turnaround Time = 13.25

Result:

The Round Robin Scheduling algorithm was successfully implemented and tested. It correctly calculated the waiting and turnaround times based on the given time quantum.