**Expr 6 d: Round Robin Scheduling**

**Code:**

#include <stdio.h>

#include <string.h>

struct Process {

    char name[10];

    int arrival\_time;

    int burst\_time;

    int remaining\_time;

    int waiting\_time;

    int turnaround\_time;

};

int main() {

    int n, time\_quantum, total\_time = 0, completed = 0;

    float total\_waiting\_time = 0, total\_turnaround\_time = 0;

    struct Process p[10];

    // Step 2: Input

    printf("Enter number of processes: ");

    scanf("%d", &n);

    printf("Enter time quantum: ");

    scanf("%d", &time\_quantum);

    // Step 3: Input each process data

    for (int i = 0; i < n; i++) {

        printf("Enter process name, arrival time and burst time for process %d: ", i+1);

        scanf("%s %d %d", p[i].name, &p[i].arrival\_time, &p[i].burst\_time);

        p[i].remaining\_time = p[i].burst\_time;

        p[i].waiting\_time = 0;

        p[i].turnaround\_time = 0;

    }

    int t = 0;

    while (completed < n) {

        int done = 1;

        for (int i = 0; i < n; i++) {

            if (p[i].remaining\_time > 0 && p[i].arrival\_time <= t) {

                done = 0;

                if (p[i].remaining\_time > time\_quantum) {

                    t += time\_quantum;

                    p[i].remaining\_time -= time\_quantum;

                } else {

                    t += p[i].remaining\_time;

                    p[i].waiting\_time = t - p[i].arrival\_time - p[i].burst\_time;

                    p[i].turnaround\_time = t - p[i].arrival\_time;

                    p[i].remaining\_time = 0;

                    completed++;

                }

            }

        }

        if (done)

            t++;

    }

    // Step 9: Calculate averages

    for (int i = 0; i < n; i++) {

        total\_waiting\_time += p[i].waiting\_time;

        total\_turnaround\_time += p[i].turnaround\_time;

    }

    float avg\_waiting\_time = total\_waiting\_time / n;

    float avg\_turnaround\_time = total\_turnaround\_time / n;

    // Step 10: Display results

    printf("\nProcess\tAT\tBT\tWT\tTAT\n");

    for (int i = 0; i < n; i++) {

        printf("%s\t%d\t%d\t%d\t%d\n", p[i].name, p[i].arrival\_time, p[i].burst\_time,

               p[i].waiting\_time, p[i].turnaround\_time);

    }

    printf("\nAverage Waiting Time: %.2f", avg\_waiting\_time);

    printf("\nAverage Turnaround Time: %.2f\n", avg\_turnaround\_time);

    return 0;

}

**Output:**

Enter number of processes: 4

Enter time quantum: 3

Enter process name, arrival time and burst time for process 1: P1 0 5

Enter process name, arrival time and burst time for process 2: P2 1 4

Enter process name, arrival time and burst time for process 3: P3 2 2

Enter process name, arrival time and burst time for process 4: P4 3 1

Process AT BT WT TAT

P1 0 5 7 12

P2 1 4 9 13

P3 2 2 4 6

P4 3 1 2 3

Average Waiting Time: 5.50

Average Turnaround Time: 8.50

**Result:**

Thus the Round Robin Scheduling Code is implemented in fedora using the C language