Expr 6 d: Round Robin Scheduling

Code:

```c
#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