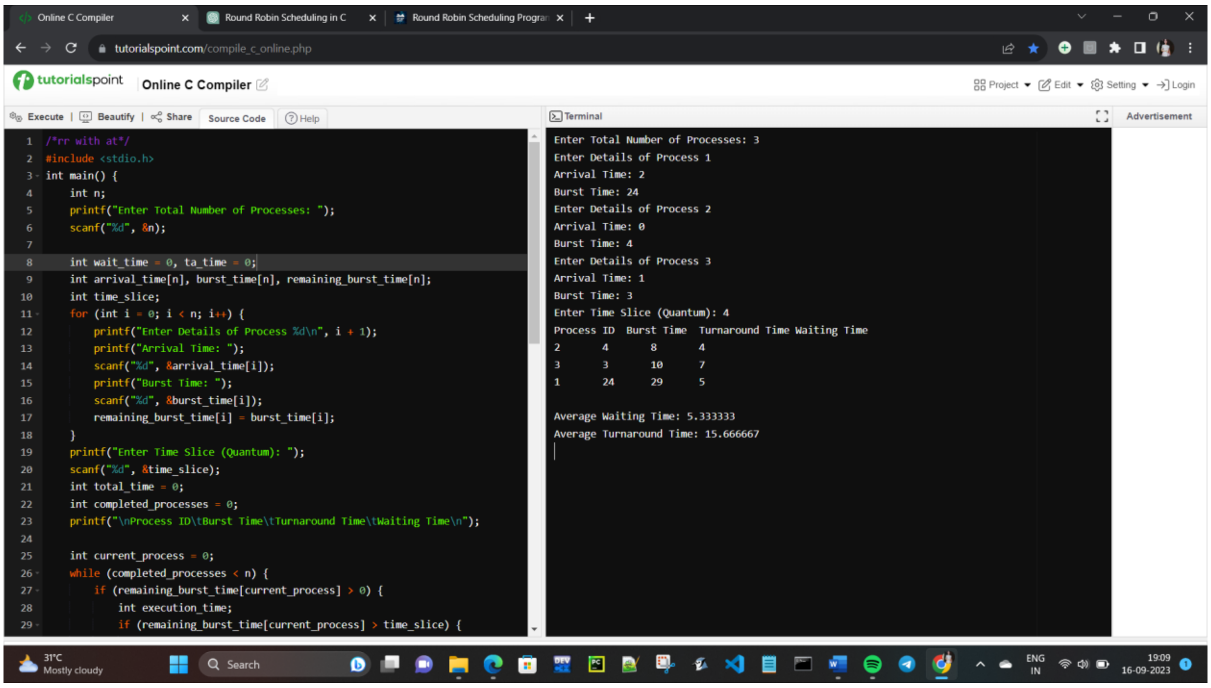
```
/*rr with at*/

#include <stdio.h>

int main() {

    int n;

    printf("Enter Total Number of Processes: ");

    scanf("%d", &n);



    int wait_time = 0, ta_time = 0;

    int arrival_time[n], burst_time[n], remaining_burst_time[n];

    int time_slice;

    for (int i = 0; i < n; i++) {

        printf("Enter Details of Process %d\n", i + 1);

        printf("Arrival Time: ");

        scanf("%d", &arrival_time[i]);

        printf("Burst Time: ");

        scanf("%d", &burst_time[i]);

        remaining_burst_time[i] = burst_time[i];

    }

    printf("Enter Time Slice (Quantum): ");

    scanf("%d", &time_slice);

    int total_time = 0;

    int completed_processes = 0;

    printf("\nProcess ID\tBurst Time\tTurnaround Time\tWaiting Time\n");



    int current_process = 0;

    while (completed_processes < n) {

        if (remaining_burst_time[current_process] > 0) {

            int execution_time;

            if (remaining_burst_time[current_process] > time_slice) {

                execution_time = time_slice;

            } else {
                execution_time = remaining_burst_time[current_process];
            }

            total_time += execution_time;
            remaining_burst_time[current_process] -= execution_time;

            if (remaining_burst_time[current_process] == 0) {
                completed_processes++;

                int turnaround_time = total_time - arrival_time[current_process];
                int waiting_time = turnaround_time - burst_time[current_process];

                printf("%d\t\t%d\t\t%d\t\t%d\n", current_process + 1, burst_time[current_process],
                       turnaround_time, waiting_time);

                wait_time += waiting_time;
                ta_time += turnaround_time;
            }

            // Move to the next process in a circular manner
            current_process = (current_process + 1) % n;
        } else {
            // If the process has already completed, move to the next process
            current_process = (current_process + 1) % n;
        }
    }

    float average_wait_time = (float)wait_time / n;
    float average_turnaround_time = (float)ta_time / n;

    printf("\nAverage Waiting Time: %f", average_wait_time);
```

```c
    printf("\nAverage Turnaround Time: %f\n", average_turnaround_time);


    return 0;
}
```



```c
/*rr with at*/
#include <stdio.h>
int main() {
    int n;
    printf("Enter Total Number of Processes: ");
    scanf("%d", &n);

    int wait_time = 0, ta_time = 0;
    int arrival_time[n], burst_time[n], remaining_burst_time[n];
    int time_slice;
    for (int i = 0; i < n; i++) {
        printf("Enter Details of Process %d\n", i + 1);
        printf("Arrival Time: ");
        scanf("%d", &arrival_time[i]);
        printf("Burst Time: ");
        scanf("%d", &burst_time[i]);
        remaining_burst_time[i] = burst_time[i];
    }
    printf("Enter Time Slice (Quantum): ");
    scanf("%d", &time_slice);
    int total_time = 0;
    int completed_processes = 0;
    printf("\nProcess ID\tBurst Time\tTurnaround Time\tWaiting Time\n");

    int current_process = 0;
    while (completed_processes < n) {
        if (remaining_burst_time[current_process] > 0) {
            int execution_time;
            if (remaining_burst_time[current_process] > time_slice) {
```

```
Enter Total Number of Processes: 3
Enter Details of Process 1
Arrival Time: 2
Burst Time: 24
Enter Details of Process 2
Arrival Time: 0
Burst Time: 4
Enter Details of Process 3
Arrival Time: 1
Burst Time: 3
Enter Time Slice (Quantum): 4
Process ID  Burst Time  Turnaround Time Waiting Time
2       4       8       4
3       3       10      7
1       24      29      5

Average Waiting Time: 5.333333
Average Turnaround Time: 15.666667
```