SRTF

```
#include <stdio.h>
#include <stdbool.h>

#define MAX_PROCESSES 10

struct Process {
    int pid;
    int arrival_time;
    int burst_time;
    int priority;
    int remaining_time;
    int turnaround_time;
    int waiting_time;
};
void sjf_preemptive(struct Process processes[], int n) {
    int total_time = 0,i;
    int completed = 0;

    while (completed < n) {
        int shortest_burst = -1;
        int next_process = -1;

        for (i = 0; i < n; i++) {
            if (processes[i].arrival_time <= total_time &&
processes[i].remaining_time > 0) {
                if (shortest_burst == -1 || processes[i].remaining_time <
shortest_burst) {
                    shortest_burst = processes[i].remaining_time;
                    next_process = i;
                }
```

```
            }
        }

        if (next_process == -1) {
            total_time++;
            continue;
        }

        processes[next_process].remaining_time--;
        total_time++;

        if (processes[next_process].remaining_time == 0) {
            completed++;
            processes[next_process].turnaround_time = total_time -
processes[next_process].arrival_time;
            processes[next_process].waiting_time =
processes[next_process].turnaround_time -
processes[next_process].burst_time;
        }
    }

    double total_turnaround_time = 0;
    double total_waiting_time = 0;

    printf("Process\tTurnaround Time\tWaiting Time\n");
    for (i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\n", processes[i].pid,
processes[i].turnaround_time, processes[i].waiting_time);

        total_turnaround_time += processes[i].turnaround_time;
        total_waiting_time += processes[i].waiting_time;
    }

    printf("Average Turnaround Time: %.2f\n", total_turnaround_time / n);
    printf("Average Waiting Time: %.2f\n", total_waiting_time / n);
```

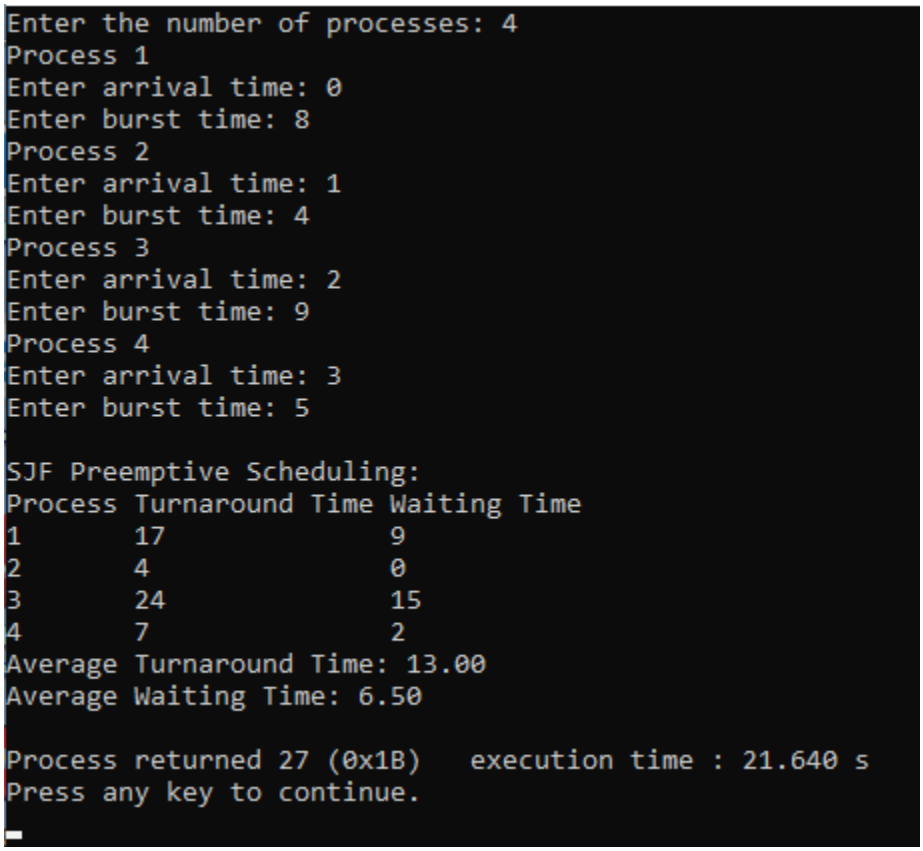
```
}

void main(){
    int n,i;
    struct Process processes[MAX_PROCESSES];

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

    for (i = 0; i < n; i++) {
        printf("Process %d\n", i + 1);
        printf("Enter arrival time: ");
        scanf("%d", &processes[i].arrival_time);
        printf("Enter burst time: ");
        scanf("%d", &processes[i].burst_time);
        processes[i].pid = i + 1;
        processes[i].remaining_time = processes[i].burst_time;
        processes[i].turnaround_time = 0;
        processes[i].waiting_time = 0;
    }
    printf("\nSJF Preemptive Scheduling:\n");
            sjf_preemptive(processes, n);
}
```

OUTPUT:


Enter the number of processes: 4
Process 1
Enter arrival time: 0
Enter burst time: 8
Process 2
Enter arrival time: 1
Enter burst time: 4
Process 3
Enter arrival time: 2
Enter burst time: 9
Process 4
Enter arrival time: 3
Enter burst time: 5
SJF Preemptive Scheduling:
Process Turnaround Time Waiting Time
1 17 9
2 4 0
3 24 15
4 7 2
Average Turnaround Time: 13.00
Average Waiting Time: 6.50
Process returned 27 (0x1B) execution time : 21.640 s
Press any key to continue.