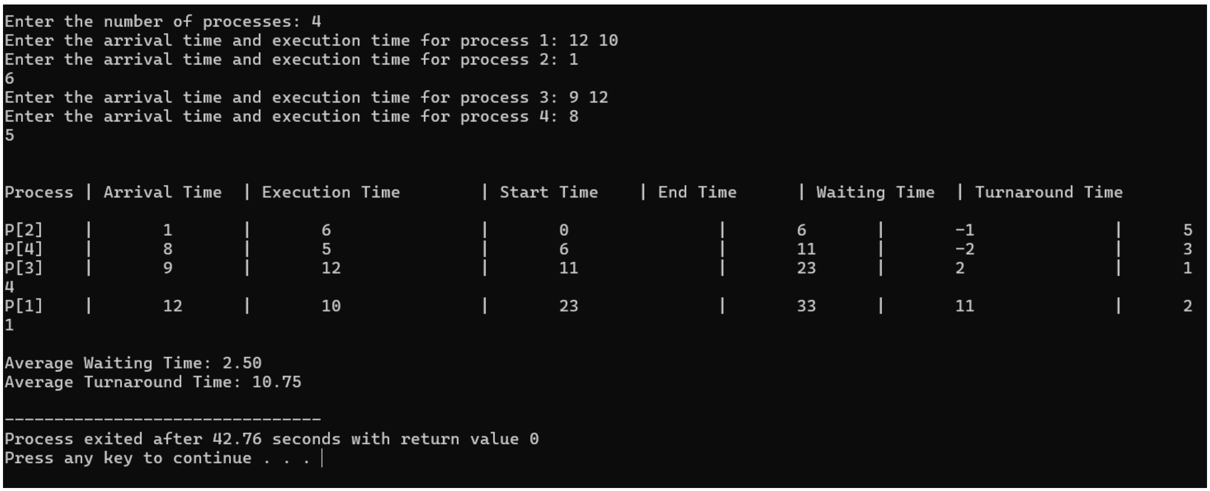
7.Construct a C program to implement a non-preemptive SJF algorithm.

A. Code:

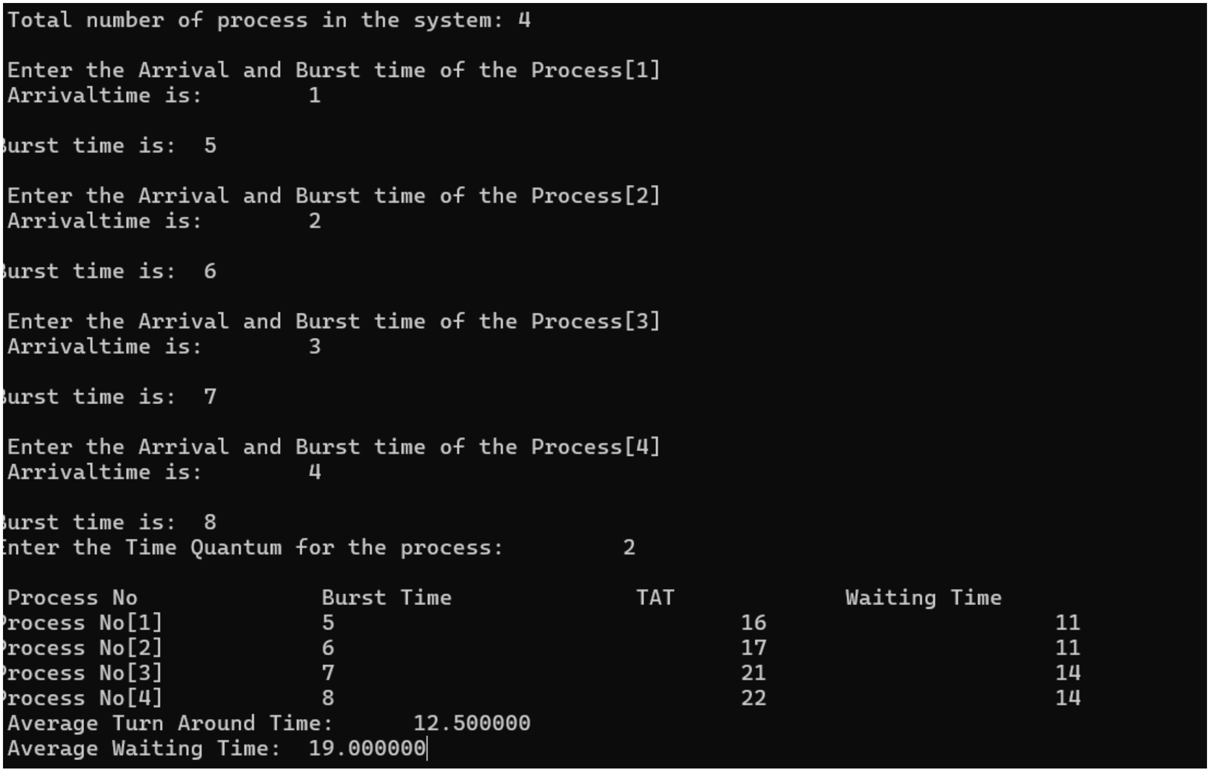
```
#include <stdio.h>

int main() {
    int at[10], bt[10], pr[10]; // Arrays for arrival time, burst time, and process IDs
    int n, i, j, temp, time = 0, count, over = 0;
    int sum_wait = 0, sum_turnaround = 0, start;
    float avgwait, avgturn;

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

    // Input arrival time and burst time for each process
    for (i = 0; i < n; i++) {
        printf("Enter the arrival time and execution time for process %d: ", i + 1);
        scanf("%d %d", &at[i], &bt[i]);
        pr[i] = i + 1; // Assign process ID
    }

    // Sort processes by arrival time
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (at[i] > at[j]) {
                // Swap arrival time
                temp = at[i];
                at[i] = at[j];
                at[j] = temp;

                // Swap burst time
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;

                // Swap process ID
                temp = pr[i];
                pr[i] = pr[j];
                pr[j] = temp;
            }
        }
    }

    printf("\n\nProcess\t| Arrival Time\t| Execution Time\t| Start Time\t| End Time\t| Waiting Time\t| Turnaround Time\n\n");

    // Execute all processes
    while (over < n) {
        count = 0;

        // Count processes that have arrived by the current time
        for (i = over; i < n; i++) {
            if (at[i] <= time) {
                count++;
            } else {
                break;
            }
        }

        // Sort the arrived processes by burst time
        if (count > 1) {
            for (i = over; i < over + count - 1; i++) {
```

```
        for (j = i + 1; j < over + count; j++) {
            if (bt[i] > bt[j]) {
                // Swap arrival time
                temp = at[i];
                at[i] = at[j];
                at[j] = temp;

                // Swap burst time
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;

                // Swap process ID
                temp = pr[i];
                pr[i] = pr[j];
                pr[j] = temp;
            }
        }
    }
}

// Process execution
start = time;
time += bt[over];

// Print process details
printf("P[%d]\t|\t%d\t|\t%d\t\t|\t%d\t\t|\t%d\t|\t%d\t\t|\t%d\n", pr[over],
      at[over], bt[over], start, time, time - at[over] - bt[over], time - at[over]);

// Update total waiting time and turnaround time
sum_wait += time - at[over] - bt[over];
```

```
        sum_turnaround += time - at[over];

        over++;

    }


    // Calculate averages

    avgwait = (float)sum_wait / (float)n;

    avgturn = (float)sum_turnaround / (float)n;


    // Print averages

    printf("\nAverage Waiting Time: %.2f", avgwait);

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


    return 0;

}
```
Output:



```
Enter the number of processes: 4
Enter the arrival time and execution time for process 1: 12 10
Enter the arrival time and execution time for process 2: 1
6
Enter the arrival time and execution time for process 3: 9 12
Enter the arrival time and execution time for process 4: 8
5


Process | Arrival Time  | Execution Time        | Start Time    | End Time      | Waiting Time  | Turnaround Time

P[2]    |       1       |       6               |       0               |       6       |       -1              |       5
P[4]    |       8       |       5               |       6               |       11      |       -2              |       3
P[3]    |       9       |       12              |       11              |       23      |       2               |       1
4
P[1]    |       12      |       10              |       23              |       33      |       11              |       2
1

Average Waiting Time: 2.50
Average Turnaround Time: 10.75

--------------------------------
Process exited after 42.76 seconds with return value 0
Press any key to continue . . .
```



```
 Total number of process in the system: 4

 Enter the Arrival and Burst time of the Process[1]
 Arrivaltime is:        1

Burst time is:  5

 Enter the Arrival and Burst time of the Process[2]
 Arrivaltime is:        2

Burst time is:  6

 Enter the Arrival and Burst time of the Process[3]
 Arrivaltime is:        3

Burst time is:  7

 Enter the Arrival and Burst time of the Process[4]
 Arrivaltime is:        4

Burst time is:  8
Enter the Time Quantum for the process:         2

 Process No              Burst Time              TAT             Waiting Time
Process No[1]            5                               16                      11
Process No[2]            6                               17                      11
Process No[3]            7                               21                      14
Process No[4]            8                               22                      14
 Average Turn Around Time:      12.500000
 Average Waiting Time:  19.000000
```