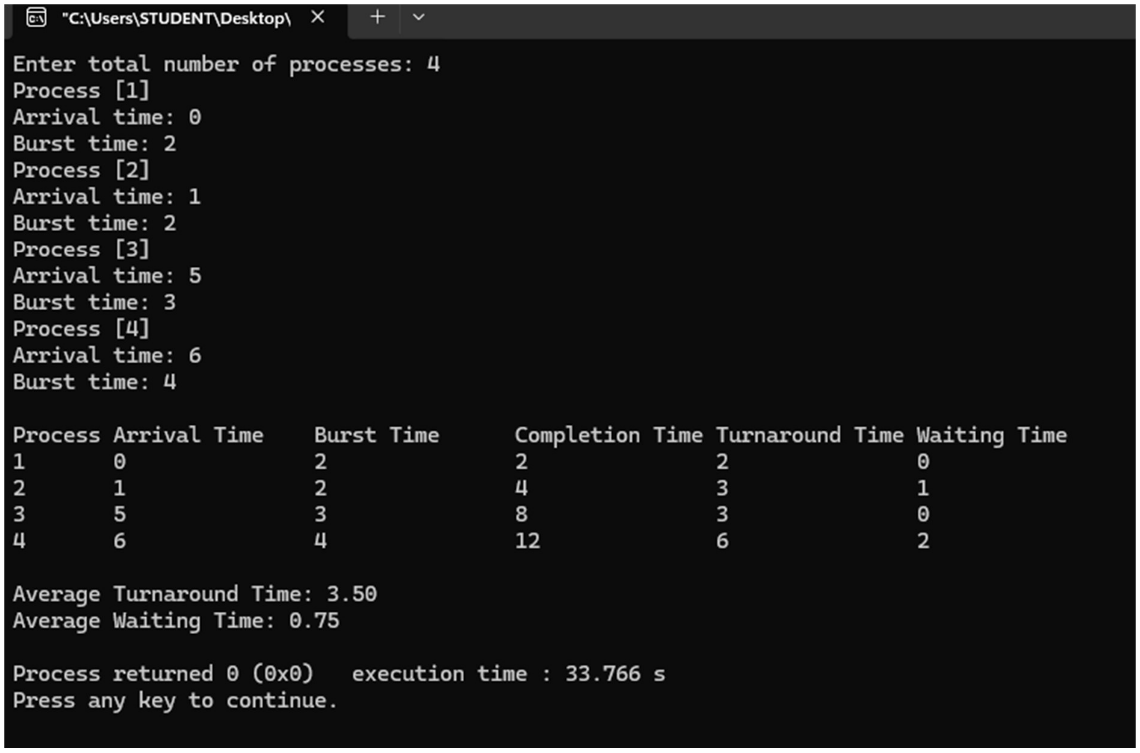
```
#include<stdio.h>

#include<stdlib.h>


#define MAX_PROCESS 30


int p[MAX_PROCESS], arrTime[MAX_PROCESS], burstTime[MAX_PROCESS],
compTime[MAX_PROCESS], TAT[MAX_PROCESS], waitTime[MAX_PROCESS];


void sortProcess(int arrTime[], int burstTime[], int n) {

   int temp;

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

      for (int j = 0; j < n - i - 1; j++) {

         if (arrTime[j] > arrTime[j + 1]) {

            // Swap arrival times

            temp = arrTime[j];

            arrTime[j] = arrTime[j + 1];

            arrTime[j + 1] = temp;


            // Swap burst times accordingly

            temp = burstTime[j];

            burstTime[j] = burstTime[j + 1];

            burstTime[j + 1] = temp;

         }

      }

   }

}


int findTurnAroundTime(int ct, int at) {

   return ct - at;

}
```

```
int waitingTime(int tat, int bt) {

    return tat - bt;

}


int main() {

    int n;

    printf("Enter total number of processes: ");

    scanf("%d", &n);


    int total_TAT = 0; // Total turnaround time

    int total_WT = 0;  // Total waiting time


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

        printf("Process [%d]\n", i + 1);

        printf("Arrival time: ");

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

        printf("Burst time: ");

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

        p[i] = i + 1; // Assigning process number

    }


    // Sort processes based on arrival time

    sortProcess(arrTime, burstTime, n);


    // Calculate completion time, turnaround time, and waiting time

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

        if (i == 0 || arrTime[i] > compTime[i - 1]) {

            compTime[i] = arrTime[i] + burstTime[i];

        } else {

            compTime[i] = compTime[i - 1] + burstTime[i];

        }
        TAT[i] = findTurnAroundTime(compTime[i], arrTime[i]);
        waitTime[i] = waitingTime(TAT[i], burstTime[i]);

        // Summing up turnaround time and waiting time
        total_TAT += TAT[i];
        total_WT += waitTime[i];
    }

    // Calculate averages
    float avg_TAT = (float)total_TAT / n;
    float avg_WT = (float)total_WT / n;

    // Displaying results including averages
    printf("\nProcess\tArrival Time\tBurst Time\tCompletion Time\tTurnaround Time\tWaiting
Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i], arrTime[i], burstTime[i], compTime[i], TAT[i],
waitTime[i]);
    }
    printf("\nAverage Turnaround Time: %.2f", avg_TAT);
    printf("\nAverage Waiting Time: %.2f\n", avg_WT);

    return 0;
}
```



```
Enter total number of processes: 4
Process [1]
Arrival time: 0
Burst time: 2
Process [2]
Arrival time: 1
Burst time: 2
Process [3]
Arrival time: 5
Burst time: 3
Process [4]
Arrival time: 6
Burst time: 4

Process Arrival Time    Burst Time      Completion Time Turnaround Time Waiting Time
1       0               2               2               2               0
2       1               2               4               3               1
3       5               3               8               3               0
4       6               4               12              6               2

Average Turnaround Time: 3.50
Average Waiting Time: 0.75

Process returned 0 (0x0)   execution time : 33.766 s
Press any key to continue.
```