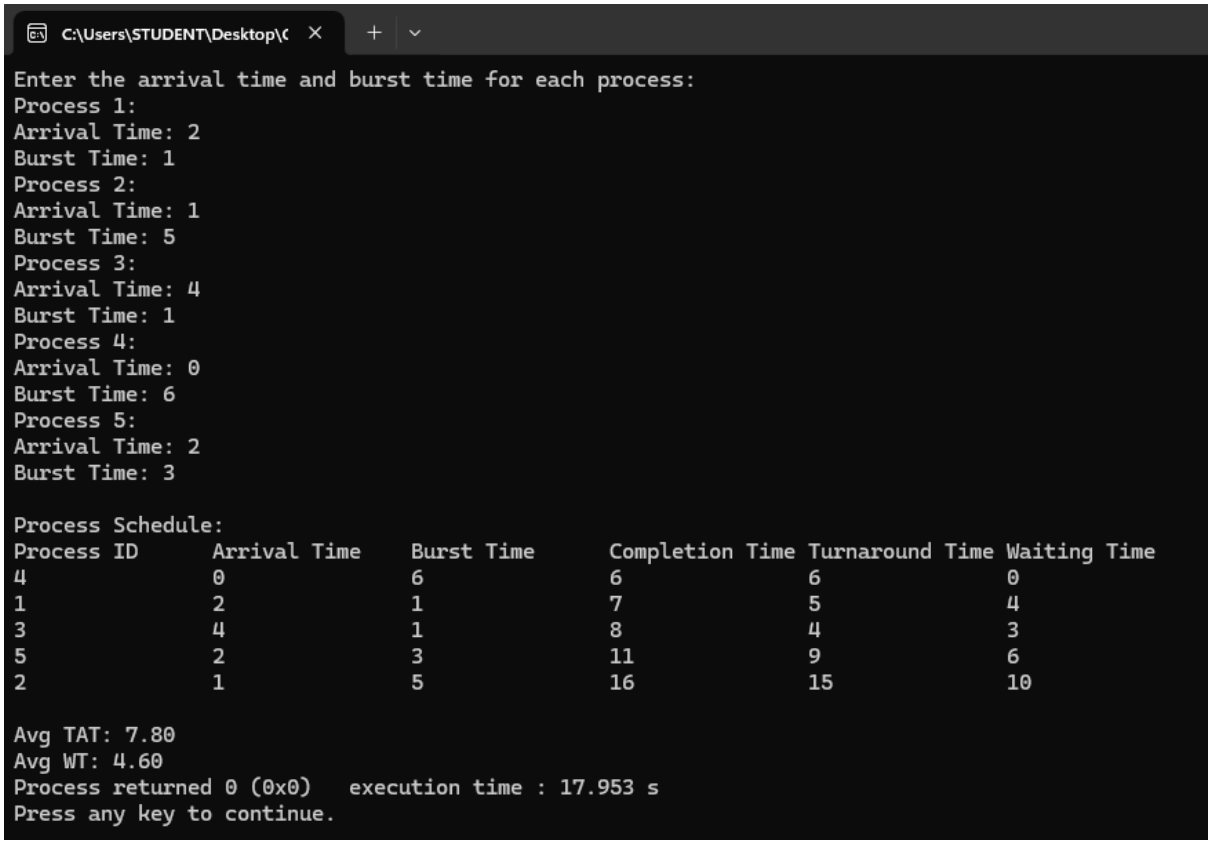
SHORTEST JOB FIRST (Non – Premptive)

```
#include <stdio.h>

struct P {
    int id;
    int at;
    int bt;
    int ct;
    int tt;
    int wt;
};

void sort(struct P p[], int n);
void sjf(struct P p[], int n);

int main() {
    int n;
    int total_tat = 0;
    int total_wt = 0;
    printf("Enter the number of processes: ");
    scanf("%d", &n);

    struct P p[n];
    printf("Enter the arrival time and burst time for each process:\n");
    for (int i = 0; i < n; i++) {
        printf("Process %d:\n", i + 1);
        p[i].id = i + 1;
        printf("Arrival Time: ");
        scanf("%d", &p[i].at);
        printf("Burst Time: ");
```

```
        scanf("%d", &p[i].bt);
    }

    sort(p, n);

    sjf(p, n);

    printf("\nProcess Schedule:\n");
    printf("Process ID\tArrival Time\tBurst Time\tCompletion Time\tTurnaround Time\tWaiting Time\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", p[i].id, p[i].at, p[i].bt, p[i].ct, p[i].tt, p[i].wt);
        total_tat += p[i].tt;
        total_wt += p[i].wt;
    }

    printf("\nAvg TAT: %.2f", (float)total_tat / n);
    printf("\nAvg WT: %.2f", (float)total_wt / n);

    return 0;
}

void sort(struct P p[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (p[j].at > p[j + 1].at || (p[j].at == p[j + 1].at && p[j].bt > p[j + 1].bt)) {
                struct P temp = p[j];
                p[j] = p[j + 1];
                p[j + 1] = temp;
            }
```

```
        }
    }
}

void sjf(struct P p[], int n) {

    int current_time = 0;
    for (int i = 0; i < n; i++) {
        int sj_index = i;
        for (int j = i + 1; j < n && p[j].at <= current_time; j++) {
            if (p[j].bt < p[sj_index].bt) {
                sj_index = j;
            }
        }
        p[sj_index].ct = current_time + p[sj_index].bt;
        p[sj_index].tt = p[sj_index].ct - p[sj_index].at;
        p[sj_index].wt = p[sj_index].tt - p[sj_index].bt;
        current_time = p[sj_index].ct;
        struct P temp = p[i];
        p[i] = p[sj_index];
        p[sj_index] = temp;
    }
}
```


C:\Users\STUDENT\Desktop\(
Enter the arrival time and burst time for each process:
Process 1:
Arrival Time: 2
Burst Time: 1
Process 2:
Arrival Time: 1
Burst Time: 5
Process 3:
Arrival Time: 4
Burst Time: 1
Process 4:
Arrival Time: 0
Burst Time: 6
Process 5:
Arrival Time: 2
Burst Time: 3
Process Schedule:
Process ID Arrival Time Burst Time Completion Time Turnaround Time Waiting Time
4 0 6 6 6 0
1 2 1 7 5 4
3 4 1 8 4 3
5 2 3 11 9 6
2 1 5 16 15 10
Avg TAT: 7.80
Avg WT: 4.60
Process returned 0 (0x0) execution time : 17.953 s
Press any key to continue.