```c
#include <stdio.h>

int main() {
    int n;

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

    int pid[n], at[n], bt[n], ct[n], tat[n], wt[n];
    int completed[n];
    float avgWT = 0, avgTAT = 0;

    // Input
    for (int i = 0; i < n; i++) {
        pid[i] = i + 1;
        printf("Enter Arrival Time and Burst Time of P%d: ", pid[i]);
        scanf("%d %d", &at[i], &bt[i]);
        completed[i] = 0;
    }

    int current_time = 0, done = 0;

    // SJF Non-Preemptive Scheduling
    while (done < n) {

        int idx = -1;
        int minBT = 999999;

        // Select shortest job available
        for (int i = 0; i < n; i++) {
            if (completed[i] == 0 && at[i] <= current_time) {
                if (bt[i] < minBT) {
                    minBT = bt[i];
                    idx = i;
                }
            }
        }

        // If no process has arrived -> CPU idle
        if (idx == -1) {
            current_time++;
            continue;
        }

        // Run selected process
        current_time += bt[idx];
        ct[idx] = current_time;
        tat[idx] = ct[idx] - at[idx];
        wt[idx] = tat[idx] - bt[idx];

        completed[idx] = 1;
        done++;
    }

    // Output table
    printf("\nPID\tAT\tBT\tCT\tTAT\tWT\n");
    for (int i = 0; i < n; i++) {
        printf("P%d\t%d\t%d\t%d\t%d\t%d\n",
               pid[i], at[i], bt[i], ct[i], tat[i], wt[i]);
        avgWT += wt[i];
        avgTAT += tat[i];
    }

    // Averages
    printf("\nAverage Waiting Time = %.2f", avgWT / n);
    printf("\nAverage Turnaround Time = %.2f\n", avgTAT / n);
```

```
    return 0;
}
```