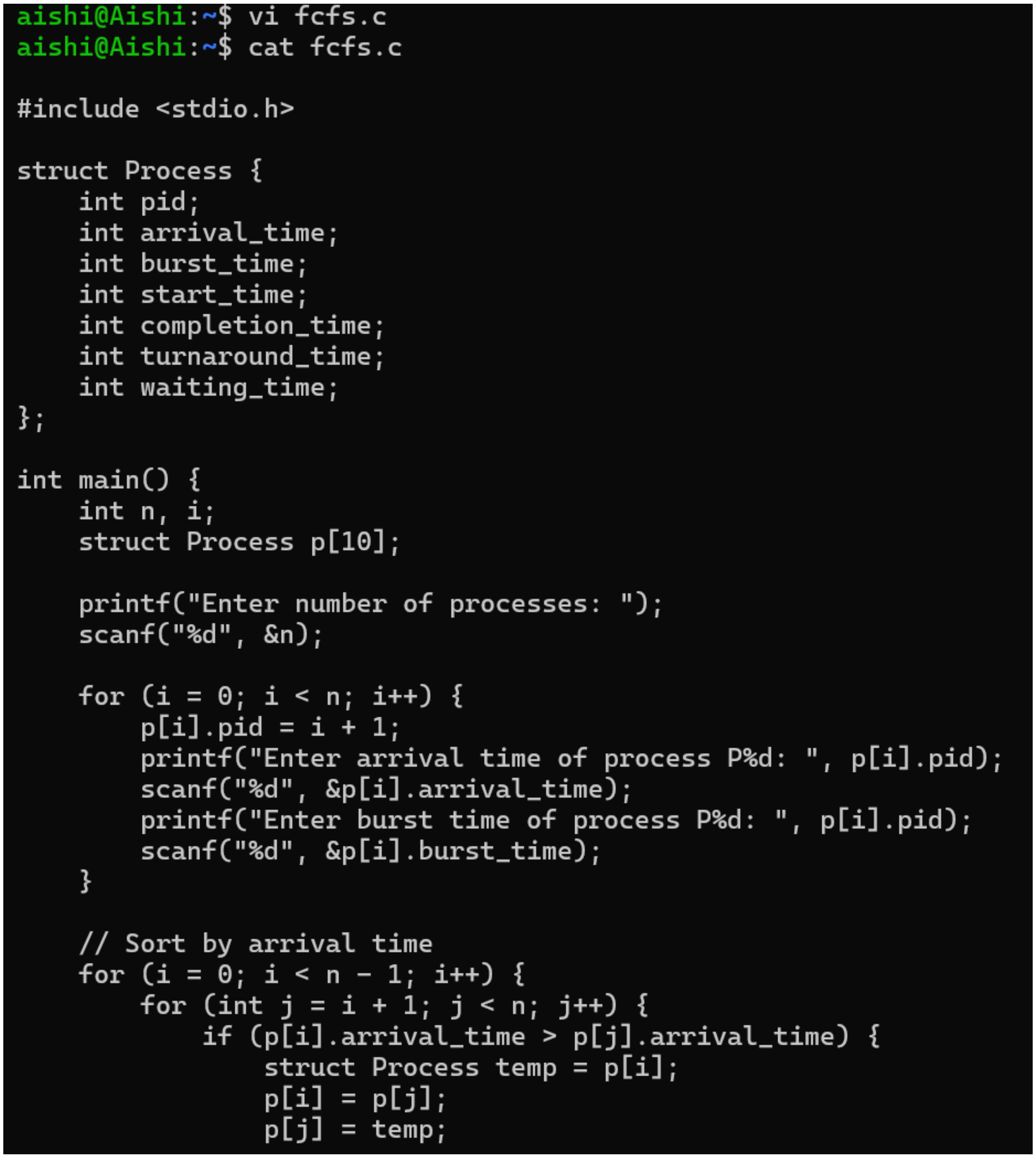
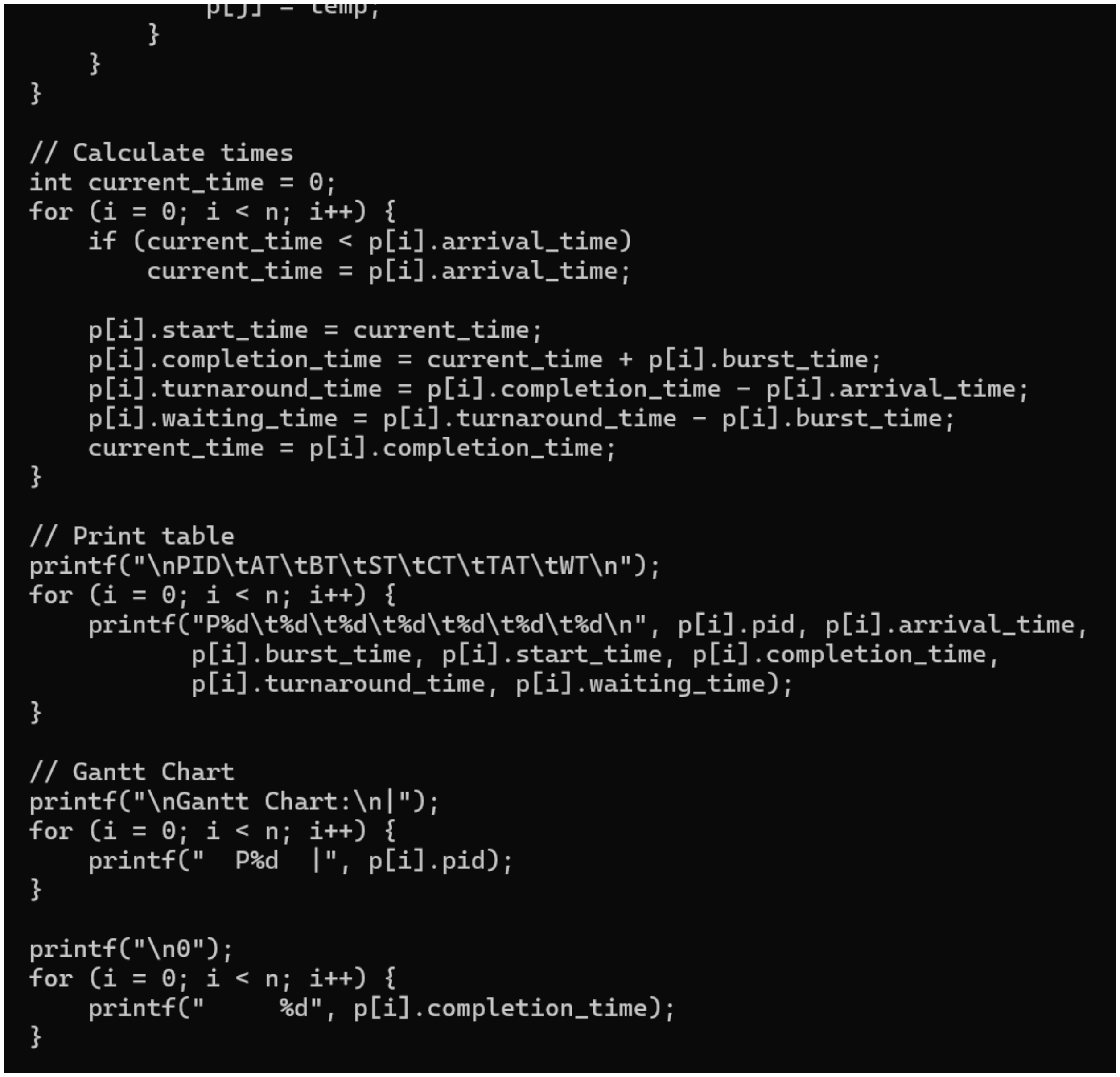
First Come First Serve Algorithm Implementation

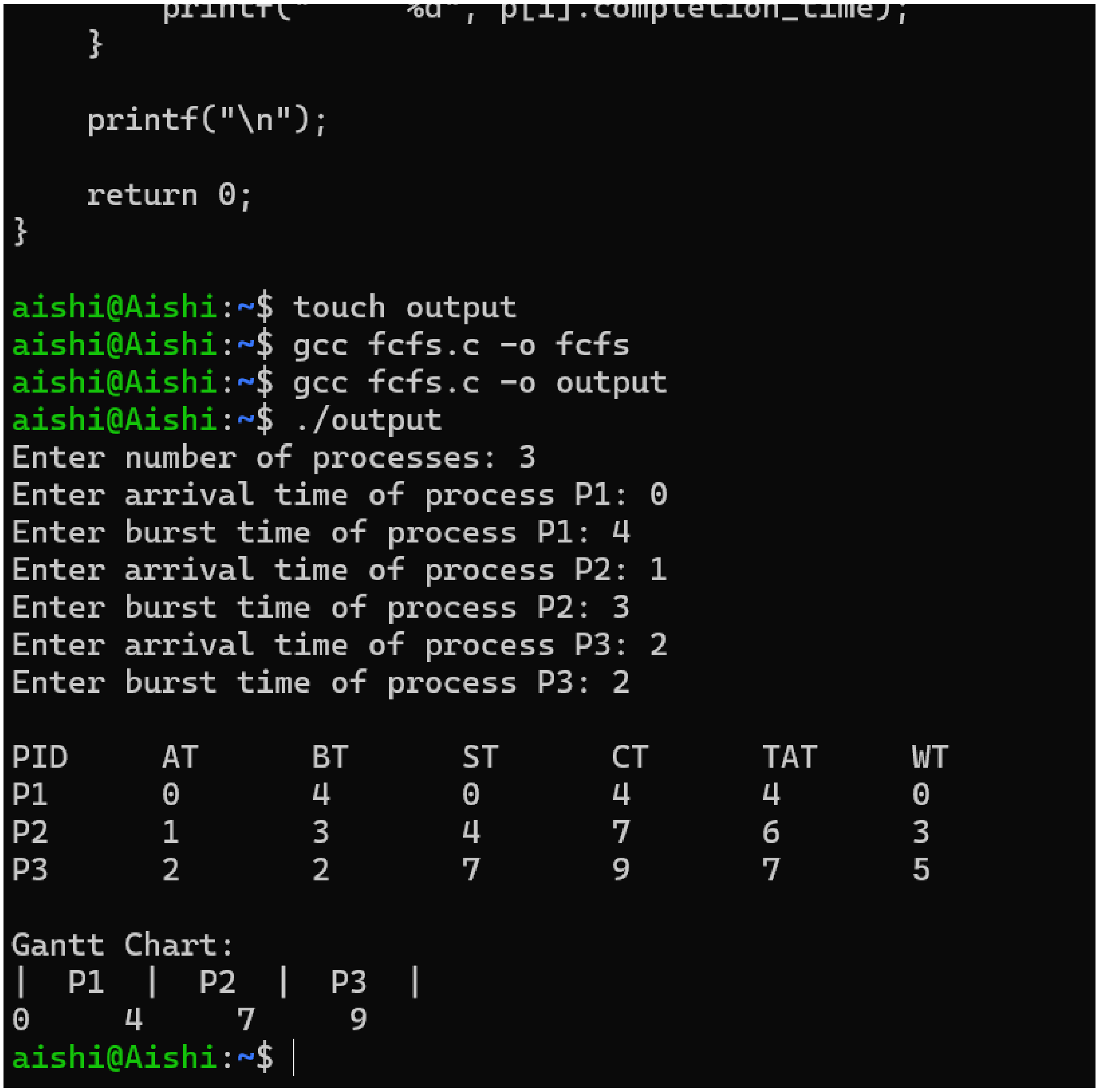
Aishi De



```
aishi@Aishi:~$ vi fcfs.c
aishi@Aishi:~$ cat fcfs.c

#include <stdio.h>

struct Process {
    int pid;
    int arrival_time;
    int burst_time;
    int start_time;
    int completion_time;
    int turnaround_time;
    int waiting_time;
};

int main() {
    int n, i;
    struct Process p[10];

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

    for (i = 0; i < n; i++) {
        p[i].pid = i + 1;
        printf("Enter arrival time of process P%d: ", p[i].pid);
        scanf("%d", &p[i].arrival_time);
        printf("Enter burst time of process P%d: ", p[i].pid);
        scanf("%d", &p[i].burst_time);
    }

    // Sort by arrival time
    for (i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (p[i].arrival_time > p[j].arrival_time) {
                struct Process temp = p[i];
                p[i] = p[j];
                p[j] = temp;
```



```
            p[j] = temp;
        }
    }
}

// Calculate times
int current_time = 0;
for (i = 0; i < n; i++) {
    if (current_time < p[i].arrival_time)
        current_time = p[i].arrival_time;

    p[i].start_time = current_time;
    p[i].completion_time = current_time + p[i].burst_time;
    p[i].turnaround_time = p[i].completion_time - p[i].arrival_time;
    p[i].waiting_time = p[i].turnaround_time - p[i].burst_time;
    current_time = p[i].completion_time;
}

// Print table
printf("\nPID\tAT\tBT\tST\tCT\tTAT\tWT\n");
for (i = 0; i < n; i++) {
    printf("P%d\t%d\t%d\t%d\t%d\t%d\t%d\n", p[i].pid, p[i].arrival_time,
           p[i].burst_time, p[i].start_time, p[i].completion_time,
           p[i].turnaround_time, p[i].waiting_time);
}

// Gantt Chart
printf("\nGantt Chart:\n|");
for (i = 0; i < n; i++) {
    printf("  P%d  |", p[i].pid);
}

printf("\n0");
for (i = 0; i < n; i++) {
    printf("     %d", p[i].completion_time);
}
```



```
        printf("     %d", p[i].completion_time);
    }

    printf("\n");

    return 0;
}

aishi@Aishi:~$ touch output
aishi@Aishi:~$ gcc fcfs.c -o fcfs
aishi@Aishi:~$ gcc fcfs.c -o output
aishi@Aishi:~$ ./output
Enter number of processes: 3
Enter arrival time of process P1: 0
Enter burst time of process P1: 4
Enter arrival time of process P2: 1
Enter burst time of process P2: 3
Enter arrival time of process P3: 2
Enter burst time of process P3: 2

PID     AT      BT      ST      CT      TAT     WT
P1      0       4       0       4       4       0
P2      1       3       4       7       6       3
P3      2       2       7       9       7       5

Gantt Chart:
|  P1  |  P2  |  P3  |
0     4     7     9
aishi@Aishi:~$
```