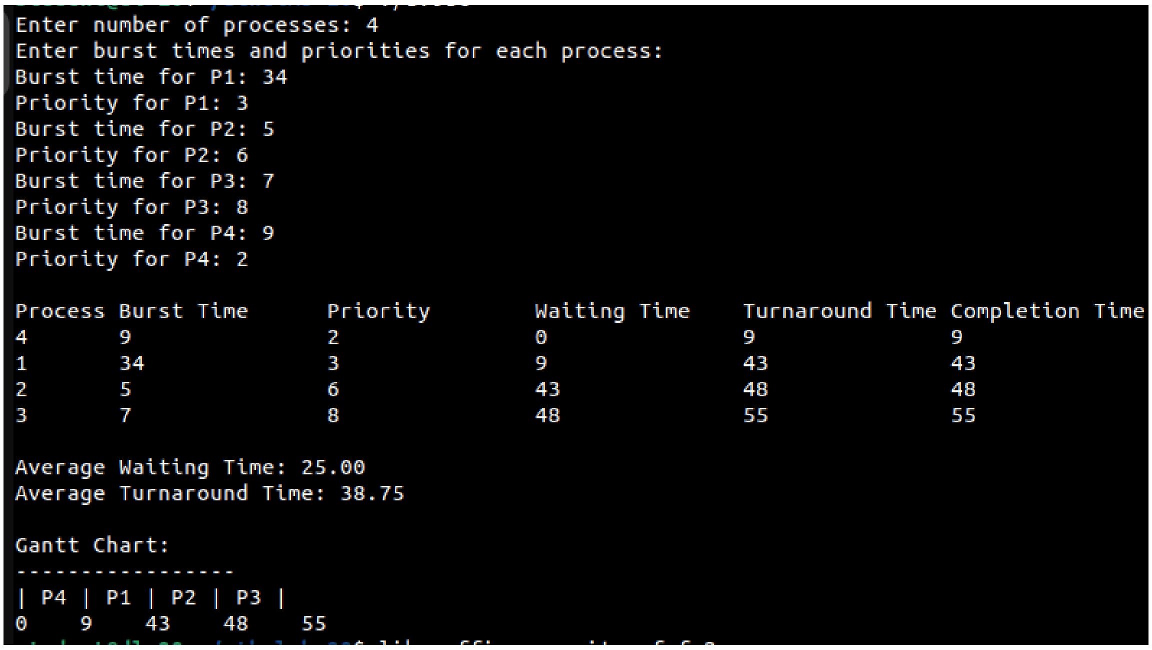
```c
#include <stdio.h>


void sortProcessesByPriority(int processes[], int bt[], int wt[], int tat[], int
ct[], int priority[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (priority[i] > priority[j]) {

                int temp = priority[i];
                priority[i] = priority[j];
                priority[j] = temp;


                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;


                temp = processes[i];
                processes[i] = processes[j];
                processes[j] = temp;
            }
        }
    }
}


void findWaitingTime(int processes[], int n, int bt[], int wt[]) {
    wt[0] = 0;
    for (int i = 1; i < n; i++) {
        wt[i] = bt[i - 1] + wt[i - 1];
    }
}


void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[]) {
    for (int i = 0; i < n; i++) {
        tat[i] = bt[i] + wt[i];
    }
}


void findCompletionTime(int processes[], int n, int bt[], int ct[]) {
    ct[0] = bt[0];
    for (int i = 1; i < n; i++) {
        ct[i] = ct[i - 1] + bt[i];
    }
}


void findAvgTime(int processes[], int n, int bt[], int priority[]) {
    int wt[n], tat[n], ct[n];


    sortProcessesByPriority(processes, bt, wt, tat, ct, priority, n);


    findWaitingTime(processes, n, bt, wt);


    findTurnAroundTime(processes, n, bt, wt, tat);
    findCompletionTime(processes, n, bt, ct);

    int total_wt = 0, total_tat = 0;

    printf("\nProcess\tBurst Time\tPriority\tWaiting Time\tTurnaround Time\
tCompletion Time\n");
    for (int i = 0; i < n; i++) {
        total_wt += wt[i];
        total_tat += tat[i];
        printf("%d\t%d\t\t%d\t\t%d\t\t%d\t\t%d\n", processes[i], bt[i],
priority[i], wt[i], tat[i], ct[i]);
    }
    printf("\nAverage Waiting Time: %.2f", (float)total_wt / n);
    printf("\nAverage Turnaround Time: %.2f\n", (float)total_tat / n);
}


void ganttChart(int processes[], int n, int bt[], int ct[]) {
    printf("\nGantt Chart:\n");
    printf("-----------------\n");

    int current_time = 0;
    for (int i = 0; i < n; i++) {
        printf("| P%d ", processes[i]);
        current_time += bt[i];
    }
    printf("|\n");

    current_time = 0;
    printf("0");
    for (int i = 0; i < n; i++) {
        current_time += bt[i];
        printf("    %d", current_time);
    }
    printf("\n");
}

int main() {
    int n;

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

    int processes[n], burst_time[n], priority[n];


    for (int i = 0; i < n; i++) {
        processes[i] = i + 1;
    }


    printf("Enter burst times and priorities for each process:\n");
    for (int i = 0; i < n; i++) {
        printf("Burst time for P%d: ", processes[i]);
        scanf("%d", &burst_time[i]);

        printf("Priority for P%d: ", processes[i]);
        scanf("%d", &priority[i]);
    }


    findAvgTime(processes, n, burst_time, priority);
    ganttChart(processes, n, burst_time, burst_time);

    return 0;
}
```



```
Enter number of processes: 4
Enter burst times and priorities for each process:
Burst time for P1: 34
Priority for P1: 3
Burst time for P2: 5
Priority for P2: 6
Burst time for P3: 7
Priority for P3: 8
Burst time for P4: 9
Priority for P4: 2

Process Burst Time      Priority        Waiting Time    Turnaround Time Completion Time
4       9               2               0               9               9
1       34              3               9               43              43
2       5               6               43              48              48
3       7               8               48              55              55

Average Waiting Time: 25.00
Average Turnaround Time: 38.75

Gantt Chart:
-----------------
| P4 | P1 | P2 | P3 |
0    9    43    48    55
```