```c
#include <stdio.h>
#include <stdlib.h>
#define MAX_PROCESSES 10
typedef struct {
    int pid;
    int arrival;
    int burst;
} Process;
void fcfsScheduling(Process queue[], int n) {
    int waiting_time = 0, turnaround_time = 0, completion_time = 0;
    int total_turnaround = 0, total_waiting = 0; // Variables to store totals
    printf("\nPID\tArrival\tBurst\tCompletion\tTurnaround\tWaiting\n");
    for (int i = 0; i < n; i++) {
        if (completion_time < queue[i].arrival) {
            completion_time = queue[i].arrival;
        }
        completion_time += queue[i].burst;
        int turnaround = completion_time - queue[i].arrival;
        int waiting = turnaround - queue[i].burst;

        total_turnaround += turnaround;
        total_waiting += waiting;

        printf("%d\t%d\t%d\t%d\t\t%d\t\t%d\n", queue[i].pid, queue[i].arrival, queue[i].burst,
               completion_time, turnaround, waiting);
    }
     printf("\nAverage Turnaround Time: %.2f\n", (float)total_turnaround / n);
    printf("Average Waiting Time: %.2f\n", (float)total_waiting / n);
}
void multiLevelQueueScheduling(Process systemQueue[], int sysCount, Process userQueue[], int userCount) {
    printf("\nExecuting System Processes (Higher Priority):");
    fcfsScheduling(systemQueue, sysCount);

    printf("\nExecuting User Processes (Lower Priority):");
    fcfsScheduling(userQueue, userCount);
}
int main() {
    int n, sysCount = 0, userCount = 0;
    Process systemQueue[MAX_PROCESSES], userQueue[MAX_PROCESSES];
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        Process p;
        int type;
        printf("\nEnter details for process %d\n", i + 1);
        printf("Process ID: ");
        scanf("%d", &p.pid);
        printf("Arrival Time: ");
        scanf("%d", &p.arrival);
        printf("Burst Time: ");
        scanf("%d", &p.burst);
        printf("Enter process type (0 - System, 1 - User): ");
        scanf("%d", &type);

        if (type == 0) {
            systemQueue[sysCount++] = p;
        } else {
            userQueue[userCount++] = p;
        }
    }
    multiLevelQueueScheduling(systemQueue, sysCount, userQueue, userCount);
    return 0;
}
```

```
PS C:\Users\STUDENT\Desktop\Archita OS lab> cd "c:\Users\STUDENT\Desktop\Archita OS lab\" ; if ($?) { gcc multi1.c -o multi1 } ; if ($?) { .\multi1 }
Enter the number of processes: 4

Enter details for process 1
Process ID: 1
Arrival Time: 0
Burst Time: 4
Enter process type (0 - System, 1 - User): 0

Enter details for process 2
Process ID: 2
Arrival Time: 0
Burst Time: 3
Enter process type (0 - System, 1 - User): 0

Enter details for process 3
Process ID: 3
Arrival Time: 0
Burst Time: 8
Enter process type (0 - System, 1 - User): 1

Enter details for process 4
Process ID: 4
Arrival Time: 0
Burst Time: 5
Enter process type (0 - System, 1 - User): 0

Executing System Processes (Higher Priority):
PID     Arrival Burst   Completion      Turnaround      Waiting
1       0       4       4               4               0
2       0       3       7               7               4
4       0       5       12              12              7

Average Turnaround Time: 7.67
Average Waiting Time: 3.67

Executing User Processes (Lower Priority):
PID     Arrival Burst   Completion      Turnaround      Waiting
3       0       8       8               8               0

Average Turnaround Time: 8.00
Average Waiting Time: 0.00
PS C:\Users\STUDENT\Desktop\Archita OS lab>
```