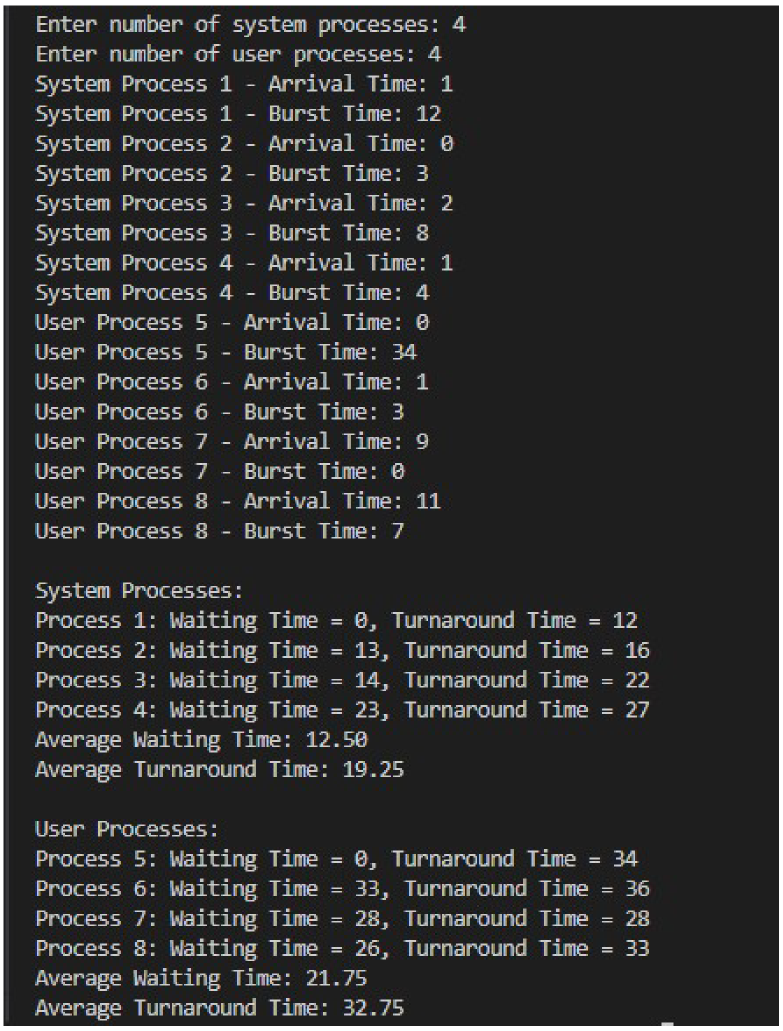
Write a C program to simulate multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided into two categories – system processes and user processes. System processes are to be given higher priority than user processes. Use FCFS scheduling for the processes in each queue.

```
#include <stdio.h>
#define MAX_PROCESSES 10


typedef struct {     int process_id, arrival_time, burst_time, completion_time, turn_around_time,
waiting_time; } PCB;


void FCFS(PCB *queue, int n) {
   int time = 0;     for (int
i = 0; i < n; i++) {
       if (queue[i].arrival_time > time) time = queue[i].arrival_time;
queue[i].completion_time = time + queue[i].burst_time;
queue[i].turn_around_time = queue[i].completion_time - queue[i].arrival_time;
queue[i].waiting_time = queue[i].turn_around_time - queue[i].burst_time;        time
= queue[i].completion_time;
   }
}


void print_results(PCB *queue, int n) {
int total_wt = 0, total_tat = 0;
   for (int i = 0; i < n; i++) {
       printf("Process %d: Waiting Time = %d, Turnaround Time = %d\n",
queue[i].process_id, queue[i].waiting_time, queue[i].turn_around_time);
total_wt += queue[i].waiting_time;

       total_tat += queue[i].turn_around_time;
   }
```

```
    printf("Average Waiting Time: %.2f\n", (float)total_wt / n);
printf("Average Turnaround Time: %.2f\n", (float)total_tat / n);
}

int main() {     int system_count,
user_count;
    PCB system_queue[MAX_PROCESSES], user_queue[MAX_PROCESSES];

    printf("Enter number of system processes: ");
scanf("%d", &system_count);     printf("Enter
number of user processes: ");     scanf("%d",
&user_count);

    for (int i = 0; i < system_count; i++) {
system_queue[i].process_id = i + 1;
printf("System Process %d - Arrival Time: ", i + 1);
scanf("%d", &system_queue[i].arrival_time);
printf("System Process %d - Burst Time: ", i + 1);
scanf("%d", &system_queue[i].burst_time);
    }

    for (int i = 0; i < user_count; i++) {         user_queue[i].process_id
= system_count + i + 1;         printf("User Process %d - Arrival
Time: ", system_count + i + 1);         scanf("%d",
&user_queue[i].arrival_time);         printf("User Process %d - Burst
Time: ", system_count + i + 1);         scanf("%d",
&user_queue[i].burst_time);
    }

    FCFS(system_queue, system_count);
    FCFS(user_queue, user_count);
    printf("\nSystem Processes:\n");
print_results(system_queue, system_count);
printf("\nUser Processes:\n");     print_results(user_queue,
user_count);


    return 0;
}
```

OUTPUT:



```
Enter number of system processes: 4
Enter number of user processes: 4
System Process 1 - Arrival Time: 1
System Process 1 - Burst Time: 12
System Process 2 - Arrival Time: 0
System Process 2 - Burst Time: 3
System Process 3 - Arrival Time: 2
System Process 3 - Burst Time: 8
System Process 4 - Arrival Time: 1
System Process 4 - Burst Time: 4
User Process 5 - Arrival Time: 0
User Process 5 - Burst Time: 34
User Process 6 - Arrival Time: 1
User Process 6 - Burst Time: 3
User Process 7 - Arrival Time: 9
User Process 7 - Burst Time: 0
User Process 8 - Arrival Time: 11
User Process 8 - Burst Time: 7

System Processes:
Process 1: Waiting Time = 0, Turnaround Time = 12
Process 2: Waiting Time = 13, Turnaround Time = 16
Process 3: Waiting Time = 14, Turnaround Time = 22
Process 4: Waiting Time = 23, Turnaround Time = 27
Average Waiting Time: 12.50
Average Turnaround Time: 19.25

User Processes:
Process 5: Waiting Time = 0, Turnaround Time = 34
Process 6: Waiting Time = 33, Turnaround Time = 36
Process 7: Waiting Time = 28, Turnaround Time = 28
Process 8: Waiting Time = 26, Turnaround Time = 33
Average Waiting Time: 21.75
Average Turnaround Time: 32.75
```