6/6/24 Week 5

write a C program to simulate multi-level queue scheduling algorithm considering the following scenario. All the processes in the system are divided in 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>
#include <stdlib.h>
#define MAX_QUEUE_SIZE 100
// structure to represent a process
typedef struct {
    int pid;
    int priority;
    int arrival_time;
    int burst_time;
} Process;
// Fn to swap two processes
void swap(Process *a, Process *b) {
    Process temp = *a;
    *a = *b;
    *b = temp;
}
void fcfs(Process queue[], int size) {
    int i, j;
    for(i=0; i<size-1; i++) {
        for(j=0; j<size-i-1; j++) {
            if (queue[j].arrival_time > queue[j+1].arrival_time) {
                swap(&queue[j], &queue[j+1]);
            }
        }
    }
}
```

```
void multi_level_queue_scheduling( Process processes[], int num_processes){
Process system_queue [MAX_QUEUE_SIZE];
Process user_queue[ MAX_QUEUE_SIZE];
int system_count =0, user_count =0;
for(int i=0; i<num_processes; i++){
    if (processes[i].priority==1){
        system_queue[system_count++]=
                processes[i];
    }else{
        user_queue[user_count++] =
                processes[i];
    }
}
fcfs( system_queue, system_count);
fcfs( user_queue, user_count);
printf("--------------------\n");
printf("|Process| Priority| Arrival Time| Burst Time|\n");
printf("--------------------\n");
for(int i=0;i<system_count;i++){
    printf("| %d | %d | %d | %d |\n",
        system_queue[i].pid, system_queue[i].priority, system_queue[i].arrival_time,
        system_queue[i].burst_time);
}
for(int i=0; i<user_count ;i++){
```

```
printf("%d | %d | %d | %d | %d | %d\n",
    user_queue[i].pid, user_queue[i].priority,
    user_queue[i].arrival_time, user_queue[i].
                                burst_time);
}
printf("-------------------------------\n");
}
int main(){
    int num_processes;
    printf("Enter the number of processes:");
    scanf("%d", &num_processes);
    Process processes[num_processes];
    for(i=0; i<num_processes; i++){
        printf("Enter details for Process %d:\n",
                                    i+1);
        processes[i].pid = i+1;
        printf("Priority (1 for system, 0 for
                                    user):");
        scanf("%d", &processes[i].priority);
        printf("Arrival Time:");
        scanf("%d", &processes[i].arrival_
                                    time);
        printf("Burst time:");
        scanf("%d", &processes[i].burst_time);
    }
    printf("\n Multilevel Queue Scheduling Result:\n");
    multi_level_queue_scheduling(processes, num_proces
    return 0;
}
```

output:

Enter the number of processes: 4

Enter details for Process 1:

Priority (1 for system, 0 for user): 1

Arrival Time: 0

Burst Time: 2

Enter details for Process 2:

Priority (1 for system, 0 for user): 1

Arrival Time: 2

Burst time: 5

Enter details for Process 3:

Priority (1 for system, 0 for user): 0

Arrival Time: 3

Burst Time: 5

Enter details for Process 4:

Priority (1 for system, 0 for user): 0

Arrival Time: 6

Burst Time: 5

Multi-level Queue Scheduling Result:

Process	Priority	Arrival Time	Burst Time
1	1	0	2
2	1	2	5
3	0	3	5
4	0	6	5

—— X ——

6/6/24