

ASSIGNMENT-4

**1. Write a C program to simulate a 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>
#include <stdlib.h>
#define MAX 10

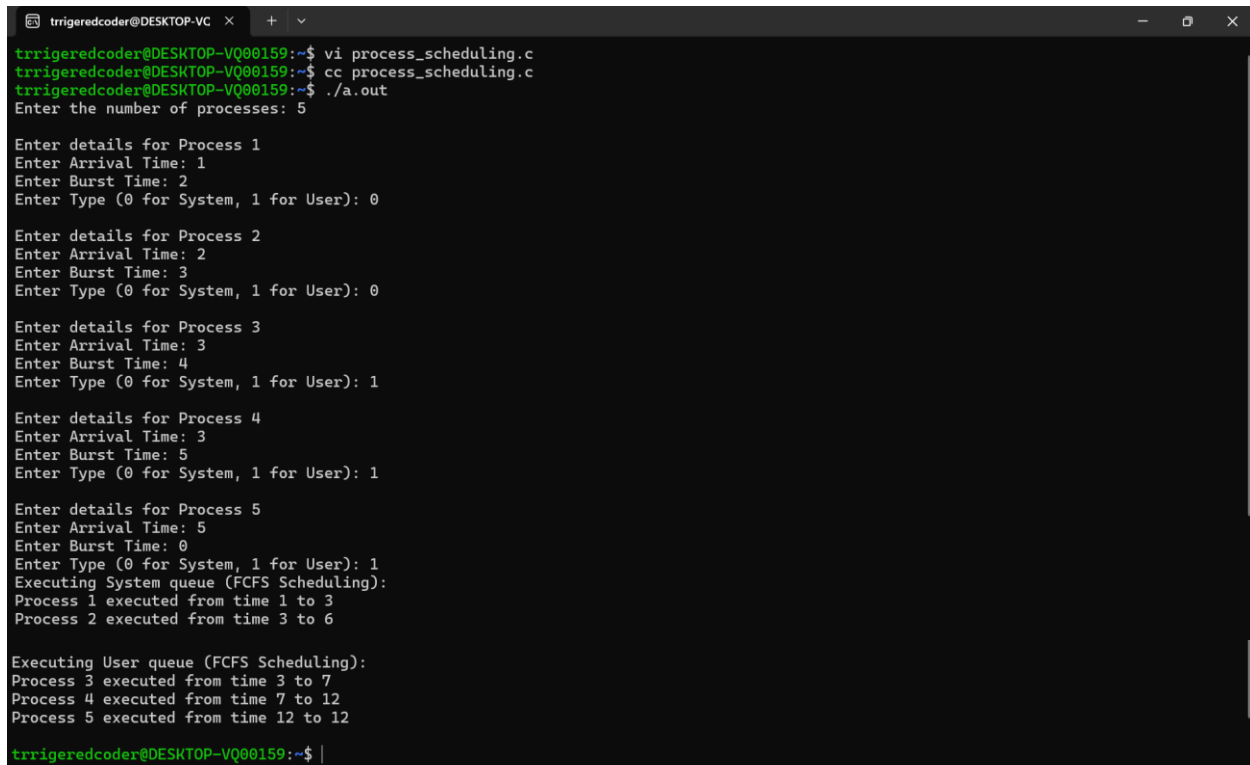
typedef struct {
    int id;
    int arrival_time;
    int burst_time;
} Process;

void sortByArrivalTime(Process queue[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (queue[j].arrival_time > queue[j + 1].arrival_time) {
                Process temp = queue[j];
                queue[j] = queue[j + 1];
                queue[j + 1] = temp;
            }
        }
    }
}

void executeQueue(Process queue[], int n, const char* queueName) {
    printf("Executing %s queue (FCFS Scheduling):\n", queueName);
    int time = 0;
    for (int i = 0; i < n; i++) {
        if (time < queue[i].arrival_time) {
            time = queue[i].arrival_time;
        }
        printf("Process %d executed from time %d to %d\n", queue[i].id, time, time + queue[i].burst_time);
        time += queue[i].burst_time;
    }
    printf("\n");
}

int main() {
    Process systemQueue[MAX], userQueue[MAX];
    int systemCount = 0, userCount = 0, n;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        int type;
        Process p;
        printf("\nEnter details for Process %d\n", i + 1);
        p.id = i + 1;
        printf("Enter Arrival Time: ");
        scanf("%d", &p.arrival_time);
        printf("Enter Burst Time: ");
        scanf("%d", &p.burst_time);
        printf("Enter Type (0 for System, 1 for User): ");
```

```
scanf("%d", &type);
if (type == 0) {
    systemQueue[systemCount++] = p;
}
else {
    userQueue[userCount++] = p;}}
sortByArrivalTime(systemQueue, systemCount);
sortByArrivalTime(userQueue, userCount);
executeQueue(systemQueue, systemCount, "System");
executeQueue(userQueue, userCount, "User");
return 0;
}
```



```
trrigeredcoder@DESKTOP-VC x + v
trrigeredcoder@DESKTOP-VQ00159:~$ vi process_scheduling.c
trrigeredcoder@DESKTOP-VQ00159:~$ cc process_scheduling.c
trrigeredcoder@DESKTOP-VQ00159:~$ ./a.out
Enter the number of processes: 5

Enter details for Process 1
Enter Arrival Time: 1
Enter Burst Time: 2
Enter Type (0 for System, 1 for User): 0

Enter details for Process 2
Enter Arrival Time: 2
Enter Burst Time: 3
Enter Type (0 for System, 1 for User): 0

Enter details for Process 3
Enter Arrival Time: 3
Enter Burst Time: 4
Enter Type (0 for System, 1 for User): 1

Enter details for Process 4
Enter Arrival Time: 3
Enter Burst Time: 5
Enter Type (0 for System, 1 for User): 1

Enter details for Process 5
Enter Arrival Time: 5
Enter Burst Time: 0
Enter Type (0 for System, 1 for User): 1
Executing System queue (FCFS Scheduling):
Process 1 executed from time 1 to 3
Process 2 executed from time 3 to 6

Executing User queue (FCFS Scheduling):
Process 3 executed from time 3 to 7
Process 4 executed from time 7 to 12
Process 5 executed from time 12 to 12
trrigeredcoder@DESKTOP-VQ00159:~$ |
```