# CPE 460 Quiz 02: CPU Scheduling

Department of Computer Engineering
Yarmouk University

Spring 2017

Last Come First Serve (LCFS) is a special purpose CPU scheduling algorithm. The LCFS scheduling algorithm serves the process that comes last first. The code in `lcfs.c` simulates the LCFS scheduling algorithm and computes the time parameters for each served process. Based on the next page, answer the following questions:

1. In `main` function there are four errors will cause improper execution for the program. Find and fix them.

2. Function `initializeProcesses` must initialize the procces array properly to avoid any uninitizlied fields in the `process` structure before the application of LCFS. Dose it initialize the process array properly? If NO, add the proper changes.

3. Function `lcfsScheduling` must apply LCFS on the process array, however, there are four *logical* errors in the algorithm. Find and fix them.

4. Complete the time table for the served processes based on the correct behaviour of the LCFS algorithm.

```
[enas@cpe]~$ cat lcfs.c

#include <stdio.h>
#include <stdlib.h>
struct process{
    int pid;
    int burstTime;
    int waitingTime;
    int turnaroundTime;
};

void initializeProcesses(struct process processes[], int processCount);
void lcfsScheduling(struct process processes[], int processCount);

void main(){
    // 1- Prompt the user to enter the number of processes
    printf("Please, enter the number of processes:");
    int n;
    scanf("%d",n);

    // 2-  Allocate an array of struct process to accomadate the number ofuser processes
    struct process P[20];

    // 3- Prompt the user to enter the burstTime for each process
    printf("Please, enter the burst time for %d processes\n",n);
    initializeProcesses(P,20);

    // 4- Call Last Come First Serve lcfsScheduling
    lcfsScheduling(P,n);

    // 5- Print the data for all processes
    printf("PID\tBurstT\tWaitT\tTurnaroundT\n");
    int i = 0;
    for(i = 1; i <= n; i++){
        printf("%d\t%d\t%d\t%d\t\n",P[i].pid,P[i].burstTime,P[i].waitingTime,P[i].turnaroundTime);
    }


}

void lcfsScheduling(struct process processes[], int processCount){
    processes[0].waitingTime = 0;
    processes[0].turnaroundTime = processes[0].burstTime;
    int i = 0;
    for(i=processCount-2; i>=0 ;i--){
        processes[i].waitingTime = processes[i-1].turnaroundTime;
        processes[i].turnaroundTime = processes[i].waitingTime + processes[i].burstTime;
    }
}
```

```
void initializeProcesses(struct process processes[], int processCount){
    int i = 0;
    for(i = 0; i < processCount; i++){
        printf("Enter the burst time for P%d:", i);
        scanf("%d",&processes[i].burstTime);

    }
}
```

```
[enas@cpe]~$  gcc -o run lcfs.c
[enas@cpe]~$ ./run
Please, enter the number of processes:4
Please, enter the burst time for 4 processes
Enter the burst time for P0:10
Enter the burst time for P1:4
Enter the burst time for P2:11
Enter the burst time for P3:6
PID     BurstT  WaitT   TurnaroundT
0       10      ---      ---
1       4       17      21
2       11      ---      ---
3       6       0       6
```