Project TA – Peter CHUNG (cspeter@cse.ust.hk): Handling questions before the deadline
Grader TA - Lingyun YANG (lyangbk@cse.ust.hk) Grading and appeal handling after the deadline

COMP3511 Operating System (Fall 2023)

PA2: Simplified Multi-level Feedback Queue (MLFQ)

Released on 12-Oct (Thu)   Due on 02-Nov (Thu) at 23:59

Introduction

Multi-level feedback queue (MLFQ) is a process scheduler that a process can move between the various queues via aging. In this project assignment, you only need to implement a 3-level feedback queue with at most 10 processes. Upon completion of the project, students should be able to understand the details of MLFQ and how to implement a 3-level MLFQ.

You are highly recommended to attend the corresponding project-related lab.

Program Usage

You need to implement a program that simulates a MLFQ to schedule several processes. The program name is `mlfq`. Here is the sample usage:

```
$> ./mlfq < input.txt > output.txt
```

`$>` represents the shell prompt.
< means input redirection, which is used to redirect the file content as the standard input
> means output redirection, which is used to redirect the standard output to a text file
Thus, you can easily use the given test cases to test your program and use the `diff` command to compare your output files with the sample output files.

Getting Started

`mlfq_skeleton.c` is provided. You should rename the file as `mlfq.c`

You can add new constants, variables, and helper functions
Necessary header files are included. You should not add extra header files.
Some constants and helper functions are provided in the starter code.
Please read the skeleton code carefully.

Assumptions

- There are at most 10 different processes
- **The process table is sorted by the arrival time**
- The arrival time of the first process is always equal to 0
- No processes have the same arrival time
- The CPU won't be idle for the entire scheduling of the processes
- There are at most 300 steps in the Gantt chart

Input and Output Format

The input parsing is given in the skeleton code.
Empty lines and lines starting with # are ignored.

A sample input file:

```
# An input file for a Simplified Multi-level Feedback Queue (MLFQ)
# Empty lines and lines starting with '#' are ignored

tq0 = 2
tq1 = 4

# The process table definition
process_table_size = 4
process_table =
P1 0 7
P2 1 1
P3 4 1
P4 6 3
```

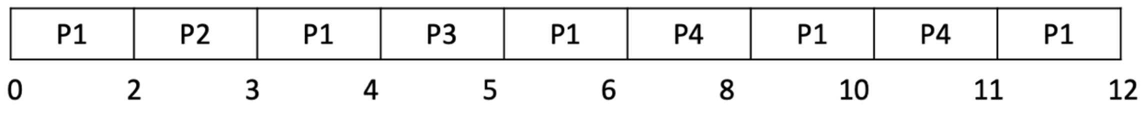
The output consists of 2 regions:

- Displaying the parsed values from the input
- Displaying the final Gantt chart

The sample output file based on the sample input file:

```
tq0 = 2
tq1 = 4
process_table =
Process Arrival Burst
P1      0       7
P2      1       1
P3      4       1
P4      6       3
Gantt Chart = 0 P1 2 P2 3 P1 4 P3 5 P1 6 P4 8 P1 10 P4 11 P1 12
```

The above Gantt chart string is equivalent to the following Gantt chart:



Project TA – Peter CHUNG (cspeter@cse.ust.hk): Handling questions before the deadline
Grader TA - Lingyun YANG (lyangbk@cse.ust.hk) Grading and appeal handling after the deadline

Compilation

The following command can be used to compile the program

```
$> gcc -std=c99 -o mlfq mlfq.c
```

The option `c99` is used to provide a more flexible coding style (e.g., you can define an integer variable anywhere within a function)

Sample test cases

Several test cases (both input files and output files) are provided. We don't have other hidden test cases.

The grader TA will probably write a grading script to mark the test cases. Please use the Linux `diff` command to compare your output with the sample output. For example:

```
$> diff --side-by-side your-outX.txt  sample-outX.txt
```

Sample Executable

The sample executable (runnable in a CS Lab 2 machine) is provided for reference. After the file is downloaded, you need to add an execution permission bit to the file. For example:

```
$> chmod u+x mlfq
```

Development Environment

CS Lab 2 is the development environment. Please use one of the following machines (`csl2wk`**`XX`**`.cse.ust.hk`), where **XX**=`01…40`. The grader will use the same platform.

In other words, *"my program works on my own laptop/desktop computer, but not in one of the CS Lab 2 machines"* is an invalid appeal reason. **Please test your program on our development environment (not on your own desktop/laptop) thoughtfully** before your submission, even you are running your own Linux OS. Remote login is supported on all CS Lab 2 machines, so you are not required to be physically present in CS Lab 2.

Project TA – Peter CHUNG (cspeter@cse.ust.hk): Handling questions before the deadline
Grader TA - Lingyun YANG (lyangbk@cse.ust.hk) Grading and appeal handling after the deadline

Marking Scheme

1. Please fill in your name, ITSC email, and declare that you do not copy from others. A template is already provided near the top of the source file.

2. (50%) Correctness of the **5** provided test cases.

   We will check the source codes to avoid students hard coding the test cases in their programs. Example of hard coding: a student may simply detect the input and then display the corresponding output without implementing the program. It may happen because some given test cases are provided. 0 marks will be given if you hardcode the test cases in your program.

3. (50%) Correctness of the **5** hidden test cases.

   They are like the provided test cases but using different input values. They help detect hardcoding. For example, the chance of hardcoding is very high if all provided test cases are passed, but all hidden test cases are failed.

4. Make sure you use the Linux diff command to check the output format.
5. Make sure to test your program in one of our CS Lab 2 machines (not your own desktop/laptop computer)

***Plagiarism**: Both parties (i.e., students providing the codes and students copying the codes) will receive 0 marks. Near the end of the semester, a plagiarism detection software (JPlag) will be used to identify cheating cases. **DON'T** do any cheating!*

Submission

File to submit:

`mlfq.c`

Please check carefully you submit the correct file. Canvas will do auto-renaming if you submit multiple times. You don't need to worry about the Canvas renaming.
In the past semesters, some students submitted the executable file instead of the source file. Zero marks will be given as the grader cannot grade the executable file.

You are not required to submit other files, such as the input test cases.

Late Submission

For late submission, please submit it via email to the grader TA.
There is a 10% deduction, and only 1 day late is allowed
(Reference: Chapter 1 of the lecture notes)