COMP3520 Operating Systems Internals Assignment 1 – Synchronization

General Instructions

This assignment is about synchronization. It consists of two **compulsory** tasks:

1. Write a multithreaded program to solve the synchronization problem as described in the section "The Problem: Group Lab Exercise"; and
2. Answer the assigned discussion document questions (which are provided in a separate document).

This assignment is an individual assignment. Whilst you are permitted to discuss this assignment with other students, the work that you submit must be your own work. You should not incorporate the work of other students (past or present) into your source code or discussion document.

You will be required to submit your discussion document to *Turnitin* for similarity checking as part of assignment submission. The examiner may use other similarity checking tools in addition to *Turnitin*. Your source code may also be checked.

Submit your source code and report to the appropriate submission inboxes in the COMP3520 Canvas website.

The Problem: Group Lab Exercise

There are ***N* students** in the class to do lab exercises. The students are divided into ***M* groups**. Each group has a group *id* ranging from 0 to *M*-1. Each group will have *N/M* students if *N* is divisible by *M*. If *N* is not divisible by *M*, the first *N%M* groups will have ⌊*N/M*⌋+1 students each and the remaining groups will have ⌊*N/M*⌋ students each. **One teacher** *randomly* assigns the students into different groups after all students have arrived.

There are ***K* lab rooms**. Each lab room has one tutor who assists students to conduct the exercise in the lab (there are a total of ***K* tutors** and tutor's *id* = lab room's *id*). A lab room can only hold one group of students at a time.

The students may enter one of the *K* lab rooms to conduct their lab exercise only when their group *id* is called by the teacher. The teacher will call a group *id* once there are empty rooms and there is a waiting group. The teacher calls groups to conduct the lab exercise in natural group *id* order, i.e., first, group 0; then group 1, 2, and so forth. A group may take a minimum of *T*/2 units of time and a maximum of *T* units of time to complete their lab exercise (in your program, a unit of time is assumed to be one second).

Once a group has completed the exercise, all students in that group must leave the lab room immediately. After this room is *completely* vacated, the tutor will report to the teacher the lab availability. The teacher will then call another group of students to enter the vacated room for their lab exercise. If there are more than one empty room, the tutors are queued in a first-in-first-out order for the teacher to assign new student groups to them.

To solve this synchronization problem, **you may use pthreads mutexes and condition variables**. **However, you must not use any other type of synchronization mechanisms.**

You program needs to ask the user to enter the following parameters:

- *N*: the total number of students in the class;
- *M*: the number of groups;
- *K*: The number of tutors, or lab rooms; and
- *T*: the time limit for each group of students to do the lab exercise.

As mentioned previously, each student thread is assigned a group *id* **by the teacher thread** (not by the main default thread when the student threads are created).

To check whether your program functions correctly, the following status messages must be printed out by the students, teacher and tutors, respectively.

When the teacher is waiting for all students to arrive:

- "Teacher: I'm waiting for all students to arrive."
- "Student [*id*]: I have arrived and wait for being assigned to a group."

When the teacher assigns group *id*s to students:

- "Teacher: All students have arrived. I start to assign group ids to students."
- "Teacher: student [*id*] is in group [*id*]."
- "Student [*id*]: OK, I'm in group [*id*] and waiting for my turn to enter a lab room."

When assigning lab rooms to groups after students are assigned to groups:

- "Teacher: I'm waiting for lab rooms to become available."
- "Tutor [*id*]: The lab room [*id*] is vacated and ready for one group."
- "Teacher: The lab [*id*] is now available. Students in group [*id*] can enter the room and start your lab exercise."
- "Student [*id*] in group [*id*]: My group is called. I will enter the lab room [*id*] now."
- "Tutor [*id*]: All students in group [*id*] have entered the room [*id*]. You can start your exercise now." **//after all students in the group have entered the room**
- "Tutor [*id*]: Students in group[*id*] have completed the lab exercise in [*t*] units of time. You may leave this room now."
- Student [*id*] in group [*id*]: Thanks Tutor [*id*]. Bye!"
- "Teacher: There are no students waiting. Tutor [*id*], you can go home now."
- "Tutor[*id*]: Thanks Teacher. Bye!"
- "Teacher: All students and tutors are left. I can now go home."

When all students have completed lab exercises and tutors have left, the main thread must print the following status message and then terminate:

- "Main thread: All students have completed their lab exercises. This is the end of simulation."

Additional Requirements

Source Code

Your solution must be implemented in the C/C++ language.

Your source code needs to be properly commented and appropriately structured to allow programmers who have a working knowledge of C/C++ to understand and easily maintain your code.

You need to include a well-structured and properly commented *makefile* that allows for the compilation of your source code using the *make* command on the School of Computer Science servers.

Testing and Debugging

You are responsible for testing and debugging your source code.

It is crucial that you ensure that the source code you submit compiles correctly on the School of Computer Science servers and that the resulting binary functions as intended. If your submitted source codes cannot be compiled on the School servers, marks will be heavily deducted.

Discussion Document

You are required to answer all assigned questions in a separate written document. The questions and requirements specific to the discussion document will be provided in a separate document.

Use of Writing Assistance and Generative AI Tools Policy

You are permitted to use writing assistance and generative artificial intelligence (AI) tools to help you develop your program and write your report. You are required to complete and submit the "Documentation of Writing Assistance and Generative AI Tool Usage Form", even if you do not use writing assistance or generative AI tools at all. You will not lose marks for using these tools as long as you properly document your use of them.

Other Matters

Marking criteria for the source code and discussion document will be provided separately.

To maximize your chances of realizing your full potential in COMP3520, **please start work on this assignment promptly.**