Programming Exercise 2 – Teacher assigns students to groups

From this week you will tackle synchronization problems directly related to Assignment 1. Therefore, no suggested solution programs will be given.

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

To solve this synchronization problem, you need to write two thread routines *teacher_routine()* and *student_routine()* in addition to *main()*.

You may use pthread mutexes and condition variables. **However, you must not use any other type of synchronization mechanisms.**

In the *main()* function, your program needs to ask the user to supply the following parameters:

- *N*: the total number of students in the class;
- *M*: the number of groups;

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

After the assignment of group ids is completed:

- "Teacher: Students can leave now."
- "Student [*id*]: Bye Teacher."
- "Teacher: I can now go home."

At the end the main thread print:

- "Main thread: This the end of simulation."