# Parallel and MultiThreaded Programming

# CSYE 7215

# Homework 9

Due: March 31, 2020

Put all your java, compiled class files and documentation into zip file named Homework9.zip and submit it via the dropbox on the blackboard before the END of due date. Put your name on all .java files. There will be a short quiz on this assignment.

1. a) Provide examples of Atomic and Non-Atomic operations. Explain

b) What are the root causes of Torn-read or Torn-write? Give examples in either Java or C++

2. Write Mutex lock mechanism to protect two Threads in this C++ Posix program:

http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html

3. This C++ program is similar to Java wait/notify using condition\_variable library.

Compile and run this code, and explain how it works.

<https://www.modernescpp.com/index.php/condition-variables>

4. C++ Atomic library provides a set of operations, atomic\_bool, atomic\_short,

atomic\_int, atomic\_long, etc: <http://www.cplusplus.com/reference/atomic/>

a) Compile and run this program:

<http://www.cplusplus.com/reference/atomic/atomic/atomic/>

b) Change atomic\_bool operation with atomic\_short in this program.

c) Compile and run program.

4. Define a Student class with instance variables name, id, midterm, project, and final-exam. Name is a string whereas others are all integers. Also add a static variable nextId which is an integer and statically initialized to 1. In each of them, the id should be assigned to the next available id given by nextId. The default constructor should set the name of the student object to “StudentX” where X is the next id.

A) Your program is to create 32 Student Threads each to be identified with name-Thread-nextId. The default constructor for each thread calls a method to randomly generate grades for midterm, project, and final-exam ranging between 50 to 100 inclusive. You need to consider 1 second wait-time between each score generation for midterm, project, and final-exam. Each student thread writes the grade scores to “Grades” file in this format: name, nextId, ThreadId, midterm, project, finalExam. All student threads share this file and you need to protect it.

B) Create GraderThread that checks “Grades” file periodically up to 30 seconds to retrieve submitted grades by all student threads. How do you protect the file? The GraderThread reads the file (format described above) and validates name, id, threadId, scores for all 25 student threads submitted scores. For any missing grade, the student will receives zero score. The GraderThread does calculateGrade() (30% midterm + 30% project + 40% final) and returns a letter grade like “A”, “B”, “C”, “D” or “F”, based on the overall score.

C) Create GradesTestDriver class to create 50, 100, 150 Student Threads and one GraderThread to test your program.

D) a) Write this program with Java and then with C++

b) Use Explicit Locking for both Java and C++

c) Evaluate the code Performance between Java and C++

d) Report Results with all performance differences

e) Discuss as to why one performance is better than the other