# Parallel and MultiThreaded Programming

# CSYE 7215

# Homework 5

# Due: February 16, 2020

Put all your java, compiled class files and documentation into zip file named Homework5.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. Provide four numbers (byte, short, int, long) number examples. Show the results for Signed and Unsigned arithmetic. In each case, in binary show the base, position, and value in that position.

2. Consider Class File data structure, Explain each case with an Example:

a) magic, b) constant\_pool, c) super\_class, d) interfaces, e) fields, f) methods, g) attributes

ClassFile {

u4 magic;

u2 minor\_version;

u2 major\_version;

u2 constant\_pool\_count;

cp\_info constant\_pool[constant\_pool\_count-1];

u2 access\_flags;

u2 this\_class;

u2 super\_class;

u2 interfaces\_count;

u2 interfaces[interfaces\_count];

u2 fields\_count;

field\_info fields[fields\_count];

u2 methods\_count;

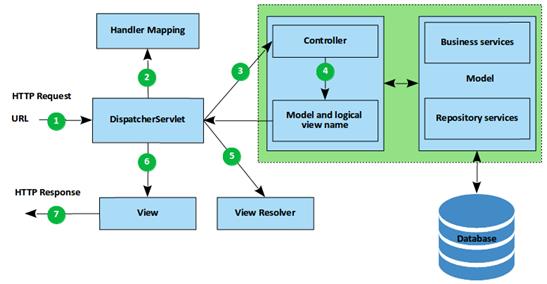
method\_info methods[methods\_count];

u2 attributes\_count;

attribute\_info attributes[attributes\_count];

}

3. Consider the following MVC model. Use this Model to identify and design three services for the University myNortheastern portal. A) You need to show the controller layer, service layer, and database layer for each of the services that you identify. B) Extend this diagram to show for all services.



4. You have learned how to create multi-threaded Student grading system using both implicit locking and explicit locking. Now you are to create student grading system using Socket client/server programming.

A) Consider example “A Network Tic-Tac-Toe Game” in the this article, analyze,

Compile and Run the code: <https://cs.lmu.edu/~ray/notes/javanetexamples/>

B) Use example model to build student grading system with 50 client student threads and one server GraderThread. Each client-student thread generates 4 scores (homework, quiz, midterm, final), and each score is randomly generated between 70 to 100. The scores are stored in ScoresMap as key/value, ({homework, score}, {quiz, score}, {midterm, score}, {final, score}). Then, each student thread connects to the GraderThread server (with incremental 100 milliseconds interval), and once it is connected to the server, it sends the scoresMap, studentId, firstName, lastName. The GraderServer receives student’s information: a) parses the ScoresMap for student scores, calculates letter grade A, B, C, D, F, and stores letter grade in its local cache GradesMap (threadId, grade), and then writes it to the file “FinalGrades”, and replies grade back to the client student thread. b) In subsequent student requests with missing/mistake grade (updated specific score in ScoresMap), the server parses the score map information, Verifies grade information in cache, calculates letter grade, updates cache, updates file, and replies the new grade back to student thread.