**Assignment 5**

*Design and implement* an application that mangers course schedules and course enrollments. Each course schedule contains information about the course number (such as CSC211, MAT306), academic year, semester (fall, spring, or summer), day (of a week), starting time and ending time, the name of the instructor, and the number of enrollments (assuming single-section classes only). Here are *essential features* of the application:

1. The application should be able to detect whether there is schedule conflict. Here is a (simplified) rule for schedule conflict: Any two courses (of the same semester within the same year) from the same academic program (CSC, MAT, ART, etc.) at the same level (100’s, 200’s, 300’s, or 400’s) can’t have overlapped times.
2. The application should provide operations to add (when the added schedule doesn’t cause schedule conflicts with other course schedules) and delete course schedules.
3. The application has the ability to modify an existing course schedule in terms of the day, the time, and/or the instructor.
4. The application has an operation for searching a schedule given a course number.
5. The application should provide an operation for enrolling in a class given necessary student information (say, name and id). As a result, the application should also track the enrolled students for each scheduled course, and provide operation to return a class roster given a course number.
6. Override toString() methods in ways that are meaningful.
7. When appropriate, override ‘equals’ and implement ‘compareTo()’ for the benefit of the application.

**Requirements:**

1. Provide a design solution for this application. Note that the specification above is still fluid, so feel free to make assumptions as you deem appropriate. In particular, none of the operation headers is given, so you design them. There is also no indication given as to what data structures to use, the collaborating classes you might need, and the service methods each class is responsible for, and this is where your design comes in.
2. You DO NOT need to provide a working program. The minimum requirement is 1. a UML diagram that include all classes/interfaces and their relations, 2. A UML sequence diagram for the operation “enroll student” (from collecting inputs to putting student info into an enrollment roster), and 3. Skeleton code that conveys the structure of the program without detailed implementation (like the skeleton code I demonstrated for assignment 2).
3. Prepare for a presentation (5 minutes) to show your design (UML diagrams and the skeleton code) in the next class. Save all artifacts in Word to a USB drive (and not forget to bring it to the class ☺).