Comp 271 – Think Before Code – OO Design

As in the course syllabus, late assignments will not be accepted unless you have completed the process for using a late pass before the due date.

1. Submit

Your complete UML class and sequence diagrams. Include commentary where helpful to understand how your design works. Your class and sequence diagrams must use meaningful names for all class attributes and methods and include parameter and return information. Sequence and class diagrams should be consistent.

Think of class diagrams as the Static / Unchanging / Piece parts of your code (and how they relate to each other). Sequence diagrams are the Dynamic / Run Time / Interworking instructions for your code.

2. Format

Use a tool of your choice such as draw.io. If you wish a more complete tool Visio is available from the department’s Microsoft Academic Alliance membership.

Submit a print of your two UML diagrams as .pdf or .jpg files. Name the files in the standard way with assignment name and your LUC uvid followed by a suitable description.

3. Content

Summary: Create the structure or architecture for a typical university registration system by dividing the system up into suitable classes, defining associations between classes to show their relationship, and defining all key members for each class. Create one or more sequence diagrams to show the complete implementation of one or more use cases (implementing one or more system events).

You do not need to complete the code for the system, only the complete design. The design must be sufficient to show how the key capabilities of the system will be implemented.

Your architecture or Object Oriented Design is to:
1. Structure a university registration system in which Departments offer Courses that are taught by Professors. Each Semester some Courses are offered as individual Sections with given meeting times, instructors, etc.

2. Students, or at least those in Good Standing, are allowed to register for their Courses by requesting specific Sections each Semester.

3. Eventually Students receive Grades for their completed courses.

4. Your system design must be sufficient for at least creating a Semester's course offerings and allowing Students to register for Courses in a Semester.

5. You should consider more capabilities as well to test your design. Grading is interesting. Ancillary functions such as generating transcripts, listing all of a Student's or Professor's courses for a Semester, calculating current GPA, usw, may be insightful for checking and improving your design.

4. Grading

Be certain to provide complete detail in your design. It must be possible to see how it will work when turned into code. It must be possible to see the steps the programmer will follow. No Magic!

20 points total. 10 points for each diagram. Up to -10 for missing details in each diagram. -5 for poor choices of encapsulation.

Assignment ThinkBeforeCode
Version 1  (java edition) Mar 2017
©W.L.Honig for Comp 271 Loyola University Chicago, Spring 2017
//end assignment