**CST316 Software Enterprise II: Construction and Transition Spring 2016 Lab: Refactoring**

**Objectives:**

1. Learn how to use your IDE to assist with Refactoring
2. Identify code smells and refactor to solve them

**Instructions:**

1. Pay attention to steps where it asks you to insert a comment in the code to identify your work. If I can’t find it, I can’t grade it!

**Task 1: Eclipse Refactorings**:

1. For this lab you will use the source code provided on Blackboard (almost the same code as you have been using). Please create a new project from the code, do not try to import it as an Eclipse archive.
2. Create a new package banking.interfaces (right-click on the project in Eclipse, select New 🡪 Package)
3. Refactor: move the source files *Asset.java, InterestBearing.java,* and *Account.java* to this new package using the Refactor utility in Eclipse (right-click on source files, select Refactor 🡪 Move…). Ensure the proper import statements were added by Eclipse. If not, use the right-click "Add Import" functionality.
4. Rename the class variable *balance* to *accountBalance* in Account.java using the context menu Refactor 🡪 Rename. Indicate your refactoring using the comment “CST316 ACTIVITY 1.4” in the code (exactly like that, in CAPS, so I can search). Ensure the project still builds correctly (you may need to make other changes along the way).
5. A common *lexical design pattern* (meaning, naming convention) is to prefix interface classes with an “I” and Abstract classes with an “A”, as in “IMyInterface” or “AMyAbstractClass”. Rename the interfaces and abstract classes in the project according to this convention, and make sure the changes propagate through the project and the project still builds and runs. Use the Refactor 🡪 Rename functionality in Eclipse.
6. Ensure your project still builds completely and correctly in Eclipse and the program runs properly (Run As…, you need the my.properties as a command-line argument as before).

CHECKPOINT 1: Zip up all of your *source tree only* and save it as labRefactoring.<asurite>\_task1.zip

**Task 2: Find Code Smells and Refactor**

1. Run your Metrics plugin before doing steps 2 and 3. Save an export of the Metrics report. You may also want to save a screenshot.
2. Find any smelly code *within a class*. Identify the smell by making a comment prefixed by “TASK 2-1 SMELL WITHIN A CLASS *<describe the smell>*”. Then proceed to refactor the code to remove the smell. Summarize your refactoring changes in your comment above the refactoring.
3. Find any smelly code *between classes*. Identify the smell by making comments in ALL RELEVANT CLASSES prefixed by “TASK 2-2 SMELL BETWEEN CLASSES *<name of smell>*”. Then proceed to refactor the code to remove the smell. Summarize your refactoring changes in your code comments.
4. Re-run the Metrics plugin and again capture an export.
5. Compare the results of your 2 metrics exports. Did any of the metrics change for the better after your 2 refactorings. Pick a metric whose value changed, and indicated why it changed and whether it changed for the better (or worse) because of the refactoring. Put your 2 exports (name them “before” and “after”) and a document with your comparison (this question) and any data or screenshots in the root of your source tree for submission.

CHECKPOINT 2: Zip up *source tree only* and save it as labRefactoring.<asurite>\_task2.zip

**Submission:**

Submit your 2 zipfiles to Blackboard by April 11 at 11:59pm. Please remember to zip your *source tree* meaning all the files under your source directory of your Eclipse project. Be sure to copy the Metrics exports and the short document answering Task 2 Step 5 questions in the root of that source tree before zipping up. It is late in the semester, we do not have time to deal with improperly formatted submissions!