Noel LeJeune e-mail: lejeunen@mscd.edu SI 1052

Assignment 01: Object Manipulation

Performance Measures

1. **Create UML Class Diagrams from Java source code**
   a. **Identify the attributes (data fields)**
   b. **Identify the constructors**
   c. **Identify the methods**
   d. **Annotate method return types and method formal parameters**
2. **Use pre-built java classes to:**
   a. **Create multiple objects**
   b. **Change the state of individual objects**
   c. **Examine (report) an objects state**
   d. **Call the methods of objects**
3. **Complete a set of test cases and execute the tests**
   a. **Determine the expected values for test cases**
   b. **Execute the test cases**
   c. **Determine whether the test cases pass or fail**
4. **Modify an existing class**
   a. **Add instance variable**
   b. **Modify constructors for additional instance variables**
   c. **Modify instance methods**
5. **Create and use local (temporary) variables**

Description (Requirements)

For this assignment, you will use the *provided* java classes. Your assignment consist of several activities.

1. Create UML Class Diagrams for each of the provided classes

The UML Class Diagram ***MUST*** be a simple text file. You may want to use jGRASP to create a text file in the same folder as the classes in this assignment.

1. List each instance variable with its access modifier (- for private) and its data type
2. List each instance method with its access modifier (- for private), its data type, and its return type

2. Add code to each test method in the ObjectTester class to fully test each of the provided classes – test the constructor(s) and all methods

The ObjectTester class has a method header with an empty body for testing each class provided.

1. Create an object of the supplied class
2. Invoke (call) each method for the object your created in Step 1 and assign the return value of the method (for methods that have return values) to a temporary variable. Then print to the screen an appropriate message including the value returned.
3. If a method does not return a method, you should display the results of the method called. For example, if the method is a setter (mutator) that changes the value of an instance variable, you should print to the screen an appropriate message displaying the value of the instance variable that was changed.
4. If a method performs some other operation, be sure you output to the screen the results (if not covered in Step 2).

Be sure you provide adequate messages to the screen to indicate what you are testing.

3. Review the output of your program to be sure each operation is correctly performed – That is, ***CHECK YOUR PROGRAM'S OUTPUT!***

4. Create a new class named Block

   Use the Rectangle class as a starting point (copy the class Rectangle and change every instance of the word Rectangle to Block and save as Block.java). The Block class is a 3-dimensional object with length, width, and height. Then,

   1. Add the instance variable height.
   2. Add getter and setter instance methods for height
   3. Modify the instance method calculateArea() changing the name to calculateSurfaceArea() and modify the formula appropriately.
   4. Modify the instance method isSquare() changing the name to isCube() and modify the boolean expression (code) appropriately.
   5. Modify the toString() method to call the methods calculateSurfaceArea() instead of calculateArea and isCube() instead of isSquare().

5. Create UML Class Diagrams for the Block class

6. Add test code as in Step 2 for the newly created Block class

SUBMIT (in a zip file with the correct name, e.g. `SueJones01.zip`)

- A ***text file*** containing a sample of the output from running the `ObjectTester` program (copy and paste). Name the file `Output01.txt`
- A ***text file*** for *each* UML class Diagram
- A ***text file*** with the "Standard Questions" (see below) with your answers. Name the file `StandardQuestions01.txt`
- Each java class including `ObjectTester.java`

Standard Questions

A. How much time did you spend on this assignment?

B. What difficulties did you experience?

C. What were the results of running your test cases/program? Run the program enough times to see that it works correctly – more about testing in subsequent assignments.

Submit - Use the "Assignments Drop Box" in Blackboard - do NOT submit via email

Submit ALL files in one zip file!