CS 441 Software Engineering

Assignment 4

Due on Wednesday, November 20, 11:59pm.

Individual Portion (50 Points)

We already know that we can create a lunar lander application of the pipe-and-filter architecture style from three independent Java classes: GetBurnRate, CalcNewValues, and DisplayValues. As mentioned in class, they can be run as: java GetBurnRate | java CalcNewValues | java DisplayValues.

Re-write Class CalcNewValues mentioned above using C++ (e.g., CalcNewValues.cpp), and run your application as: java GetBurnRate | CalcNewValues | java DisplayValues. Note that the second filter (i.e., CalcNewValues) is a C++ application. Your lunar lander application must behave in the same way as the original application above.

Submit (1) your CalcNewValues.cpp file; (2) a document that includes a screenshot of running the application (from the command line) and briefly explains why this would work and what framework(s) is involved.

Group Portion (50 Points)

Create test cases to test the Tetris application, Tetris-Testing.zip (will be sent to you in a separate announcement). Download the zip file, unzip it, and import the project into your Eclipse workspace.

Create a JUnit testing class for Class *Tetris* and Class *BoardPanel* respectively. For each class, test the following methods.

- Tetris:
  -public void updateGame()

- BoardPanel.
  -public boolean isValidAndEmpty(TileType type, int x, int y, int rotation)
  -public void addPiece(TileType type, int x, int y, int rotation)
  -public int checkLines()

Create a JUnit test suite, and add both test classes into the test suite. Put all your test cases and test suite into a different folder (e.g. test). After you finish your assignment, add your project into a zip file.

In addition, you need to write a report. The report should include the following content.

1. What problems did you find in the code? For each problem, further explain how you found it (e.g. using which test case).

2. Specifically explain the test case that you have created for the *updateGame* method of Class *Tetris*. What is your input, and what is your expected output? What is your logic of testing this method?

3. Include a screenshot of the result of running your test suite.

At the end, submit two files to the Cougar Course system: (1) the zipped project file that includes the source code of your test cases; (2) the pdf version of your report.

Hint: 1) Look at the comment in the code for the functional descriptions of those methods. You will also need to understand the source code that you are testing. Keep in mind: we are doing unit testing, or white box testing. 2) Integration testing is essentially involved in this assignment. 3) Some methods that we are going to test do not have a return value. You will need to figure out your own way to test them.