ECE 325 OBJECT-ORIENTED SOFWARE DES (LEC A1 Fa18)

Assignment 4: Static Code Analysis and Unit Testing

·Source code: *CodingHorror.java*; *UnitTesting.zip*.

·Due date: **Thursday 11th of October (5:00 pm)**. A working copy of your solution must be submitted to eClass before this date.

Import the project from the zip file to read the source codes.

Part 1: Static Code Analysis

Consider the `CodingHorror` program, although it compiles, it fails to do what the programmer wants. The programmer intends to read a string from the user, substitute all "e" characters with "o" characters, then check whether the substitution results in the string "pool". This will never be the case, no matter what input we provide, because of faults in the code.

Your task is to utilize the static analysis tool *FindBugs*, a popular open source static code checker for Java, to find the FIVE bugs reported by the tool and contained within the code.

Hint: Sometimes one bug hides another bug; and the second bug is only revealed by removing the first bug. That is, think iteration.

Note 1: Findbugs is likely to report a sixth error, Internationalization. It is something that goes beyond the scope of previous Java classes so we will ignore this one.

Note 2: If you choose Firebugs version 3.0.0 or earlier (now the newest version is 3.0.1, as of Sept. 30, 2018), you can only get four bugs detected.

Compile the*CodingHorror.java*source code into bytecodes file, and then start up the FindBugs program to analyze it: *New Project* > *Classpath for analysis* > *Add* > *Analyze*. The application will then produce a list of bugs that are present within your code.

Your task is to provide a brief summary of each bug identified:
1.What is the bug;

2.Why it is happening;

3.Provide a solution to fix the bug.

Part 2: Unit Testing

In this part, you will explore Unit Testing (UT) by using the technique in conjunction with the Eclipse IDE to develop a trivial application. This unit test will provide 4 test suites (Question 1 to 4), which will be used to drive the development of the `Calculator` class.

Your task is to complete the 5 following checkpoints and submitting your final answers as a Java project.

1. Implement functionality defined by Test Suite "Question 1"

Navigate to *Test Suites* > *Question1.java*, and run it as JUnit Test. An output will come up and inform you that there are errors! This is to be expected, don't panic! Now you need to fix these errors.

Your task in this step is to fix the errors and implement only the required functionality(*src* > *Calculator.jav* and re-run the Question 1 test suite.

CHECKPOINT 1: Test Suite "Question 1" runs without any problems, but other test suites fail.

2. Implement functionality defined by Test Suite "Question 2"

Navigate to*Test Suites* > *Question2.java*, and run it as JUnit Test. Your task here is to fix the errors and implement only the required functionality and re-run the Question 2 test suite.

CHECKPOINT 2: Test Suite "Question 2" runs without any problems, but higher test suites fail.

3. Implement functionality defined by Test Suite "Question 3"

Navigate to*Test Suites* > *Question3.java*, and run it as JUnit Test. Fix the errors and implement only the required functionality and re-run the Question 3 test suite.

CHECKPOINT 3: Test Suite "Question 3" runs without any problems, but higher test suites fail.

4. Create Test Cases within "*SqureRootTests.java*"

Navigate to *Test Suites* > *Question4.java*, and run it as JUnit Test. Question 4 will run with no

problems because there are no test cases defined in*SquareRootTests.java*. So open *Test Cases* > *SquareRootTests.java*. Fill in the following four test cases with proper asserts:

```
testRandomPositiveSquareRoot()

testRandomNegitiveSquareRoot()

testSquareRootofZero()

testSquareRootofOne()
```

CHECKPOINT 4: Test Suite "Question 4" fails on the 4 defined test cases

5. Implement functionality defined by Test Suite "Question 4"

Navigate to *Test Suites* > *Question4.java* again, and run it as JUnit Test. Now errors are found. Fix them and implement the required functionality and re-run the Question 4 test suite.

CHECKPOINT 5: Test Suite "Question 4" runs without any errors.

CodingHorror.java
UnitTesting.zip

Submission status

Attempt number	This is attempt 1 (1 attempts allowed).
Submission status	Submitted for grading
Grading status	Graded
Due date	Thursday, 11 October 2018, 5:00 PM
Time remaining	Assignment was submitted 1 day 1 hour early
Last modified	Wednesday, 10 October 2018, 3:23 PM
File submissions	Assignment4.zip Export to portfolio
Submission comments	Comments (0)

Edit submission

Make changes to your submission

Feedback

Grade	20.00 / 20.00
Graded on	Tuesday, 16 October 2018, 9:52 PM
Graded by	Mona Nashaat Ali Elmowafy

You are logged in as **Arun Woosaree** (**Log out**)
ECE 325 (LEC A1 Fa18)

Help
Email