# Faculty of Computing

**SE-314: Software Construction**

**Class: BESE 13AB**

# Lab 03: Unit Testing

# CLO-03: Design and develop solutions based on Software Construction principles. CLO-04: Use modern tools such as Eclipse, NetBeans etc. for software construction.

# 

**Date: 23rd Sep 2024**

**Time: 10:00 AM** **- 12:50 PM   
 02:30 PM – 04:50 PM**

# Instructor: Dr. Mehvish Rashid

# Lab Engineer: Mr. Aftab Farooq

### 

**Introduction:**

# Lab 03: Turtle Graphics

Students will have hands-on experience of JUnit and Git.

Material:

https://ocw.mit.edu/ans7870/6/6.005/s16/psets/ps0/

**Lab Tasks**

Solve problem 0 -3 of part 1 listed on the link.

**Problem 0: Install and Setup Problem 1: Clone and Import Problem 2: Unit Testing**

**Look at the source code contained in RulesOf6005.java in package rules . Your warm-up task is to implement**

A white background with black text

Description automatically generated

**Once you’ve implemented this method, run the main method in RulesOf6005.java .**

public static void main(String[] args) is the entry point for Java programs. In this case, the main method calls the mayUseCodeInAssignment method with input parameters. To run main in RulesOf6005 , right click on the file RulesOf6005.java in either your Package Explorer, Project View, or Navigator View, go to the *Run As* option, and click on *Java Application* .

Run the unit tests.

To run the tests in RulesOf6005Test , right click on the RulesOf6005Test.java file in either your Package Explorer, Project View, or Navigator View, and go to the *Run As* option. Click on *JUnit Test* , and you should see the JUnit view appear.

If your implementation of mayUseCodeInAssignment is correct, you should see a green bar, indicating that all the tests (there’s only 1 test, containing 2 assertions) passed.

Try *breaking* your implementation and running RulesOf6005Test again.

You should see a red bar in the JUnit view, and if you click on testMayUseCodeIn- Assignment , you will see a *stack trace* in the bottom box, which provides a brief explanation of what went wrong. Double-clicking on a line in the failure stack trace will bring up the code for that frame in the trace. This is most useful for lines that correspond to your code; this stack trace will also contain lines for Java libraries or JUnit itself.

fix your implementation so it’s correct again. Make sure the tests pass.

Passing the JUnit tests we provide does **not** necessarily mean that your code is perfect. You need to review the function specifications carefully, and **always write your own JUnit tests** to verify your code.

**Tasks Steps:**  
  
 The lab 3 tasks are as follows.

1. Setup your eclipse and install git. Show proof of git install by adding a screenshot of git in cmd.

2. Download the code from the link given in lab manual: <https://ocw.mit.edu/ans7870/6/6.005/s16/psets/ps0/>

3. Clone the project and show screenshot.

4. Open project on eclipse.

5. Implement the method mayUseCodeInAssignment according to the rules given in: <https://ocw.mit.edu/ans7870/6/6.005/s16/general/collaboration.html>

6. Run the method to see output. Screenshot output.

7. Run the unit tests given in the RulesOf6005Test. Screenshot output.

8. Break the implementation and run the test to fail them. Screenshot output.

9. Make a min of two more unit tests in RulesOf6005Test file.

10. Also at the end add commits and push the changes into the repository. Screen shot.

11. Zip code and upload both report and code.

**Tasks Steps 2:**1Practice all git commands (see the document that I have shared with you via email)

2. Take the screen shot and add in report.

## Answer:

|  |
| --- |
| **Task 2** |

**Deliverables**

Compile a single word document by filling in the solution part and submit this Word file on LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to [aftab.farooq@seecs.edu.pk.](mailto:aftab.farooq@seecs.edu.pk.)