# Department of Computing

**SE-314: Software Construction**

**Class: BESE 12AB**

# Lab 03: Unit Testing

# 

**Date: 06th Oct 2023**

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

# Instructor: Ms. Naema Asif

# 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.

## Answer:

|  |
| --- |
| Solution |
| **TASK1****GIT VERSION****TASK2****GIT ADD****TASK3****GIT STATUS****TASK4**Junit Test**TASK5**  |  | | --- | | CODE: /\* Copyright (c) 2007-2016 MIT 6.005 course staff, all rights reserved.\* Redistribution of original or derived work requires permission of course staff.\*/package rules;/\*\*\* RulesOf6005 represents the collaboration policy of 6.005 as described by the\* general information on Stellar.\*/public class RulesOf6005 {/\*\*\* Judge whether a given piece of code may be used in an assignment (problem\* set or team project) or not, according to the 6.005 collaboration policy.\*\* @param writtenByYourself true if the code in question was written by\* yourself or, in the case of a team project, your teammates,\* otherwise false.\* @param availableToOthers if not writtenByYourself, whether or not the\* code in question is available to all other students in the class.\* Otherwise ignored.\* @param writtenAsCourseWork if not writtenByYourself, whether or not the\* code in question was written specifically as part of a solution to\* a 6.005 assignment, in the current or past semesters. Otherwise\* ignored.\* @param citingYourSource if not writtenByYourself, whether or not you\* properly cite your source. Otherwise ignored.\* @param implementationRequired whether the assignment specifically asks\* you to implement the feature in question.\* @return Whether or not, based on the information provided in the\* arguments, you are likely to be allowed to use the code in\* question in your assignment, according to the 6.005 collaboration\* policy for the current semester.\*/public static boolean mayUseCodeInAssignment(boolean writtenByYourself,boolean availableToOthers, boolean writtenAsCourseWork,boolean citingYourSource, boolean implementationRequired) {// Case 1: Code is written by yourself or your teammates for the assignment.if (writtenByYourself || (!writtenByYourself && !availableToOthers && !writtenAsCourseWork && citingYourSource && implementationRequired)) {return true; // You can use the code.}// Case 2: Code is not written by yourself, but it's not available to others.else if (!writtenByYourself && !availableToOthers && !writtenAsCourseWork && !citingYourSource && !implementationRequired) {return false; // You cannot use the code.}// Case 3: Code is not written by yourself, but it's part of the current semester's coursework.else if (!writtenByYourself && !availableToOthers && writtenAsCourseWork && !citingYourSource && !implementationRequired) {return false; // You cannot use the code.}// Case 4: Code is not written by yourself, but proper citation is provided.else if (!writtenByYourself && availableToOthers && !writtenAsCourseWork && citingYourSource && !implementationRequired) {return true; // You can use the code.}// Case 5: Code is not written by yourself, and implementation is required.else if (!writtenByYourself && availableToOthers && !writtenAsCourseWork && citingYourSource && implementationRequired) {return false; // You cannot use the code.}// Case 6: Default case, where none of the above conditions apply.return false; // You cannot use the code.}/\*\*\* Main method of the class.\*\* Runs the mayUseCodeInAssignment method.\*\* @param args unused\*/public static void main(String[] args) {System.out.println("You may certainly use code you wrote yourself: " +RulesOf6005.mayUseCodeInAssignment(true, false, true, true, true));}} |  **TASK6****TASK7****TASK8****TASK9****TASK10** |

**Deliverables**

Compile a single word document by filling in the solution part and submit this Word file on LMS. This lab grading policy is as follows: The lab is graded between 0 to 10 marks. Insert the solution/answer in this document. You must show the implementation of the tasks in the designing tool, along with your completed Word document to get your work graded. You must also submit this Word document on the 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.)