The University of the West Indies, St. Augustine
COMP 3607 Object Oriented Programming II
2020/2021 Semester 1
Lab Tutorial - Week 10

This tutorial focuses on JUnit testing. A software test is a piece of software, which executes another piece of software. It validates if that code results in the expected state (state testing) or executes the expected sequence of events (behaviour testing).

A formal written unit test case is characterized by a known input and an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post-condition.

Learning Objectives:

- Set up a Maven project
- Design software tests for programs/classes
- Create JUnit 5 test classes using VS Code
- Run JUnit 5 test classes

Resources/ Useful Links:
- https://www.tutorialspoint.com/junit/junit_overview.htm
- https://junit.org/junit5/docs/current/user-guide/#overview-getting-started
- https://code.visualstudio.com/docs/java/java-testing
- https://maven.apache.org/guides/mini/guide-naming-conventions.html

Instructions
Download the Java classes for Week 10 (retrieve from myElearning). We will set up a Maven project and then design and build tests for the two domain classes: Lecturer and Department.

1. Create a Maven project for your work using the VS Code editor. Press: Ctrl+ Shift + P (Windows, Linux) or Command + Shift + P (Mac). Follow the naming conventions for groupId, artifactId, and version from here: https://maven.apache.org/guides/mini/guide-naming-conventions.html

2. Design the following software tests for the **Lecturer** class:

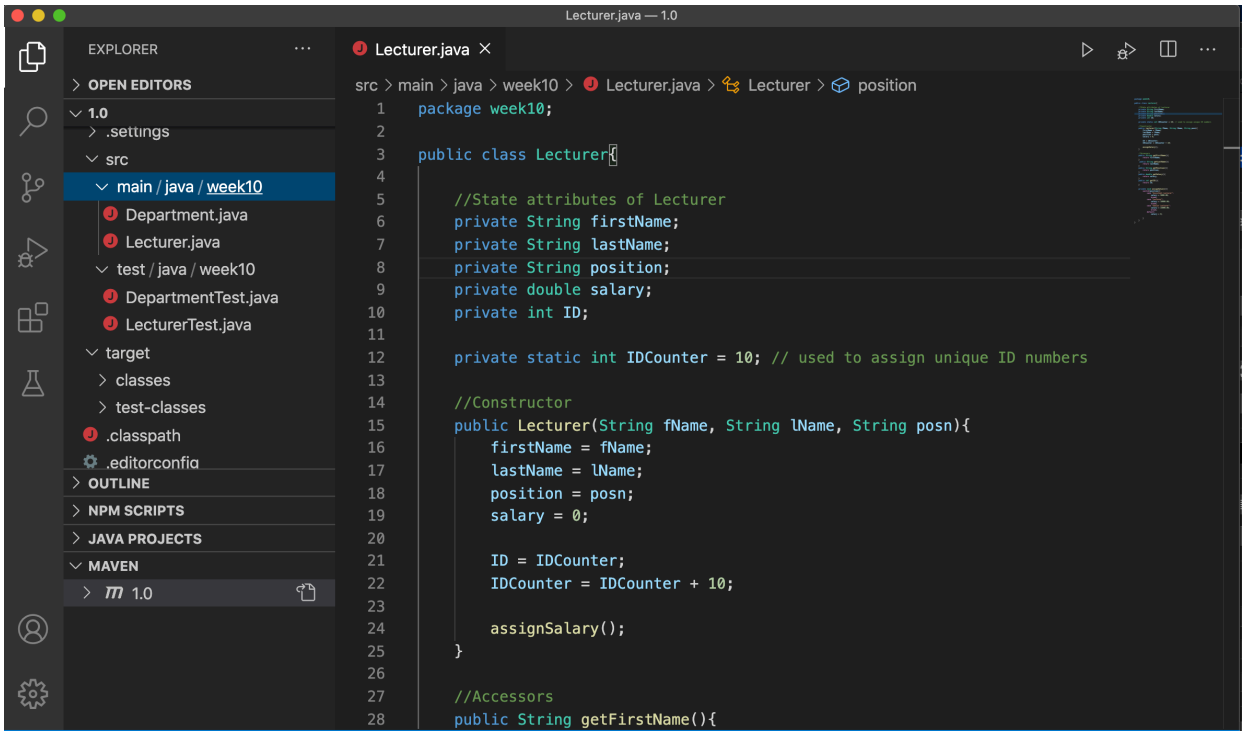
Position	Salary
Senior Lecturer	15000
Lecturer	10000
Assistant Lecturer	5000

   (i) Check that Lecturer objects are created properly (state set/ retrieved properly)

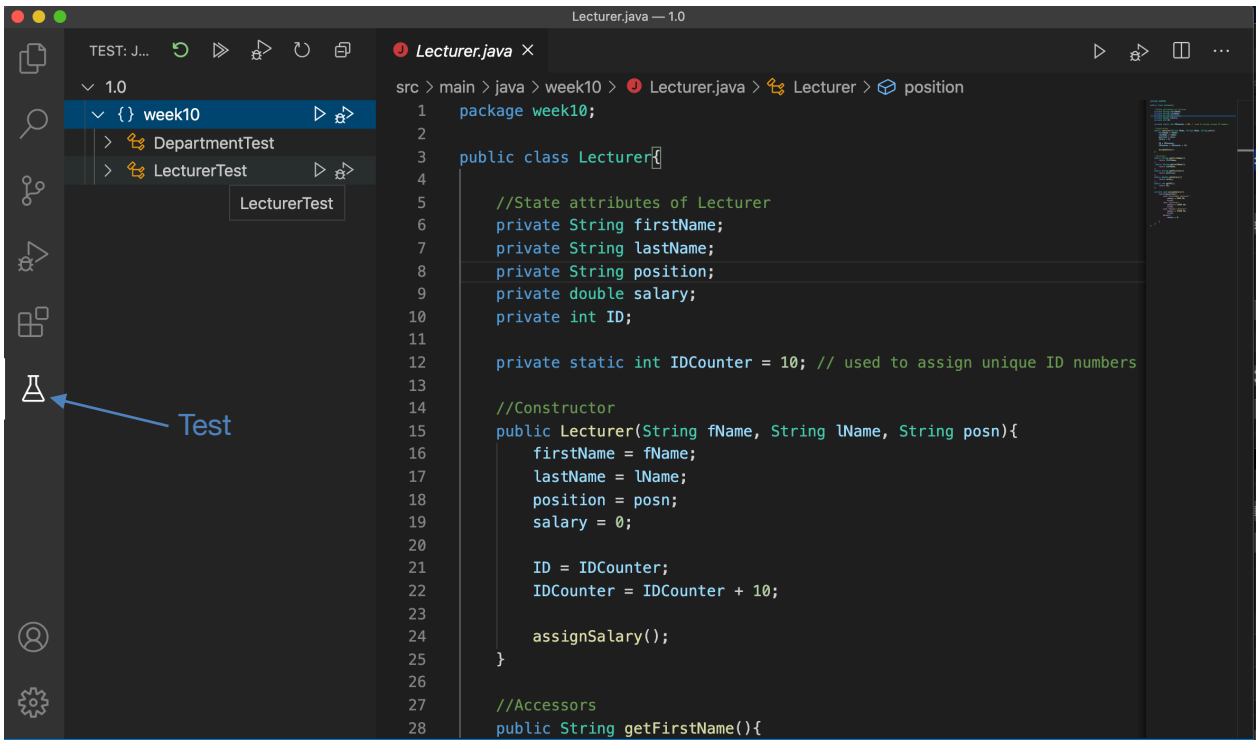
   (ii) IDs are correctly generated

   (iii) Salary is correctly assigned based on position

3. Create a JUnit test class, **LecturerTest**, for the Lecturer class: There must be at least two unit test cases for each requirement – one positive test and one negative test. If a requirement has sub-requirements, each sub-requirement must have at least two test cases as positive and negative. Organise your test classes within the test folder of your Maven project.

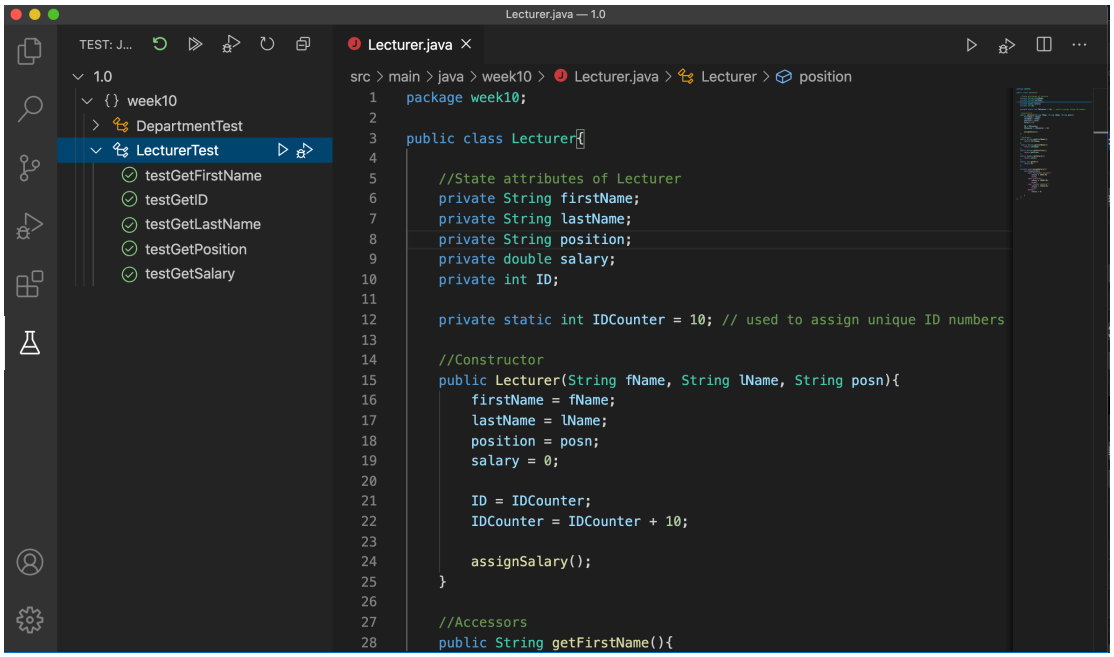


Ensure that you have set up your editor with all required extensions (Language support, Debugger). See third link in the resource list above.



4. For the **LecturerTest** class:
   (i) Create a global Lecturer object instance for use in each test
   (ii) Create the test methods with suitable names/ create appropriate ones.
   (iii) Fill in the assertions for each of the test methods to reflect your software tests in Q2.

   From the side panel, click on the Test Icon options, select Run -> Test Project. When you have successfully completed the tests for the Lecturer class, you should see all of the tests passing (green ticks).



5. Repeat steps 2-4 for the **Department** class where your tests should evaluate if:
   (i) a Department object was created and initialised properly.
   (ii) The `addLecturer( )` method is correctly creating a Lecturer object and storing it properly
   (iii) The `getLecturer( )` method is correctly returning a Lecturer object corresponding to the ID supplied as a parameter.

   All tests for both classes should pass when you're finished.