Lab TDD 2: Fixture Methods

Objectives

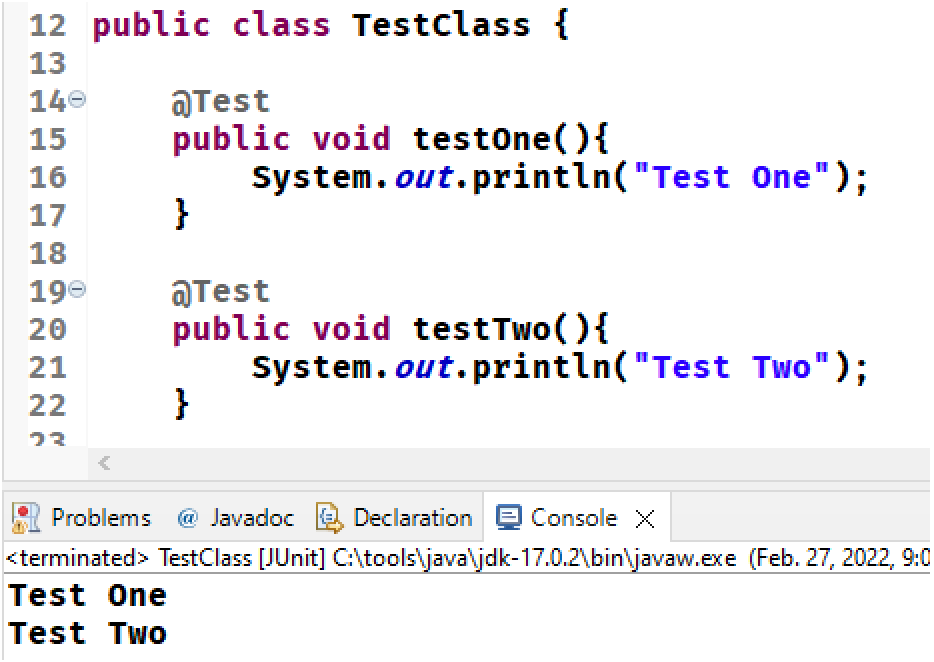
In this lab you will use the fixture methods

The lab continues from where TDD 1 lab leftoff

Instructions Project Setup

Step 1: Add a second test method

1. Create a copy of the testOne() method and rename it to testTwo() and have it print out an appropriate message

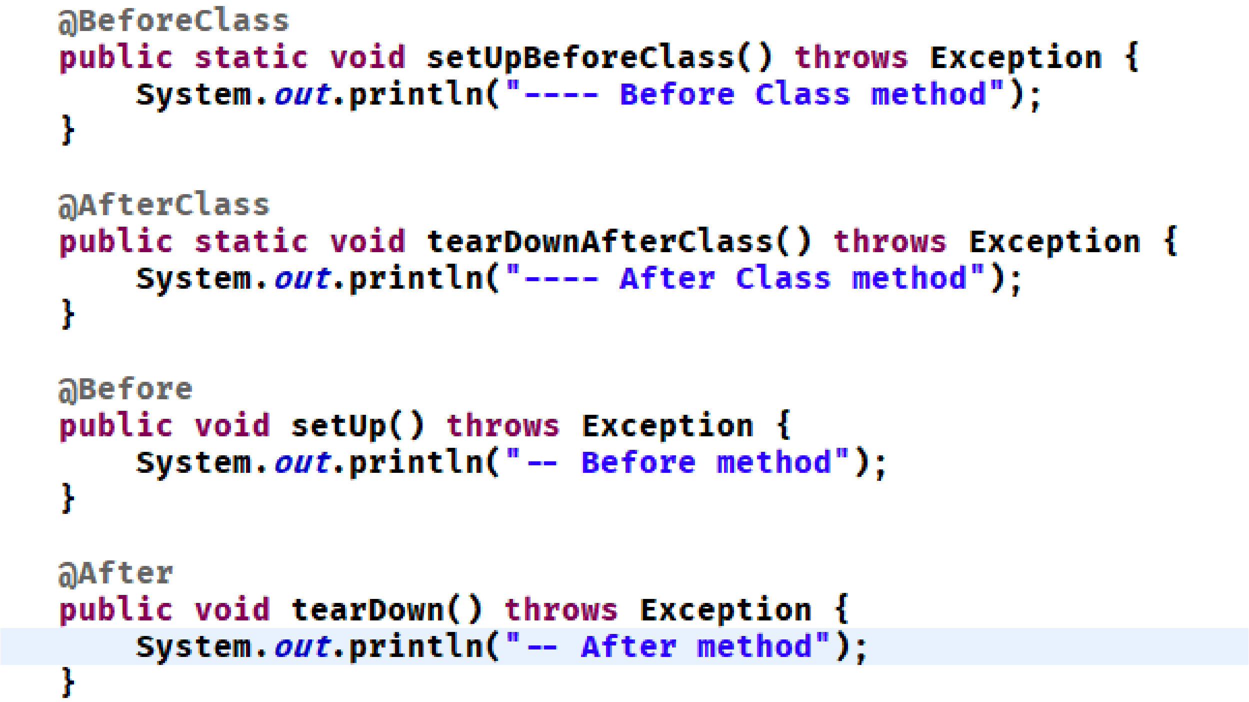


```
public class TestClass {

    @Test
    public void testOne(){
        System.out.println("Test One");
    }

    @Test
    public void testTwo(){
        System.out.println("Test Two");
    }
```

```
<terminated> TestClass [JUnit] C:\tools\java\jdk-17.0.2\bin\javaw.exe  (Feb. 27, 2022, 9:0
Test One
Test Two
```

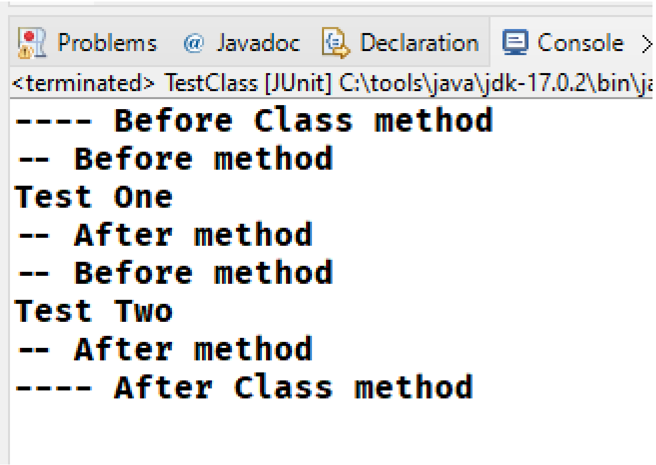
Step 2: Impement fixtures

1. For each of the fixture stubs, implement an output method as shown on the next page



```
@BeforeClass
public static void setUpBeforeClass() throws Exception {
    System.out.println("---- Before Class method");
}

@AfterClass
public static void tearDownAfterClass() throws Exception {
    System.out.println("---- After Class method");
}

@Before
public void setUp() throws Exception {
    System.out.println("-- Before method");
}

@After
public void tearDown() throws Exception {
    System.out.println("-- After method");
}
```

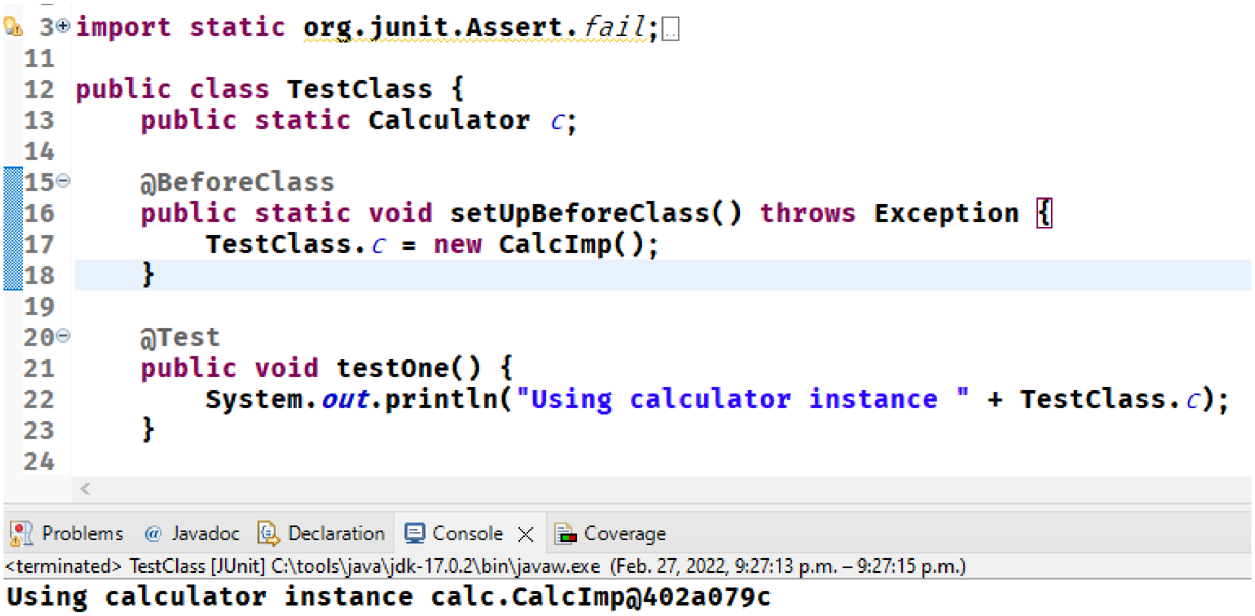
2. Run your code as a test case and explain the output you see
3. It should look like the following



```
<terminated> TestClass [JUnit] C:\tools\java\jdk-17.0.2\bin\ja
---- Before Class method
-- Before method
Test One
-- After method
-- Before method
Test Two
-- After method
---- After Class method
```

Step 3: Set up a test CalcImp

1. Since the calculator is stateless, we can run all of the tests in a single instance of the implementation
2. Delete all the fixture methods except for the @BeforeClass method
3. Create a static class variable of type Calculator
4. Create an instance of CalcImp and assign it to the static variable in the @BeforeClass method.
5. Have the test method print out the address of the calculator



```
import static org.junit.Assert.fail;

public class TestClass {
    public static Calculator c;

    @BeforeClass
    public static void setUpBeforeClass() throws Exception {
        TestClass.c = new CalcImp();
    }

    @Test
    public void testOne() {
        System.out.println("Using calculator instance " + TestClass.c);
    }

```

```
<terminated> TestClass [JUnit] C:\tools\java\jdk-17.0.2\bin\javaw.exe (Feb. 27, 2022, 9:27:13 p.m. – 9:27:15 p.m.)
Using calculator instance calc.CalcImp@402a079c
```

Save your lab because the next lab picks up from here.