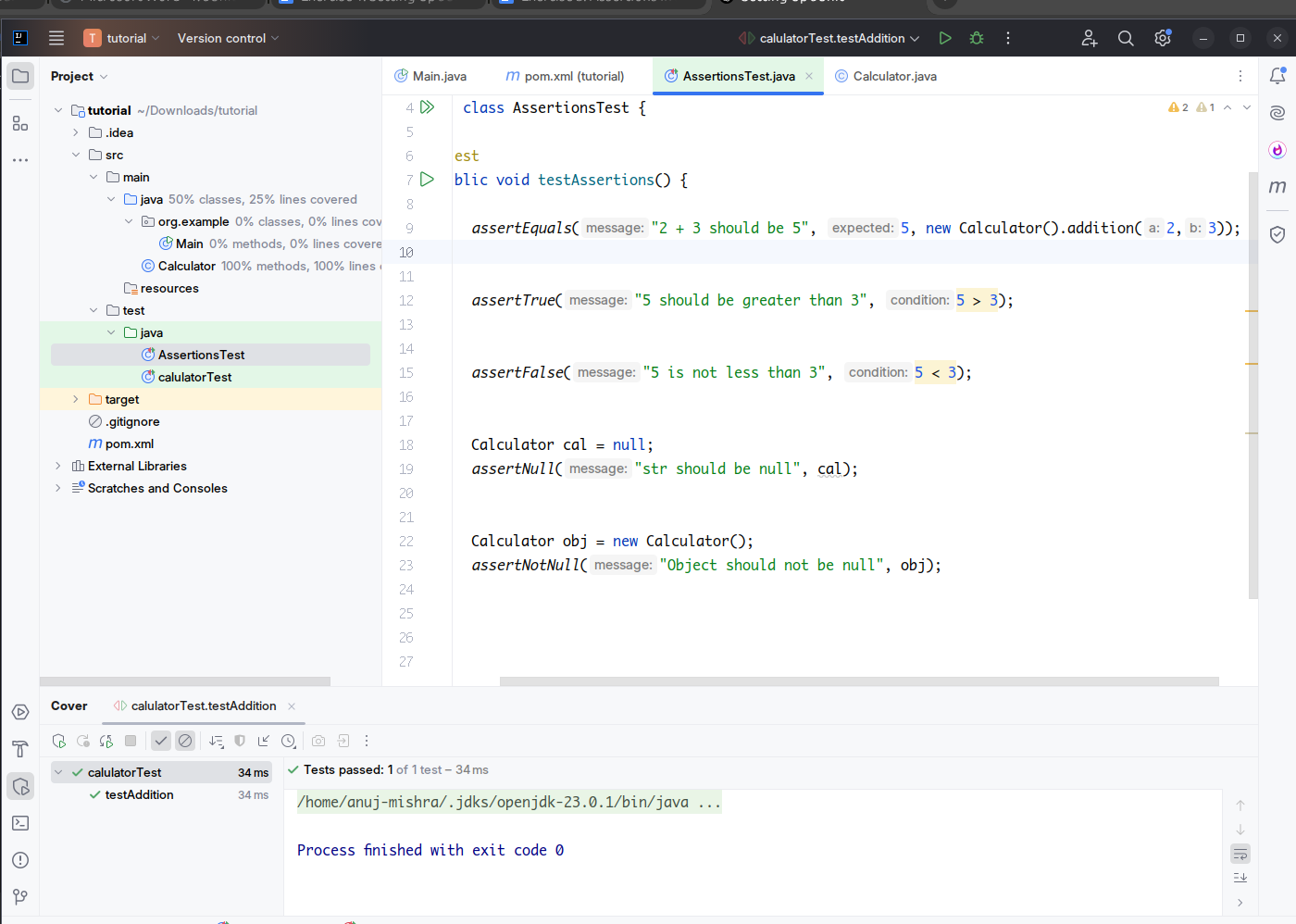
| *Exercise 3: Assertions in JUnit* |
| --- |

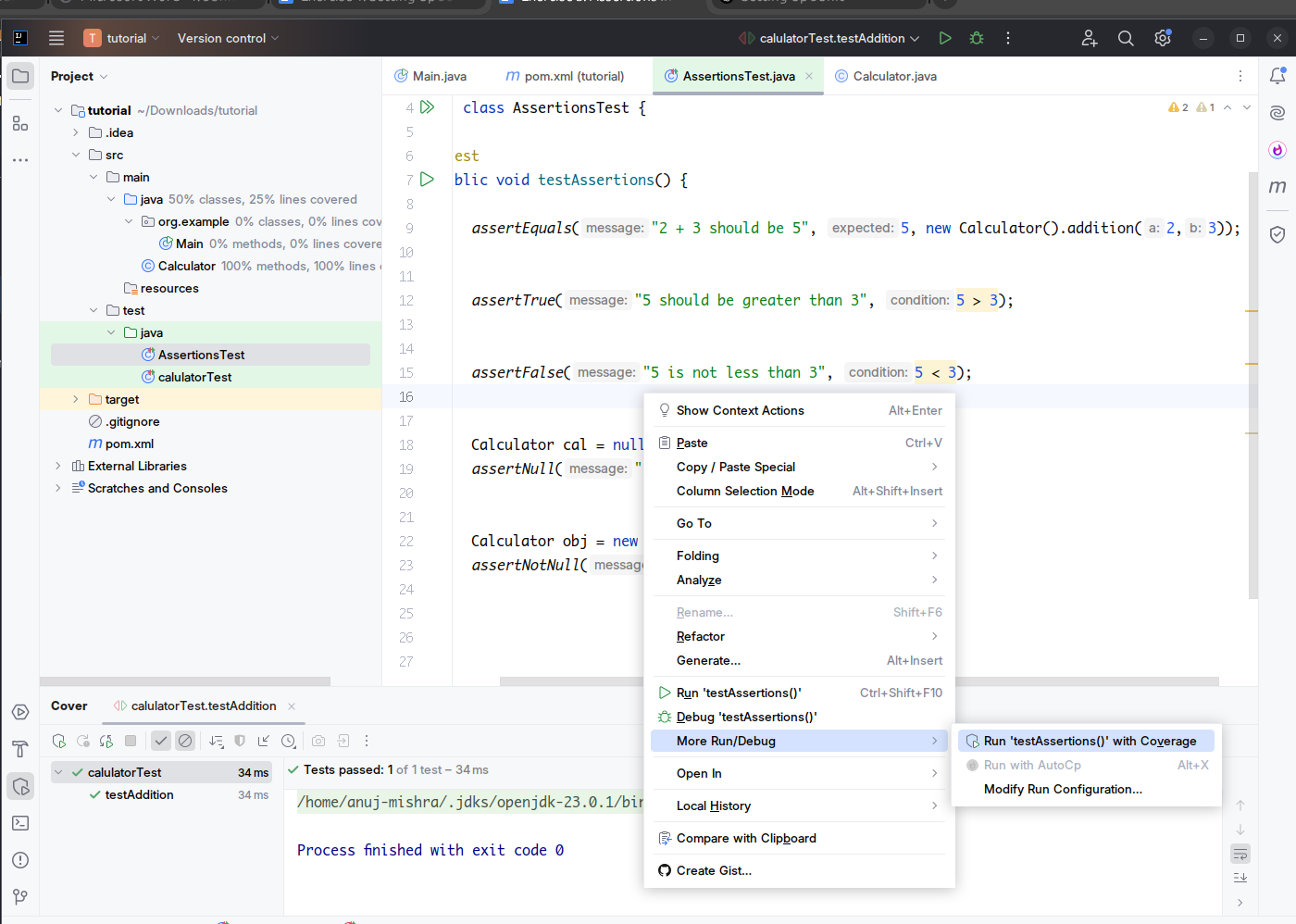
### **1. Create a test class**

Creating a new Java class in your test directory. Name it something meaningful like AssertionsTest. This class will contain all your test methods where you will use different assertions provided by JUnit to verify logic and expected outcomes.



### **2. Write a method with the @Test annotation**

Inside the test class, define a public method and annotate it with @Test from org.junit.Test. This annotation informs JUnit that the method is a test case and should be executed when tests are run. You can name the method testAssertions or something descriptive that indicates its purpose.



### **3. Use assertEquals()**

This assertion checks whether the expected value and actual result are equal. If they are not, the test fails and shows a comparison error.  
 Example: assertEquals(5, 2 + 3); verifies that the sum of 2 and 3 equals 5.

### **4. Use assertTrue()**

This assertion is used to confirm that a given boolean condition is true. If the condition evaluates to false, the test fails.  
 Example: assertTrue(5 > 3); ensures the condition is logically correct.

### **5. Use assertFalse()**

Opposite of assertTrue, this assertion checks that a condition is false. If it turns out true, the test fails.  
 Example: assertFalse(5 < 3); ensures that the statement is logically incorrect as intended.

### **6. Use assertNull()**

This assertion checks that an object or variable is null. If it is not null, the test fails.  
 Example: assertNull(str); when String str = null; confirms that the object was not initialized.

### **7. Use assertNotNull()**

This assertion checks that the object is not null. If the object is null, the test fails.  
 Example: assertNotNull(new Object()); ensures that the object was created and is not empty.

