**JUnit, Mockito and SL4J Hands-on**

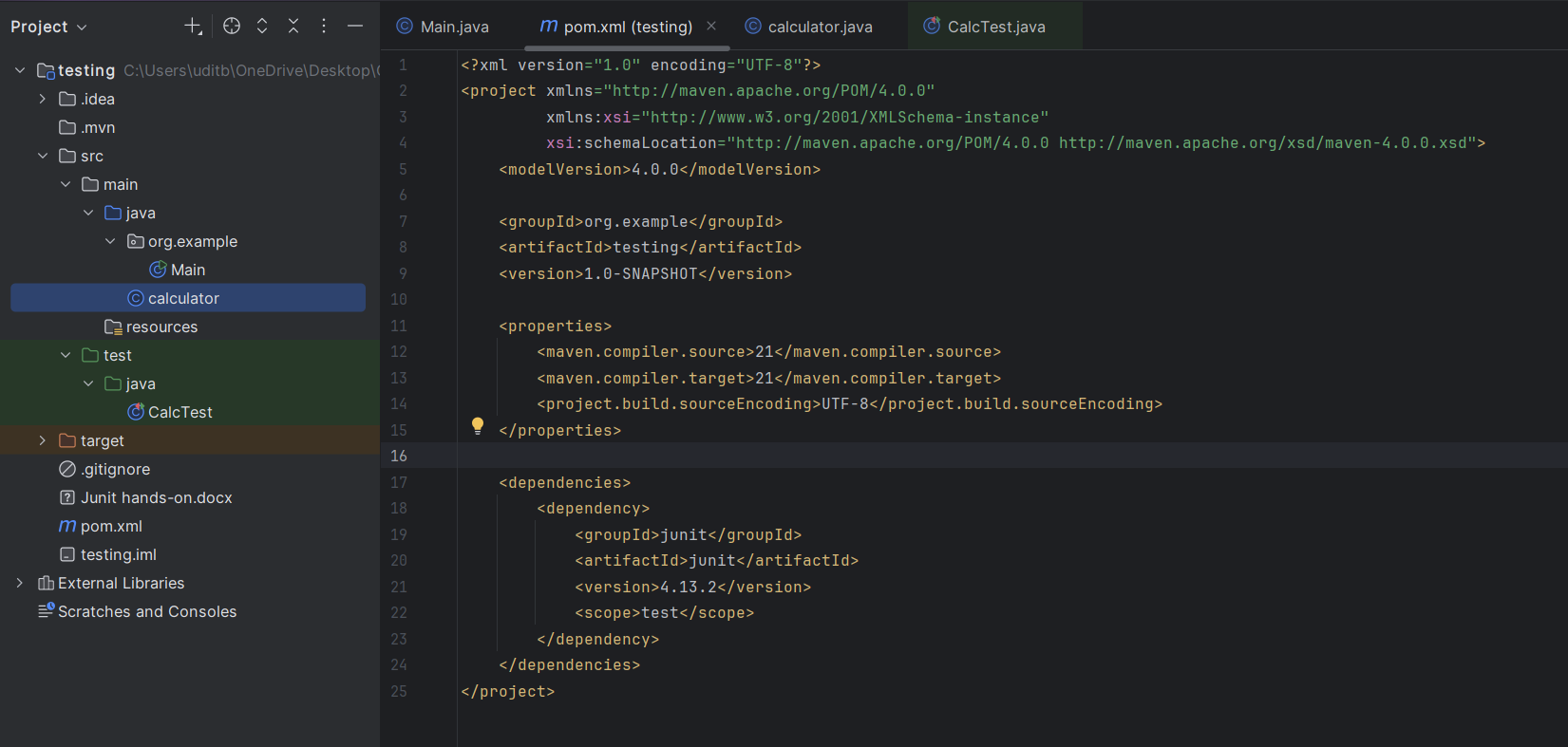
**Name : Udit Bhargava**

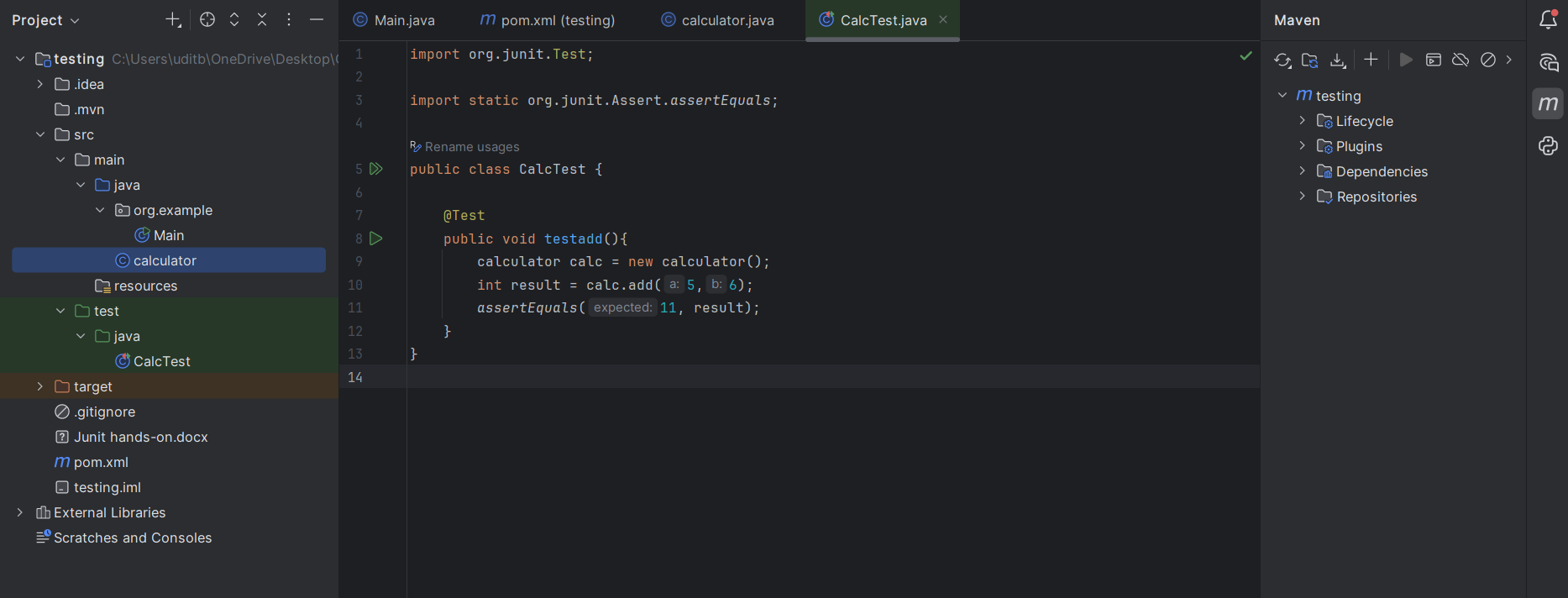
**Superset Id : 6386801**

**1. Junit Hands-on Exercises :**

**Exercise 1: Setting Up Junit**

**Output :**

****

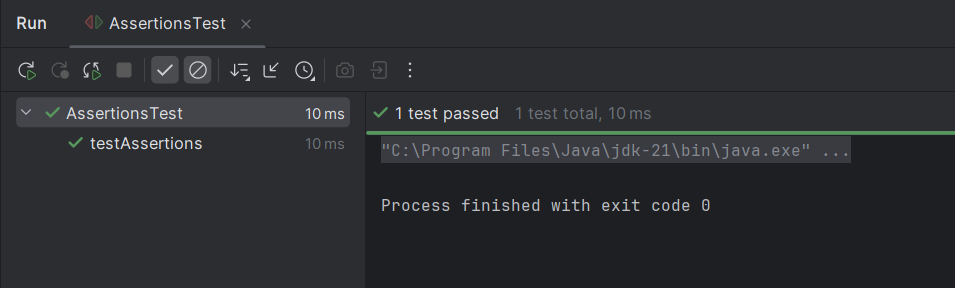


**Exercise 3: Assertions in Junit**

**CODE :**

import org.junit.Test;  
  
import static org.junit.Assert.\*;  
  
public class AssertionsTest {  
 @Test  
 public void testAssertions() {  
 *assertEquals*(5, 2 + 3);  
  
 *assertTrue*(5 > 3);  
  
 *assertFalse*(5 < 3);  
  
 *assertNull*(null);  
  
 *assertNotNull*(new Object());  
  
 String str = "hello";  
 String str2 = str;  
 *assertSame*("Both references should be the same", str, str2);  
  
 String str3 = new String("hello");  
 *assertNotSame*("Different objects should not be same", str, str3);  
  
 }  
}

**Output :**

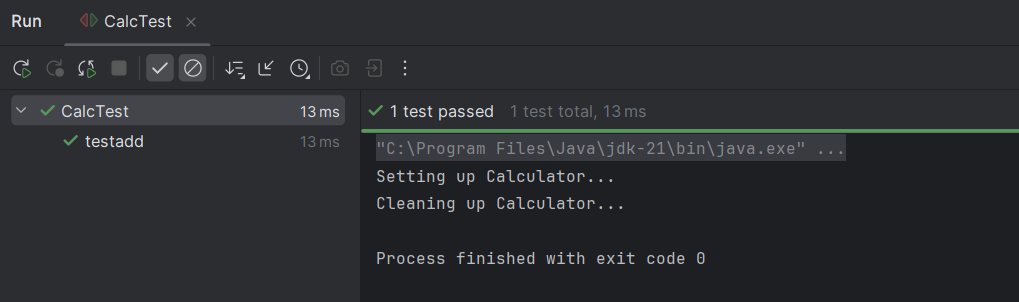


**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

**CODE :**

import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
  
import static org.junit.Assert.*assertEquals*;  
  
public class CalcTest {  
   
 private calculator calc;  
   
 @Before  
 public void setUp() {  
 System.*out*.println("Setting up Calculator...");  
 calc = new calculator(); // Arrange  
 }  
   
 @After  
 public void tearDown() {  
 System.*out*.println("Cleaning up Calculator...");  
 calc = null;  
 }  
   
 @Test  
 public void testadd(){  
 calculator calc = new calculator();  
 int result = calc.add(5,6);  
 *assertEquals*(11, result);  
 }  
}

**Output :**

****

**2. Mockito Hands-on Exercises :**

**Exercise 1: Mocking and Stubbing**

**CODE :**

**ExternalApi.java :**

public interface ExternalApi {

String getData();

}

**MyService.java :**

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java :**

import org.junit.Test;

import org.mockito.Mockito;

import static org.junit.Assert.assertEquals;

import static org.mockito.Mockito.when;

public class MyServiceTest {

@Test

public void testExternalApi(){

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

when(mockApi.getData()).thenReturn("Mock Data");

MyService service = new MyService(mockApi);

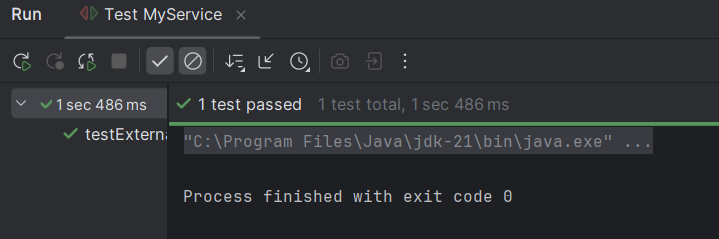
String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

**Output :**

****

**Exercise 2: Verifying Interactions**

**CODE :**

**ExternalApi.java :**

public interface ExternalApi {

String getData();

}

**MyService.java :**

public class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java :**

import org.junit.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

ExternalApi mockApi = Mockito.mock(ExternalApi.class);

MyService service = new MyService(mockApi);

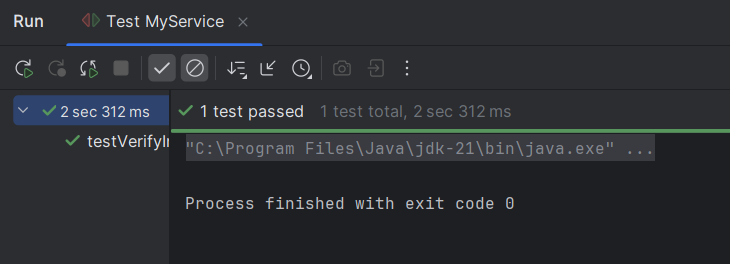
service.fetchData();

verify(mockApi).getData();

}

}

**Output :**

****

**3. SL4J Exercises :**

**Exercise 1: Logging Error Messages and Warning Levels :**

**CODE :**

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

public class LoggingExample {

private static final Logger log = LoggerFactory.getLogger(LoggingExample.class);

public static void main(String[] args){

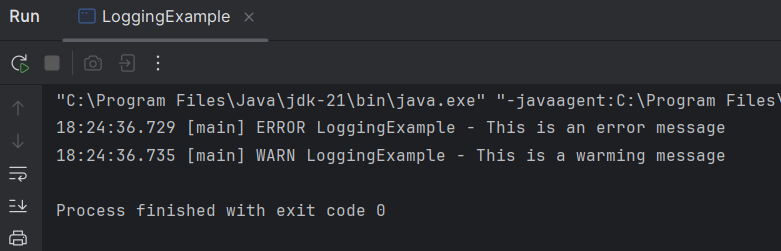
log.error("This is an error message");

log.warn("This is a warming message");

}

}

**Output :**

****