**Week 2 JUnit Testing Hands on**

**Name-Adi Jain**

**Superset ID- 6390400**

**Exercise 1: Setting Up JUnit**

**Scenario**: You need to set up JUnit in your Java project to start writing unit tests.

**Steps**:

1. Create a new Java project in your IDE (e.g., IntelliJ IDEA, Eclipse).

2. Add JUnit dependency to your project.

3. Create a new test class in your project.

**Code**

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>org.example</groupId>  
 <artifactId>testing</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>21</maven.compiler.source>  
 <maven.compiler.target>21</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
  
 <dependencies>  
 <!-- JUnit 5 API -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter-api</artifactId>  
 <version>5.10.2</version>  
 <scope>test</scope>  
 </dependency>  
  
 <!-- JUnit 5 Engine -->  
 <dependency>  
 <groupId>org.junit.jupiter</groupId>  
 <artifactId>junit-jupiter-engine</artifactId>  
 <version>5.10.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
  
  
</project>

**Pom.xml**

package org.example;  
  
public class Main {  
 public static void main(String[] args) {  
 System.*out*.printf("Hello and welcome!");  
 for (int i = 1; i <= 5; i++) {  
 System.*out*.println("i = " + i);  
 }  
 }  
}

**Main.java**

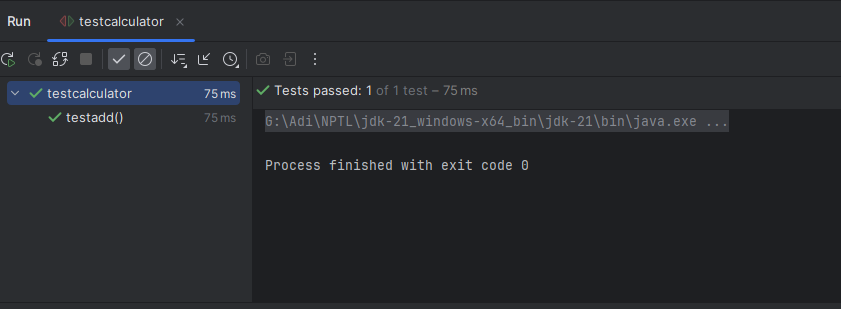
public class calculator {  
 public int add(int a, int b){  
 return a+b;  
 }  
}

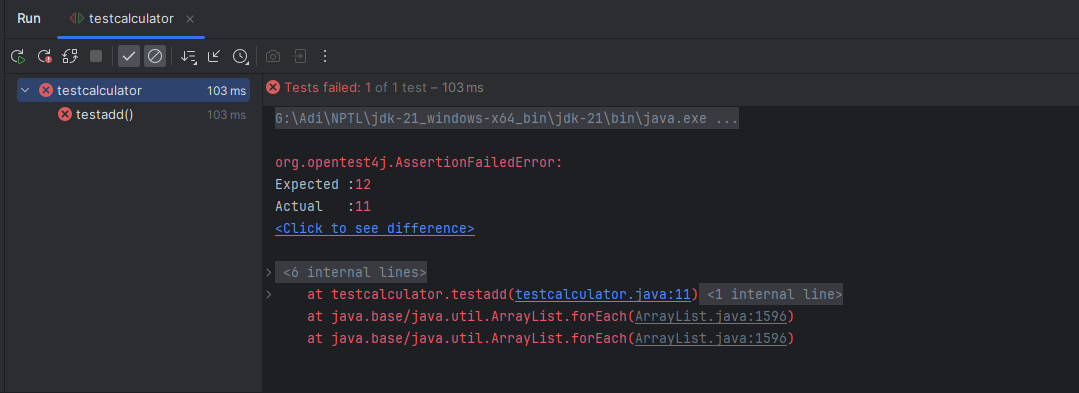
**calculator.java**

import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.*assertEquals*;  
  
public class testcalculator {  
 calculator calc= new calculator();  
 @Test  
 public void testadd(){  
 int result= calc.add(5,6);  
  
 *assertEquals*(11,result);  
  
 }  
}

**testcalculator.java**

**Output**





**Exercise 3**: **Assertions in JUnit**

**Scenario**: You need to use different assertions in JUnit to validate your test results.

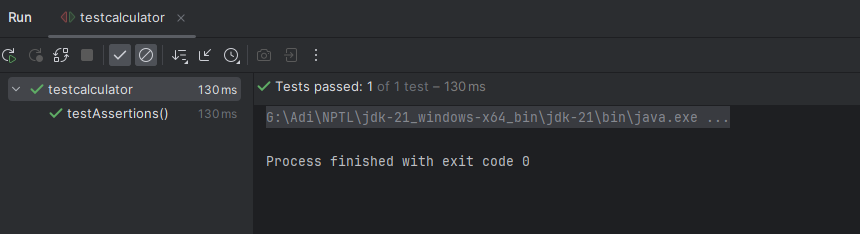
Write tests using various JUnit assertions.

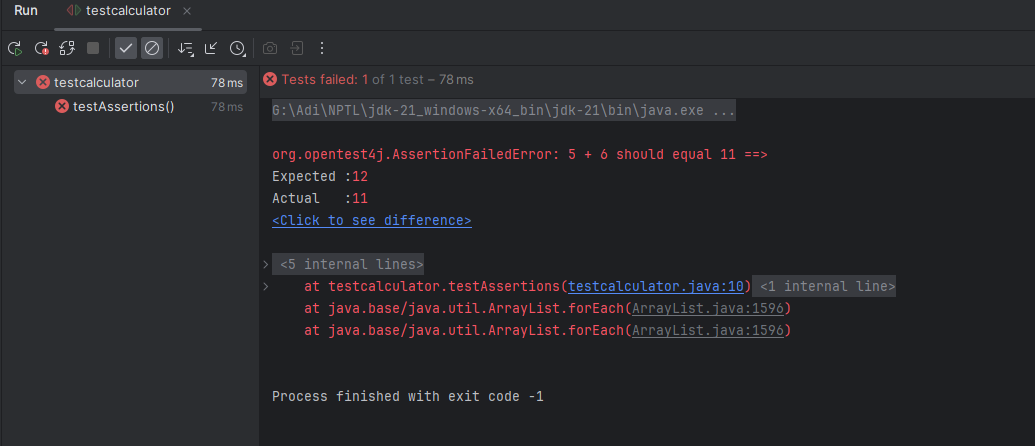
**Code**

import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
public class testcalculator {  
 calculator calc = new calculator();  
  
 @Test  
 public void testAssertions() {  
 *assertEquals*(11, calc.add(5, 6), "5 + 6 should equal 11");  
  
 *assertNotEquals*(10, calc.add(5, 6), "5 + 6 should not equal 10");  
  
 *assertTrue*(calc.add(2, 2) == 4, "2 + 2 should be 4");  
  
 *assertFalse*(calc.add(2, 2) == 5, "2 + 2 should not be 5");  
  
  
 String str = null;  
 *assertNull*(str, "str should be null");  
  
 Object obj = new Object();  
 *assertNotNull*(obj, "obj should not be null");  
  
  
 String a = "hello";  
 String b = "hello";  
 *assertSame*(a, b, "Strings with same value should be same (due to string pool)");  
  
  
 String x = new String("hello");  
 String y = new String("hello");  
 *assertNotSame*(x, y, "Different object instances should not be same");  
  
 int[] expected = {1, 2, 3};  
 int[] actual = {1, 2, 3};  
 *assertArrayEquals*(expected, actual, "Arrays should be equal");  
  
  
 *assertThrows*(ArithmeticException.class, () -> {  
 int z = 5 / 0;  
 }, "Division by zero should throw ArithmeticException");  
  
  
 *assertDoesNotThrow*(() -> {  
 int z = 5 / 1;  
 }, "Division by 1 should not throw");  
  
  
 *assertTimeout*(java.time.Duration.*ofMillis*(100), () -> {  
 Thread.*sleep*(50);  
 }, "Operation should complete within 100ms");  
 }  
}

**testcalculator.java**

**Output**





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

**Scenario**: You need to organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup

and teardown methods.

**Code**

import org.junit.jupiter.api.\*;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
public class testcalculator {  
  
 private calculator calc;  
  
 @BeforeEach  
 public void setUp() {  
 calc = new calculator();  
 System.*out*.println("Setup: Calculator initialized.");  
 }  
  
 @AfterEach  
 public void tearDown() {  
 System.*out*.println("Teardown: Test completed.\n");  
 }  
  
 @Test  
 public void testAddition() {  
 int a = 5;  
 int b = 6;  
   
 int result = calc.add(a, b);  
   
 *assertEquals*(11, result, "5 + 6 should be 11");  
 }  
  
 @Test  
 public void testAdditionWithNegativeNumbers() {  
 int a = -4;  
 int b = -6;  
   
 int result = calc.add(a, b);  
   
 *assertEquals*(-10, result, "-4 + (-6) should be -10");  
 }  
  
 @Test  
 public void testAdditionWithZero() {  
 int a = 0;  
 int b = 5;  
   
 int result = calc.add(a, b);  
   
 *assertEquals*(5, result, "0 + 5 should be 5");  
 }  
  
 @Test  
 public void testAdditionSymmetry() {  
 int a = 3;  
 int b = 7;  
   
 *assertEquals*(calc.add(a, b), calc.add(b, a), "Addition should be commutative");  
 }  
  
 @Test  
 public void testNotNullCalculator() {  
 *assertNotNull*(calc, "Calculator instance should not be null");  
 }  
}

**testcalculator.java**

**Output**

