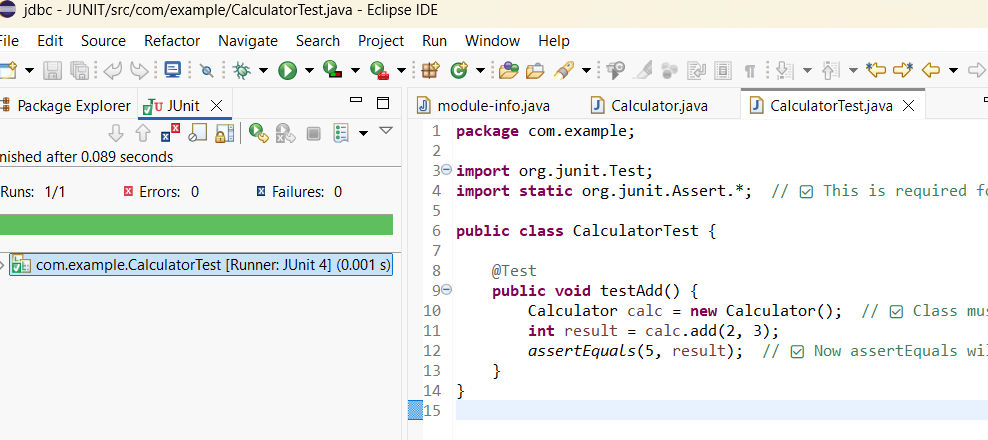
**1.JUnit\_Basic Testing Exercises**

**Exercise 1: Setting Up Junit**

**OUTPUT**



**Exercise 2: Writing Basic JUnit Tests**

**CODE**

***File name: Calculator.java***

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

public int multiply(int a, int b) {

return a \* b;

}

public int divide(int a, int b) {

if (b == 0) throw new IllegalArgumentException("Division by zero");

return a / b;

}

}

***File name:CalculatorTest.java***

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

Calculator calc = new Calculator();

@Test

public void testAdd() {

assertEquals(7, calc.add(3, 4));

}

@Test

public void testSubtract() {

assertEquals(2, calc.subtract(5, 3));

}

@Test

public void testMultiply() {

assertEquals(20, calc.multiply(4, 5));

}

@Test

public void testDivide() {

assertEquals(5, calc.divide(10, 2));

}

@Test(expected = IllegalArgumentException.class)

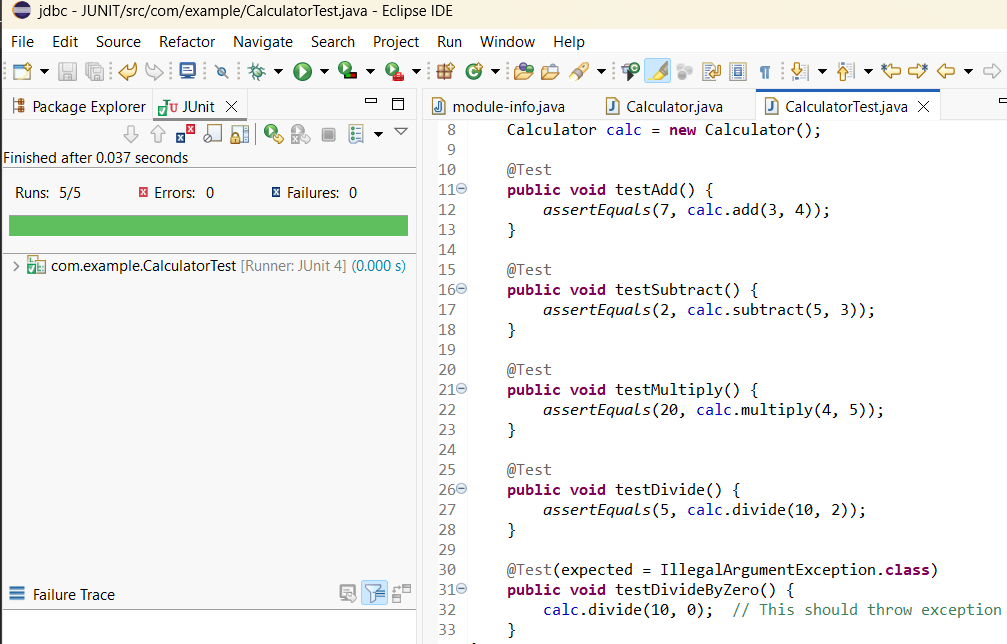
public void testDivideByZero() {

calc.divide(10, 0); // This should throw exception

}

}

**OUTPUT**



**Exercise 3: Assertions in Junit**

**CODE**

***File name: AssertionsTest.java***

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

@Test

public void testAssertions() {

// assertEquals - check for equality

assertEquals(5, 2 + 3);

// assertTrue - check if condition is true

assertTrue(10 > 5);

// assertFalse - check if condition is false

assertFalse(3 > 10);

// assertNull - check if value is null

String nullStr = null;

assertNull(nullStr);

// assertNotNull - check if value is not null

String name = "JUnit";

assertNotNull(name);

// assertSame - check if two references point to the same object

String a = "Hello";

String b = a;

assertSame(a, b);

// assertNotSame - check if two references do not point to the same object

String x = new String("World");

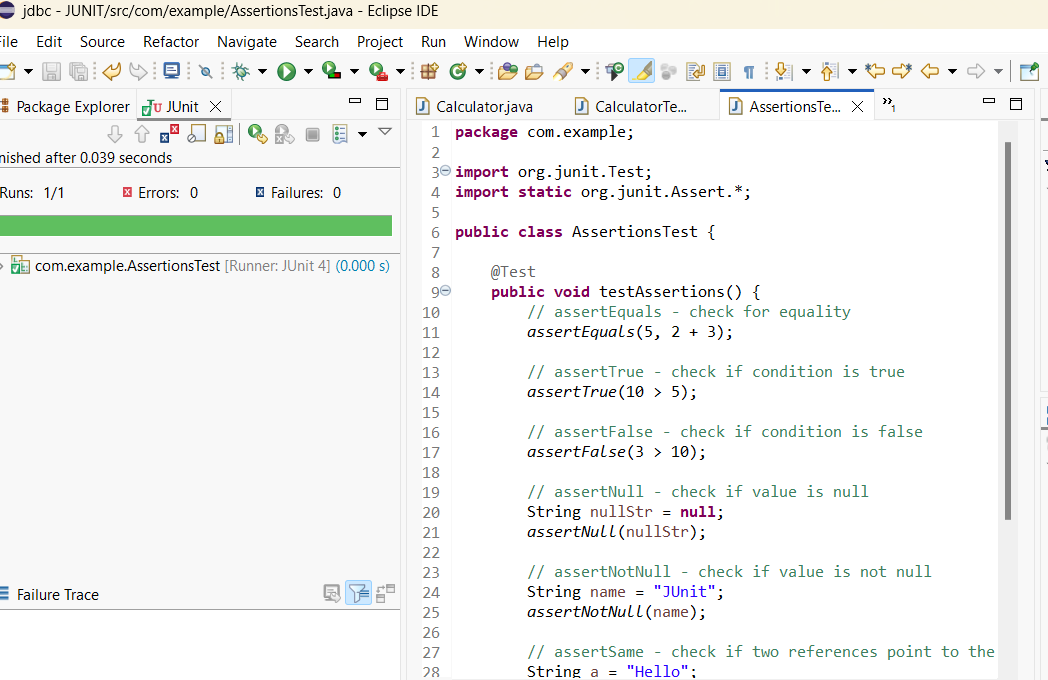
String y = new String("World");

assertNotSame(x, y);

}

}

**OUTPUT**



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

**CODE**

***File name: Calculator.java***

package com.example;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

**File name: CalculatorTest.java**

package com.example;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calc;

// setup method - runs BEFORE each test

@Before

public void setUp() {

System.out.println("Setting up Calculator...");

calc = new Calculator();

}

// Teardown method - runs AFTER each test

@After

public void tearDown() {

System.out.println("Cleaning up...");

calc = null; //

}

@Test

public void testAddition() {

// Act

int result = calc.add(5, 3);

// Assert

assertEquals(8, result);

}

@Test

public void testSubtraction() {

int result = calc.subtract(10, 4);

assertEquals(6, result);

}

}

**OUTPUT**

