**JUnit Testing Exercises**

**Exercise 1: Setting Up Junit**

Program: Calculator.java

public class Calculator {

public int add(int a, int b) {

return a + b;

}

}

Program: JunitTestClass

CalculatorTest.java

import static org.junit.Assert.\*;

import org.junit.Test;

public class CalculatorTest {

Calculator calc = new Calculator();

@Test

public void testAdd() {

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

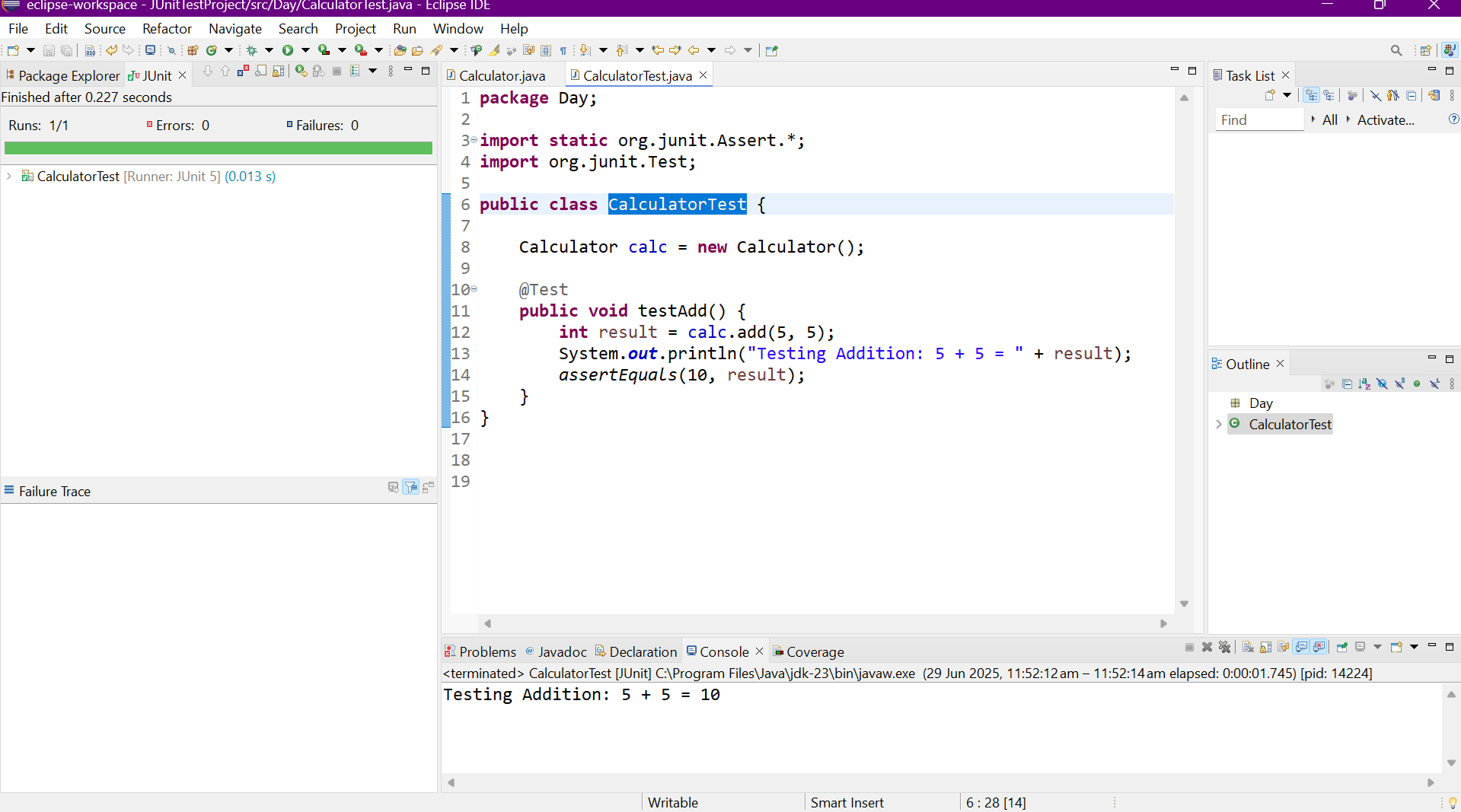
System.out.println("Testing Addition: 5 + 5 = " + result);

assertEquals(10, result);

}

}

OUTPUT:



**Exercise 3: Assertions in Junit**

Program:

import org.junit.Test;

import static org.junit.Assert.\*;

public class MyLogicTest {

MyLogic logic = new MyLogic();

@Test

public void testAllAssertions() {

// 1. assertEquals

assertEquals(10, logic.add(7, 3));

// 2. assertTrue

assertTrue(logic.isPositive(5));

// 3. assertFalse

assertFalse(logic.isPositive(-1));

// 4. assertNull

assertNull(logic.returnNull());

// 5. assertNotNull

assertNotNull(logic.returnObject());

// 6. assertEquals with strings

assertEquals("JUnit", logic.echo("JUnit"));

// 7. assertTrue with string logic

assertTrue(logic.echo("hello").startsWith("h"));

// 8. assertFalse with empty string check

assertFalse(logic.isStringEmpty("Not Empty"));

// 9. assertSame

Integer x = 100;

assertSame(x, logic.sameReference(x));

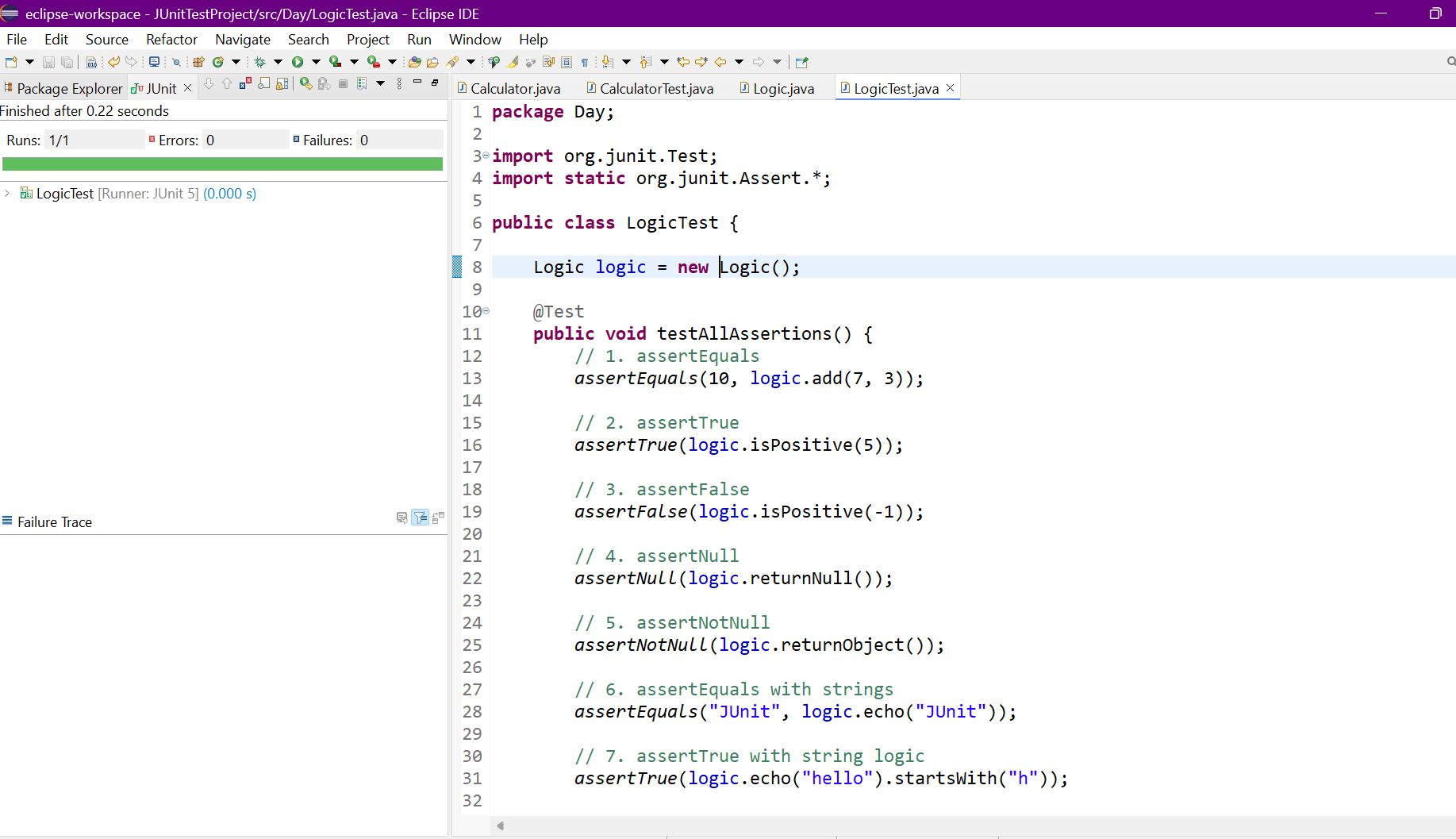
// 10. assertNotSame

assertNotSame(new String("a"), new String("a"));

}

}

Output:



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

Program:

Calculator.java;

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 ArithmeticException("Divide by zero");

return a / b;

}

}

CalculatorTest.java:

Program:

import org.junit.After;

import org.junit.Before;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calculator;

// Setup method runs before each test

@Before

public void setUp() {

calculator = new Calculator();

System.out.println("Setup: New Calculator created");

}

// Teardown method runs after each test

@After

public void tearDown() {

System.out.println("Teardown: Test complete\n");

}

@Test

public void testAddition() {

// Arrange

int a = 5, b = 3;

// Act

int result = calculator.add(a, b);

// Assert

assertEquals(8, result);

}

@Test

public void testSubtraction() {

// Arrange

int a = 10, b = 4;

// Act

int result = calculator.subtract(a, b);

// Assert

assertEquals(6, result);

}

@Test

public void testMultiplication() {

// Arrange

int a = 6, b = 7;

// Act

int result = calculator.multiply(a, b);

// Assert

assertEquals(42, result);

}

@Test

public void testDivision() {

// Arrange

int a = 20, b = 5;

// Act

int result = calculator.divide(a, b);

// Assert

assertEquals(4, result);

}

@Test(expected = ArithmeticException.class)

public void testDivideByZero() {

// Arrange

int a = 10, b = 0;

// Act

calculator.divide(a, b);

// Assert handled by @Test(expected = ...)

}

}

OUTPUT:

