JUnit\_Basic Testing Exercises

Exercise 1: Setting Up Junit

CalculatorTest.java

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

@Test

public void testAdd() {

Calculator calc = new Calculator();

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

}

}

Calculator.java

package com.example;

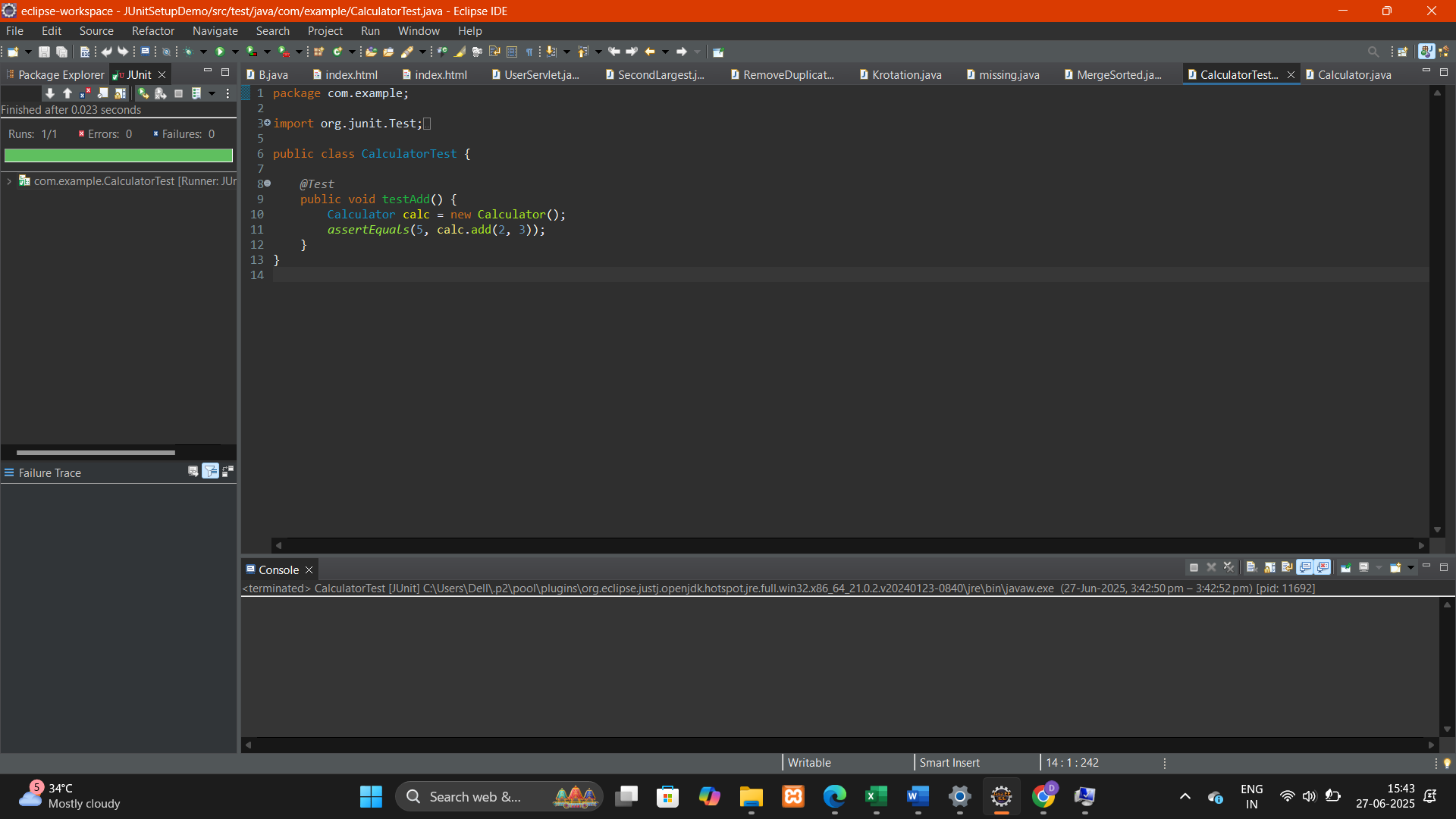
public class Calculator {

public int add(int a, int b) {

return a + b;

}

}



Exercise 3: Assertions in JUnit

Scenario:

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

Steps:

1. Write tests using various JUnit assertions.

Solution Code:

public class AssertionsTest {

@Test

public void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3);

// Assert true

assertTrue(5 > 3);

// Assert false

assertFalse(5 < 3);

// Assert null

assertNull(null);

// Assert not null

assertNotNull(new Object());

}

}

AssertionsTest.java

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class AssertionsTest {

@Test

void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3, "2 + 3 should equal 5");

// Assert true

assertTrue(5 > 3, "5 is greater than 3");

// Assert false

assertFalse(5 < 3, "5 is not less than 3");

// Assert null

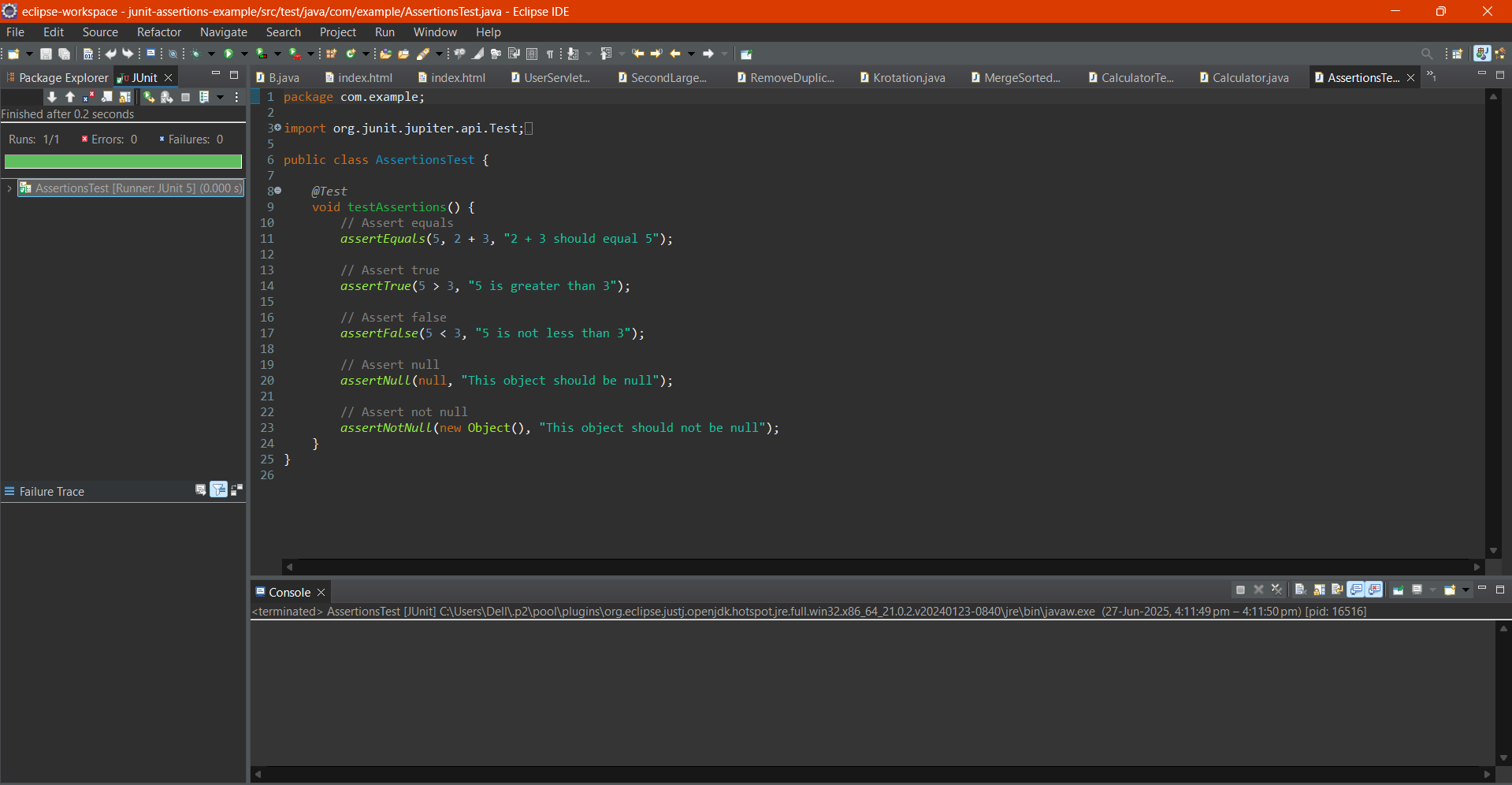
assertNull(null, "This object should be null");

// Assert not null

assertNotNull(new Object(), "This object should not be null");

}

}



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.

Steps:

1. Write tests using the AAA pattern.

2. Use @Before and @After annotations for setup and teardown methods.

AssertionsTest.java

package com.example;

import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.\*;

public class AssertionsTest {

@Test

void testAssertions() {

// Assert equals

assertEquals(5, 2 + 3, "2 + 3 should equal 5");

// Assert true

assertTrue(5 > 3, "5 is greater than 3");

// Assert false

assertFalse(5 < 3, "5 is not less than 3");

// Assert null

assertNull(null, "This object should be null");

// Assert not null

assertNotNull(new Object(), "This object should not be null");

}

}

CalculatorTest.java

package com.example;

import org.junit.jupiter.api.\*;

import static org.junit.jupiter.api.Assertions.\*;

public class CalculatorTest {

private Calculator calculator;

@BeforeEach

void setUp() {

// Setup - runs before each test

calculator = new Calculator();

}

@AfterEach

void tearDown() {

// Teardown - runs after each test

calculator = null;

}

@Test

void testAddition() {

// Arrange

int a = 10;

int b = 5;

// Act

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

// Assert

assertEquals(15, result, "10 + 5 should equal 15");

}

@Test

void testSubtraction() {

// Arrange

int a = 10;

int b = 3;

// Act

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

// Assert

assertEquals(7, result, "10 - 3 should equal 7");

}

}

class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}