JUnit Testing Exercises

Exercise 1: Setting Up JUnit

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

Step 1: Create a New Java Project

Create a Maven Project:
Eclipse: File > New > Maven Project

Step 2: Add JUnit Dependency

Add the following to your pom.xml inside the <dependencies> section:

```
<dependencies>
<dependency>
<groupId>junit</groupId>
<artifactId>junit</artifactId>
<version>4.13.2</version>
<scope>test</scope>
</dependency>
</dependencies>
```

Step 3: Create a New Java Class
Let’s assume you're writing a simple utility class.

src/main/java/com/example/Calculator.java

```java
package com.example;

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

```

Step 4: Create a JUnit Test Class

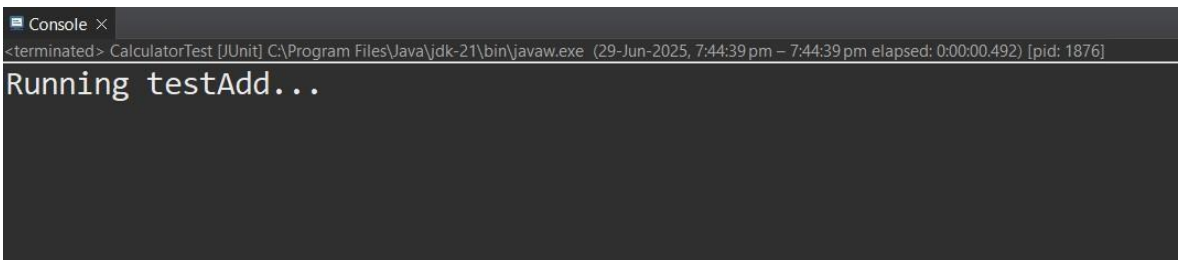
src/test/java/com/example/CalculatorTest.java

```java
package com.example;

import org.junit.Test;
import static org.junit.Assert.*;

public class CalculatorTest {
    @Test

    public void testAdd() {
        System.out.println("Running testAdd...");
        assertEquals(5, new Calculator().add(2, 3));
    }
}

```

How to Run the Test

In Eclipse: Right-click > Run As > JUnit Test

Expected Output



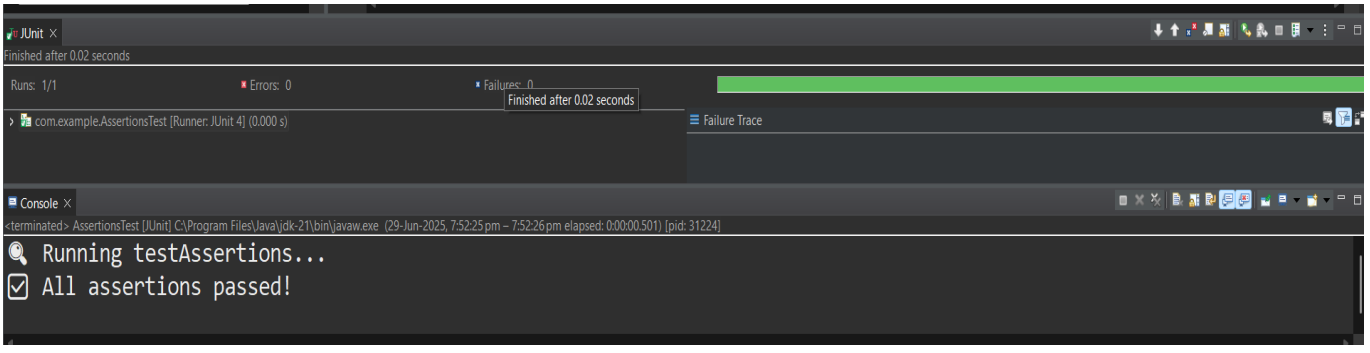
Exercise 3: Assertions in JUnit

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

1. Write tests using various JUnit assertions.

```java
package com.example;

import org.junit.Test;
import static org.junit.Assert.*;

public class AssertionsTest {

    @Test
    public void testAssertions() {
        System.out.println("🔍 Running testAssertions...");

        assertEquals(5, 2 + 3);
        assertTrue(5 > 3);
        assertFalse(5 < 3);
        assertNull(null);
        assertNotNull(new Object());

        System.out.println("☑ All assertions passed!");
    }
}
```

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.

Steps: 1. Write tests using the AAA pattern.

Step 1: Create the Class Under Test

Create a simple class to test:

- src/main/java/com/example/Calculator.java

```
1 package com.example;
2
3 public class Calculator {
4     public int add(int a, int b) {
5         return a + b;
6     }
7
8     public int multiply(int a, int b) {
9         return a * b;
10    }
11 }
12
```

Step 2: Create the Test Class

- src/test/java/com/example/CalculatorTest.java

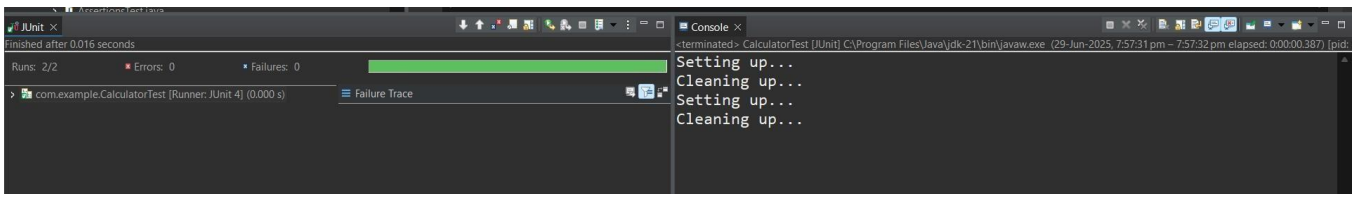
```java
package com.example;
import org.junit.After;
public class CalculatorTest {
    private Calculator calculator;

    @Before
    public void setUp() {
        System.out.println("Setting up...");
        calculator = new Calculator(); // Arrange
    }

    @After
    public void tearDown() {
        System.out.println("Cleaning up...");
        calculator = null;
    }

    @Test
    public void testAdd() {
        int result = calculator.add(2, 3);

        assertEquals(5, result);
    }

    @Test
    public void testMultiply() {
        int result = calculator.multiply(4, 5);
        assertEquals(20, result);
    }
}
```

Run the Tests

In Eclipse:

Right-click on CalculatorTest.java → Run 'CalculatorTest'

Expected Console Output



KANMANI MURUGHAIYAN
Superset id: 6407636