**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. If you are using Maven, add the following to your pom.xml: junit junit 4.13.2 test

3. Create a new test class in your project.

**CODE:**

**Pom.xml**

<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

https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>week2</groupId>

<artifactId>junit</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- JUnit 4 dependency -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**Src/main/java/Sample.java**

**package** junit;

**public** **class** Sample {

**public** **int** add(**int** a, **int** b) {

**return** a + b;

}

}

**Src/test/java/SampleTest.java**

**package** junit;

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

**import** org.junit.Test;

**public** **class** SampleTest {

@Test

**public** **void** testAdd() {

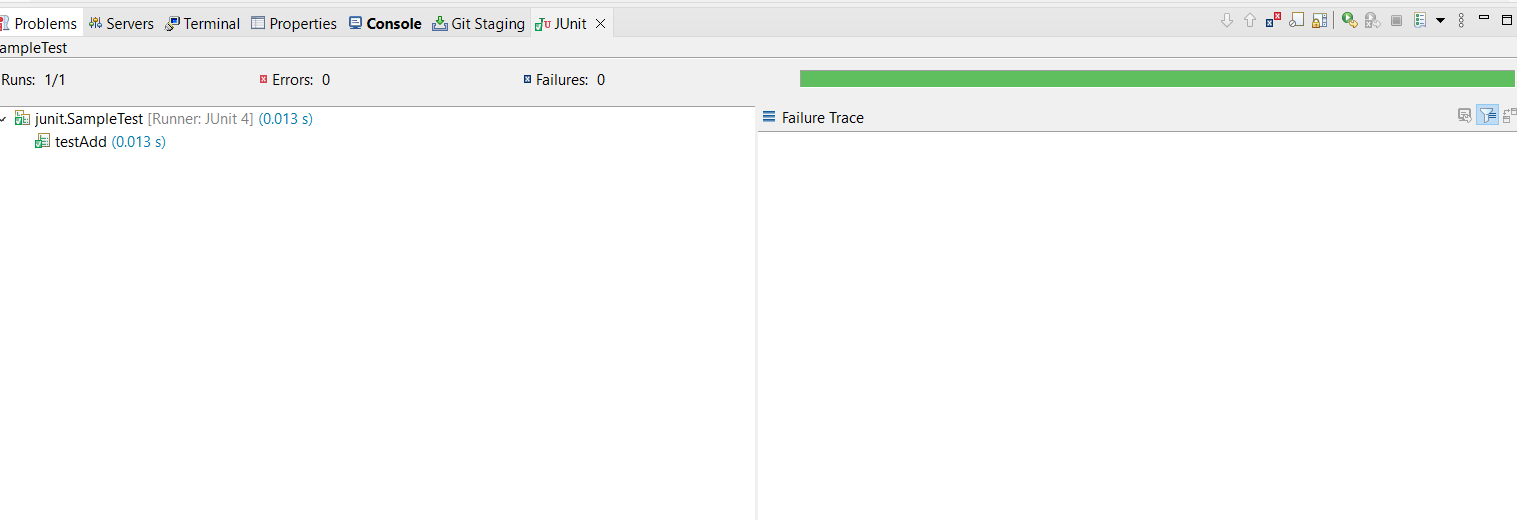
Sample sample = **new** Sample();

**int** result = sample.add(10, 5);

*assertEquals*(15, result);

}

}

****

**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());

}

}

**Src/test/java/AssertionTest.java**

**package** junit;

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

**import** org.junit.Test;

**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

Object obj1 = **null**;

*assertNull*(obj1);

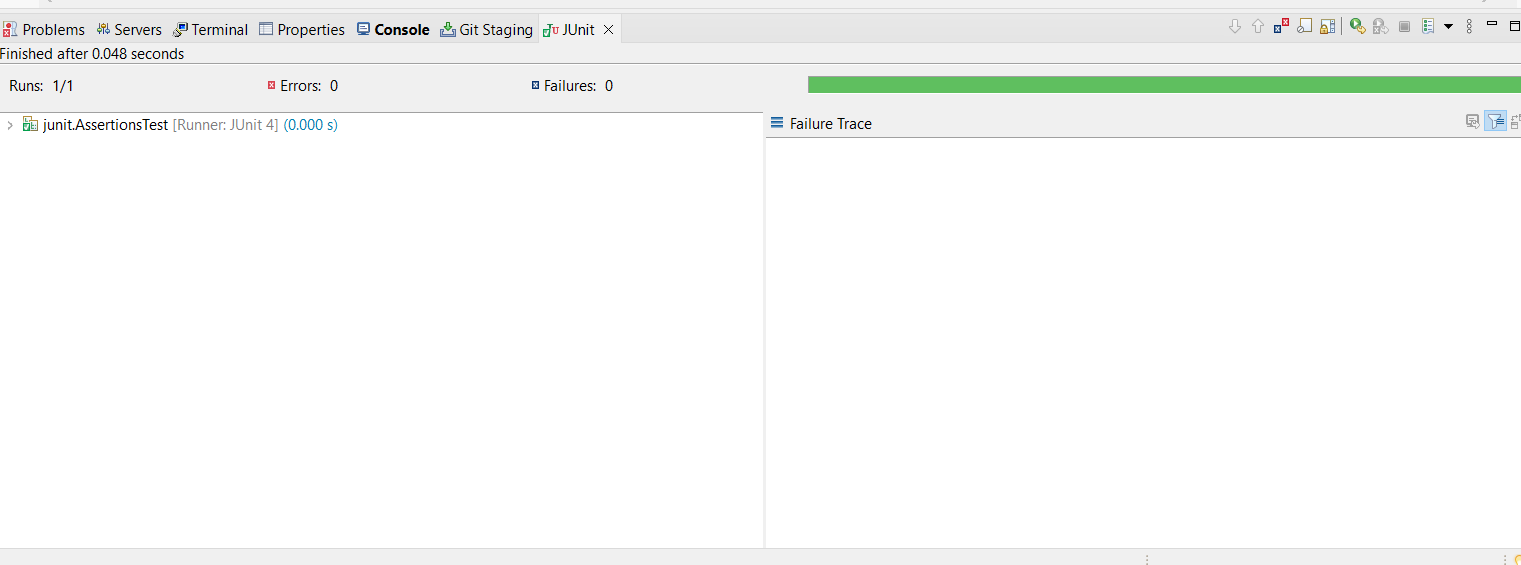
// Assert not null

Object obj2 = **new** Object();

*assertNotNull*(obj2);

}

}

****

**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.

**Src/main/java/Calculator.java**

**package** junit;

**public** **class** Calculator {

**public** **int** add(**int** a, **int** b) {

**return** a + b;

}

**public** **int** subtract(**int** a, **int** b) {

**return** a - b;

}

}

**Src/test/java/CalculatorTest.java**

**package** junit;

**import** org.junit.After;

**import** org.junit.Before;

**import** org.junit.Test;

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

**public** **class** CalculatorTest {

**private** Calculator calculator;

@Before

**public** **void** setUp() {

calculator = **new** Calculator();

System.***out***.println("Setup done");

}

@After

**public** **void** tearDown() {

calculator = **null**;

System.***out***.println("Tear down done");

}

@Test

**public** **void** testAddition() {

**int** a = 10;

**int** b = 5;

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

*assertEquals*(15, result);

}

@Test

**public** **void** testSubtraction() {

**int** a = 10;

**int** b = 3;

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

*assertEquals*(7, result);

}

}

****