# Exercise 1: Setting Up JUnit

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

* Maven Dependency (pom.xml):

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

* Create a test class (ExampleTest.java):

import org.junit.Test;  
import static org.junit.Assert.\*;  
  
public class ExampleTest {  
 @Test  
 public void sampleTest() {  
 assertEquals(2, 1 + 1);  
 }  
}

* Output:
* -------------------------------------------------------
* T E S T S
* -------------------------------------------------------
* Running ExampleTest
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.005 sec
* Results :
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

# Exercise 3: Assertions in JUnit

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

* Code (AssertionsTest.java):

import org.junit.Test;  
import static org.junit.Assert.\*;  
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());  
 }  
}

* Output:
* -------------------------------------------------------
* T E S T S
* -------------------------------------------------------
* Running AssertionsTest
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.006 sec
* Results :
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

# Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup andTeardown Methods in JUnit

Scenario: Organize your tests using the Arrange-Act-Assert (AAA) pattern and use setup and teardown methods.

* Code (CalculatorTest.java):

import org.junit.\*;  
import static org.junit.Assert.\*;  
public class CalculatorTest {  
 private Calculator calc;  
  
 @Before  
 public void setUp() {  
 calc = new Calculator();  
 System.out.println("Setup complete");  
 }  
  
 @After  
 public void tearDown() {  
 System.out.println("Teardown complete");  
 }  
  
 @Test  
 public void testAdd() {  
 // Arrange is done in setUp()  
 // Act  
 int result = calc.add(2, 3);  
 // Assert  
 assertEquals(5, result);  
 }  
}  
  
class Calculator {  
 public int add(int a, int b) {  
 return a + b;  
 }  
}

* Output:
* Setup complete
* Teardown complete
* -------------------------------------------------------
* T E S T S
* -------------------------------------------------------
* Running CalculatorTest
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.007 sec
* Results :
* Tests run: 1, Failures: 0, Errors: 0, Skipped: 0