# JUnit Testing Exercises

## 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:

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

3. Create a new test class in your project.

## Exercise 2: Writing Basic JUnit Tests

Scenario:  
You need to write basic JUnit tests for a simple Java class.

Steps:  
1. Create a new Java class with some methods to test.  
2. Write JUnit tests for these methods.

## 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());  
 }  
}

## 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.

Example Code:

public class CalculatorTest {  
 private Calculator calculator;  
  
 @Before  
 public void setUp() {  
 calculator = new Calculator();  
 }  
  
 @After  
 public void tearDown() {  
 calculator = null;  
 }  
  
 @Test  
 public void testAddition() {  
 // Arrange  
 int a = 2;  
 int b = 3;  
  
 // Act  
 int result = calculator.add(a, b);  
  
 // Assert  
 assertEquals(5, result);  
 }  
}