# 1. JUnit\_Basic Testing Exercises

## Exercise 1: Setting Up JUnit

Create a simple Java project and set up JUnit 5.  
- Include the JUnit 5 dependency in your build tool (e.g., Maven or Gradle).  
- Write a basic test class with one test method that checks if 2 + 2 equals 4.

import org.junit.jupiter.api.Test;  
import static org.junit.jupiter.api.Assertions.assertEquals;  
  
public class BasicTest {  
  
 @Test  
 void testAddition() {  
 assertEquals(4, 2 + 2);  
 }  
}

## Exercise 3: Assertions in JUnit

Write test methods using various assertions like:  
- assertEquals  
- assertTrue  
- assertFalse  
- assertNull  
- assertNotNull

import org.junit.jupiter.api.Test;  
import static org.junit.jupiter.api.Assertions.\*;  
  
public class AssertionTests {  
  
 @Test  
 void testAssertions() {  
 assertEquals(5, 2 + 3);  
 assertTrue(10 > 1);  
 assertFalse(1 > 10);  
 assertNull(null);  
 assertNotNull("JUnit");  
 }  
}

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

Create a test class that demonstrates:  
- The Arrange-Act-Assert (AAA) pattern  
- Setup using @BeforeEach  
- Teardown using @AfterEach

import org.junit.jupiter.api.\*;  
  
import static org.junit.jupiter.api.Assertions.assertEquals;  
  
public class CalculatorTest {  
  
 Calculator calc;  
  
 @BeforeEach  
 void setUp() {  
 calc = new Calculator();  
 }  
  
 @AfterEach  
 void tearDown() {  
 calc = null;  
 }  
  
 @Test  
 void testAddition() {  
 // Arrange is done in setUp()  
 // Act  
 int result = calc.add(2, 3);  
 // Assert  
 assertEquals(5, result);  
 }  
}  
  
class Calculator {  
 int add(int a, int b) {  
 return a + b;  
 }  
}