Exercise 3: Assertions in Junit

import org.junit.Test;

import static org.junit.Assert.\*;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

public class AssertionsTest {

@Test

public void testBasicAssertions() {

assertEquals("Addition should work", 5, 2 + 3);

assertEquals(0.333, 1.0/3.0, 0.001);

assertTrue("5 should be greater than 3", 5 > 3);

assertFalse("5 should not be less than 3", 5 < 3);

Object nullObj = null;

Object realObj = new Object();

assertNull("Object should be null", nullObj);

assertNotNull("Object should not be null", realObj);

}

@Test

public void testArrayAndCollectionAssertions() {

// Array equality

int[] expectedArray = {1, 2, 3};

int[] actualArray = {1, 2, 3};

assertArrayEquals("Arrays should be equal", expectedArray, actualArray);

List<String> expectedList = Arrays.asList("Java", "JUnit", "Testing");

List<String> actualList = new ArrayList<>(expectedList);

assertEquals("Lists should be equal", expectedList, actualList);

}

@Test

public void testReferenceAssertions() {

String str1 = "hello";

String str2 = "hello";

String str3 = new String("hello");

assertSame("References should point to same object", str1, str2);

assertNotSame("References should not point to same object", str1, str3);

}

@Test

public void testExceptionAssertions() {

try {

int result = 1 / 0;

fail("Should have thrown ArithmeticException");

} catch (ArithmeticException e) {

assertEquals("/ by zero", e.getMessage());

}

try {

int[] arr = new int[5];

int val = arr[10];

fail("Should have thrown ArrayIndexOutOfBoundsException");

} catch (ArrayIndexOutOfBoundsException e) {

assertTrue(e.getMessage().contains("10"));

}

}

@Test(expected = NullPointerException.class)

public void testExpectedException() {

String str = null;

str.length();

}

@Test

public void testCombinedAssertions() {

String message = "Hello JUnit";

assertAll("Multiple assertions",

() -> assertTrue(message.startsWith("H")),

() -> assertTrue(message.endsWith("t")),

() -> assertEquals(11, message.length())

);

}

}