Testing Final Lab

Name: Uzair Ali

REG NO: SP21-BSE-025

Submitted To: Sir, Mukhtiar Zamin

Question 1: pick any algorithm of your choice and using EF*Boundary value analysis write 5 test (Effective) and automate with junit.

Ans:

My Algorithm:

```
public static int findMax(int[] arr) {
    int max = arr[0];
    for (int i = 1; i < arr.length; i++) {
        if (arr[i] > max) {
            max = arr[i];
        }
    }
    return max;
}
```

Test Case ID	Test Case	Description	Input	Expected outcome	Actual Outcome	Status
TC-01	Valid Input with a Single Element	Verify that the algorithm returns the correct maximum value when the input array has a single element.	**[1]**	**1**	1	Pass

TC-02	Valid Input with Multiple Elements	Verify that the algorithm returns the correct maximum value when the input array has multiple elements.	`[1, 2, 3, 4, 5]`	`5`	5	Pass
TC-03	Invalid Input (Empty Array)	Verify that the algorithm throws an exception when the input array is empty.	`[]`	`throws ArrayIndexOutOfBoundsException`	`throws ArrayIndexOutOfBoundsException`	Pass
TC-04	Edge Value (Two Elements)	Verify that the algorithm returns the correct maximum value when the input array has two elements.	`[1, 2]`	`2`	2	Pass
TC-05	Maximum Value	Verify that the algorith	`[1, 2, 3,`	`100`	100	Pass

	(Large Array)	m returns the correct maximum value when the input array has a large number of elements	..., 100]			

Automation with junit:

```
import org.junit.Test;
import static org.junit.Assert.*;

public class FindMaxTest {

    @Test
    public void testSingleElement() {
        int[] arr = {1};
        assertEquals(1, FindMax.findMax(arr));
    }

    @Test
    public void testMultipleElements() {
        int[] arr = {1, 2, 3, 4, 5};
        assertEquals(5, FindMax.findMax(arr));
    }

    @Test(expected = ArrayIndexOutOfBoundsException.class)
    public void testEmptyArray() {
        int[] arr = {};
        FindMax.findMax(arr);
    }

    @Test
```

```
    public void testEdgeValue() {
        int[] arr = {1, 2};
        assertEquals(2, FindMax.findMax(arr));
    }

    @Test
    public void testMaximumValue() {
        int[] arr = new int[100];
        for (int i = 0; i < 100; i++) {
            arr[i] = i + 1;
        }
        assertEquals(100, FindMax.findMax(arr));
    }
}
```

Verify Test Result:

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;

public class VerifyTestResultTest {

    @Test
    public void testVerifyTestResult() {
        // Arrange
        FindMax findMax = new FindMax();
        int[] singleElementArray = {1};
        int[] multipleElementsArray = {1, 2, 3, 4, 5};
        int[] emptyArray = {};
        int[] edgeValueArray = {2, 2, 2, 2, 2};
        int[] maximumValueArray = {100, 20, 30, 40, 50};

        // Act
        int singleElementResult = findMax.findMax(singleElementArray);
        int multipleElementsResult = findMax.findMax(multipleElementsArray);
        int emptyArrayResult = 0; // expected to throw
ArrayIndexOutOfBoundsException
        int edgeValueResult = findMax.findMax(edgeValueArray);
        int maximumValueResult = findMax.findMax(maximumValueArray);

        // Assert
        assertEquals(1, singleElementResult);
        assertEquals(5, multipleElementsResult);
```

```
        try {
            findMax.findMax(emptyArray);
            assert false; // expected to throw ArrayIndexOutOfBoundsException
        } catch (ArrayIndexOutOfBoundsException e) {
            // expected exception
        }
        assertEquals(2, edgeValueResult);
        assertEquals(100, maximumValueResult);
    }
}
```

Verify Test Question Count:

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;

public class VerifyTestQuestionCountTest {

    @Test
    public void testVerifyTestQuestionCount() {
        // Arrange
        int expectedTestQuestionCount = 5;

        // Act
        int actualTestQuestionCount = FindMaxTest.class.getMethods().length;

        // Assert
        assertEquals(expectedTestQuestionCount, actualTestQuestionCount);
    }
}
```

Verify Class Analytics:

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;

public class VerifyClassAnalyticsTest {

    @Test
    public void testVerifyClassAnalytics() {
        // Arrange
        int expectedMethodCount = 1;
        int expectedTestCasesCount = 5;
```

```
        double expectedMethodCoverage = 100.0;

        // Act
        int actualMethodCount = FindMax.class.getMethods().length;
        int actualTestCasesCount = FindMaxTest.class.getMethods().length;
        double actualMethodCoverage = (double) actualTestCasesCount /
actualMethodCount * 100;

        // Assert
        assertEquals(expectedMethodCount, actualMethodCount);
        assertEquals(expectedTestCasesCount, actualTestCasesCount);
        assertEquals(expectedMethodCoverage, actualMethodCoverage, 0.01);
    }
}
```