Experiment-7

Author: Atharva Paliwal

Roll no: B2-40

Date: 07 April 2021

Aim: To perform White-Box Testing to test the functionalities using JUnit testing tool.

Problem Statement:

[1] Implement a Class Quadratic representing solution involving finding roots of a quadratic equation. Create a JUnit Test Suite incorporating all possible test cases covering independent paths.

[2] Choose any class from your Class Diagram (one which involves substantial computational logic) and implement it. Now create JUnit suit to test this class appropriately.

Description:

White Box Testing is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing and Glass box testing.

Code:

[Problem Statement 1]

(quadratic.java)

```
import static java.lang.Math.*;


class quadratic {


    private int a , b , c;


    public quadratic(int x, int y, int z){

        a = x;

        b = y;

        c = z;

    }
```

```
    public String findRoots(){
        if (a == 0) {
            System.out.println("Invalid");
            return "Invalid";
        }
        int d = b * b - 4 * a * c;
        double sqrt_val = sqrt(abs(d));
        if (d > 0) {
            System.out.println("Roots are real and different \n");
            return((double)(-b + sqrt_val) / (2 * a) + "\n"+
(double)(-b - sqrt_val) / (2 * a));
        }
        else if (d == 0) {
            System.out.println("Roots are real and same \n");
            return(-(double)b / (2 * a) + "\n"+ -(double)b / (2 *
a));
        }
        else {
            System.out.println("Roots are complex \n");
            return(-(double)b / (2 * a) + " + i"+ sqrt_val + "\n"+ -
(double)b / (2 * a)+ " - i" + sqrt_val);
        }
    }
}
```

(quadraticTest.java)

```
import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
```

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

public class quadraticTest {

    public quadraticTest() {
    }

    @BeforeClass
    public static void setUpClass() {
    }

    @AfterClass
    public static void tearDownClass() {
    }

    @Before
    public void setUp() {
    }

    @After
    public void tearDown() {
    }

    /**
     * Test of findRoots method, of class quadratic.
     */
    @Test
    public void testCaseI(){
        quadratic q = new quadratic(0,4,4);
        String expected = "Invalid" ;
        String actual = q.findRoots();
```

```
        assertEquals(expected, actual);
    }

    @Test
    public void testCaseII(){
        quadratic q = new quadratic(1,-7,12);

        String expected = "4.0" + "\n" + "3.0" ;
        String actual = q.findRoots();
        assertEquals(expected, actual);
    }

    @Test
    public void testCaseIII(){
        quadratic q = new quadratic(1,2,1);

        String expected = "-1.0" + "\n" + "-1.0" ;
        String actual = q.findRoots();
        assertEquals(expected, actual);
    }

    @Test
    public void testCaseIV(){
        quadratic q = new quadratic(2,2,1);

        String expected = "-0.5 + i2.0" + "\n" + "-0.5 - i2.0" ;
        String actual = q.findRoots();
        assertEquals(expected, actual);
    }
}
```

[Problem Statement 2]

(Invoice.java)

```
package invoice;


/**
 *
 * @author ACER
 */
public class Invoice {
    public double generateAmount(String type,double weight,double
distance){


        switch(type){
            case "air":
                if (weight <= 8) {
                    return weight * 3;
                }
                else if ((weight >= 9) && (weight <= 16)) {
                    return weight * 4;
                }
                else if (weight >= 17){
                    return weight * 6;
                }
                break;
            case "road":
                if (weight <= 8) {
                    return weight*1.50 + distance*0.5;
                }
                else {
                    if ((weight >= 9) && (weight <= 16)) {
                        return 2.35 + distance*0.5;
                    }
```

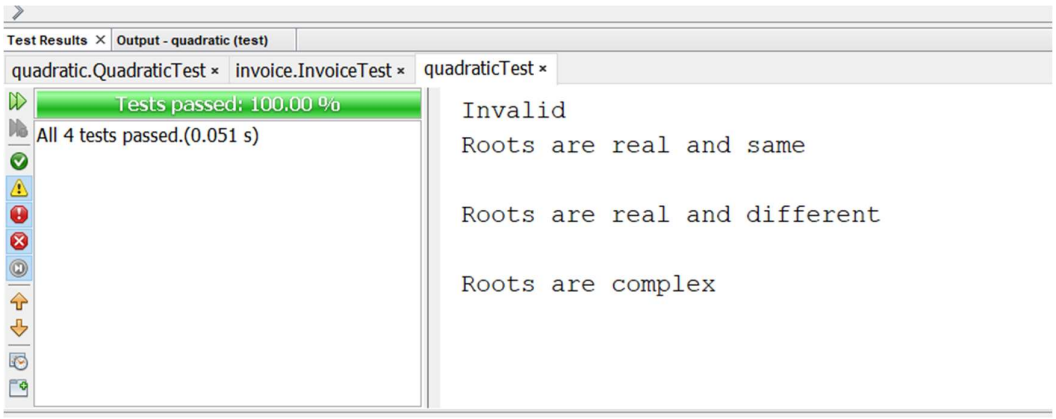
```
                    else if (weight >= 17) {
                        return weight*3.25 + distance*0.5;
                    }
                }
                break;
            case "water":
                if (weight <= 8) {
                    return weight*0.50;
                }
                else {
                    if ((weight >= 9) && (weight <= 16)) {
                        return weight*1.50;
                    }
                    else if (weight >= 17) {
                        return weight*2.50;
                    }
                }
                break;
        }
        return 0;
    }
}
```

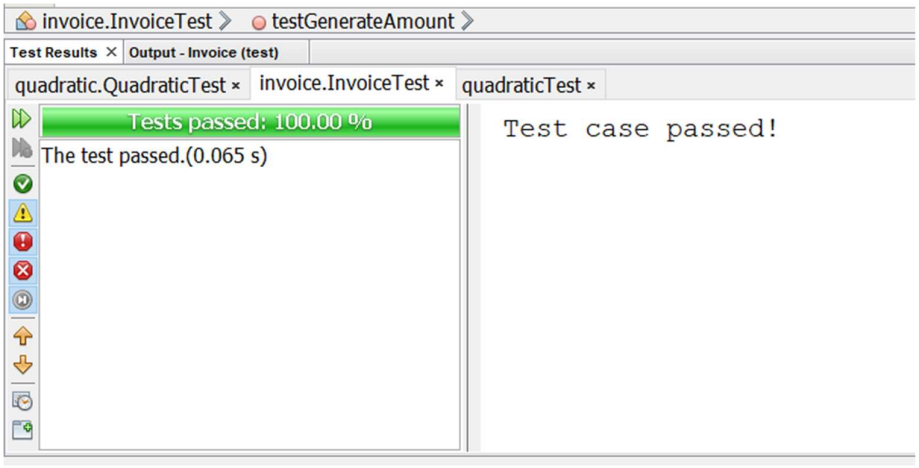
(InvoiceTest.java)

```
package invoice;

import org.junit.After;
import org.junit.AfterClass;
import org.junit.Before;
import org.junit.BeforeClass;
import org.junit.Test;
import static org.junit.Assert.*;

public class InvoiceTest {

    public InvoiceTest() {
    }

    @BeforeClass
    public static void setUpClass() {
    }

    @AfterClass
    public static void tearDownClass() {
    }

    @Before
    public void setUp() {
    }

    @After
```

```
    public void tearDown() {
    }

    @Test
    public void testGenerateAmount() {
        Invoice instance = new Invoice();

        //AIR
        double expResult = 48.0;
        double result = instance.generateAmount("air", 12.0 , 0.0);
        assertEquals(expResult, result, 0.0);

        // ROAD
        double expected = 310.5;
        double actual = instance.generateAmount("road",7,600);
        assertEquals(expected,actual,0.0);

        // WATER
        double expectedw = 62.5;
        double actualw = instance.generateAmount("water",25,0);
        assertEquals(expectedw,actualw,0.0);

        System.out.println("Test case passed!");
    }

}
```

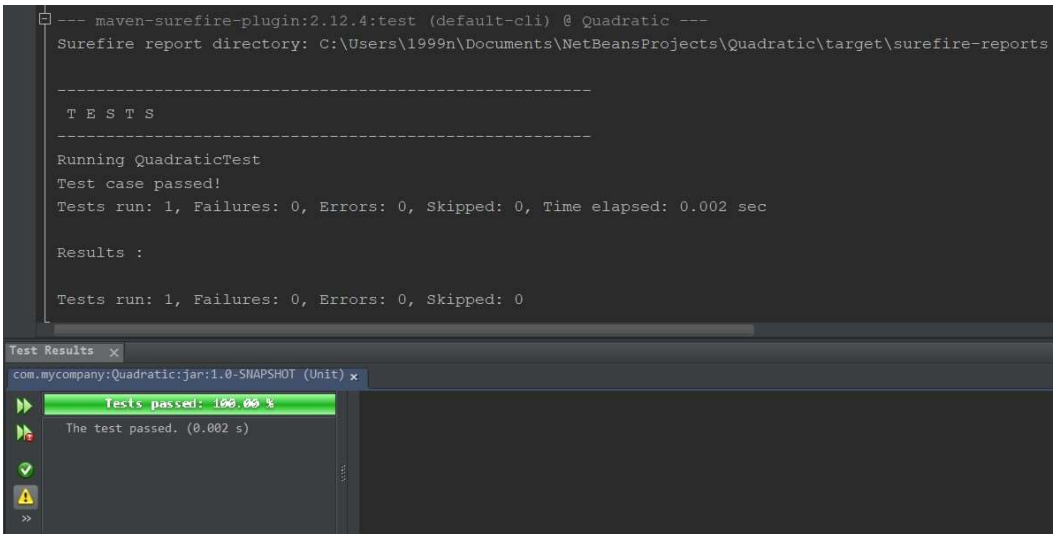
OUTPUT


Test Results
Output - quadratic (test)
quadratic.QuadraticTest ×
invoice.InvoiceTest ×
quadraticTest ×
Tests passed: 100.00 %
All 4 tests passed.(0.051 s)
Invalid
Roots are real and same
Roots are real and different
Roots are complex

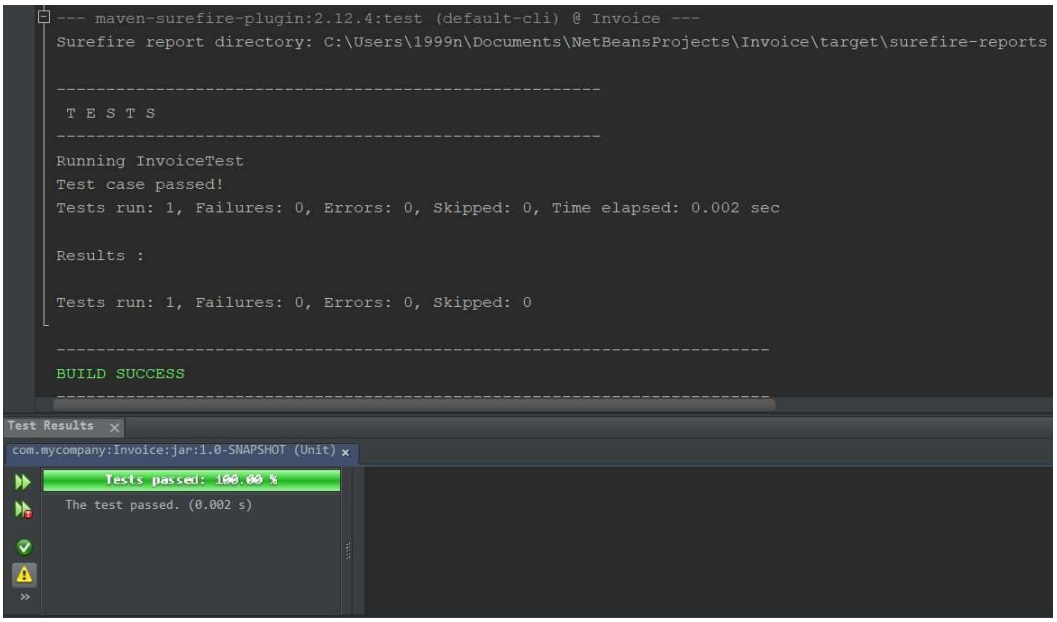

invoice.InvoiceTest
testGenerateAmount
Test Results
Output - Invoice (test)
quadratic.QuadraticTest ×
invoice.InvoiceTest ×
quadraticTest ×
Tests passed: 100.00 %
The test passed.(0.065 s)
Test case passed!

Output:

[Problem Statement 1]



[Problem Statement 2]


--- maven-surefire-plugin:2.12.4:test (default-cli) @ Invoice ---
Surefire report directory: C:\Users\1999n\Documents\NetBeansProjects\Invoice\target\surefire-reports
T E S T S
Running InvoiceTest
Test case passed!
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.002 sec
Results :
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
BUILD SUCCESS
Test Results
com.mycompany:Invoice:jar:1.0-SNAPSHOT (Unit)
Tests passed: 100.00 %
The test passed. (0.002 s)