espol

ESCUELA SUPERIOR

PÓLITECNICA DEL LITORAL

WORKSHOP ABOUT EMPIRICAL SOFTWARE TESTING

AUTORES:

- AGUILAR MORA OSWALDO
- BERMUDEZ MOREIRA KAREN
- BERNAL MOREIRA GUILLERMO
- ORTIZ HOLGUIN EDUARDO
- WONG PAVON HUGO

SUBJECT: ING. SOFTWARE II

TUTOR: DR. MERA CARLOS

DEADLINE: 2020/06/11

1. Abstract

This document contains the technical report corresponding to the first group workshop called **"WORKSHOP ABOUT EMPIRICAL SOFTWARE TESTING"** of **GROUP#4** belonging to the **SOFTWARE II ENGINEERING** course of **2020-PAO I**.

The report contains a description of the workshop, the pseudocode, the testing approach, the JAVA implementation and the test result.

2. Description

The triangle problem is presented together with a pseudocode that provides a solution to the identification of the type of triangle, or if the values entered do not form this figure, with their respective validations, in this workshop it is requested to implement the pseudocode, design the cases of tests deemed necessary and finally execute the test cases.

2.1. Triangle problem

It is one of the most used problems for teaching **software tests**, it consists of a program that must take as input three integer numerical values that correspond to the sides of a triangle, must evaluate the necessary conditions and finally determine if the entered values correspond to a **scalene triangle**, an **equilateral** triangle, an **isosceles** triangle, or ultimately the values do not form a triangle.

To determine it, there are the following conditions:

- The entered values must be in the range of [1,200].
- Values must comply with triangular inequality.
- The equality of the sides determines whether they are equilateral, scalene or isosceles.

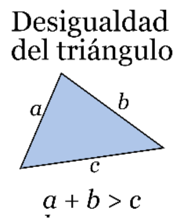


Figure 1.- Triangular Inequality

2.2. Implementation

It is requested to carry out the implementation using as a JAVA language additionally to use some tool to carry out the tests and the collaborative development tool Git.

3. Pseudocode

```
Program triangle'
Dim a, b, c As Integer
Dim c1, c2, c3, IsATriangle As Boolean

'Step 1: Get Input
Do
     Output("Enter 3 integers which are sides of a triangle")
     Input(a, b, c)
     c1 = (1 ≤ a) AND (a ≤ 200)
     c2 = (1 ≤ b) AND (b ≤ 200)
     c3 = (1 ≤ c) AND (c ≤ 200)
     If NOT(c1)
          Then Output("Value of a is not in the range of permitted values")
     EndIf
     If NOT(c2)
          Then Output("Value of b is not in the range of permitted values")
     EndIf
     If NOT(c3)
          ThenOutput("Value of c is not in the range of permitted values")
     EndIf
Until c1 AND c2 AND c3
Output("Side A is",a)
Output("Side B is",b)
Output("Side C is",c)

'Step 2: Is A Triangle?
If (a < b + c) AND (b < a + c) AND (c < a + b)
     Then IsATriangle = True
     Else IsATriangle = False
EndIf

'Step 3: Determine Triangle Type
If IsATriangle
     Then If (a = b) AND (b = c)
          Then Output ("Equilateral")
          Else If (a ≠ b) AND (a ≠ c) AND (b ≠ c)
               Then Output ("Scalene")
               Else Output ("Isosceles")
          EndIf
     EndIf
     Else Output("Not a Triangle")
EndIf
End triangle
```

Figure 2.- Triangle Problem Pseudocode

4. Design of Test Cases

It has been decided to divide the tests into the following categories:

- Data Type Tests
- Range of Values Tests
- Results Tests

4.1. Data Type Tests

These tests will evaluate the **robustness** of the program, that is, if it does not crash due to the values entered and if it notifies the user of the error. They will be tested with non-integer values either **floating point** or text **strings**.

4.2. Range of Values Tests

In these tests the correct validation of the variables will be evaluated, if they are in the designated range between **1** and **200**, if they show the correct messages.

4.3. Results Tests

In these tests, the final results will be evaluated after passing the previous validations, correct values will be used, and the results will be verified among the 4 possible cases **(Equilateral, Scalene, Isosceles, Non-Triangle).**

4.4. Detail of test cases

# Case	Commentary	a	b	c	Result
Data Type Tests					
1	String 1	30	b	2	**Error 1**
2	String 2	sk	1	Sd	**Error 1**
3	Floating point 1	21	5	3,4	**Error 1**
4	Floating point 2	5,4	2,1	199,2	**Error 1**
5	Mix	4	7,5	abc	**Error 1**
Range of Values Tests					
6	Exceedance a	274	12	97	**Error 2**
7	Exceedance b	76	599	3	**Error 2**
8	Exceedance c	55	55	201	**Error 2**
9	Insufficient a	0	2	2	**Error 2**
10	Insufficient b	12	-45	90	**Error 2**
11	Insufficient c	124	125	-1	**Error 2**
12	Mix 1	156	500	-12	**Error 2**
13	Mix 2	0	-999	999	**Error 2**
Results Tests					
14	Equilateral 1	5	5	5	Equilateral

15	Equilateral 2	200	200	200	Equilateral
16	Scalene 1	5	3	7	Scalene
17	Scalene 2	70	120	170	Scalene
18	Isosceles 1	6	3	6	Isosceles
19	Isosceles 2	132	132	140	Isosceles
20	No triangle 1	6	13	6	No triangle
21	No triangle 2	40	24	199	No triangle
22	No triangle 3	1	2	3	No triangle

5. Source Code

5.1. Repository

Github was used as a collaboration tool for the development of the workshop, the link is:

- https://github.com/kbermude/Taller1_Software

5.2. Development Considerations

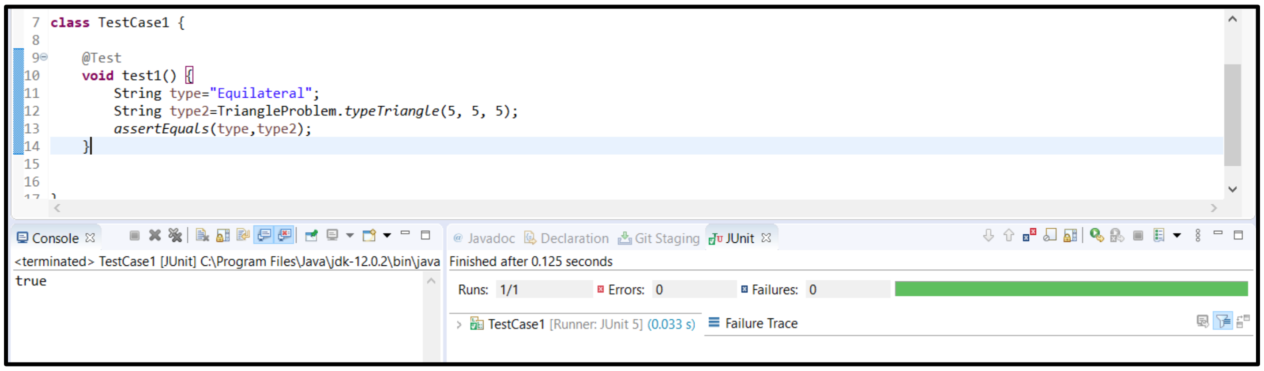
For the development of the activity and its implementation, the following points were considered:

- **Eclipse** was used as Java IDE.
- The **jUnit** tool was used for the tests.

6. Tests and Results

# Case	Commentary	a	b	c	Theorycal Result	Test Result
Data Type Tests						
1	String 1	30	b	2	**Error 1**	**System Crash**
2	String 2	sk	1	Sd	**Error 1**	**System Crash**
3	Floating point 1	21	5	3,4	**Error 1**	**System Crash**
4	Floating point 2	5,4	2,1	199,2	**Error 1**	**System Crash**
5	Mix	4	7,5	abc	**Error 1**	**System Crash**
Range of Values Tests						
6	Exceedance a	274	12	97	**Error 2**	**Error 2**
7	Exceedance b	76	599	3	**Error 2**	**Error 2**
8	Exceedance c	55	55	201	**Error 2**	**Error 2**
9	Insufficient a	0	2	2	**Error 2**	**Error 2**
10	Insufficient b	12	-45	90	**Error 2**	**Error 2**
11	Insufficient c	124	125	-1	**Error 2**	**Error 2**
12	Mix 1	156	500	-12	**Error 2**	**Error 2**
13	Mix 2	0	-999	999	**Error 2**	**Error 2**
Results Tests						
14	Equilateral 1	5	5	5	**Equilateral**	**Equilateral**
15	Equilateral 2	200	200	200	**Equilateral**	**Equilateral**
16	Scalene 1	5	3	7	**Scalene**	**Scalene**
17	Scalene 2	70	120	170	**Scalene**	**Scalene**
18	Isosceles 1	5	2	5	**Isosceles**	**Isosceles**
19	Isosceles 2	132	132	140	**Isosceles**	**Isosceles**
20	No triangle 1	6	13	6	**No triangle**	**No triangle**
21	No triangle 2	40	24	199	**No triangle**	**No triangle**
22	No triangle 3	1	2	3	**No triangle**	**No triangle**

7. Evidence

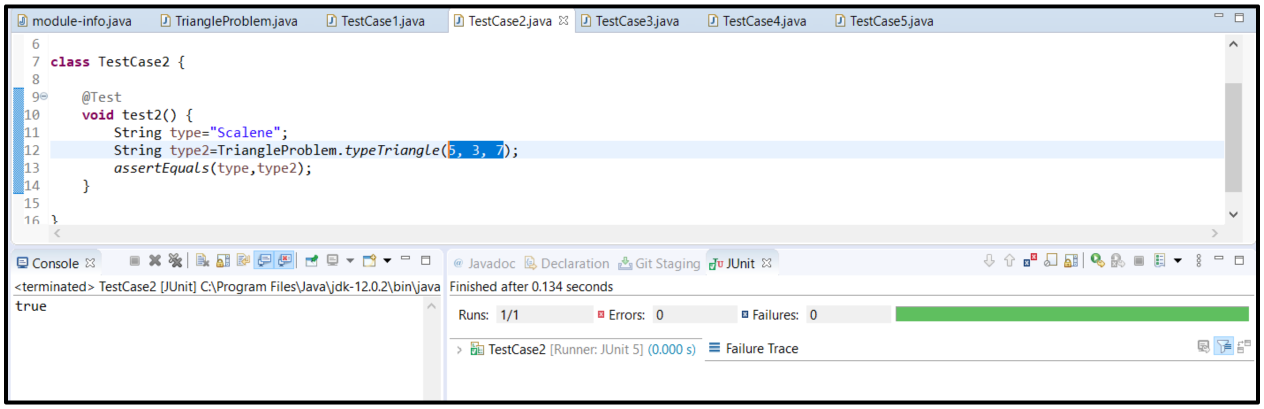


```java
class TestCase1 {

    @Test
    void test1() {
        String type="Equilateral";
        String type2=TriangleProblem.typeTriangle(5, 5, 5);
        assertEquals(type,type2);
    }


}
```

```
<terminated> TestCase1 [JUnit] C:\Program Files\Java\jdk-12.0.2\bin\java
true
```

```
Finished after 0.125 seconds
Runs: 1/1    Errors: 0    Failures: 0
TestCase1 [Runner: JUnit 5] (0.033 s)
```

Figure 3.- Test #14 (5,5,5)

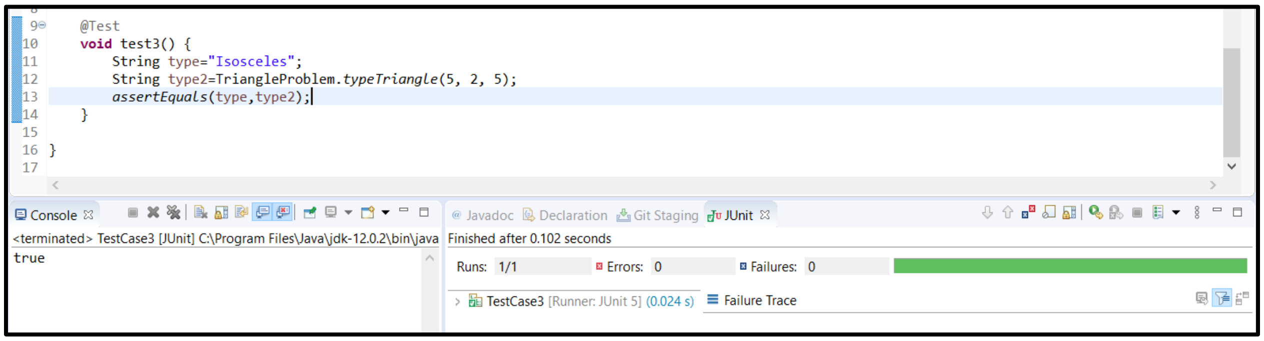


```java

class TestCase2 {

    @Test
    void test2() {
        String type="Scalene";
        String type2=TriangleProblem.typeTriangle(5, 3, 7);
        assertEquals(type,type2);
    }

}
```

```
<terminated> TestCase2 [JUnit] C:\Program Files\Java\jdk-12.0.2\bin\java
true
```

```
Finished after 0.134 seconds
Runs: 1/1    Errors: 0    Failures: 0
TestCase2 [Runner: JUnit 5] (0.000 s)
```

Figure 4.- Test #16 (5,3,7)

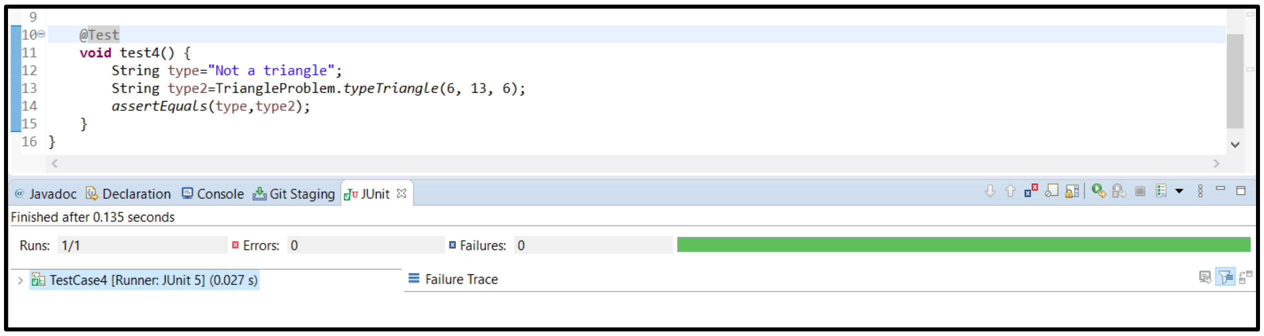


```java
    @Test
    void test3() {
        String type="Isosceles";
        String type2=TriangleProblem.typeTriangle(5, 2, 5);
        assertEquals(type,type2);
    }

}

```

```
<terminated> TestCase3 [JUnit] C:\Program Files\Java\jdk-12.0.2\bin\java
true
```

```
Finished after 0.102 seconds
Runs: 1/1    Errors: 0    Failures: 0
TestCase3 [Runner: JUnit 5] (0.024 s)
```

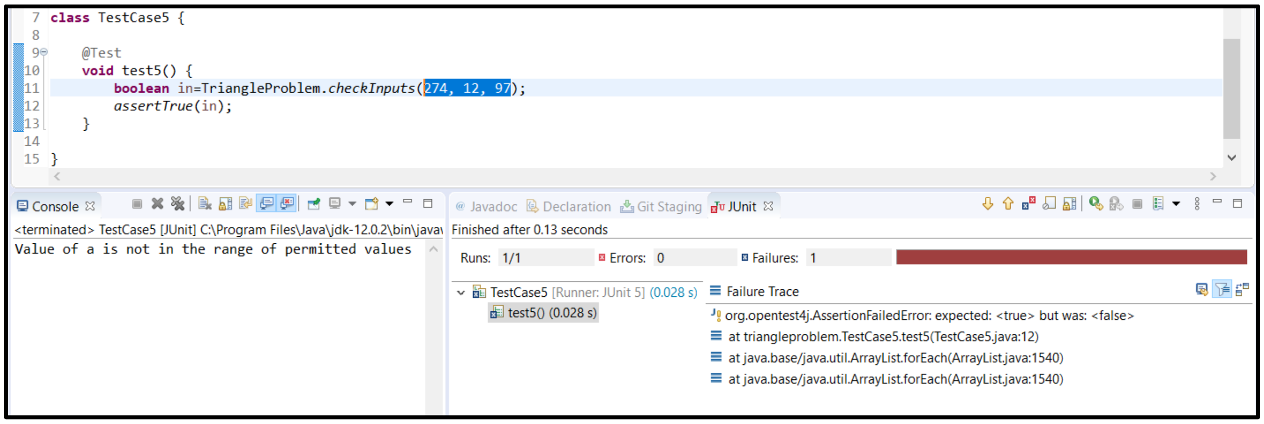
Figure 5.- Test #18 (5,2,5)



```java
@Test
void test4() {
    String type="Not a triangle";
    String type2=TriangleProblem.typeTriangle(6, 13, 6);
    assertEquals(type,type2);
}
}
```

```
Finished after 0.135 seconds
Runs: 1/1    Errors: 0    Failures: 0
TestCase4 [Runner: JUnit 5] (0.027 s)
Failure Trace
```

Figure 6.- Test #20 (6,13,6)

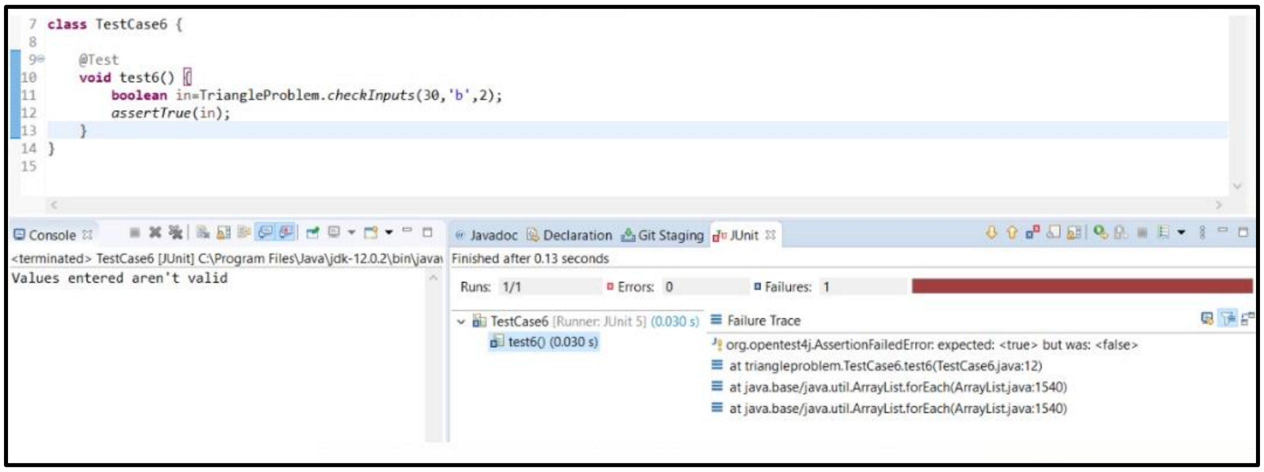


```java
class TestCase5 {

    @Test
    void test5() {
        boolean in=TriangleProblem.checkInputs(274, 12, 97);
        assertTrue(in);
    }

}
```

```
<terminated> TestCase5 [JUnit] C:\Program Files\Java\jdk-12.0.2\bin\java
Value of a is not in the range of permitted values
```

```
Finished after 0.13 seconds
Runs: 1/1    Errors: 0    Failures: 1
TestCase5 [Runner: JUnit 5] (0.028 s)
  test5() (0.028 s)
Failure Trace
org.opentest4j.AssertionFailedError: expected: <true> but was: <false>
at triangleproblem.TestCase5.test5(TestCase5.java:12)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1540)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1540)
```

Figure 7.- Test #6 (274,12,97)



```java
class TestCase6 {

    @Test
    void test6() {
        boolean in=TriangleProblem.checkInputs(30,'b',2);
        assertTrue(in);
    }
}
```

```
<terminated> TestCase6 [JUnit] C:\Program Files\Java\jdk-12.0.2\bin\java
Values entered aren't valid
```

```
Finished after 0.13 seconds
Runs: 1/1    Errors: 0    Failures: 1
TestCase6 [Runner: JUnit 5] (0.030 s)
  test6() (0.030 s)
Failure Trace
org.opentest4j.AssertionFailedError: expected: <true> but was: <false>
at triangleproblem.TestCase6.test6(TestCase6.java:12)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1540)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1540)
```

Figure 8.- Test #1 (30,'b',2)

8. Conclusions

- The program pass **17/22** tests.
- The **5 faults** are in the first group of tests **"Data type Tests"** , the program input as defined by integer values, if we pass other data type the program present a system crash.
- The program is correctly implemented in input range, in case that input is over or under the range the system presents a message predefined.
- The program presents the correct result in case that the input values correspond to equilateral, scalene or isosceles triangle.
- The program presents the correct message in case that the input values do not correspond to a triangle.
- **JUnit** is a useful tool to test the software.