RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Course Number	891
Course Title	Software Testing and Quality Assurance
Semester/Year	W2023

Instructor	Dr. Reza Samavi

Lab No.	2

Lab Title	Test Suite, Advanced JUnit, Unit Parameterized Tests, and Theories

Submission Date	February 6th, 2023
Due Date	February 6th, 2023

Student Name	Student ID	Signature*
Abdulrehman Khan	500968727	A.K.

**By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work.*

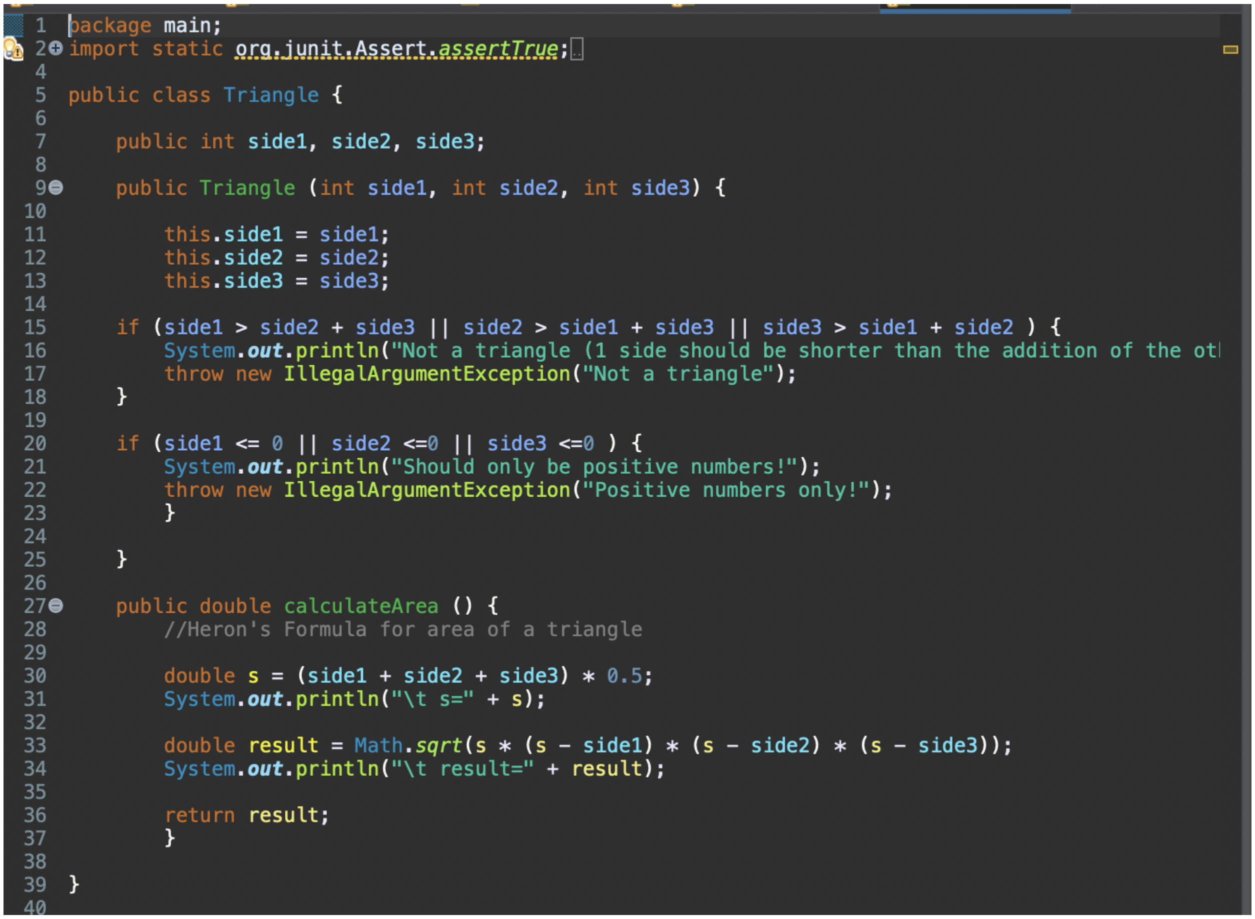
PART 1

Q1:

What will happen if the user creates a triangle with new Triangle(3,4,100)? Is this something you should or can test in JUnit? How might you do so?

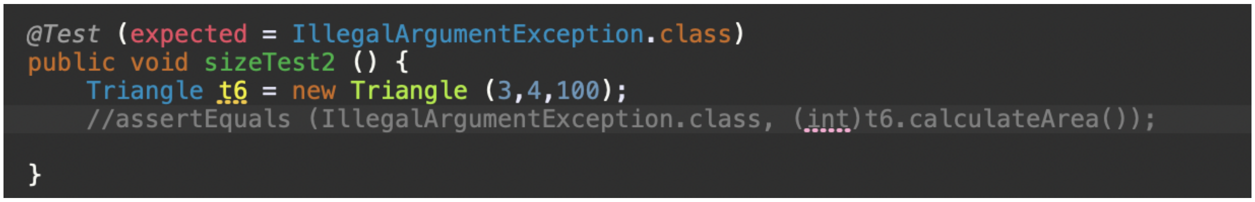
An exception is expected if a user creates a new Triangle (3,4,100) because this is not a triangle, one side (100) is bigger than the addition of the other two sides (3 and 4). Hence violating what it means to be a triangle! This can be seen below via the testing method sizeTest2()

Triangle.java:



```java
package main;
import static org.junit.Assert.assertTrue;

public class Triangle {

    public int side1, side2, side3;

    public Triangle (int side1, int side2, int side3) {

        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;

    if (side1 > side2 + side3 || side2 > side1 + side3 || side3 > side1 + side2 ) {
        System.out.println("Not a triangle (1 side should be shorter than the addition of the ot
        throw new IllegalArgumentException("Not a triangle");
    }

    if (side1 <= 0 || side2 <=0 || side3 <=0 ) {
        System.out.println("Should only be positive numbers!");
        throw new IllegalArgumentException("Positive numbers only!");
        }

    }

    public double calculateArea () {
        //Heron's Formula for area of a triangle

        double s = (side1 + side2 + side3) * 0.5;
        System.out.println("\t s=" + s);

        double result = Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
        System.out.println("\t result=" + result);

        return result;
        }

}
```

TriangleTest.java



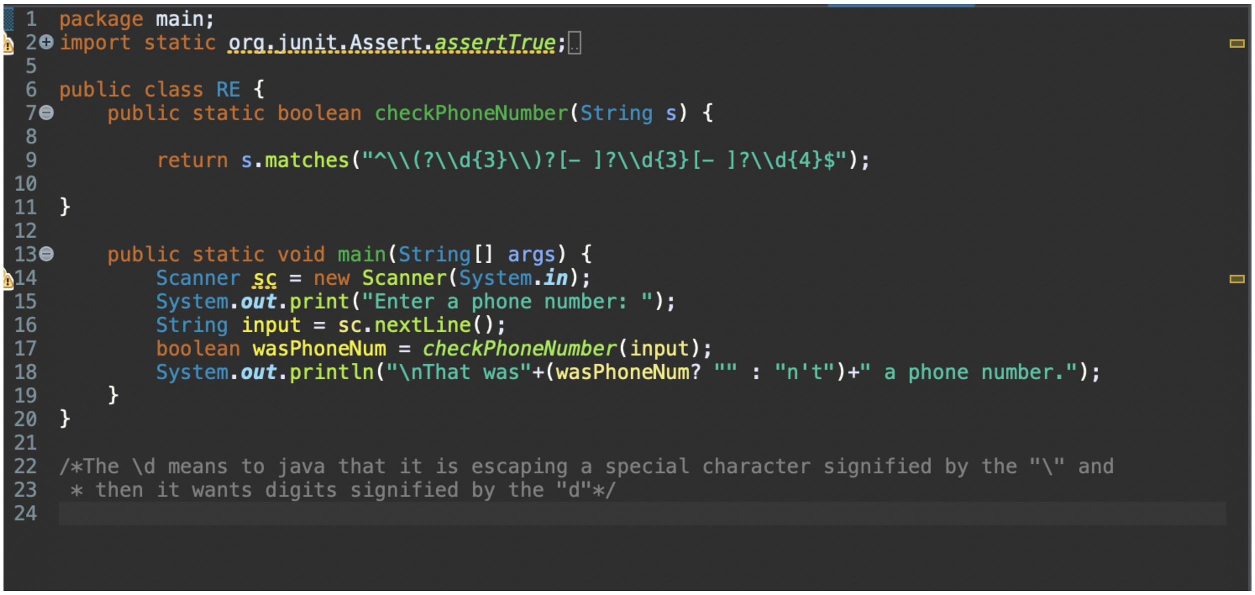
```java
@Test (expected = IllegalArgumentException.class)
public void sizeTest2 () {
    Triangle t6 = new Triangle (3,4,100);
    //assertEquals (IllegalArgumentException.class, (int)t6.calculateArea());

}
```

Q2:

What does the “\d” mean to Java when you are creating a String?

The \d means to java that it is escaping a special character signified by the “\” and then it wants digits signified by the “d”

RE.java



```java
package main;
import static org.junit.Assert.assertTrue;

public class RE {
    public static boolean checkPhoneNumber(String s) {

        return s.matches("^\\(?\\d{3}\\)?[- ]?\\d{3}[- ]?\\d{4}$");

}

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a phone number: ");
        String input = sc.nextLine();
        boolean wasPhoneNum = checkPhoneNumber(input);
        System.out.println("\nThat was"+(wasPhoneNum? "" : "n't")+" a phone number.");
    }
}

/*The \d means to java that it is escaping a special character signified by the "\" and
 * then it wants digits signified by the "d"*/

```

Test the current regular expression with the following inputs as the "VALID" inputs: (123)123−1234 and (123) 456−7890
Why are not these working?

The given regex did not have the correct symbols to allow for spaces between the 3 digits which was added inside the []

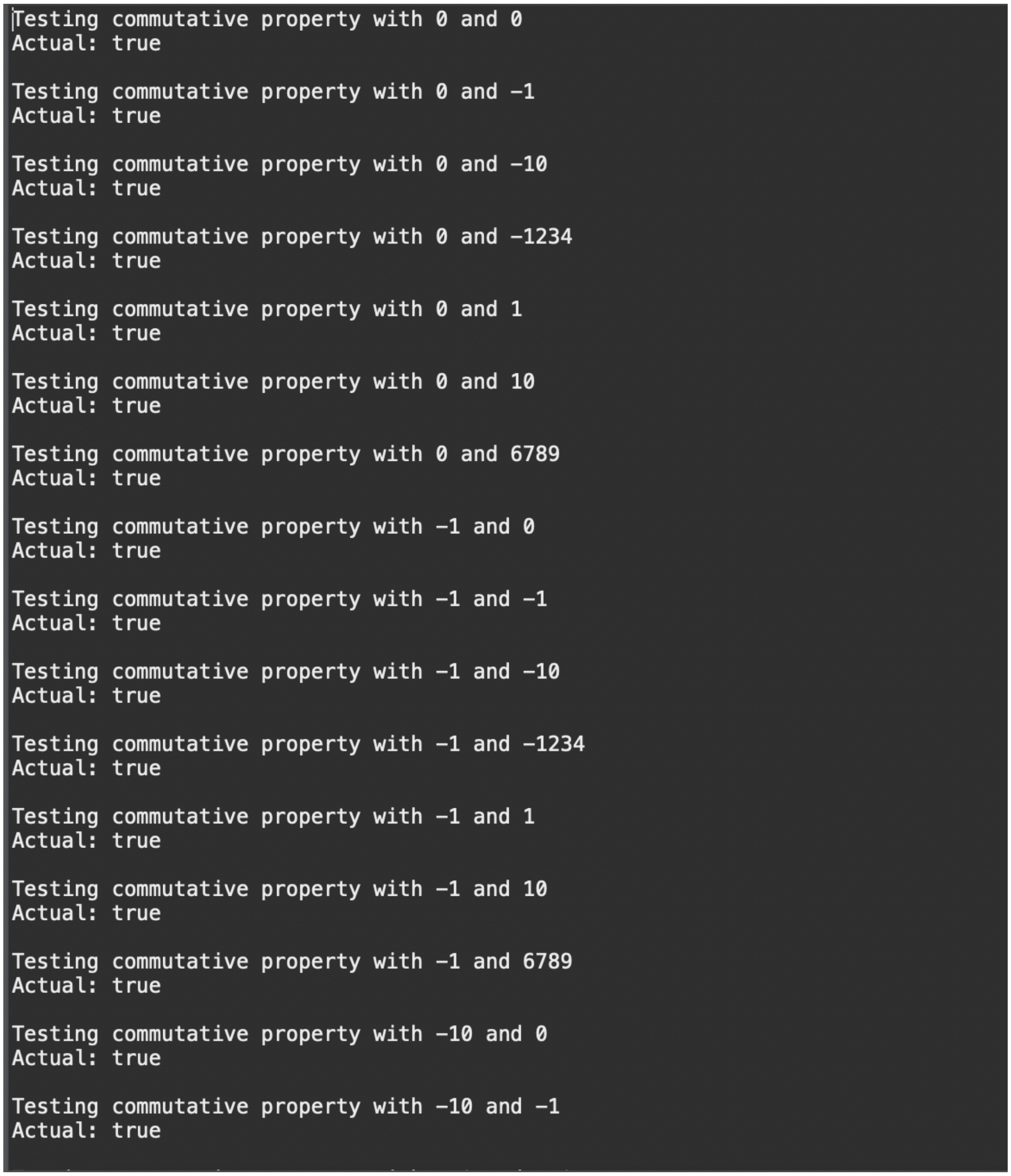
PART 2

Q3:

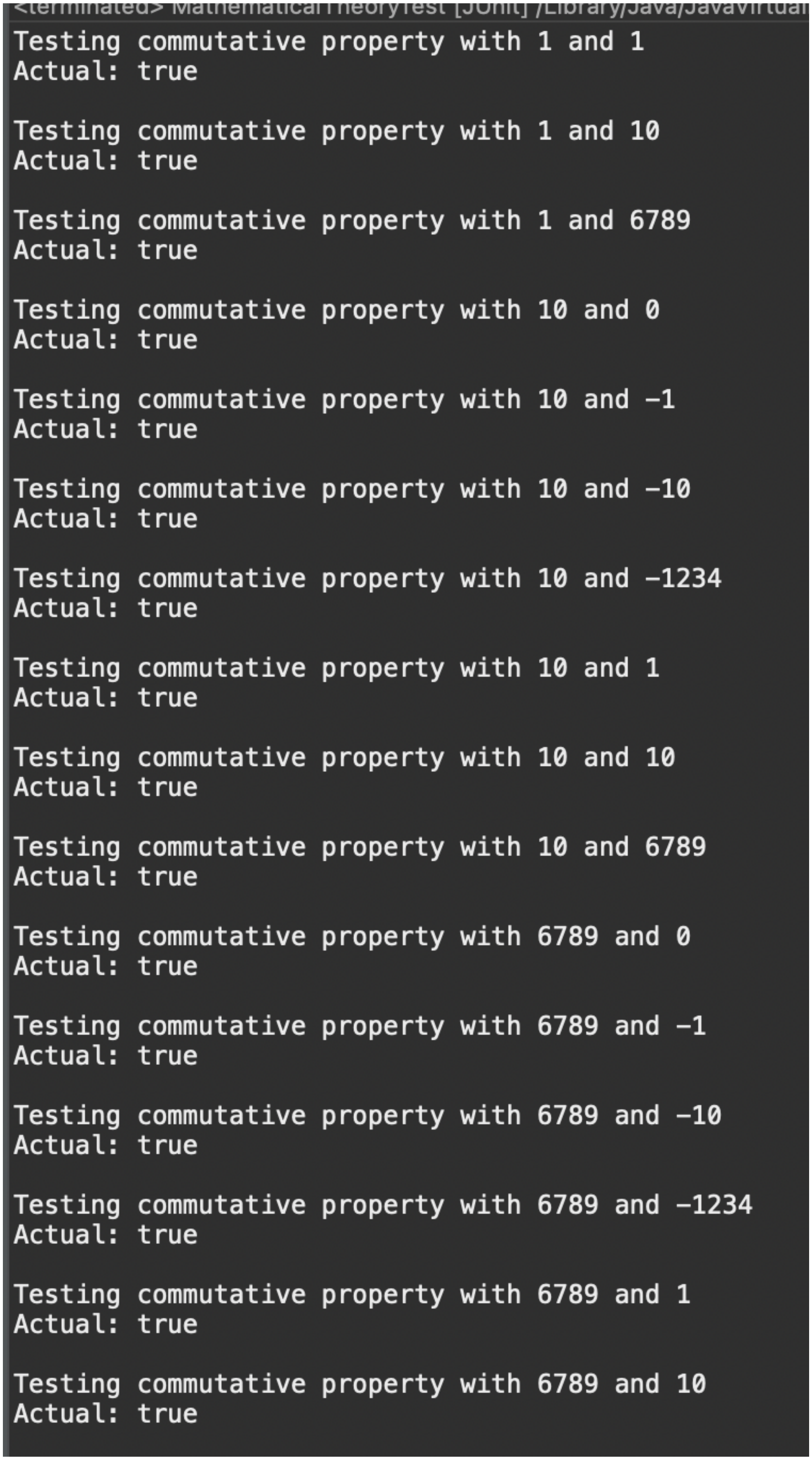
3.

Consider a new value set newval= {0, -1, -10, -1234, 1, 10, 6789} as your data points for input values in your testing program for every possible pair of a and b. What will be your results, and why? Demonstrate and justify your achieved results

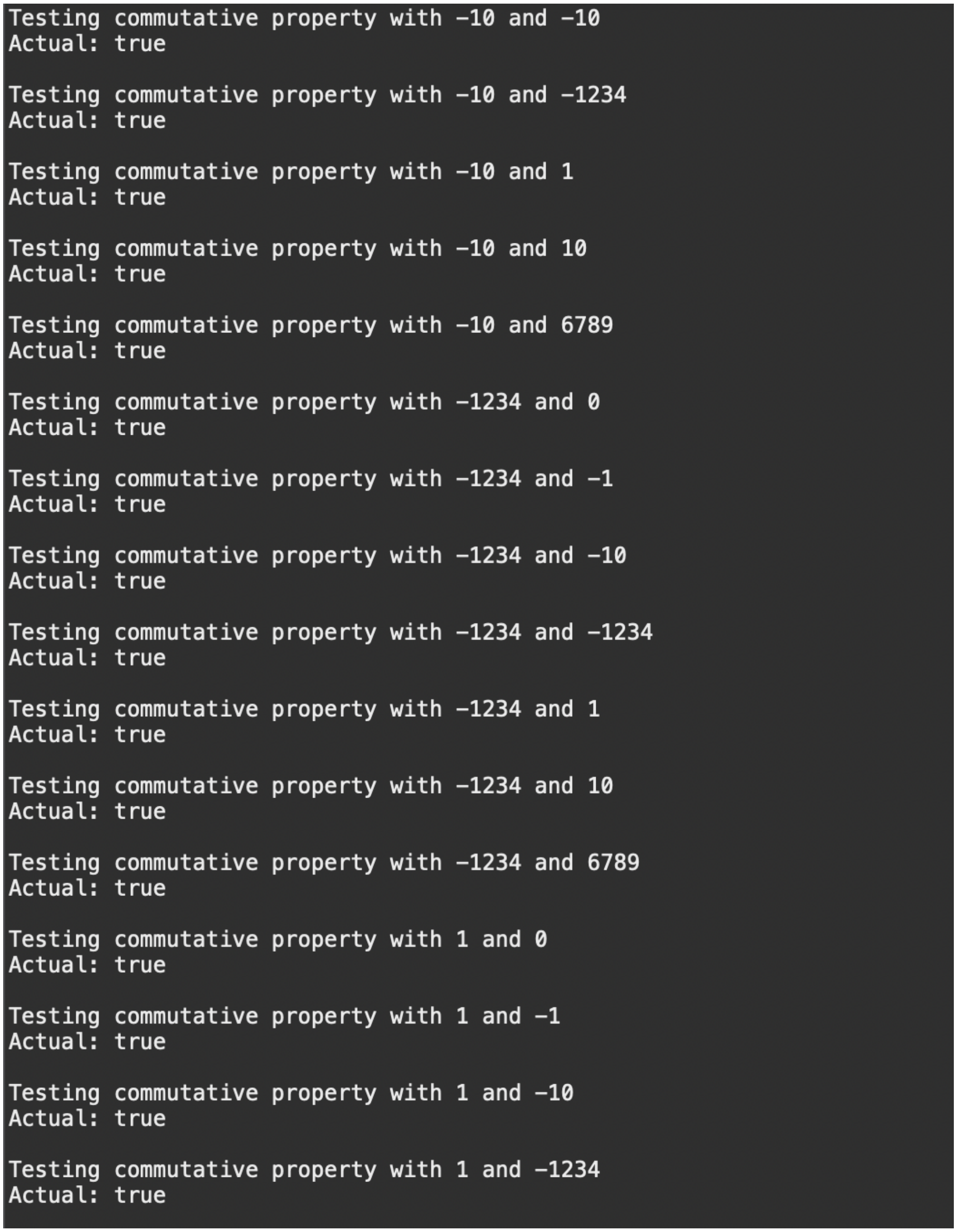
The commutative property testing property should always produce an actual value of true and actually was as can be witnessed below:



```
Testing commutative property with 0 and 0
Actual: true

Testing commutative property with 0 and -1
Actual: true

Testing commutative property with 0 and -10
Actual: true

Testing commutative property with 0 and -1234
Actual: true

Testing commutative property with 0 and 1
Actual: true

Testing commutative property with 0 and 10
Actual: true

Testing commutative property with 0 and 6789
Actual: true

Testing commutative property with -1 and 0
Actual: true

Testing commutative property with -1 and -1
Actual: true

Testing commutative property with -1 and -10
Actual: true

Testing commutative property with -1 and -1234
Actual: true

Testing commutative property with -1 and 1
Actual: true

Testing commutative property with -1 and 10
Actual: true

Testing commutative property with -1 and 6789
Actual: true

Testing commutative property with -10 and 0
Actual: true

Testing commutative property with -10 and -1
Actual: true
```

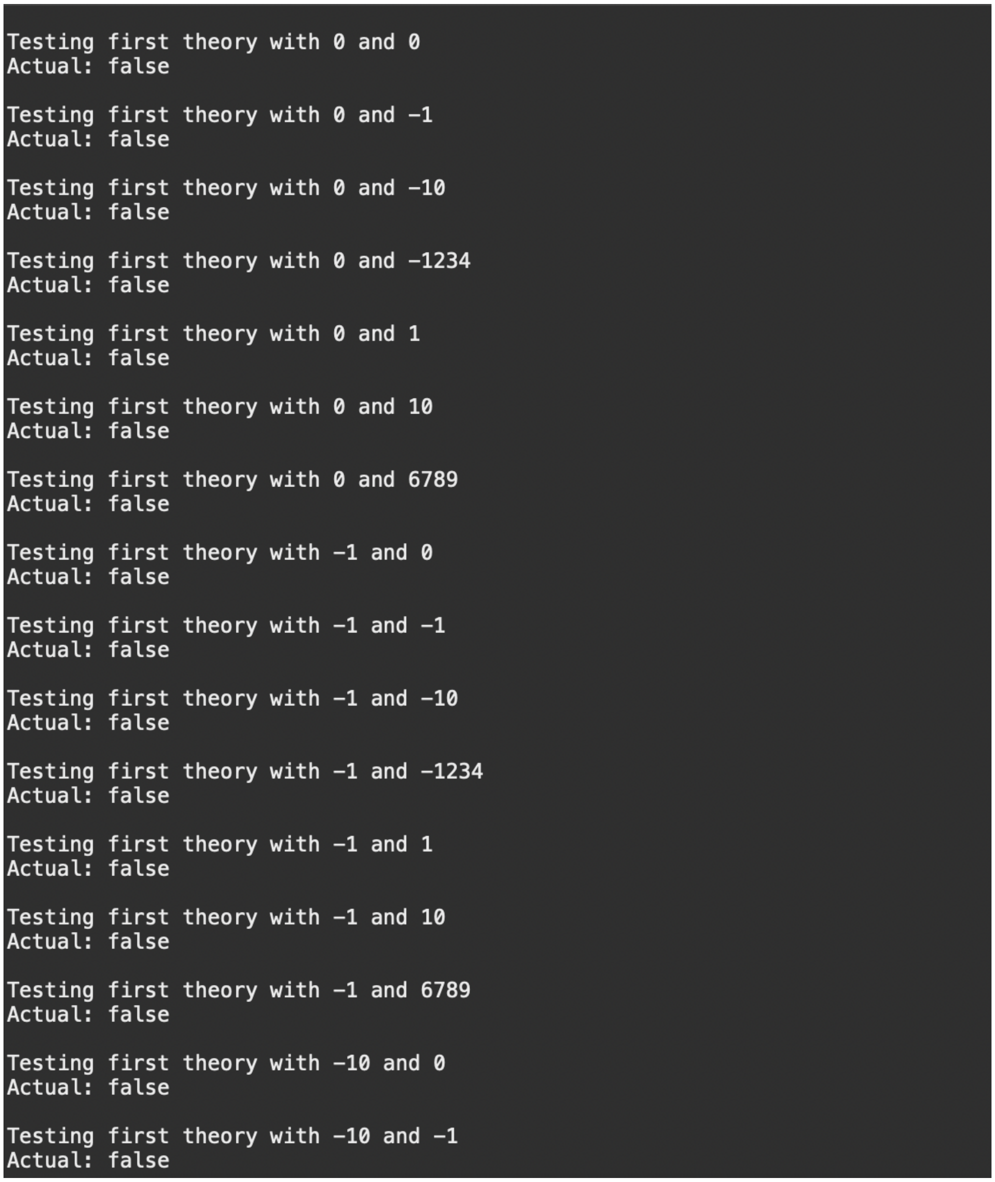


```
<terminated> MathematicalTheoryTest [JUnit] /Library/Java/JavaVirtual
Testing commutative property with 1 and 1
Actual: true

Testing commutative property with 1 and 10
Actual: true

Testing commutative property with 1 and 6789
Actual: true

Testing commutative property with 10 and 0
Actual: true

Testing commutative property with 10 and -1
Actual: true

Testing commutative property with 10 and -10
Actual: true

Testing commutative property with 10 and -1234
Actual: true

Testing commutative property with 10 and 1
Actual: true

Testing commutative property with 10 and 10
Actual: true

Testing commutative property with 10 and 6789
Actual: true

Testing commutative property with 6789 and 0
Actual: true

Testing commutative property with 6789 and -1
Actual: true

Testing commutative property with 6789 and -10
Actual: true

Testing commutative property with 6789 and -1234
Actual: true

Testing commutative property with 6789 and 1
Actual: true

Testing commutative property with 6789 and 10
Actual: true
```

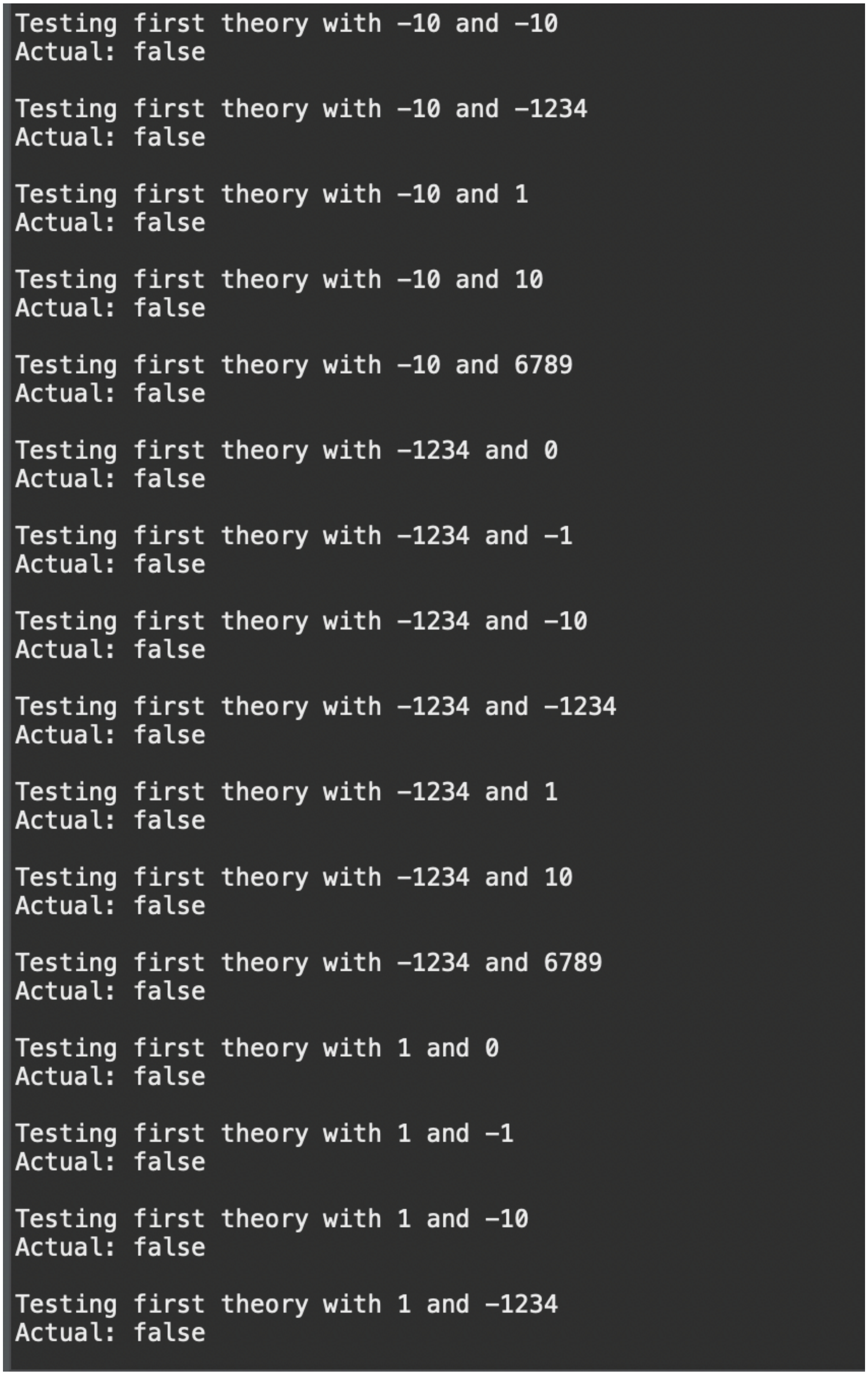


```
Testing commutative property with -10 and -10
Actual: true

Testing commutative property with -10 and -1234
Actual: true

Testing commutative property with -10 and 1
Actual: true

Testing commutative property with -10 and 10
Actual: true

Testing commutative property with -10 and 6789
Actual: true

Testing commutative property with -1234 and 0
Actual: true

Testing commutative property with -1234 and -1
Actual: true

Testing commutative property with -1234 and -10
Actual: true

Testing commutative property with -1234 and -1234
Actual: true

Testing commutative property with -1234 and 1
Actual: true

Testing commutative property with -1234 and 10
Actual: true

Testing commutative property with -1234 and 6789
Actual: true

Testing commutative property with 1 and 0
Actual: true

Testing commutative property with 1 and -1
Actual: true

Testing commutative property with 1 and -10
Actual: true

Testing commutative property with 1 and -1234
Actual: true
```

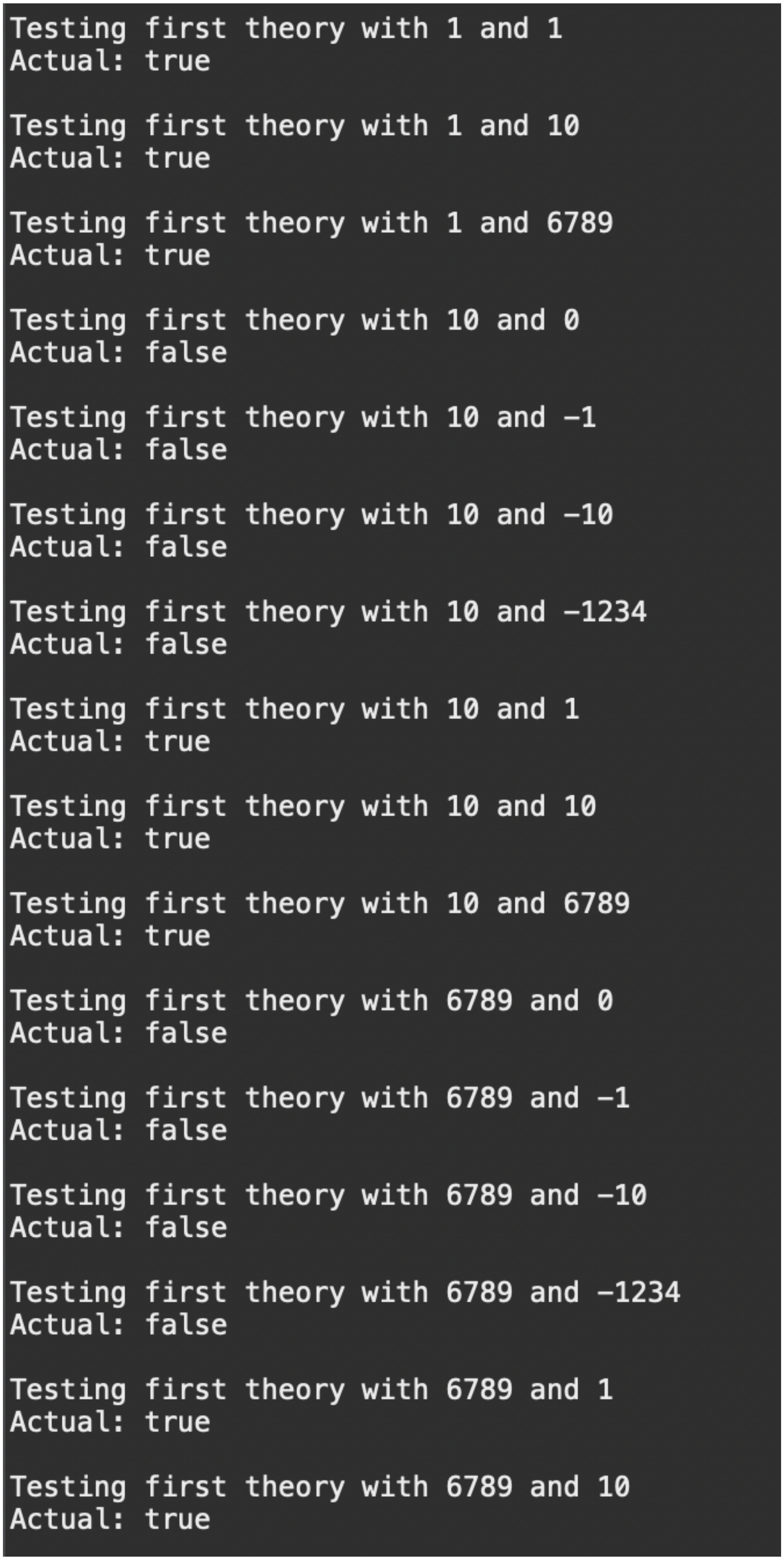
The firstTheory at first was producing negative results and not passing the JUnit test until an assume(True) and assert(True) was inserted and produced and can be witnessed as follows:



```
Testing first theory with 0 and 0
Actual: false

Testing first theory with 0 and -1
Actual: false

Testing first theory with 0 and -10
Actual: false

Testing first theory with 0 and -1234
Actual: false

Testing first theory with 0 and 1
Actual: false

Testing first theory with 0 and 10
Actual: false

Testing first theory with 0 and 6789
Actual: false

Testing first theory with -1 and 0
Actual: false

Testing first theory with -1 and -1
Actual: false

Testing first theory with -1 and -10
Actual: false

Testing first theory with -1 and -1234
Actual: false

Testing first theory with -1 and 1
Actual: false

Testing first theory with -1 and 10
Actual: false

Testing first theory with -1 and 6789
Actual: false

Testing first theory with -10 and 0
Actual: false

Testing first theory with -10 and -1
Actual: false
```



```
Testing first theory with -10 and -10
Actual: false

Testing first theory with -10 and -1234
Actual: false

Testing first theory with -10 and 1
Actual: false

Testing first theory with -10 and 10
Actual: false

Testing first theory with -10 and 6789
Actual: false

Testing first theory with -1234 and 0
Actual: false

Testing first theory with -1234 and -1
Actual: false

Testing first theory with -1234 and -10
Actual: false

Testing first theory with -1234 and -1234
Actual: false

Testing first theory with -1234 and 1
Actual: false

Testing first theory with -1234 and 10
Actual: false

Testing first theory with -1234 and 6789
Actual: false

Testing first theory with 1 and 0
Actual: false

Testing first theory with 1 and -1
Actual: false

Testing first theory with 1 and -10
Actual: false

Testing first theory with 1 and -1234
Actual: false
```



```
Testing first theory with 1 and 1
Actual: true

Testing first theory with 1 and 10
Actual: true

Testing first theory with 1 and 6789
Actual: true

Testing first theory with 10 and 0
Actual: false

Testing first theory with 10 and -1
Actual: false

Testing first theory with 10 and -10
Actual: false

Testing first theory with 10 and -1234
Actual: false

Testing first theory with 10 and 1
Actual: true

Testing first theory with 10 and 10
Actual: true

Testing first theory with 10 and 6789
Actual: true

Testing first theory with 6789 and 0
Actual: false

Testing first theory with 6789 and -1
Actual: false

Testing first theory with 6789 and -10
Actual: false

Testing first theory with 6789 and -1234
Actual: false

Testing first theory with 6789 and 1
Actual: true

Testing first theory with 6789 and 10
Actual: true
```