JUNIT

Testing

Fernando Guerrero Nuez

Proyecto base

```java
package com.fergn06.eq2g_java_testing;
import static java.lang.StrictMath.sqrt;
import java.util.Arrays;
import java.util.Scanner;

public class Ecu2grado {

    public static void main(String[] args) {

        double a, b, c, x1, x2;
        double[] coeficientes = new double[3];
        pedirCoeficientes(coeficientes);

        a = coeficientes[0];
        b = coeficientes[1];
        c = coeficientes[2];

        double[] raices = new double[2];
        int numSols = resolverEcu(a, b, c, raices);
        x1 = raices[0];
        x2 = raices[1];
        System.out.println(Arrays.toString(raices));
        mostrarSoluciones(a, b, c, numSols, x1, x2);

    }


    public static void pedirCoeficientes(double[] oCoeficientes) {
        Scanner myKeyboard = new Scanner(System.in);
        do {
            System.out.println("Introduce coeficiente a: ");
            oCoeficientes[0] = myKeyboard.nextDouble();
            if (oCoeficientes[0] == 0) {
                System.out.println("El coeficiente a no puede ser 0");
            }
        } while (oCoeficientes[0] == 0);
        System.out.println("Introduce coeficiente b: ");
        oCoeficientes[1] = myKeyboard.nextDouble();
        System.out.println("Introduce coeficiente c: ");
        oCoeficientes[2] = myKeyboard.nextDouble();
    }


    public static int resolverEcu(double iA, double iB, double iC, double[] oRaices) {
        double discriminante, x1, x2;
        int numSols = 0;
```

```
        discriminante = (iB * iB) - (4 * iA * iC);
        if (discriminante < 0) {
            numSols = 0;
        } else if (discriminante == 0) {
            numSols = 1;
            x1 = -iB / (2 * iA);
            oRaices[0] = x1;
        } else {
            numSols = 2;
            x1 = (-iB + sqrt(discriminante)) / (2 * iA);
            x2 = (-iB - sqrt(discriminante)) / (2 * iA);
            oRaices[0] = x1;
            oRaices[1] = x2;
        }
        return numSols;
    }


    public static void mostrarSoluciones(double iA, double iB, double iC, int iNumSols, double
iX1, double iX2) {
        System.out.println("La ecuacion de grado 2 con coeficientes a = " + iA + ",b = " + iB +
" y c = " + iC);
        switch (iNumSols) {
            case 0:
                System.out.println("No tiene solución");
                break;
            case 1:
                System.out.println("Tiene 1 solución: " + iX1);
                break;
            case 2:
                System.out.println("Tiene 2 soluciones: " + iX1 + " y " + iX2);
                break;
        }
    }
    public static int suma(int a, int b){
        return (a + b);
    }
}
```

Código de pruebas

```java
import com.fergn06.eq2g_java_testing.Ecu2grado;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

/**
 *
 * @author fernandoguenue
 */
public class EQ2GJUnitTest {

    public EQ2GJUnitTest() {
    }

//    @org.junit.jupiter.api.BeforeAll
//    public static void setUpClass() throws Exception {
//    }
//
//    @org.junit.jupiter.api.AfterAll
//    public static void tearDownClass() throws Exception {
//    }
//
//    @org.junit.jupiter.api.BeforeEach
//    public void setUp() throws Exception {
//    }
//
//    @org.junit.jupiter.api.AfterEach
//    public void tearDown() throws Exception {
//    }

    @BeforeAll
    public static void setUpClass() {
    }

    @AfterAll
    public static void tearDownClass() {
    }

    @BeforeEach
    public void setUp() {
    }

    @AfterEach
    public void tearDown() {
```

```
    }
    @Test
    public void testMostrarSoluciones0() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, 2, 5, raices);
        assertEquals(0, numSols, "Debe haber 0 soluciones");
    }

    @Test
    public void testMostrarSoluciones1() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, -2, 1, raices);
        assertEquals(1, numSols, "Debe haber 1 solución");
        assertEquals(1.0, raices[0], "La raíz debe ser 1.0");
    }

    @Test
    public void testMostrarSoluciones2() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, -3, 2, raices);
        assertEquals(2, numSols, "Debe haber 2 soluciones");
        assertEquals(2.0, raices[0], "Raíz 1 incorrecta");
        assertEquals(1.0, raices[1], "Raíz 2 incorrecta");
    }
}
```

Test 0 soluciones

```
    @Test
    public void testMostrarSoluciones0() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, 2, 5, raices);
        assertEquals(0, numSols, "Debe haber 0 soluciones");
    }
```

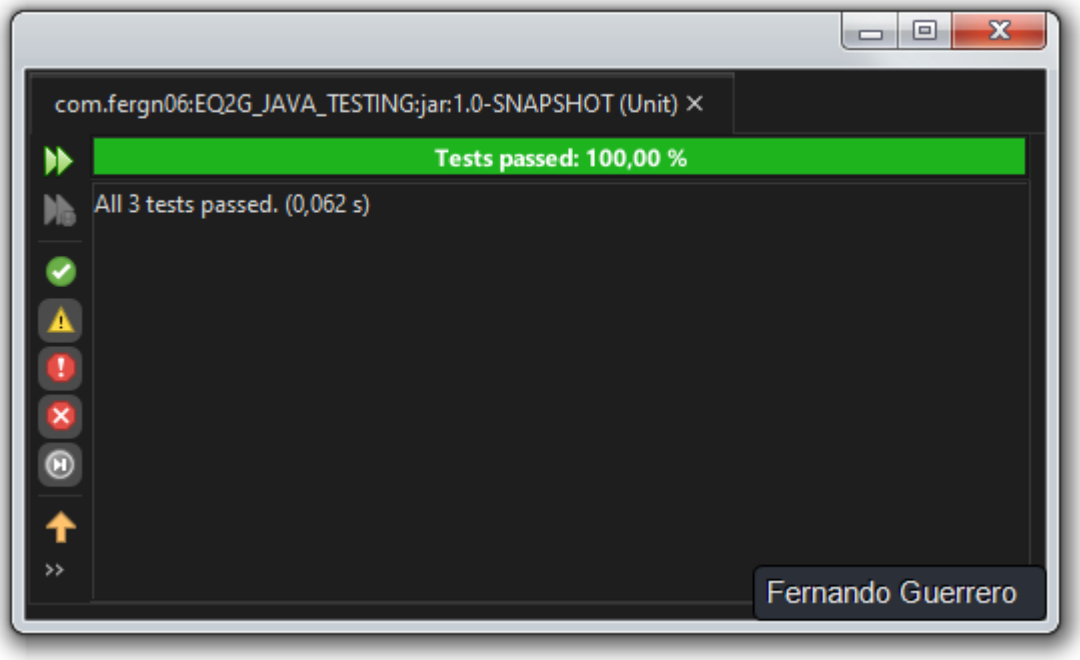
Test 1 solucion

```
    @Test
    public void testMostrarSoluciones1() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, -2, 1, raices);
        assertEquals(1, numSols, "Debe haber 1 solución");
        assertEquals(1.0, raices[0], "La raíz debe ser 1.0");
    }
```

Test 2 soluciones

```
    @Test
    public void testMostrarSoluciones2() {
        double[] raices = new double[2];
        int numSols = Ecu2grado.resolverEcu(1, -3, 2, raices);
        assertEquals(2, numSols, "Debe haber 2 soluciones");
        assertEquals(2.0, raices[0], "Raíz 1 incorrecta");
        assertEquals(1.0, raices[1], "Raíz 2 incorrecta");
    }
```

Resultados


com.fergn06:EQ2G_JAVA_TESTING:jar:1.0-SNAPSHOT (Unit)
Tests passed: 100,00 %
All 3 tests passed. (0,062 s)
Fernando Guerrero