



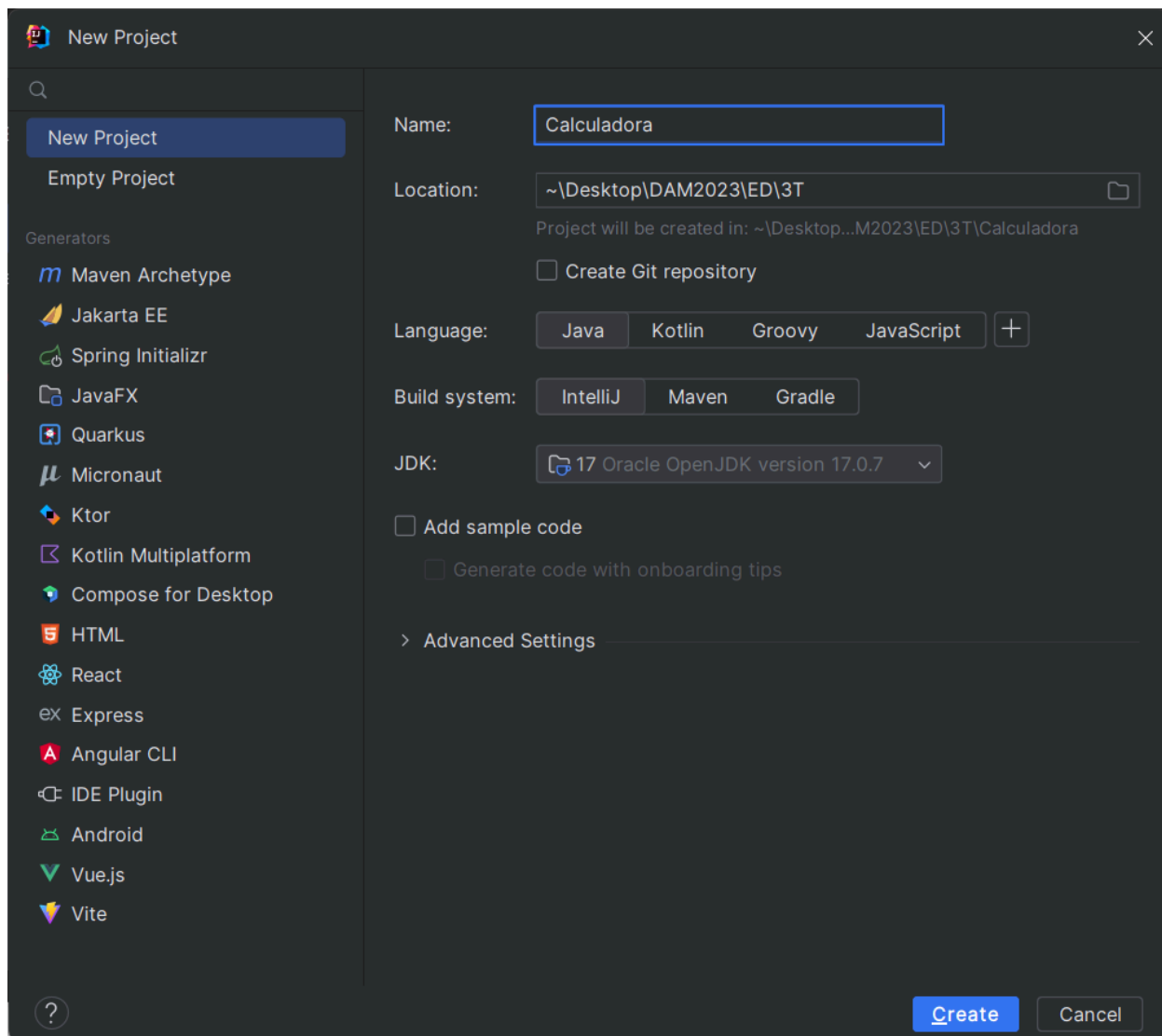
ENTORNOS DE DESARROLLO

PRÁCTICA 18

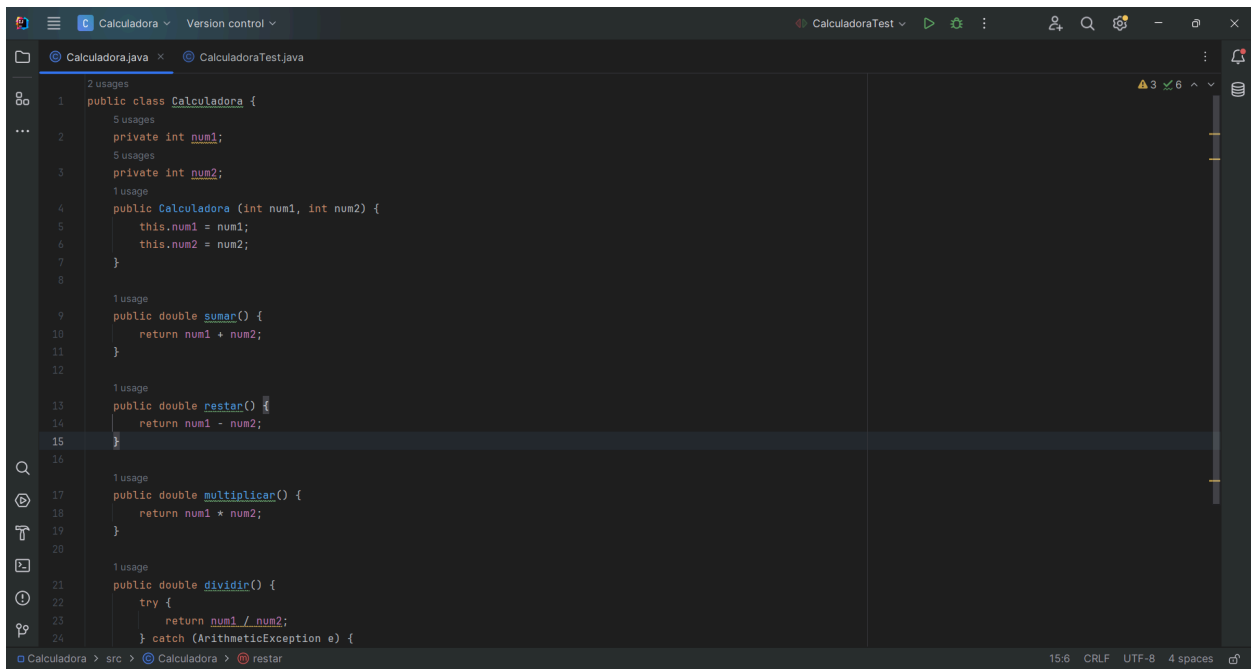
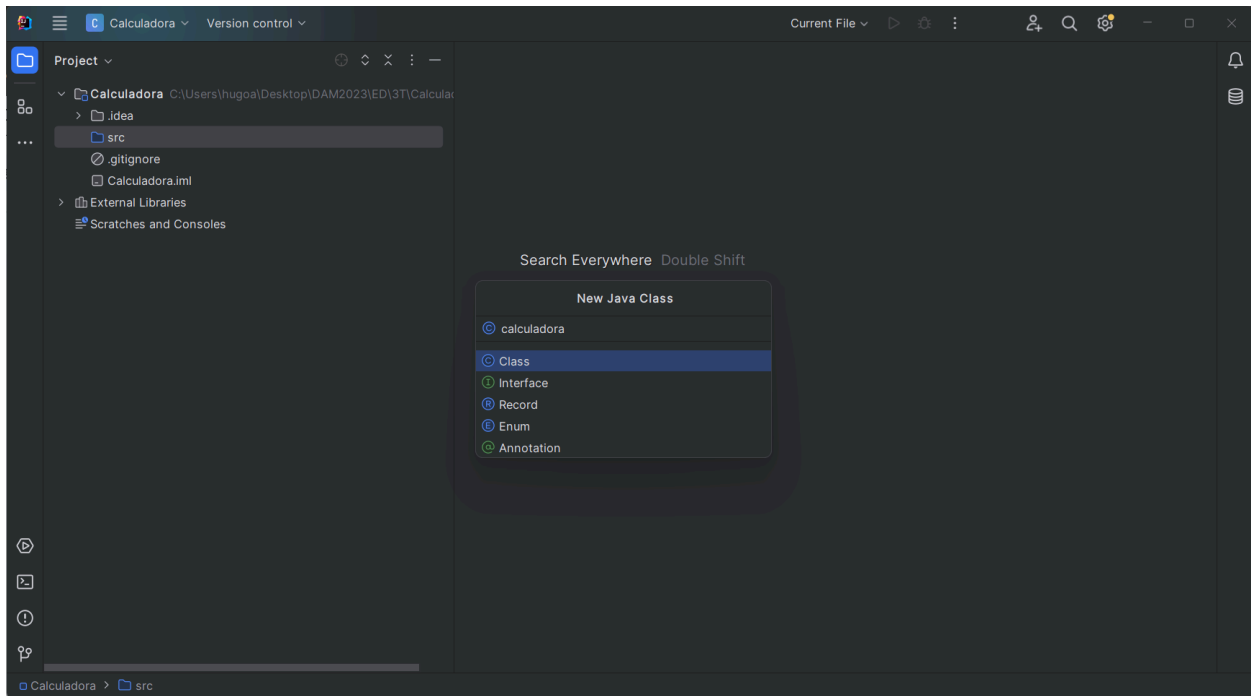


EJERCICIO 1.

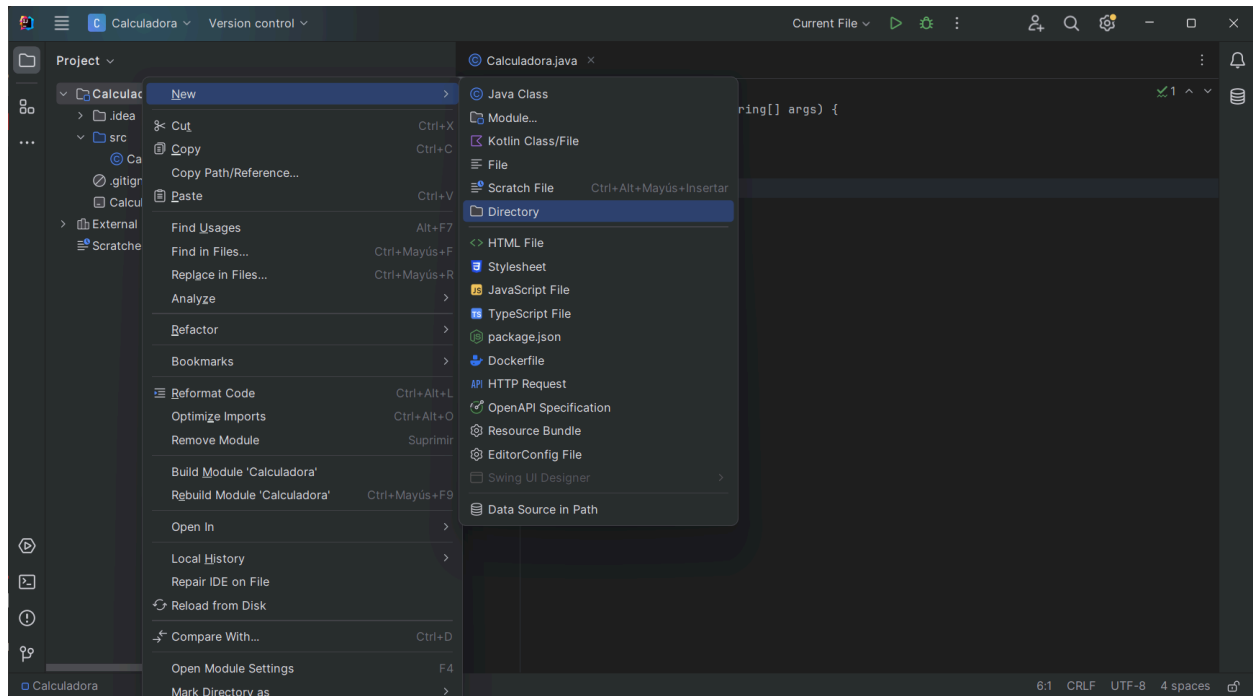
Primero deberemos crear el proyecto.



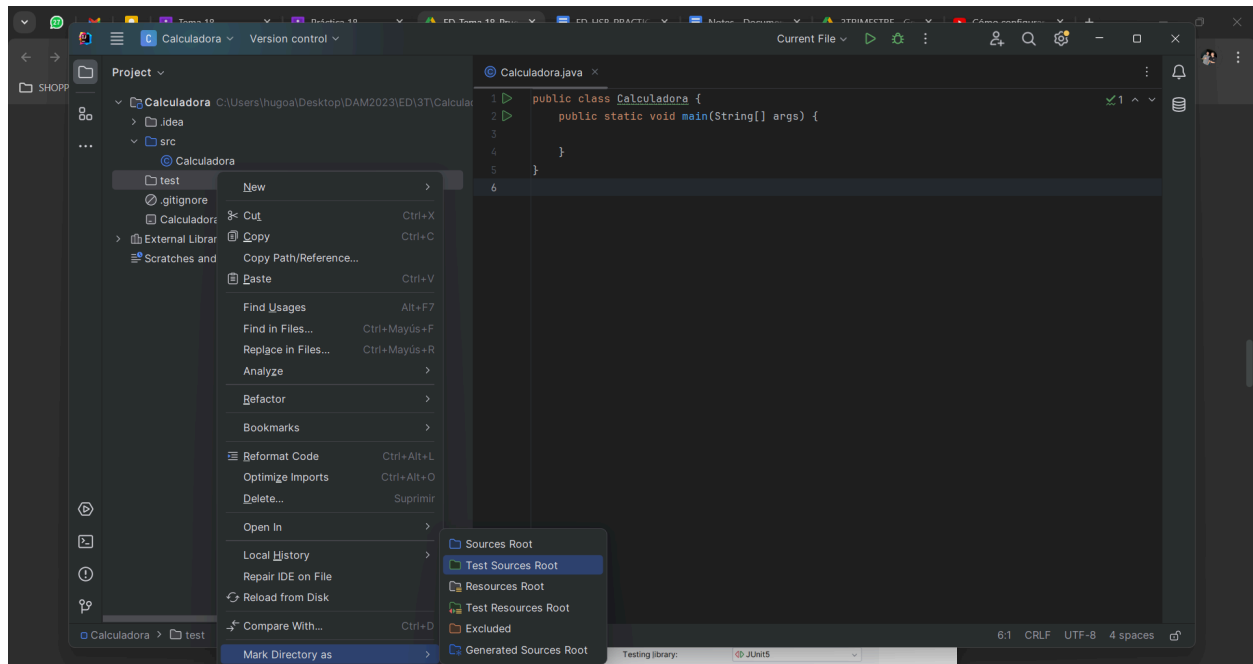
El segundo paso será crear la clase calculadora y rellenarla.



Una vez creada la clase y rellenada, crearemos un directorio donde haremos los tests.

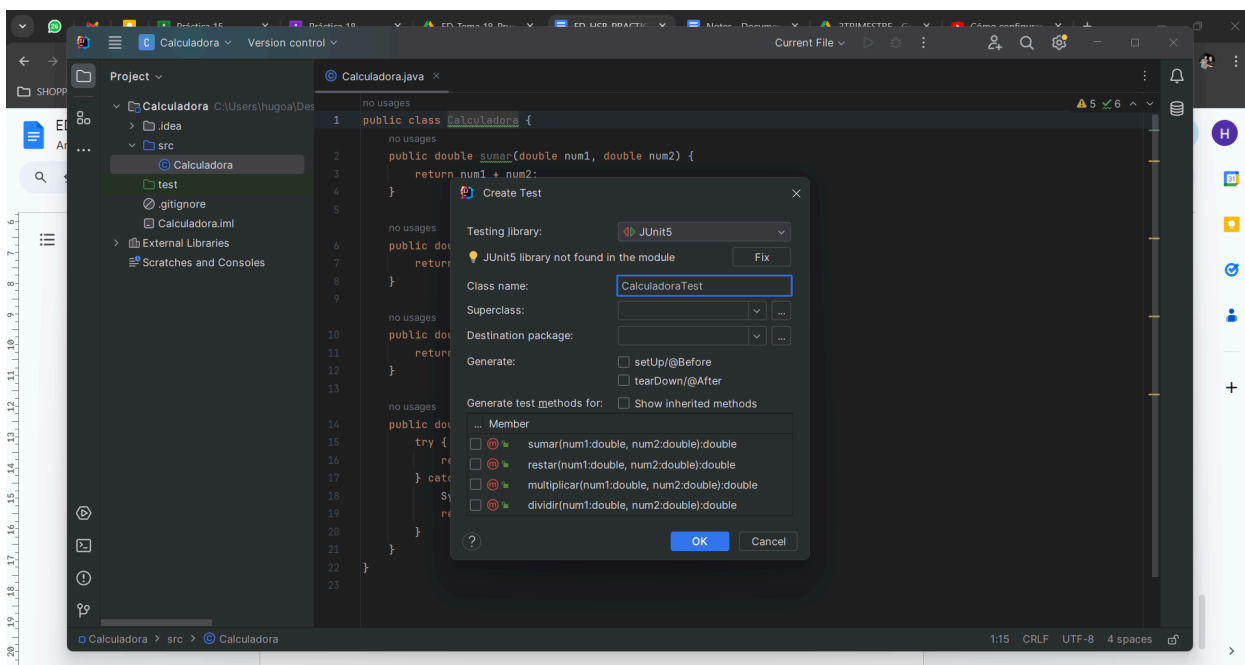
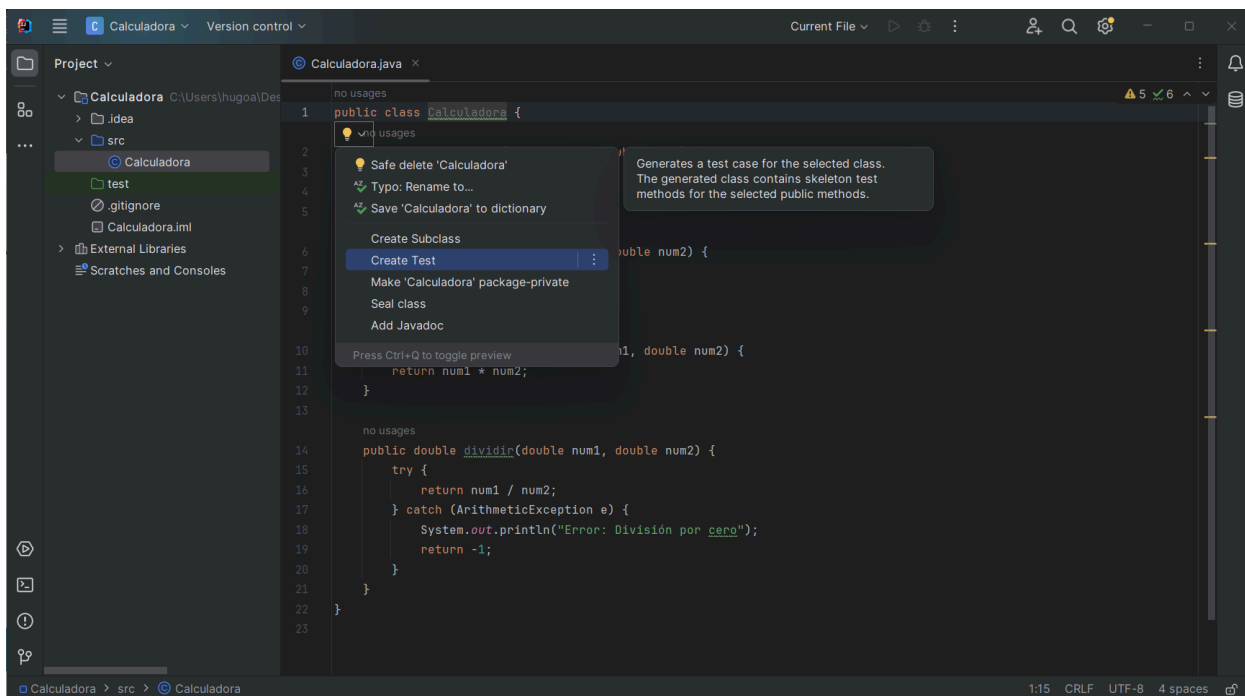


Le indicamos que queremos que sea un directorio para los tests.

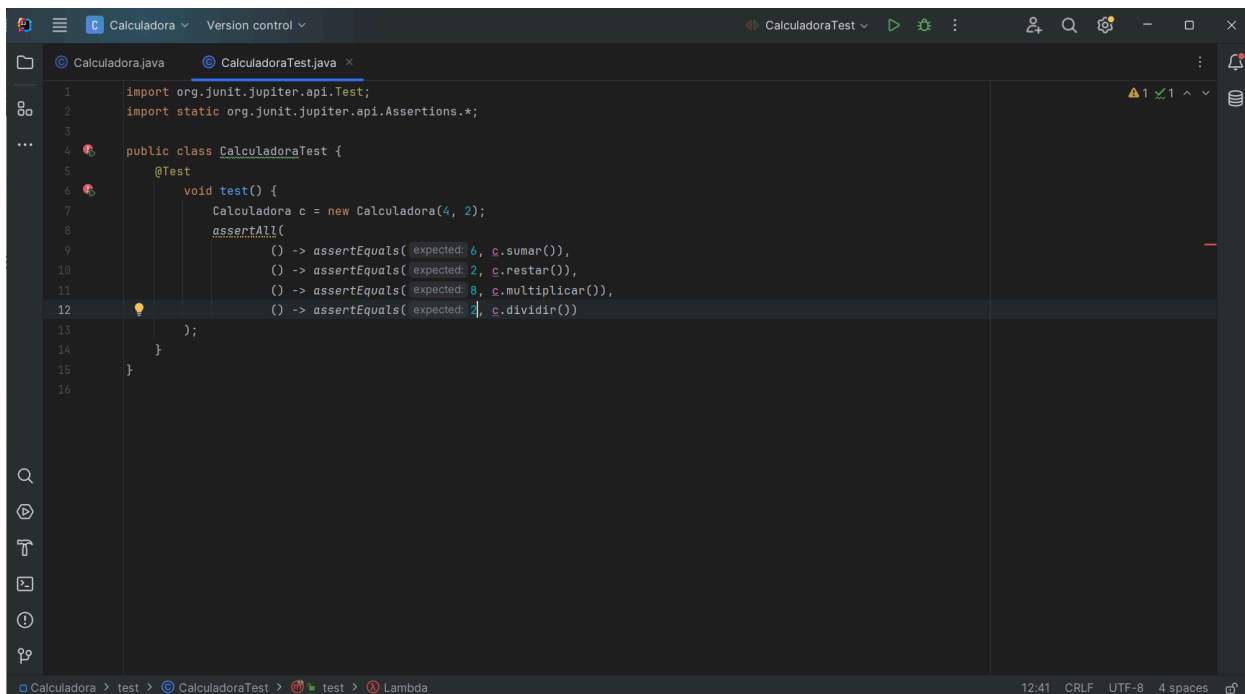


EJERCICIO 2.

Creamos una clase test de la clase Calculadora.



Una vez creada deberemos rellenarla con los tests de las funciones correspondientes de la clase Calculadora.

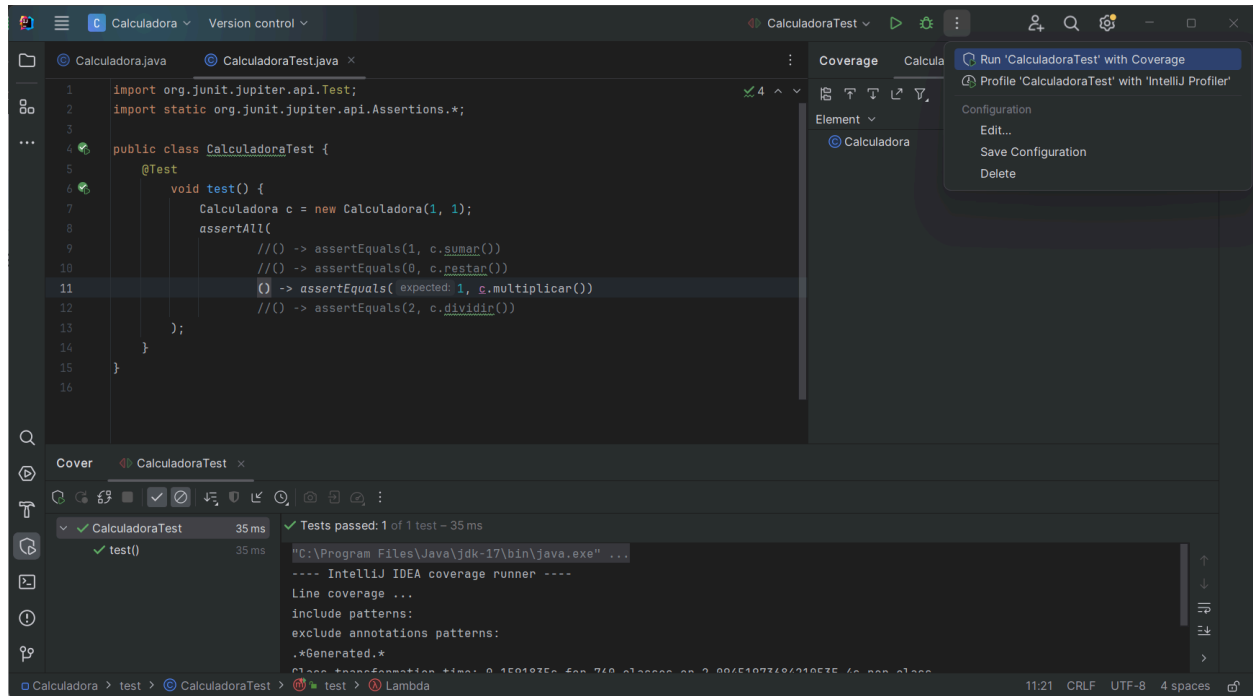


```
1  import org.junit.jupiter.api.Test;
2  import static org.junit.jupiter.api.Assertions.*;
3
4  public class CalculadoraTest {
5      @Test
6      void test() {
7          Calculadora c = new Calculadora(4, 2);
8          assertEquals(6, c.sumar());
9          assertEquals(2, c.restar());
10         assertEquals(8, c.multiplicar());
11         assertEquals(2, c.dividir());
12     }
13 }
14
15
16
```

Calculadora > test > CalculadoraTest > test > Lambda 12:41 CRLF UTF-8 4 spaces

EJERCICIO 3.

Es importante que a la hora de hacer pruebas de nuestro código, ejecutemos el código de la siguiente manera:



Pruebas método sumar

Pruebas correctas:

The screenshot shows the IntelliJ IDEA interface with the `CalculadoraTest.java` file open. The test method `test()` is highlighted, showing the following code:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(4, 2);
8         assertEquals(6, c.sumar());
9     }
10 }
11
12 //() -> assertEquals(2, c.restar());
13 //() -> assertEquals(8, c.multiplicar());
14 //() -> assertEquals(2, c.dividir());
```

The test run results are displayed at the bottom, showing "Tests passed: 1 of 1 test - 37 ms". The coverage window on the right shows the following data:

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)

The screenshot shows the IntelliJ IDEA interface with the `CalculadoraTest.java` file open. The test method `test()` is highlighted, showing the following code:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(1, 2);
8         assertEquals(3, c.sumar());
9     }
10 }
11
12 //() -> assertEquals(6, c.restar());
13 //() -> assertEquals(1, c.multiplicar());
14 //() -> assertEquals(2, c.dividir());
```

The test run results are displayed at the bottom, showing "Tests passed: 1 of 1 test - 36 ms". The coverage window on the right shows the following data:

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)

The screenshot shows the IntelliJ IDEA interface with the `CalculadoraTest.java` file open. The test `test()` has passed successfully. The coverage window shows 100% class coverage and 40% method coverage.

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(6, 3);
8         assertEquals(9, c.sumar());
9         //() -> assertEquals(6, c.restar());
10        //() -> assertEquals(1, c.multiplicar());
11        //() -> assertEquals(2, c.dividir());
12    }
13 }
14
15
16
```

Coverage window:

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)

Test results:

```
✓ CalculadoraTest 35 ms
✓ test() 35 ms
Tests passed: 1 of 1 test - 35 ms
```

IntelliJ IDEA coverage runner output:

```
Line coverage ...
include patterns:
exclude annotations patterns:
.*Generated.*
```

Pruebas incorrectas:

The screenshot shows the IntelliJ IDEA interface with the `CalculadoraTest.java` file open. The test `test()` has failed. The coverage window shows 100% class coverage and 40% method coverage.

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(4, 2);
8         assertEquals(5, c.sumar());
9         //() -> assertEquals(2, c.restar());
10        //() -> assertEquals(8, c.multiplicar());
11        //() -> assertEquals(2, c.dividir());
12    }
13 }
14
15
16
```

Coverage window:

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)

Test results:

```
✗ CalculadoraTest 38 ms
✗ test() 36 ms
Tests failed: 1 of 1 test - 36 ms
```

org.opentest4j.AssertionFailedError:

```
Expected :5.0
Actual   :6.0
<Click to see difference>
```

The screenshot shows an IDE window with the following components:

- Editor:** Displays `CalculadoraTest.java`. The code is as follows:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(6, 3);
8         assertEquals(
9             () -> assertEquals(expected: 1, c.sumar())
10             //() -> assertEquals(6, c.restar())
11             //() -> assertEquals(1, c.multiplicar())
12             //() -> assertEquals(2, c.dividir())
13         );
14     }
15 }
16
```
- Coverage:** A table showing coverage for `Calculadora`.

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)
- Run/Debug Console:** Shows the test result.

CalculadoraTest 39 ms

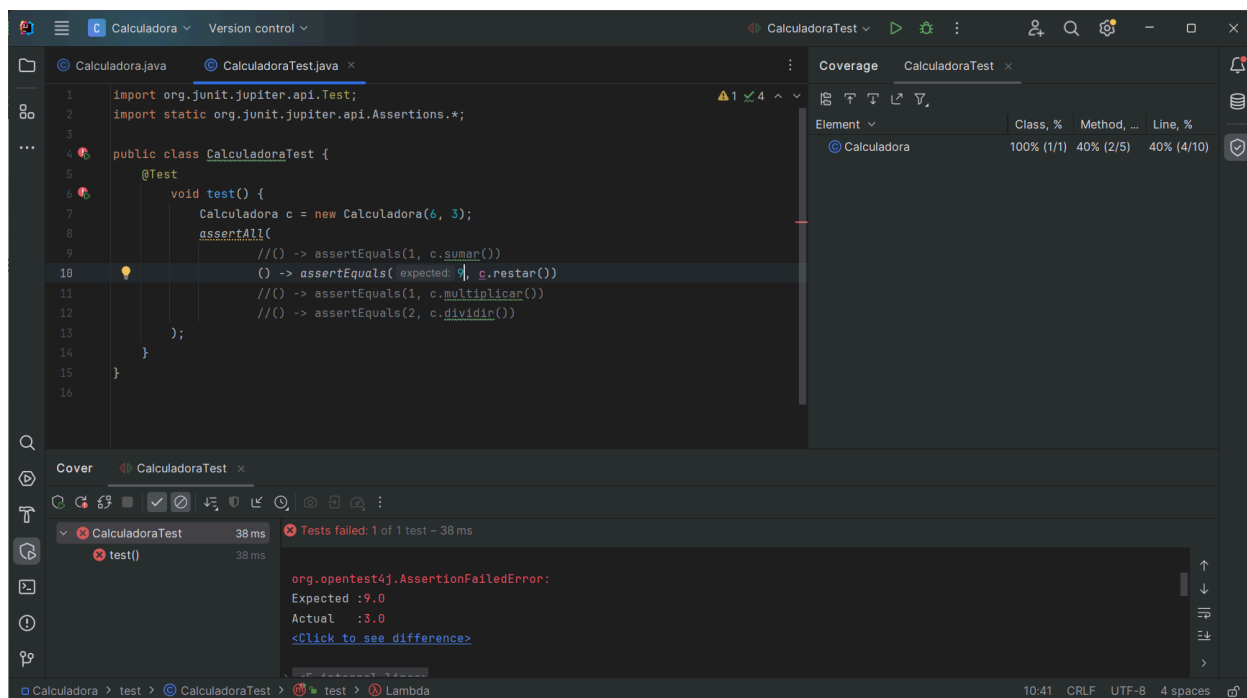
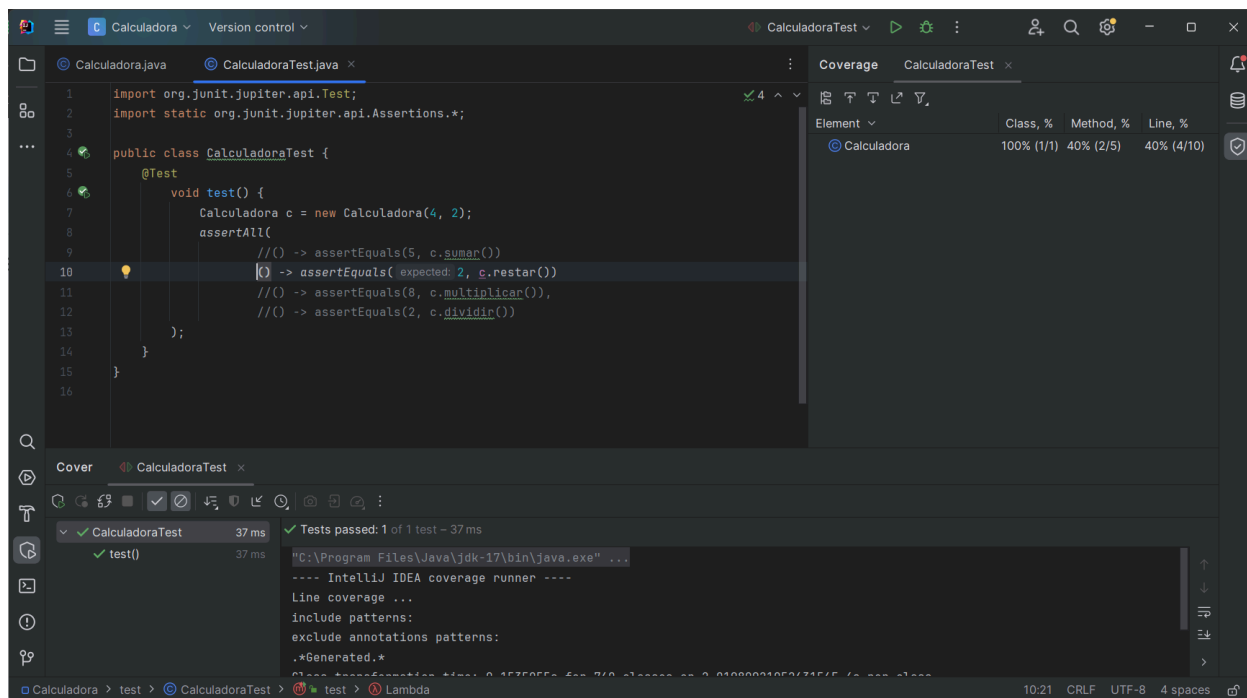
Tests failed: 1 of 1 test - 39 ms

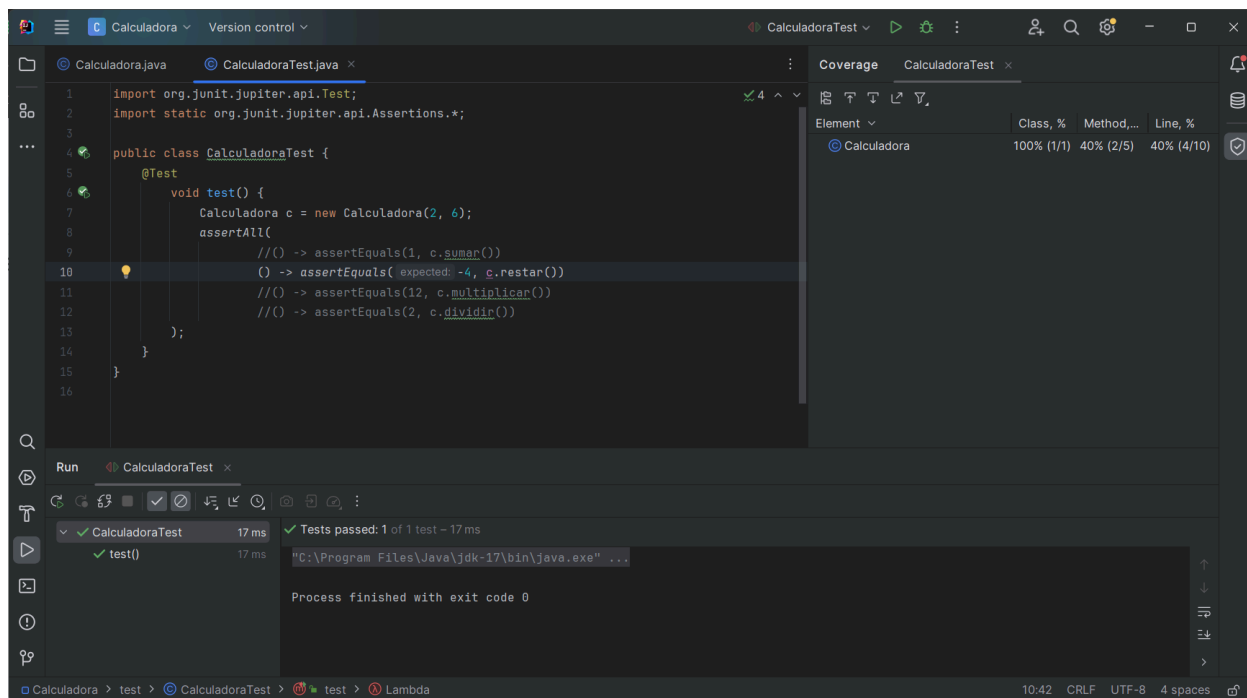
test() 39 ms

org.opentest4j.AssertionFailedError:
Expected :1.0
Actual :9.0
[<Click to see difference>](#)
- Bottom Bar:** Shows the breadcrumb `Calculadora > test > CalculadoraTest > test > Lambda` and the status `9:53 CRLF UTF-8 4 spaces`.

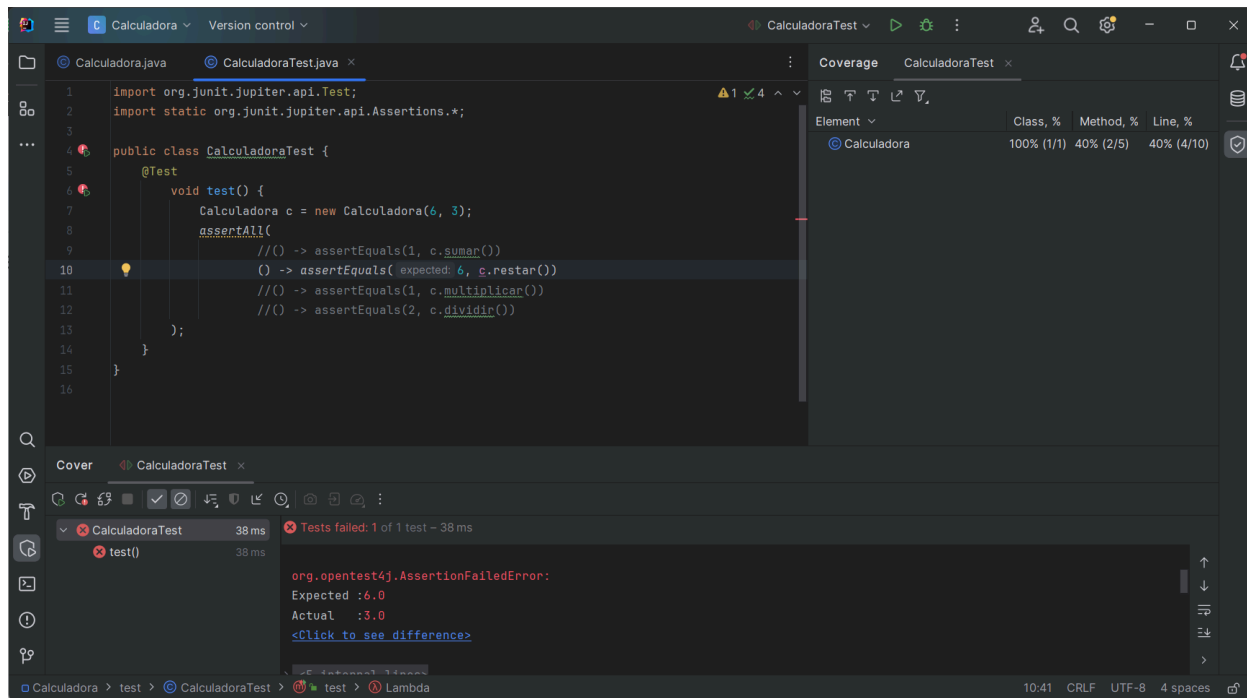
Pruebas método restar

Pruebas correctas:





Pruebas incorrectas:



The screenshot shows an IDE with the following components:

- Editor:** Displays `CalculadoraTest.java` with the following code:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(4, 2);
8         assertEquals(5, c.sumar());
9         //() -> assertEquals(5, c.sumar())
10        () -> assertEquals( expected: 6, c.restar());
11        //() -> assertEquals(8, c.multiplicar());
12        //() -> assertEquals(2, c.dividir());
13    }
14 }
15
16
```

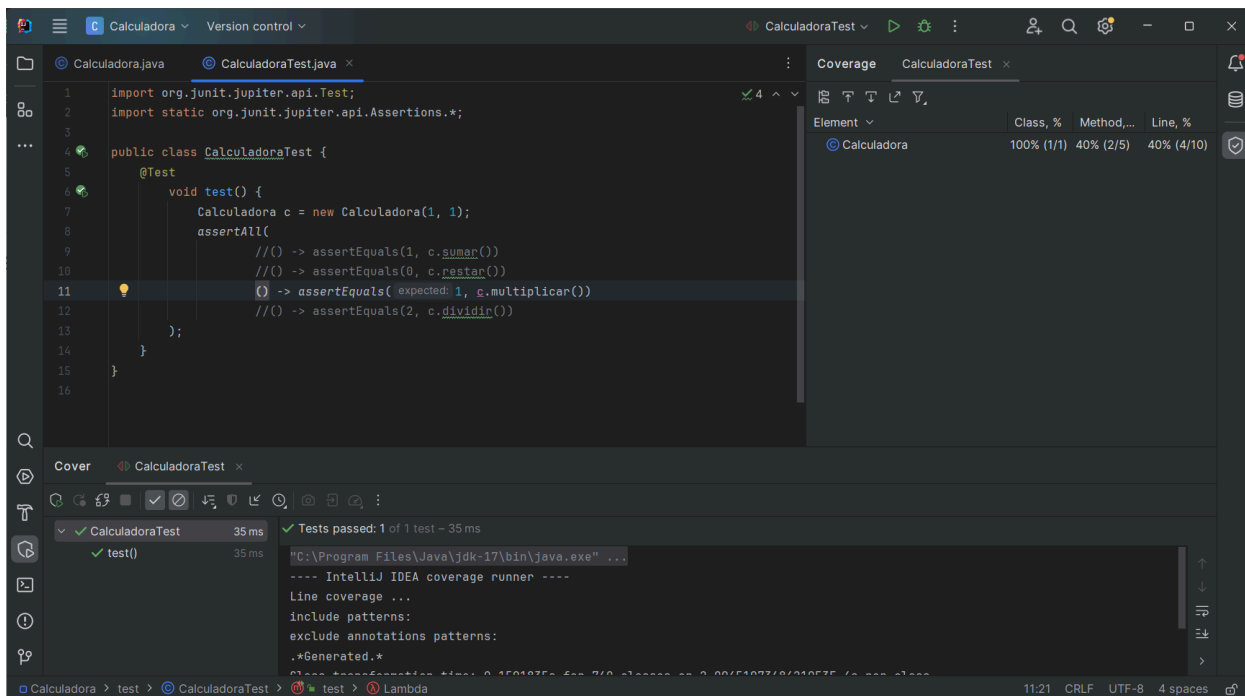
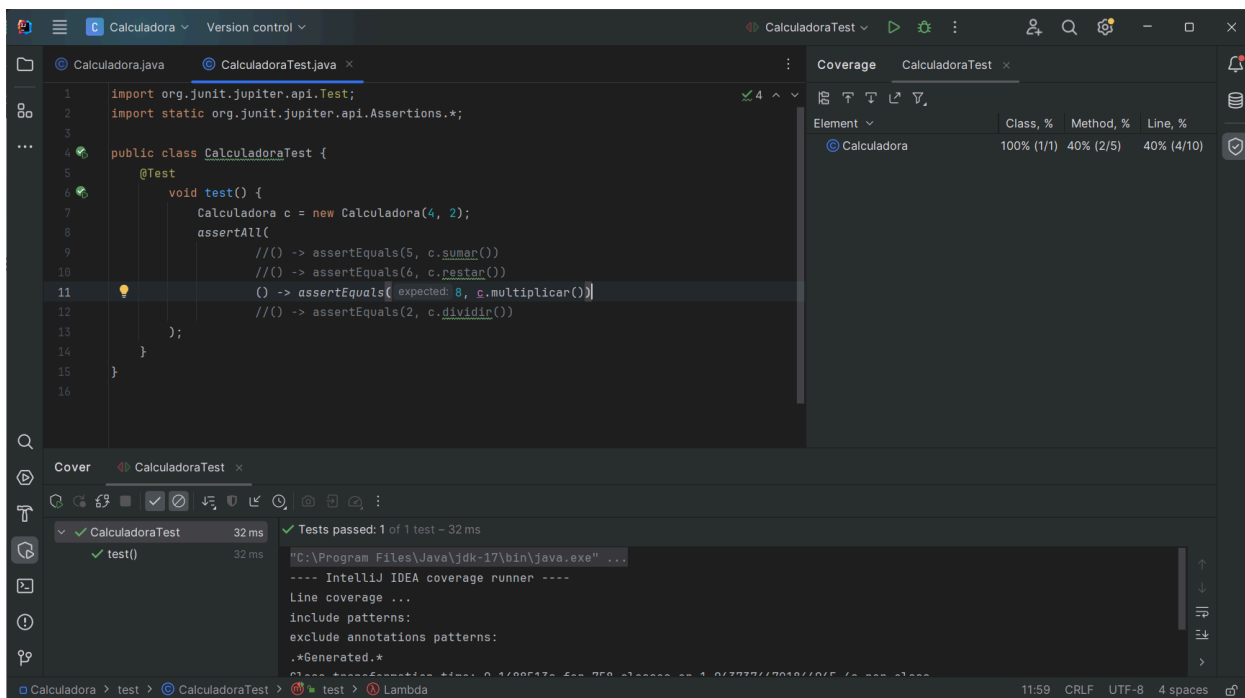
A yellow lightbulb icon is next to line 10, indicating a suggestion or warning.
- Coverage:** A table showing coverage for `Calculadora`:

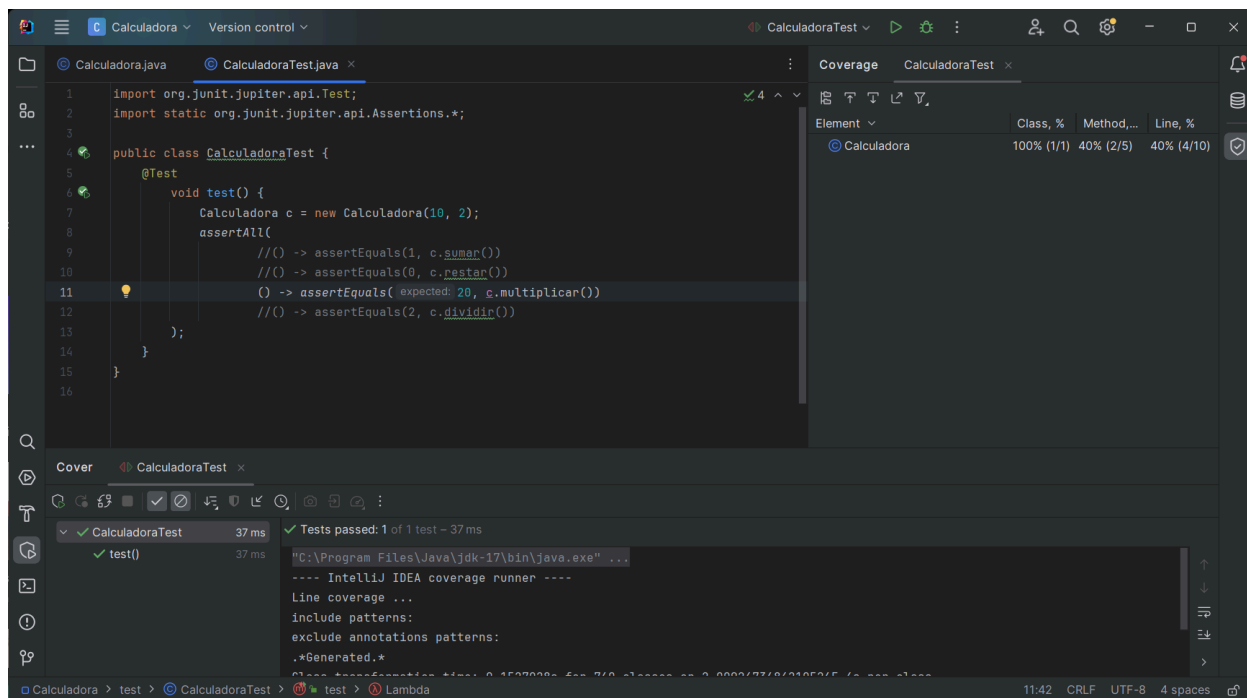
Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)
- Test Results:** Shows a failure for `CalculadoraTest.test()` (40 ms). The error message is:

```
org.opentest4j.AssertionFailedError:
Expected :6.0
Actual   :2.0
<Click to see difference>
```
- Bottom Bar:** Shows the breadcrumb `Calculadora > test > CalculadoraTest > test > Lambda` and the status `10:41 CRLF UTF-8 4 spaces`.

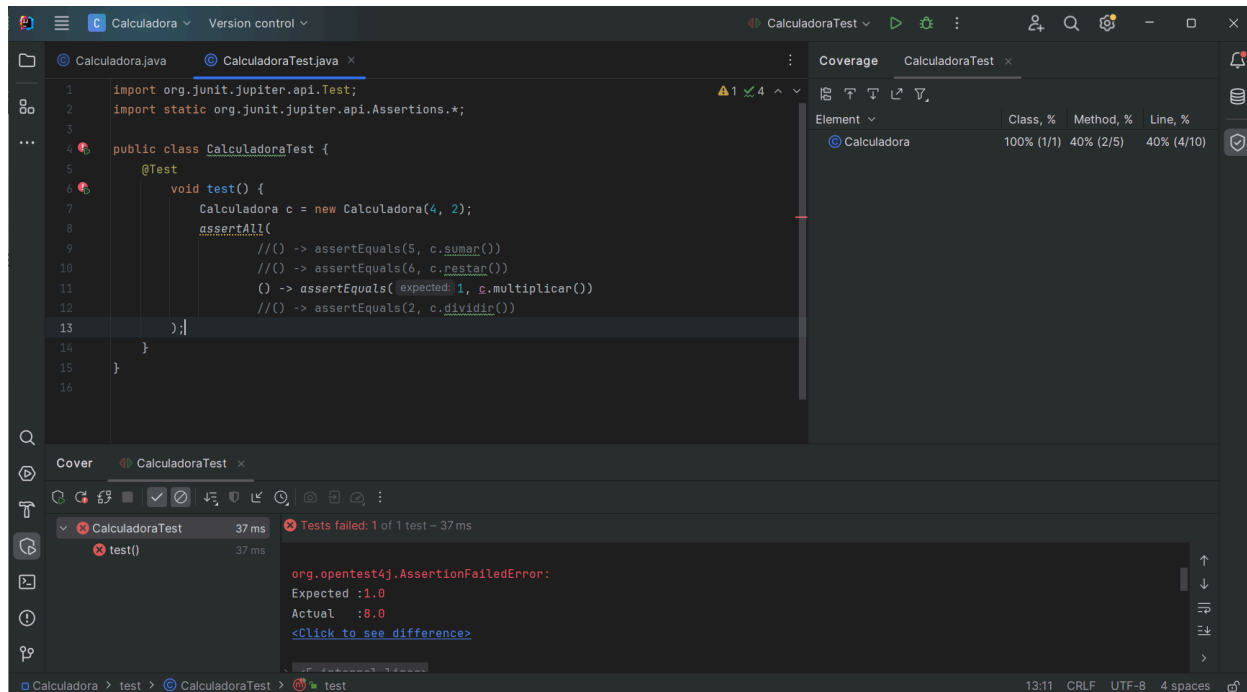
Prueba método multiplicar

Pruebas correctas:





Prueba incorrecta:



The screenshot shows an IDE with the following components:

- Editor:** Displays `CalculadoraTest.java`. The code is as follows:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(8, 2);
8         assertEquals(1, c.sumar());
9         assertEquals(0, c.restar());
10        () -> assertEquals( expected: 12, c.multiplicar())
11
12        //() -> assertEquals(2, c.dividir())
13    }
14 }
15
16
```

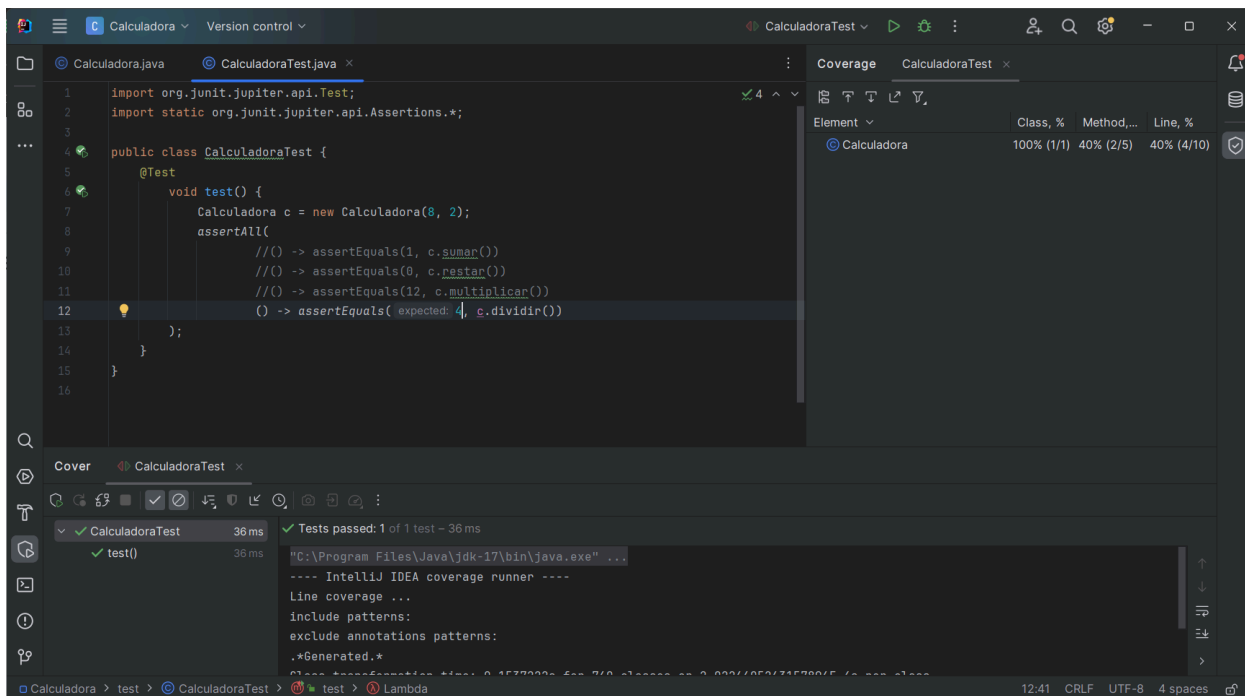
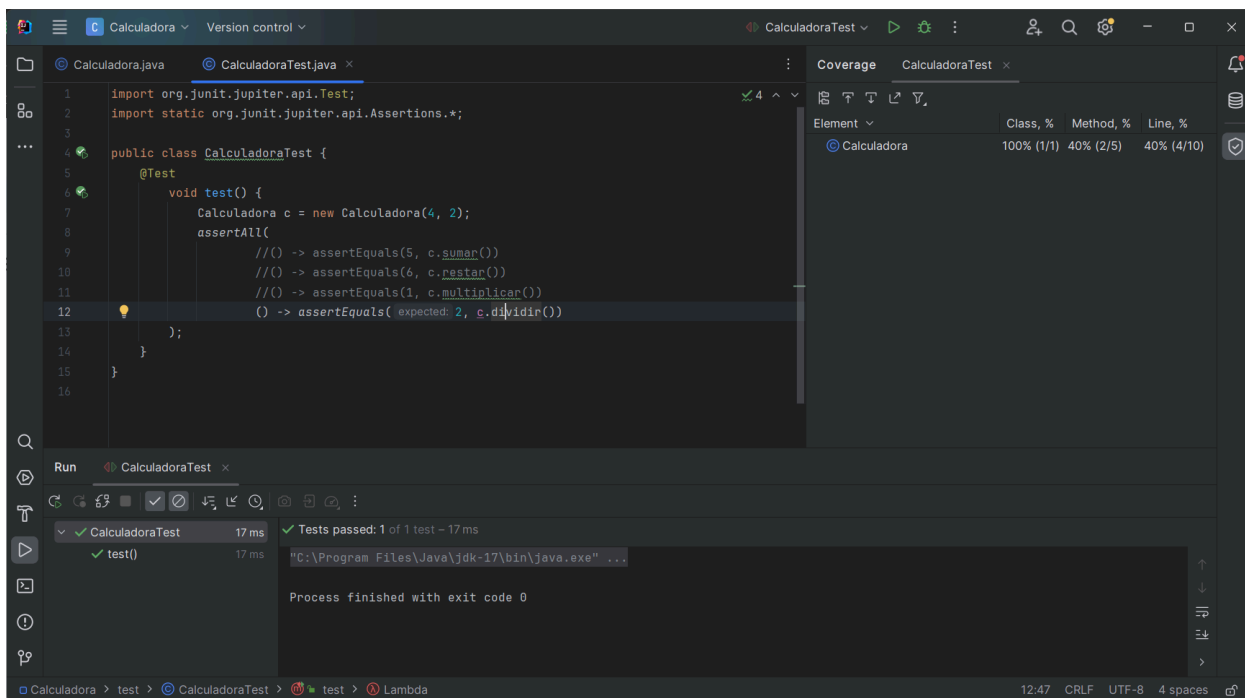
A lightbulb icon is present next to line 11, indicating a suggestion or error.
- Coverage:** A panel on the right showing coverage for `CalculadoraTest`. It includes a table with the following data:

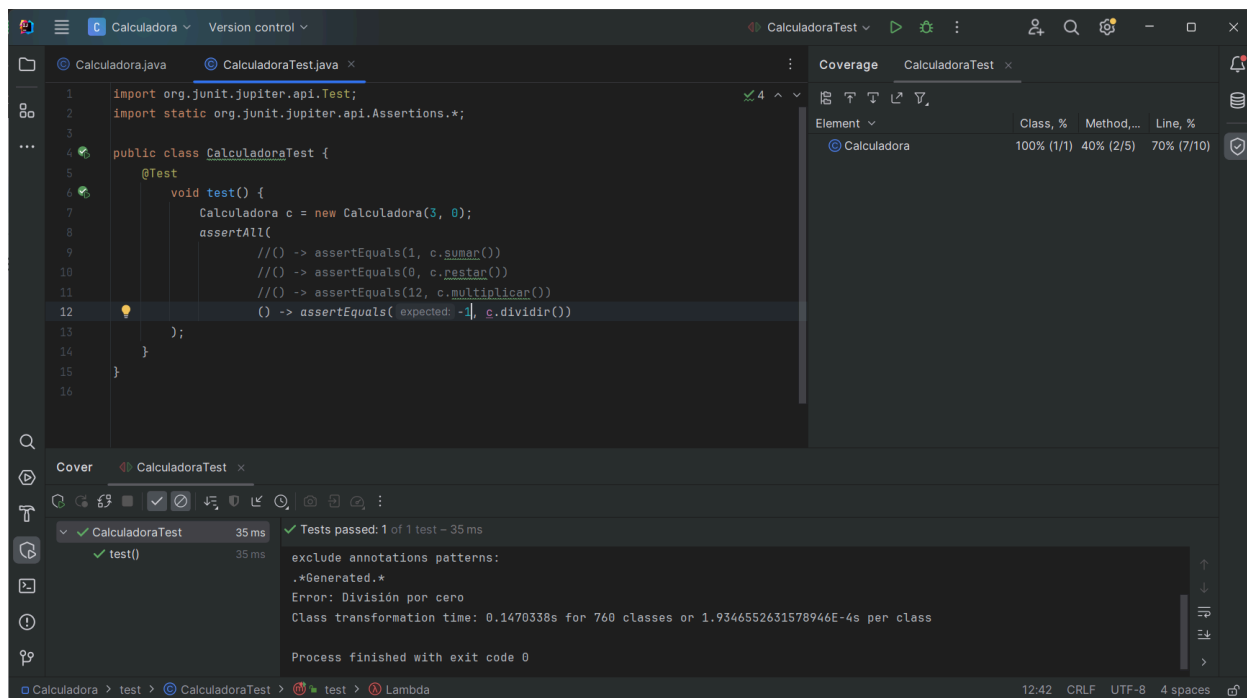
Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	40% (4/10)
- Run and Debug:** A panel at the bottom showing the execution of the `test()` method. It indicates that the test failed with the following error message:

```
org.opentest4j.AssertionFailedError:
Expected :12.0
Actual   :16.0
<Click to see difference>
```

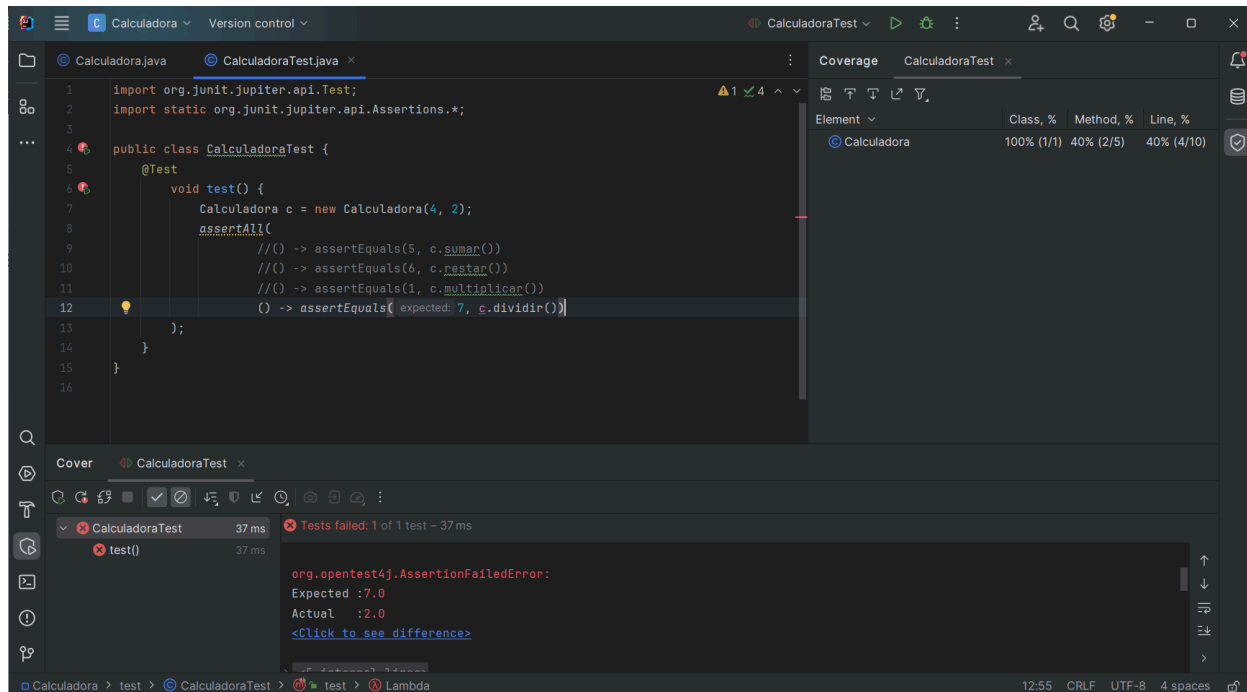

Prueba método dividir

Prueba correcta:





Prueba incorrecta:



The screenshot shows an IDE window with the following components:

- Editor:** Displays `CalculadoraTest.java`. The code is as follows:

```
1 import org.junit.jupiter.api.Test;
2 import static org.junit.jupiter.api.Assertions.*;
3
4 public class CalculadoraTest {
5     @Test
6     void test() {
7         Calculadora c = new Calculadora(7, 0);
8         assertEquals(1, c.sumar());
9         //() -> assertEquals(0, c.restar())
10        //() -> assertEquals(12, c.multiplicar())
11        //() -> assertEquals(expected: 2, c.dividir())
12    }
13 }
14
15
16
```
- Coverage:** A table showing coverage for `Calculadora`.

Element	Class, %	Method, %	Line, %
Calculadora	100% (1/1)	40% (2/5)	70% (7/10)
- Test Results:** Shows a failed test.
 - Test: `CalculadoraTest.test()` (41 ms)
 - Failure: `org.opentest4j.AssertionFailedError: Expected :2.0 Actual :-1.0`
- Bottom Bar:** Shows the file path `Calculadora > test > CalculadoraTest` and the method `test`. The status bar indicates `7:46 CRLF UTF-8 4 spaces`.

CÓDIGO CLASE CALCULADORA

```
public class Calculadora {  
  
    private int num1;  
  
    private int num2;  
  
    public Calculadora (int num1, int num2) {  
  
        this.num1 = num1;  
  
        this.num2 = num2;  
  
    }  
  
  
    public double sumar() {  
  
        return num1 + num2;  
  
    }  
  
  
    public double restar() {  
  
        return num1 - num2;  
  
    }  
  
  
    public double multiplicar() {  
  
        return num1 * num2;  
  
    }  
  
  
    public double dividir() {
```

```
    try {  
        return num1 / num2;  
    } catch (ArithmeticException e) {  
        System.out.println("Error: División por cero");  
        return -1;  
    }  
}  
}
```

CÓDIGO CLASE CALCULADORATEST

```
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.*;

public class CalculadoraTest {

    @Test

    void test() {

        Calculadora c = new Calculadora(7, 0);

        assertAll(

            () -> assertEquals(1, c.sumar())

            () -> assertEquals(0, c.restar())

            () -> assertEquals(12, c.multiplicar())

            () -> assertEquals(2, c.dividir())

        );

    }

}
```
