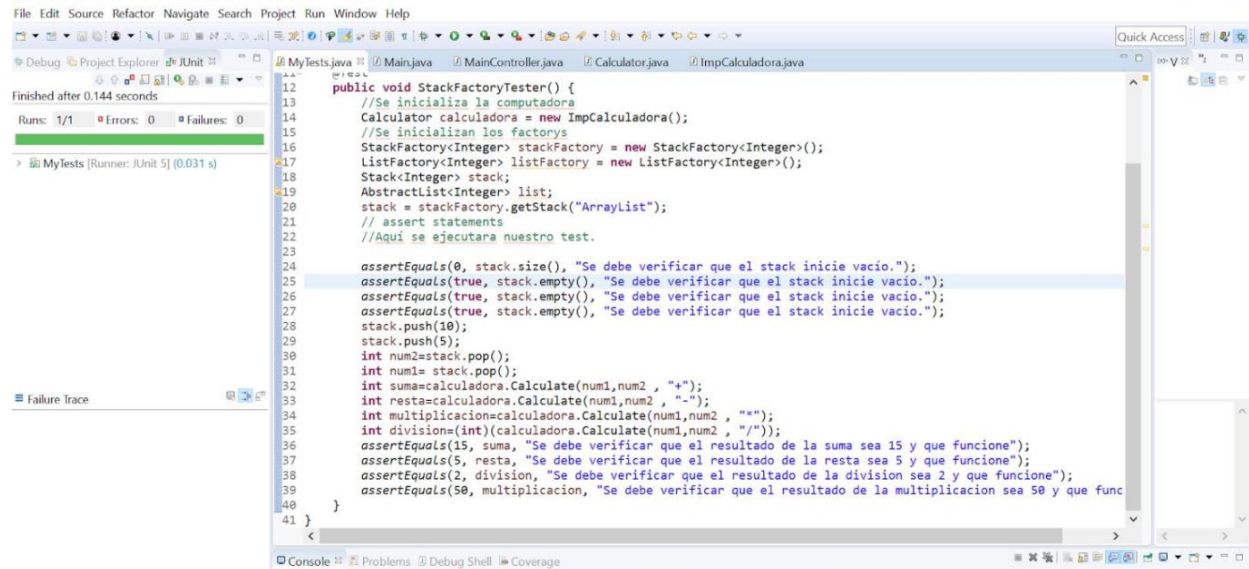


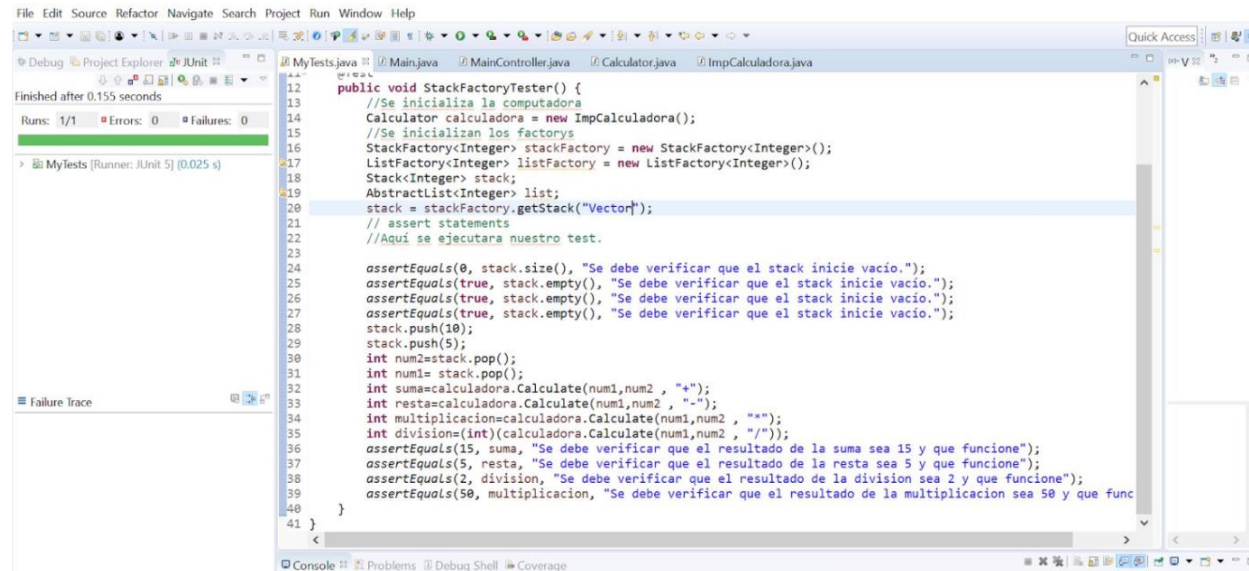
- Error a la hora de dividir...

Daniela Villamar 19086

Luis Rosales



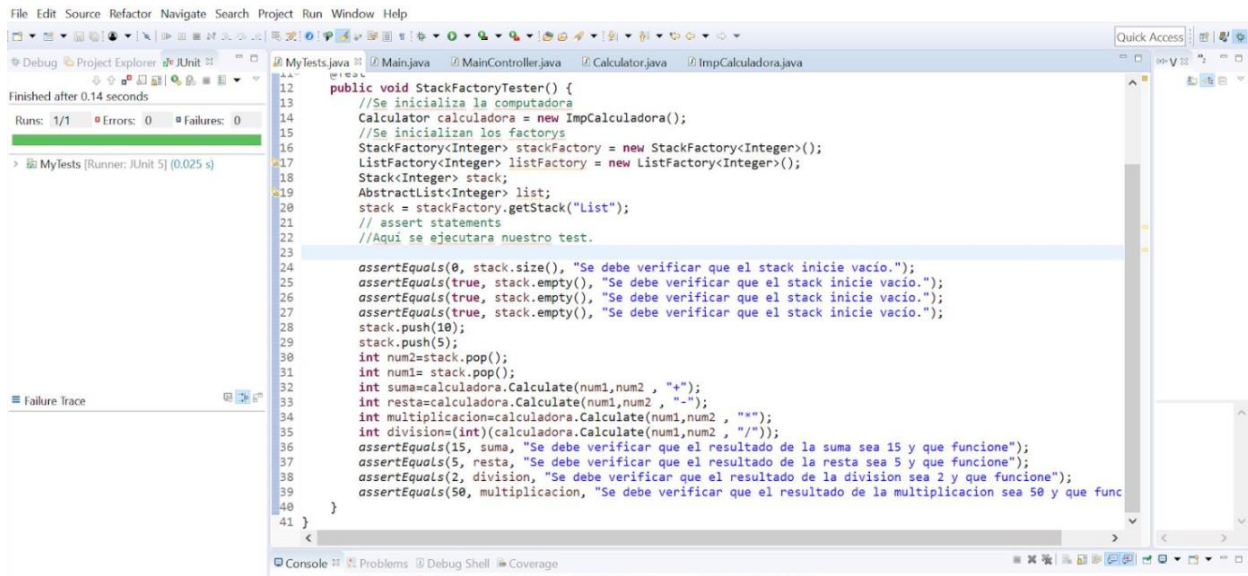
## Prueba Vector



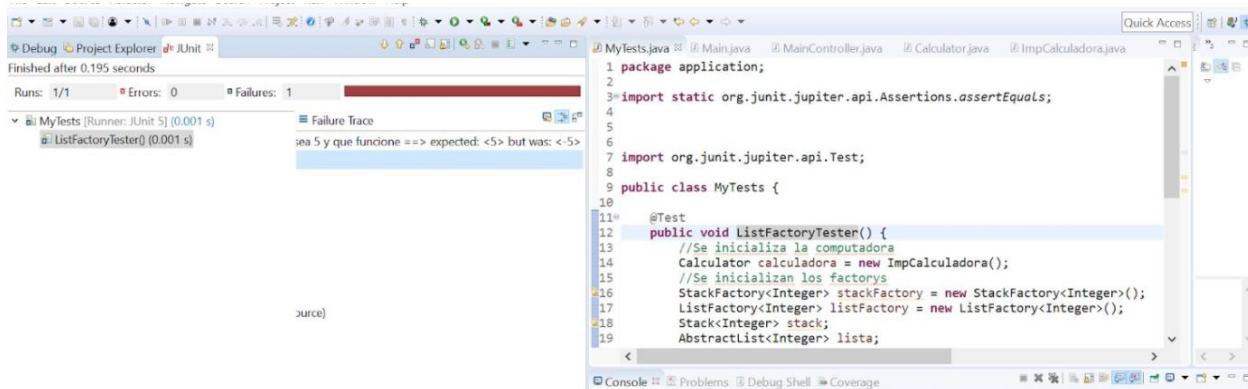
Daniela Villamar 19086

Luis Rosales

## Prueba List



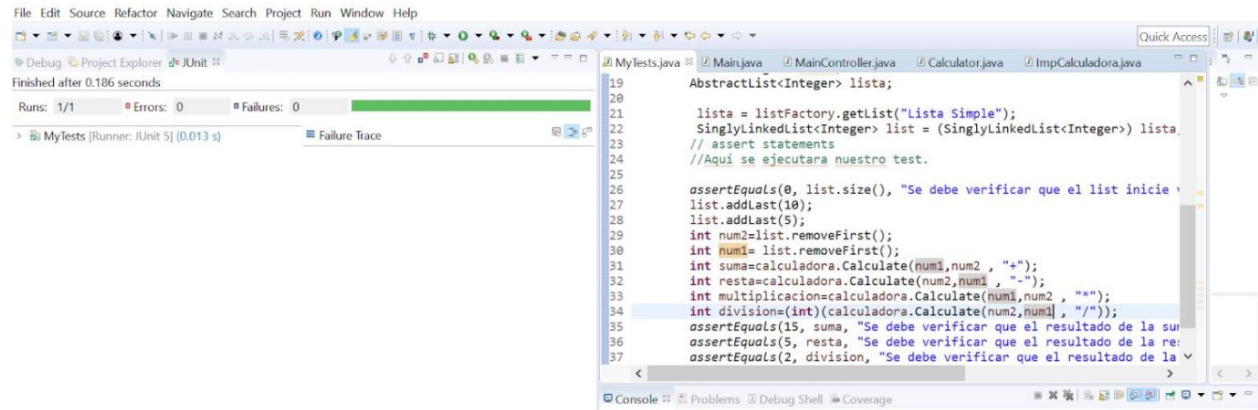
## ListFactory-SingleLinkedTest



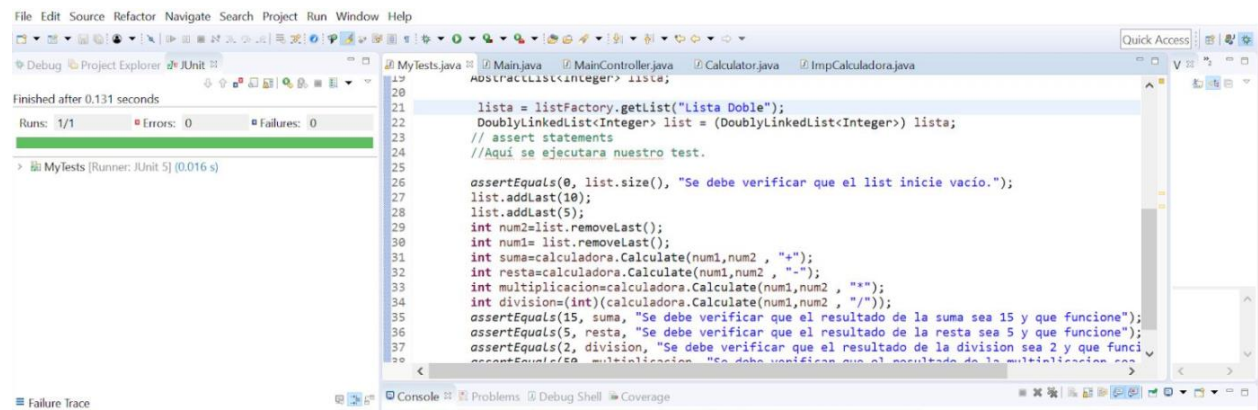
- Error porque saca el primero de la lista y no el ultimo...

Daniela Villamar 19086

Luis Rosales



## DoublyLinkedList

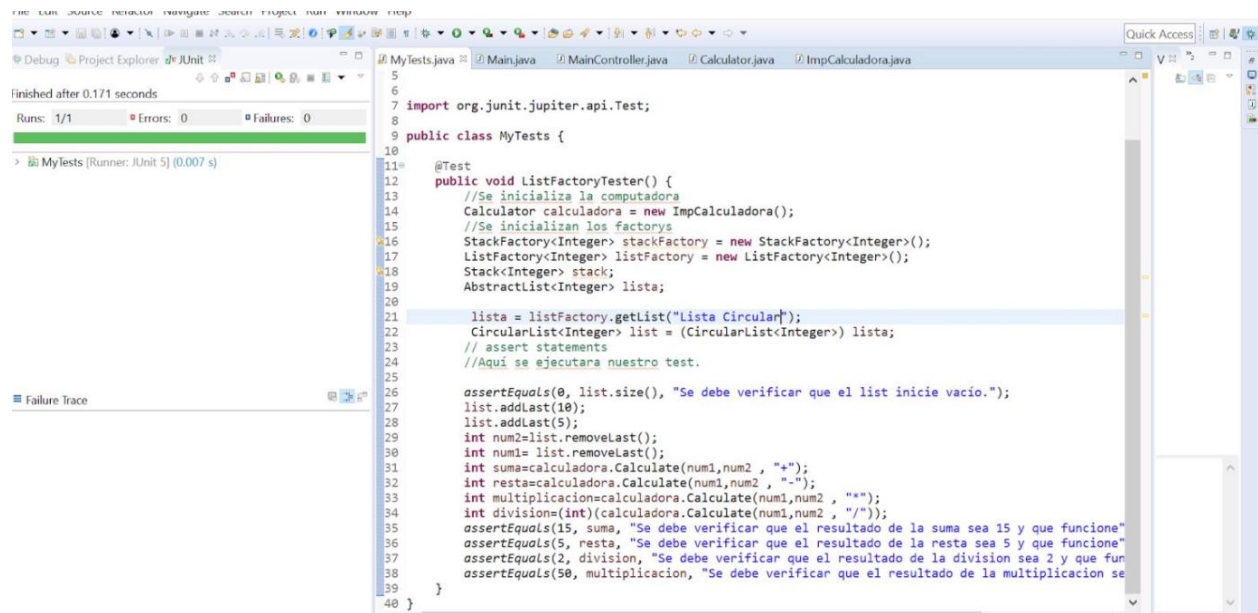




Daniela Villamar 19086

Luis Rosales

## CircularList



The screenshot shows an IDE window with a Java test class `MyTests.java`. The class is annotated with `@Test` and contains a `public void ListFactoryTester()` method. The method initializes a `Calculator` and a `ListFactory`, then creates a `CircularList` and performs various operations and assertions. The IDE also shows a `Failure Trace` panel at the bottom left, indicating that the test failed after 0.171 seconds.

```
5
6
7 import org.junit.jupiter.api.Test;
8
9 public class MyTests {
10
11     @Test
12     public void ListFactoryTester() {
13         //Se inicializa la computadora
14         Calculator calculadora = new ImpCalculadora();
15         //Se inicializan los factories
16         StackFactory<Integer> stackFactory = new StackFactory<Integer>();
17         ListFactory<Integer> listFactory = new ListFactory<Integer>();
18         Stack<Integer> stack;
19         AbstractList<Integer> lista;
20
21         lista = listFactory.getList("Lista Circular");
22         CircularList<Integer> list = (CircularList<Integer>) lista;
23         // assert statements
24         //Aqui se ejecutara nuestro test.
25
26         assertEquals(0, list.size(), "Se debe verificar que el list inicie vacio.");
27         list.addLast(10);
28         list.addLast(5);
29         int num2=list.removeLast();
30         int num1= list.removeLast();
31         int suma=calculadora.Calculate(num1,num2 , "+");
32         int resta=calculadora.Calculate(num1,num2 , "-");
33         int multiplicacion=calculadora.Calculate(num1,num2 , "*");
34         int division=(int)(calculadora.Calculate(num1,num2 , "/"));
35         assertEquals(15, suma, "Se debe verificar que el resultado de la suma sea 15 y que funcione");
36         assertEquals(5, resta, "Se debe verificar que el resultado de la resta sea 5 y que funcione");
37         assertEquals(2, division, "Se debe verificar que el resultado de la division sea 2 y que funcione");
38         assertEquals(50, multiplicacion, "Se debe verificar que el resultado de la multiplicacion sea 50");
39     }
40 }
```

Failure Trace