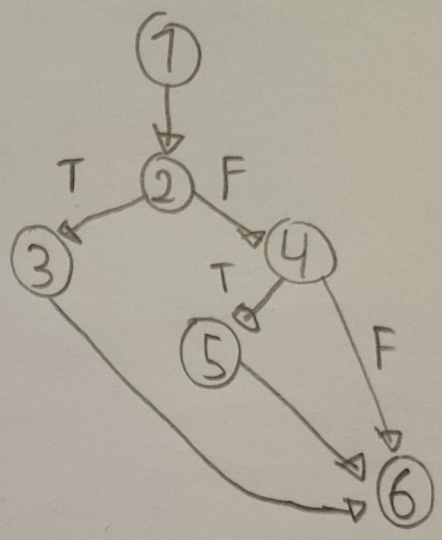
Exercicio 1

```
public int comprobarPositivo(int num) {
1  int res = 0;
2-  if (num < 0) {
3-    res = -1;
4-  } else if (num > 0) {
5     res = 1;
    }
6   return res;
}
```



$v(G) = A - N + 2 = 7 - 6 + 2 = 3$

Camiños:

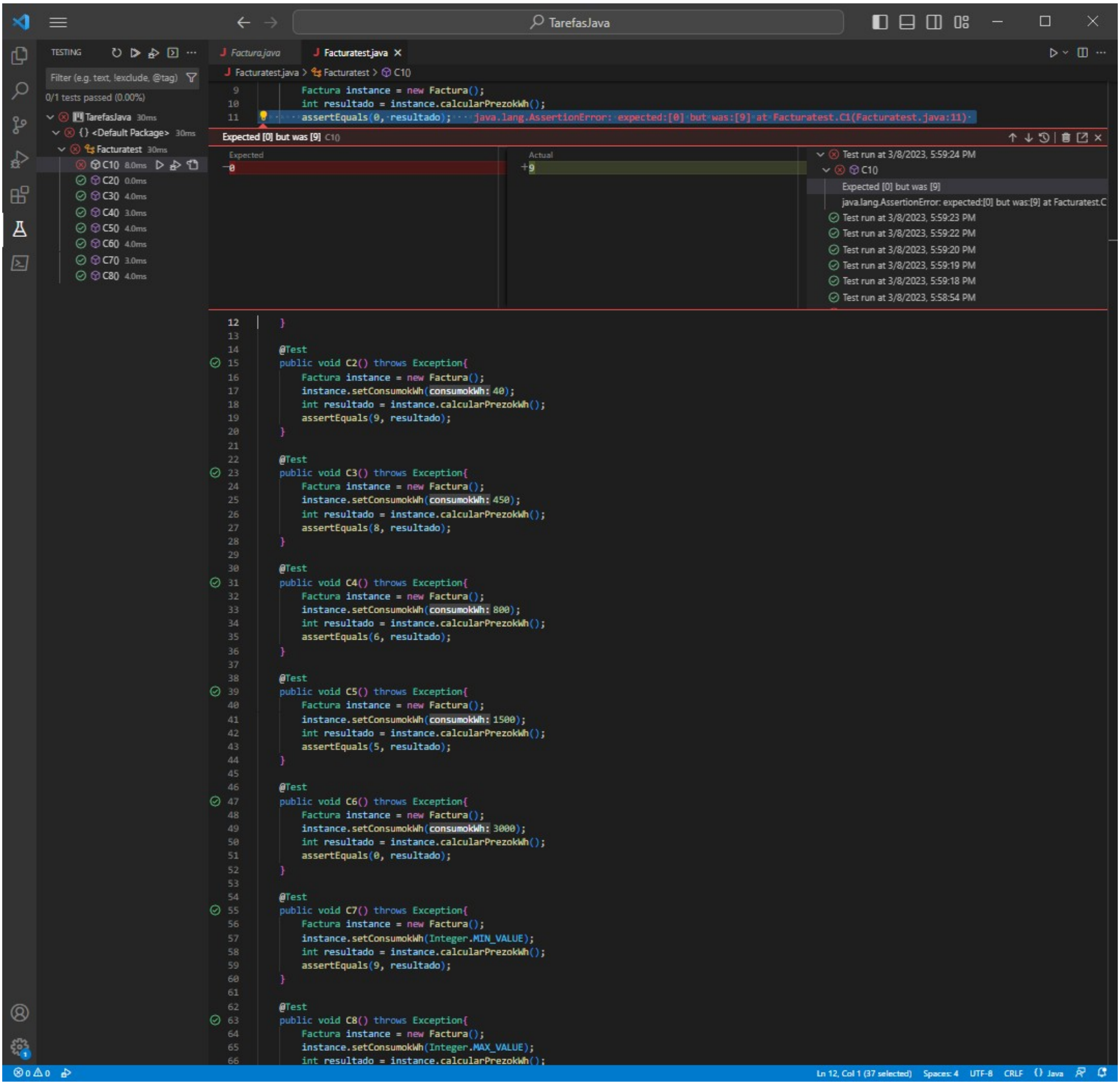
C1: 1-2-3-6

C2: 1-2-4-5-6

C3: 1-2-4-6

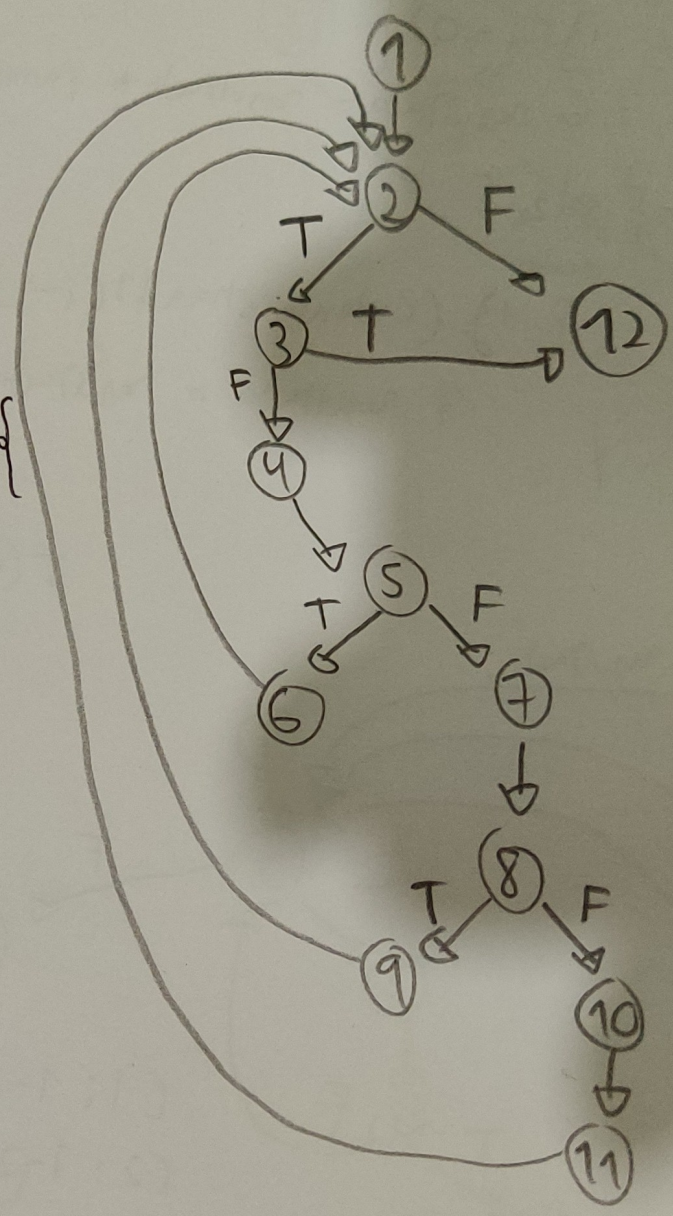
Exercicio 2

Identificador Caso de proba	Entrada	Valores límite	Resultado Esperado	Resultado Obtido
C1	null	null	0	9
C2	40	integer.minValue – 300	9	9
C3	450	301 – 600	8	8
C4	800	601 – 1000	6	6
C5	1500	1001 – 2000	5	5
C6	3000	3000 – Integer.maxvalue	0	0
C7	Integer.minValue	-	9	9
C8	Integer.maxValue	-	0	0
No exercicio pon que se atopamos algún error o corrixamos. Diría que hai 2 errores, o primeiro sendo que acepta números negativos e o Segundo que debería ser "else if (consumokWh>2000) en vez de <=				



Exercicio3

```
public boolean busca (char c, char[] V) {
1|  int a, z, m;
    a = 0;
    z = V.length - 1;
    boolean resultado = false;
2   3
While (a <= z && resultado == false) {
4|  m = (a + z) / 2;
5|  if (V[m] == c) {
6|      resultado = true;
    }
7   else {
8       if (V[m] < c) {
9           a = m + 1;
        }
10      else {
11          z = m - 1;
        }
    }
12  return resultado;
}
```



V(G) = 15 + 2 - 12 = 5

C1: 1 - 2 - 12

C2: 1 - 2 - 3 - 12

C3: 1 - 2 - 3 - 4 - 5 - 6 - 2 - 12

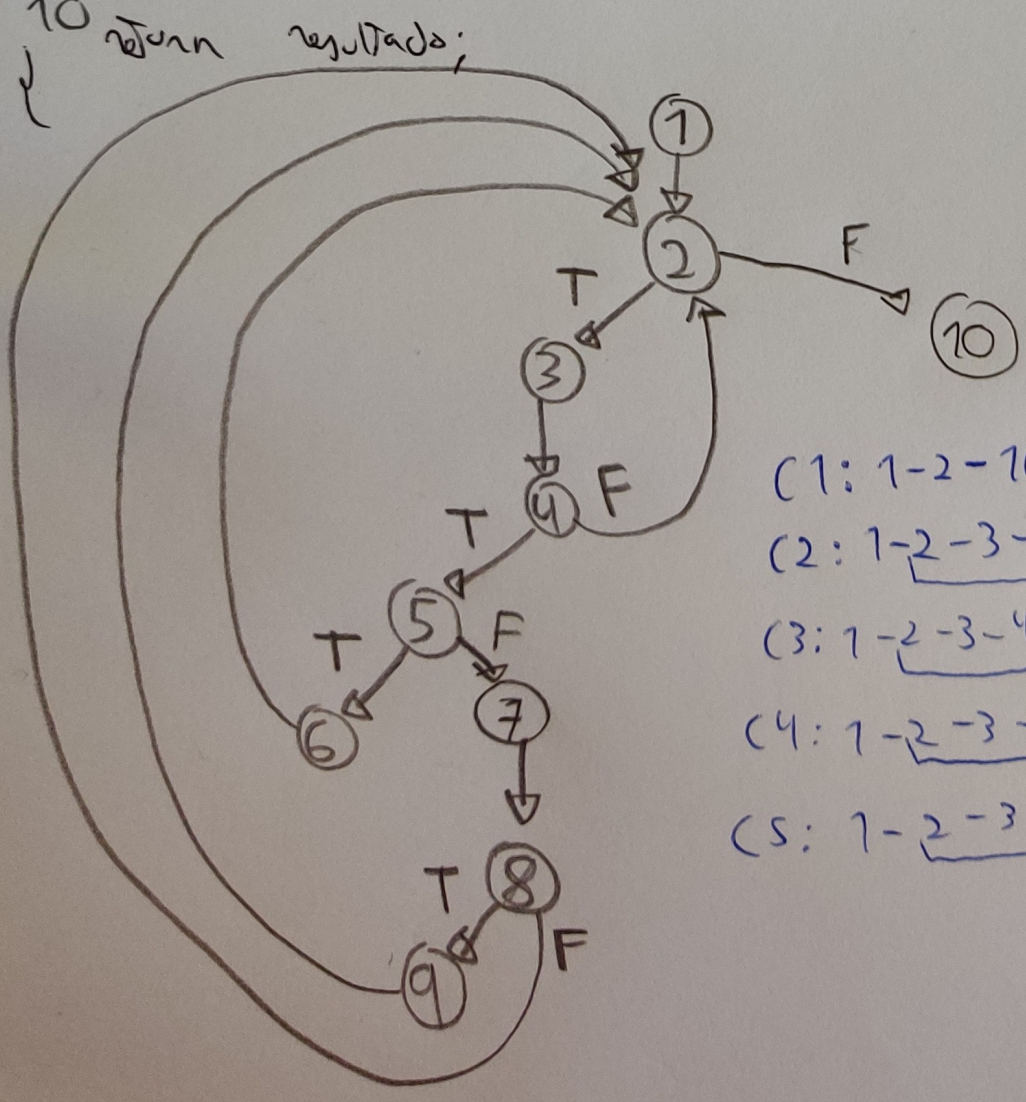
C4: 1 - 2 - 3 - 4 - 5 - 7 - 8 - 9 - 2 - 12

C5: 1 - 2 - 3 - 4 - 5 - 7 - 8 - 10 - 11 - 2 - 12

Exercicio 4:

```
public String obterAcrónimo(String cadea) {
1   String resultado = "";
    char caracter;
2   int n = cadea.length();
    for(int i=0; i<n; i++){
3     caracter = cadena.charAT(i);
4     if(caracter != ' '){
5       if(i=0){
6         resultado = resultado + caracter + '.';
7     }else{
8       if(cadena.charAT(i-1) == ' '){
9         resultado = resultado + caracter + '.';
    }
10  return resultado;
```

V(G) = 13 + 2 - 10 = 5



C1: 1-2-10

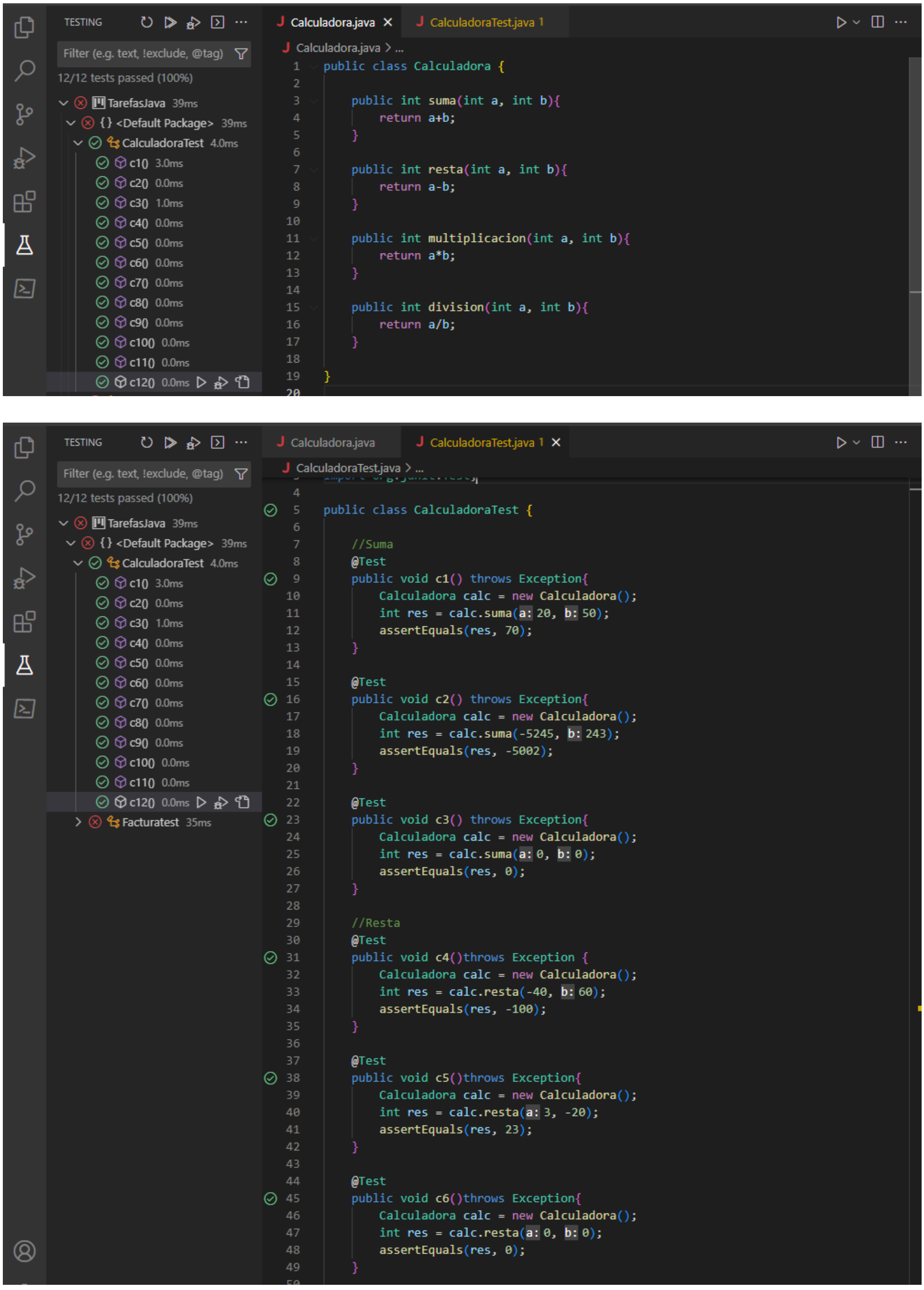
C2: 1-2-3-4-2-10

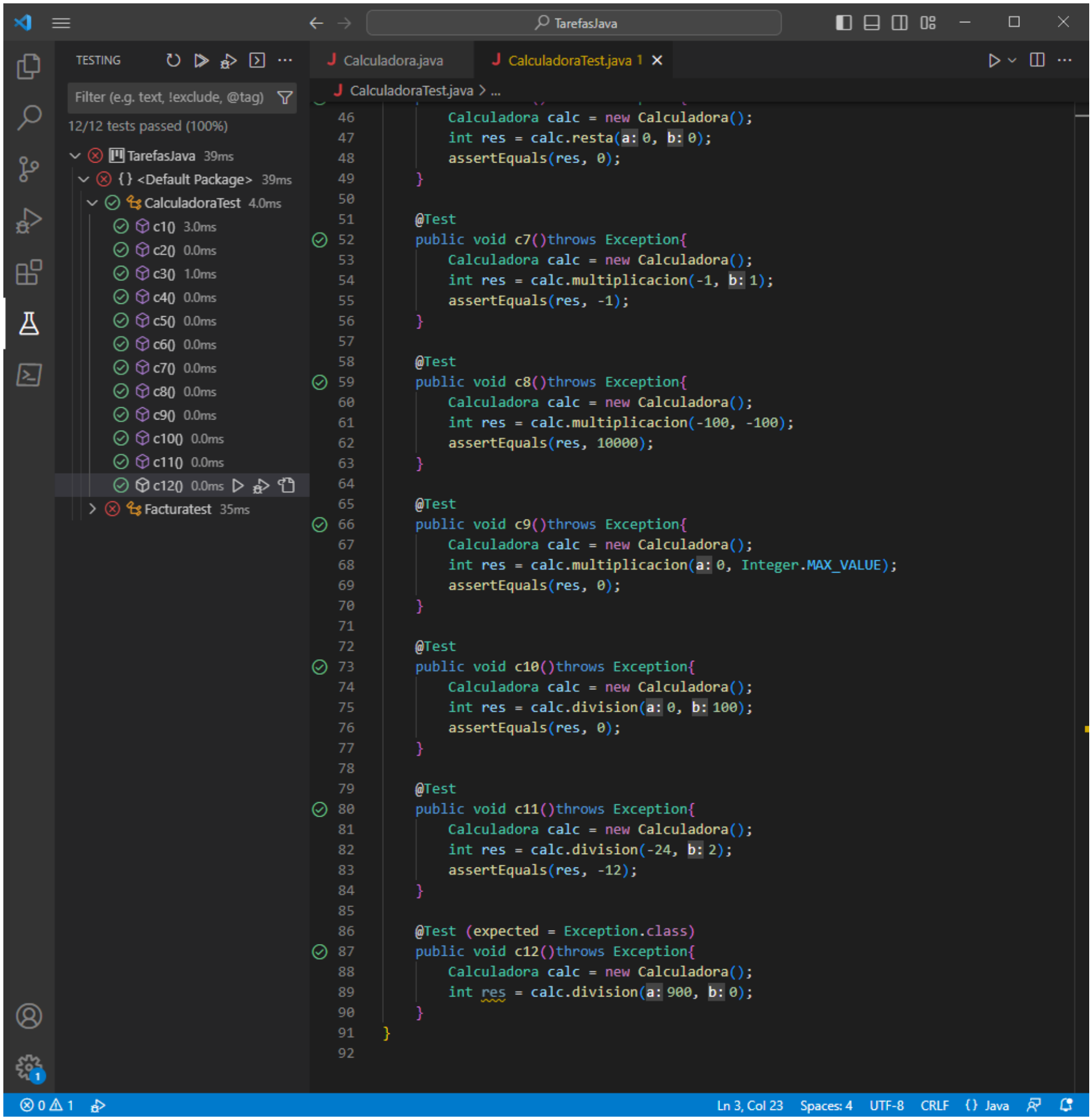
C3: 1-2-3-4-5-6-2-10

C4: 1-2-3-4-5-7-8-2-10

C5: 1-2-3-4-5-7-8-9-2-10

Exercicio 5


TESTING
Filter (e.g. text, !exclude, @tag)
12/12 tests passed (100%)
TarefasJava 39ms
{} <Default Package> 39ms
CalculadoraTest 4.0ms
c1() 3.0ms
c2() 0.0ms
c3() 1.0ms
c4() 0.0ms
c5() 0.0ms
c6() 0.0ms
c7() 0.0ms
c8() 0.0ms
c9() 0.0ms
c10() 0.0ms
c11() 0.0ms
c12() 0.0ms
Facturatest 35ms
Calculadora.java
CalculadoraTest.java 1
Calculadora.java > ...
public class Calculadora {
public int suma(int a, int b){
return a+b;
}
public int resta(int a, int b){
return a-b;
}
public int multiplicacion(int a, int b){
return a*b;
}
public int division(int a, int b){
return a/b;
}
}
CalculadoraTest.java > ...
public class CalculadoraTest {
//Suma
@Test
public void c1() throws Exception{
Calculadora calc = new Calculadora();
int res = calc.suma(a: 20, b: 50);
assertEquals(res, 70);
}
@Test
public void c2() throws Exception{
Calculadora calc = new Calculadora();
int res = calc.suma(-5245, b: 243);
assertEquals(res, -5002);
}
@Test
public void c3() throws Exception{
Calculadora calc = new Calculadora();
int res = calc.suma(a: 0, b: 0);
assertEquals(res, 0);
}
//Resta
@Test
public void c4()throws Exception {
Calculadora calc = new Calculadora();
int res = calc.resta(-40, b: 60);
assertEquals(res, -100);
}
@Test
public void c5()throws Exception{
Calculadora calc = new Calculadora();
int res = calc.resta(a: 3, -20);
assertEquals(res, 23);
}
@Test
public void c6()throws Exception{
Calculadora calc = new Calculadora();
int res = calc.resta(a: 0, b: 0);
assertEquals(res, 0);
}




```
46          Calculadora calc = new Calculadora();
47          int res = calc.resta(a: 0, b: 0);
48          assertEquals(res, 0);
49      }
50
51      @Test
52      public void c7()throws Exception{
53          Calculadora calc = new Calculadora();
54          int res = calc.multiplicacion(-1, b: 1);
55          assertEquals(res, -1);
56      }
57
58      @Test
59      public void c8()throws Exception{
60          Calculadora calc = new Calculadora();
61          int res = calc.multiplicacion(-100, -100);
62          assertEquals(res, 10000);
63      }
64
65      @Test
66      public void c9()throws Exception{
67          Calculadora calc = new Calculadora();
68          int res = calc.multiplicacion(a: 0, Integer.MAX_VALUE);
69          assertEquals(res, 0);
70      }
71
72      @Test
73      public void c10()throws Exception{
74          Calculadora calc = new Calculadora();
75          int res = calc.division(a: 0, b: 100);
76          assertEquals(res, 0);
77      }
78
79      @Test
80      public void c11()throws Exception{
81          Calculadora calc = new Calculadora();
82          int res = calc.division(-24, b: 2);
83          assertEquals(res, -12);
84      }
85
86      @Test (expected = Exception.class)
87      public void c12()throws Exception{
88          Calculadora calc = new Calculadora();
89          int res = calc.division(a: 900, b: 0);
90      }
91  }
92
```