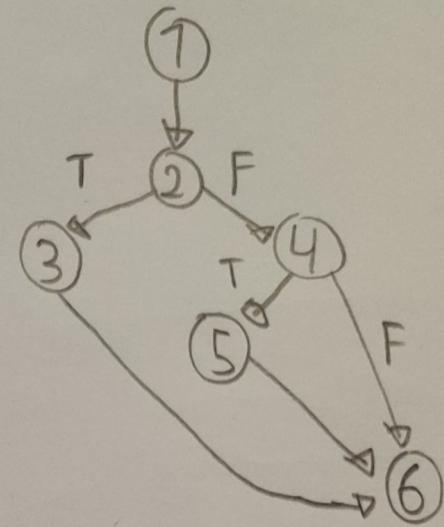


## Exercicio 1

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



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

Camino:

(1: 1-2-3-6

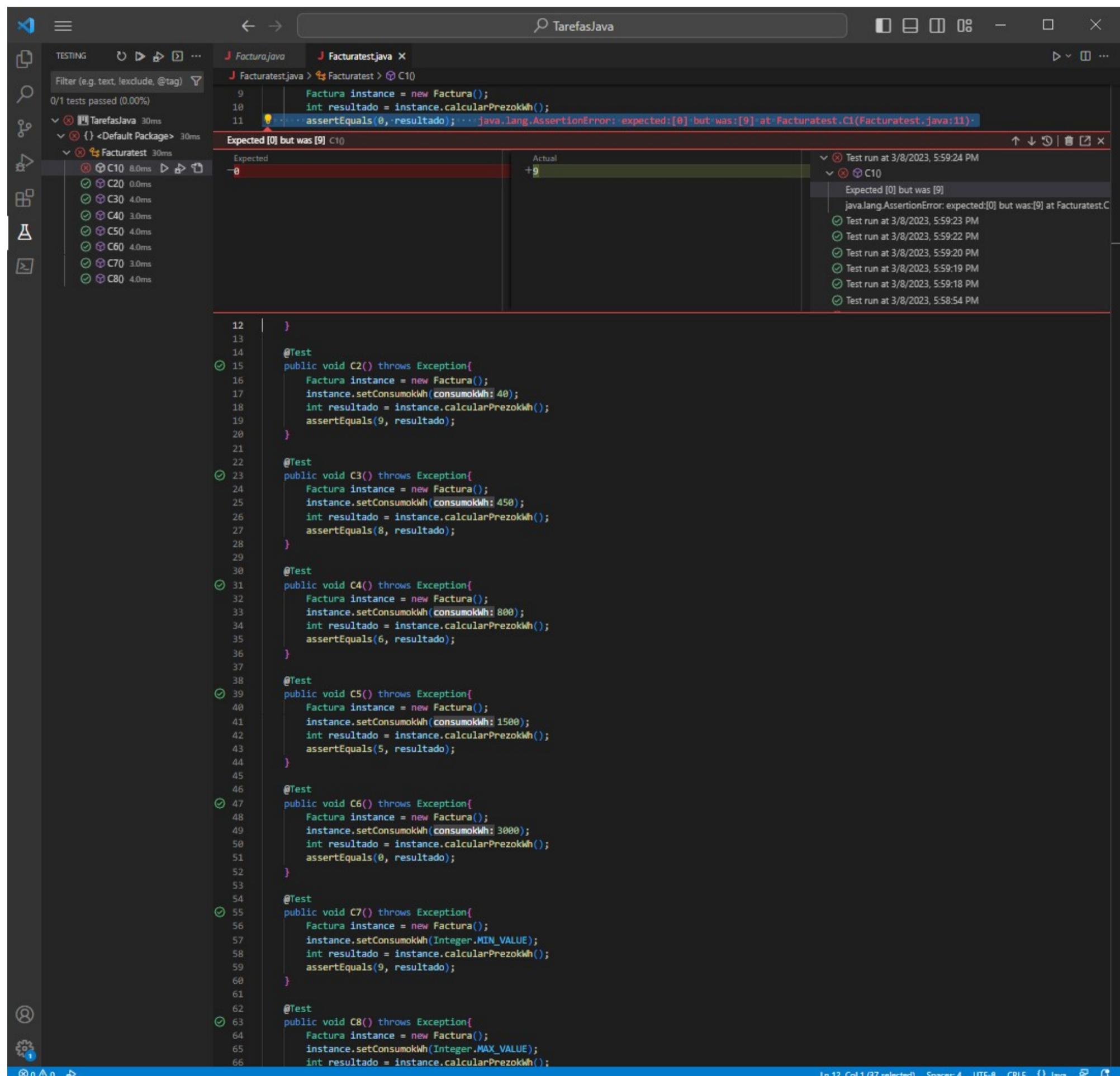
(2: 1-2-4-5-6

(3: 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 <=



```

TESTING      ⏪ ⏴ ⏵ ⏷ ⏸ ... J Facturajava J Facturatest.java ×
Filter (e.g. text, exclude, @tag) T
0/1 tests passed (0.00%)
✗ ⚡ TareasJava 30ms
✗ ⚡ {} <Default Package> 30ms
✗ ⚡ Facturatest 30ms
✗ ⚡ C10 8.0ms
✗ ⚡ C20 0.0ms
✗ ⚡ C30 4.0ms
✗ ⚡ C40 3.0ms
✗ ⚡ C50 4.0ms
✗ ⚡ C60 4.0ms
✗ ⚡ C70 3.0ms
✗ ⚡ C80 4.0ms
J Facturatestjava > Facturatest > C10
    Factura instance = new Factura();
    int resultado = instance.calcularPrezokWh();
    assertEquals(0, resultado); ... java.lang.AssertionError: expected:[0] but was:[9] at Facturatest.C1(Facturatest.java:11)
Expected [0] but was [9] C10
    Expected: 0
    Actual: +9
    ✓ Test run at 3/8/2023, 5:59:24 PM
    ✓ C10
        Expected [0] but was [9]
        java.lang.AssertionError: expected:[0] but was:[9] at Facturatest.C1(Facturatest.java:11)
        ✓ Test run at 3/8/2023, 5:59:23 PM
        ✓ Test run at 3/8/2023, 5:59:22 PM
        ✓ Test run at 3/8/2023, 5:59:20 PM
        ✓ Test run at 3/8/2023, 5:59:19 PM
        ✓ Test run at 3/8/2023, 5:59:18 PM
        ✓ Test run at 3/8/2023, 5:59:17 PM
        ✓ Test run at 3/8/2023, 5:58:54 PM
12     }
13
14     @Test
15     public void C2() throws Exception{
16         Factura instance = new Factura();
17         instance.setConsumokWh(consumokWh: 40);
18         int resultado = instance.calcularPrezokWh();
19         assertEquals(9, resultado);
20     }
21
22     @Test
23     public void C3() throws Exception{
24         Factura instance = new Factura();
25         instance.setConsumokWh(consumokWh: 450);
26         int resultado = instance.calcularPrezokWh();
27         assertEquals(8, resultado);
28     }
29
30     @Test
31     public void C4() throws Exception{
32         Factura instance = new Factura();
33         instance.setConsumokWh(consumokWh: 800);
34         int resultado = instance.calcularPrezokWh();
35         assertEquals(6, resultado);
36     }
37
38     @Test
39     public void C5() throws Exception{
40         Factura instance = new Factura();
41         instance.setConsumokWh(consumokWh: 1500);
42         int resultado = instance.calcularPrezokWh();
43         assertEquals(5, resultado);
44     }
45
46     @Test
47     public void C6() throws Exception{
48         Factura instance = new Factura();
49         instance.setConsumokWh(consumokWh: 3000);
50         int resultado = instance.calcularPrezokWh();
51         assertEquals(0, resultado);
52     }
53
54     @Test
55     public void C7() throws Exception{
56         Factura instance = new Factura();
57         instance.setConsumokWh(Integer.MIN_VALUE);
58         int resultado = instance.calcularPrezokWh();
59         assertEquals(9, resultado);
60     }
61
62     @Test
63     public void C8() throws Exception{
64         Factura instance = new Factura();
65         instance.setConsumokWh(Integer.MAX_VALUE);
66         int resultado = instance.calcularPrezokWh();

```

Ln 12, Col 1 (37 selected) Spaces: 4 UTF-8 CRLF ⓘ Java ⌂ ⌃ ⌄

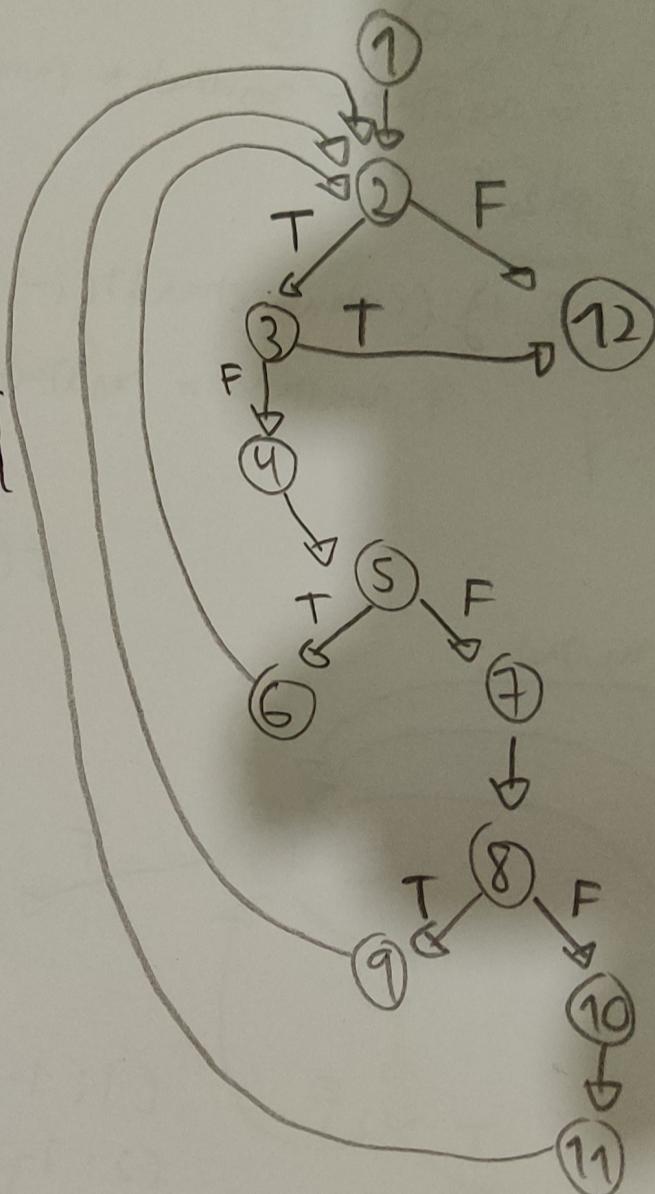
### Exercício3

```

public boolean busca(char c, char[] v) {
    int a, z, m;
    a = 0;
    z = v.length - 1;
    boolean resultado = false; 3
    while (a <= z && resultado == false) {
        m = (a + z) / 2;
        if (v[m] == c) {
            if (resultado = true);
        } else {
            if (v[m] < c) {
                a = m + 1;
            } else {
                z = m - 1;
            }
        }
    }
    return resultado;
}

```

$$V(G) = 18 + 2 - 12 = 14$$



C1: 1-2-12

C2: 1-2-3-12

C3: 1-2-3-4-5-6-2-12 | ~~unb~~

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

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

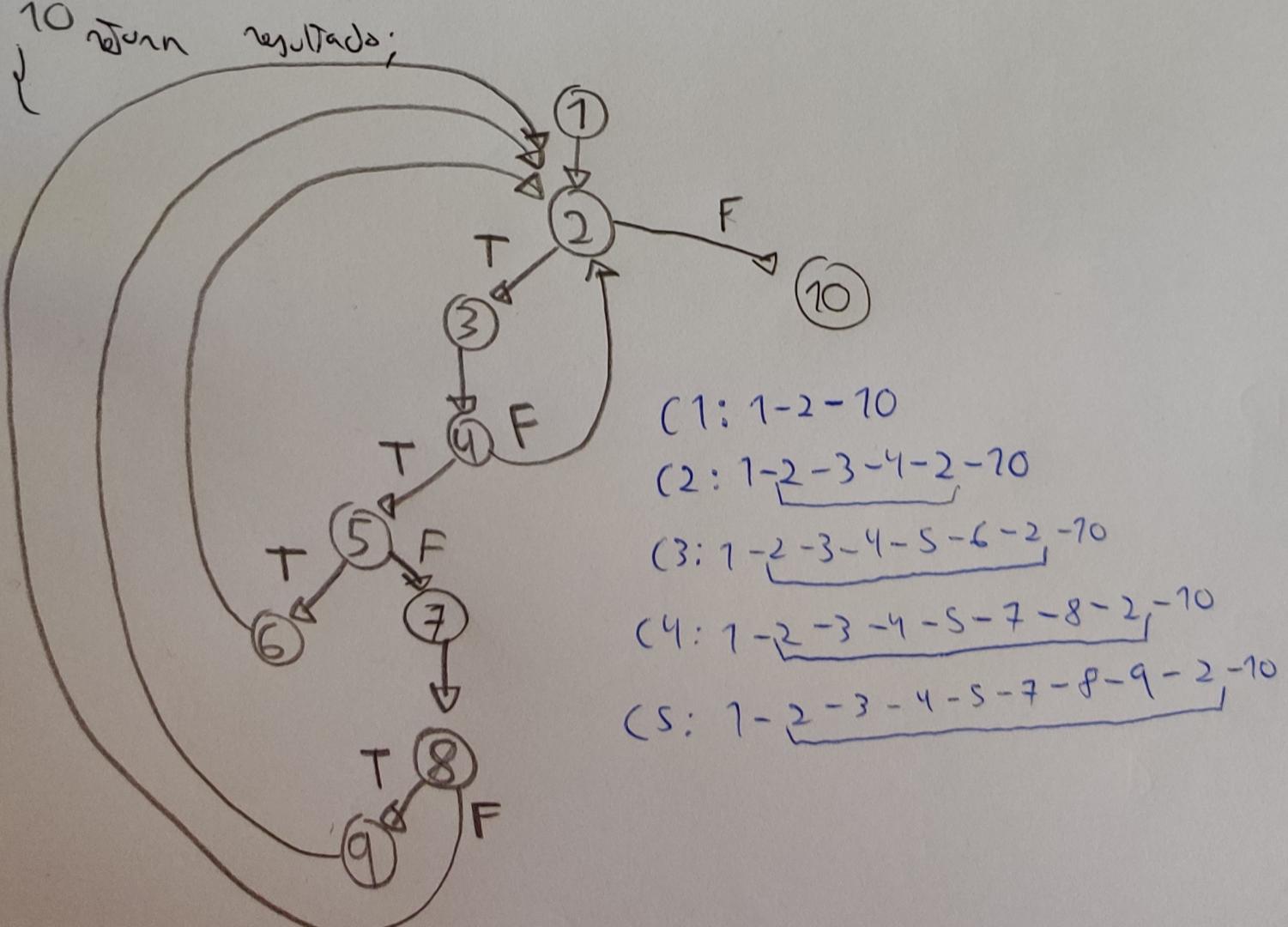
#### Exercicio 4:

```

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

```

$$V(G) = 73 + 2 - 10 = 5$$



## Exercicio 5

The screenshot shows the Java IDE interface with the 'TESTING' view open. The status bar indicates '12/12 tests passed (100%)'. The left sidebar has icons for file, search, and other tools. The main area displays two tabs: 'Calculadora.java' and 'CalculadoraTest.java 1'. The code for 'Calculadora.java' is:

```
1 public class Calculadora {  
2     public int suma(int a, int b){  
3         return a+b;  
4     }  
5     public int resta(int a, int b){  
6         return a-b;  
7     }  
8     public int multiplicacion(int a, int b){  
9         return a*b;  
10    }  
11    public int division(int a, int b){  
12        return a/b;  
13    }  
14}  
15}
```

The screenshot shows the Java IDE interface with the 'TESTING' view open. The status bar indicates '12/12 tests passed (100%)'. The left sidebar has icons for file, search, and other tools. The main area displays two tabs: 'Calculadora.java' and 'CalculadoraTest.java 1'. The code for 'CalculadoraTest.java' is:

```
1 public class CalculadoraTest {  
2     //Suma  
3     @Test  
4     public void c1() throws Exception{  
5         Calculadora calc = new Calculadora();  
6         int res = calc.suma(a: 20, b: 50);  
7         assertEquals(res, 70);  
8     }  
9     //Resta  
10    @Test  
11    public void c2() throws Exception{  
12        Calculadora calc = new Calculadora();  
13        int res = calc.suma(-5245, b: 243);  
14        assertEquals(res, -5002);  
15    }  
16    @Test  
17    public void c3() throws Exception{  
18        Calculadora calc = new Calculadora();  
19        int res = calc.suma(a: 0, b: 0);  
20        assertEquals(res, 0);  
21    }  
22    //Multiplicacion  
23    @Test  
24    public void c4()throws Exception {  
25        Calculadora calc = new Calculadora();  
26        int res = calc.resta(-40, b: 60);  
27        assertEquals(res, -100);  
28    }  
29    //Division  
30    @Test  
31    public void c5()throws Exception{  
32        Calculadora calc = new Calculadora();  
33        int res = calc.resta(a: 3, -20);  
34        assertEquals(res, 23);  
35    }  
36    @Test  
37    public void c6()throws Exception{  
38        Calculadora calc = new Calculadora();  
39        int res = calc.resta(a: 0, b: 0);  
40        assertEquals(res, 0);  
41    }  
42}
```

The screenshot shows a Java development environment with a dark theme. On the left, there's a sidebar with various icons for navigation and configuration. The main area has two tabs at the top: 'Calculadora.java' and 'CalculadoraTest.java'. The 'CalculadoraTest.java' tab is active, displaying a class with several test methods. The code uses annotations like @Test and throws Exception. It includes methods for addition, subtraction, multiplication, division, and exception handling. The test runner on the left shows 12/12 tests passed in 39ms. The status bar at the bottom provides information about the current file ('CalculadoraTest.java'), including line number (Ln 3), column (Col 23), and encoding (UTF-8). It also shows options for spaces (Spaces: 4), CRLF, and Java.

```
CalculadoraTest.java > ...
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 }
```