

## Atividade 2: JUNIT Testes Automatizados

### Relatório de Erros para a classe Calculadora.java

**Aluna:** Isabela Queiroz Marinho

Repositório do Github: [IsabelaQM/atividadeCalculadora \(github.com\)](https://github.com/IsabelaQM/atividadeCalculadora)

Observação: A memória foi definida como um valor inteiro no código fornecido para a classe Calculadora. Com isso, os resultados obtidos dos casos de teste também serão retornados como valores inteiros, mesmo se o método avaliado tenha sido implementado com a lógica correta.

#### 1) Teste para o construtor sem parâmetro de entrada

Método a ser Testado	Calculadora()
Cenário de teste (entradas)	Construtor sem parâmetro
Resultado Esperado	0
Resultado Obtido	1

Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testConstrutorSemParametro()
11 class CalculadoraTest {
25 // Teste para o construtor sem parâmetro de entrada
26 @Test
27 void testConstrutorSemParametro() throws Exception{
28     calc = new Calculadora();
29     assertEquals(expected:0, calc.getMemoria()); ... Expected [0] but was [1]

Expected [0] but was [1] testConstrutorSemParametro()
Expected
-0
Actual
+1

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS
Começou o teste!!!!!!
Inicializando caso de teste
Finalizando caso de teste
Fim do teste!!!!
```

## 2) Teste para o construtor com parâmetro: recebendo o valor 3

Método a ser Testado	Calculadora(int)
Cenário de teste (entradas)	memória = 3
Resultado Esperado	3
Resultado Obtido	3

Captura da Tela do Resultado do Teste:

The screenshot displays an IDE with two tabs: 'Calculadora.java' and 'CalculadoraTest.java'. The 'CalculadoraTest.java' tab is active, showing the following code:

```
11 class CalculadoraTest {
27     void testConstrutorSemParametro() throws Exception{
29         assertEquals(expected:0, calc.getMemoria());
30     }
31
32     // Teste para o construtor com parâmetro de entrada = 3
33     @Test
34     void testConstrutorComParametro() throws Exception{
35         assertEquals(expected:3, calc.getMemoria());
36     }
37
38     // Teste do método somar: somar um número negativo (construtor com valor = 3)
39     @Test
40     void testSomarNumeroNegativo() throws Exception{
41         calc.somar(-5);
42         assertEquals(expected:3, calc.getMemoria());
43     }
44 }
```

The bottom of the IDE shows the 'TEST RESULTS' panel. It lists the following test results:

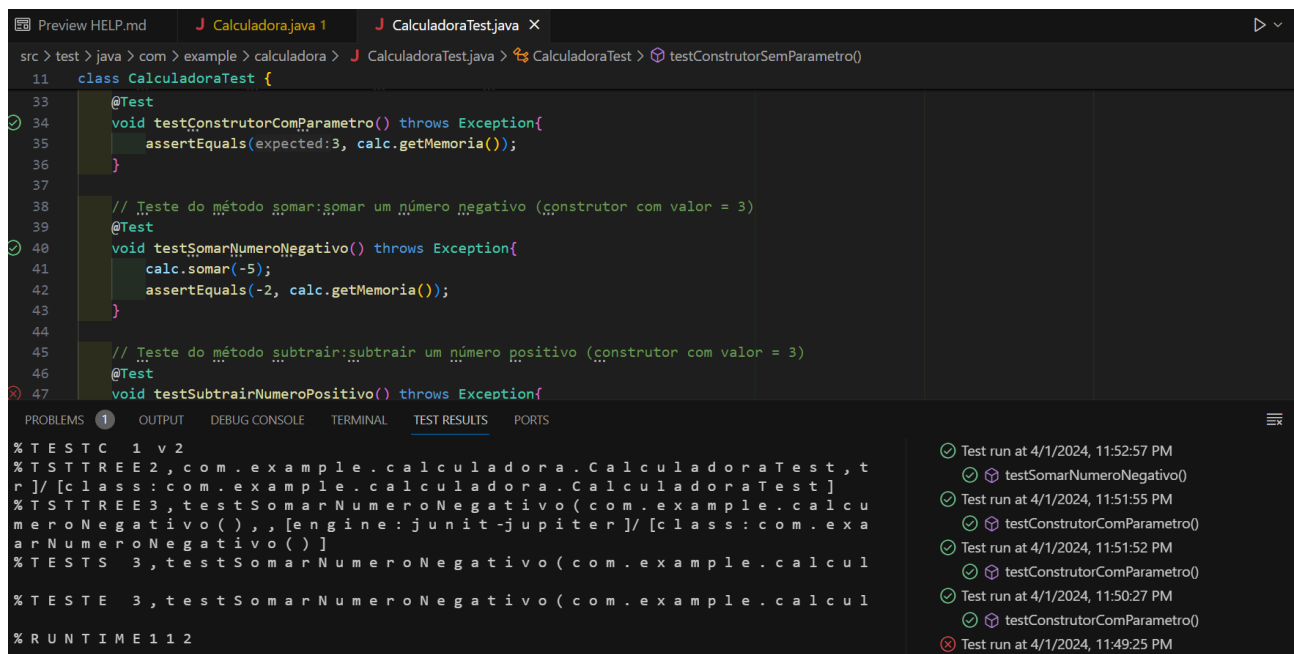
- Test run at 4/1/2024, 11:51:55 PM: testConstrutorComParametro() (Success)
- Test run at 4/1/2024, 11:51:52 PM: testConstrutorComParametro() (Success)
- Test run at 4/1/2024, 11:50:27 PM: testConstrutorComParametro() (Success)
- Test run at 4/1/2024, 11:49:25 PM: testConstrutorSemParametro() (Failure)

The failure message for the last test is: 'Expected [0] but was [1]'.

### 3) Teste do método somar: somar um número negativo

Método a ser Testado	somar(int)
Cenário de teste (entradas)	memória = 3, valor = -5
Resultado Esperado	-2
Resultado Obtido	-2

Captura da Tela do Resultado do Teste:



```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testConstrutorSemParametro()
11 class CalculadoraTest {
33     @Test
34     void testConstrutorComParametro() throws Exception{
35         assertEquals(expected:3, calc.getMemoria());
36     }
37
38     // Teste do método somar:somar um número negativo (construtor com valor = 3)
39     @Test
40     void testSomarNumeroNegativo() throws Exception{
41         calc.somar(-5);
42         assertEquals(-2, calc.getMemoria());
43     }
44
45     // Teste do método subtrair:subtrair um número positivo (construtor com valor = 3)
46     @Test
47     void testSubtrairNumeroPositivo() throws Exception{
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```
% TEST C 1 v 2
% TSTREE 2, com.example.calculadora.CalculadoraTest, t
r ]/[class:com.example.calculadora.CalculadoraTest]
% TSTREE 3, testSomarNumeroNegativo(com.example.calcu
meroNegativo()), , [engine:junit-jupiter]/[class:com.exa
rNumeroNegativo()]
% TESTS 3, testSomarNumeroNegativo(com.example.calcul
% TESTE 3, testSomarNumeroNegativo(com.example.calcul
% RUNTIME 112
```

Test run at 4/1/2024, 11:52:57 PM  
testSomarNumeroNegativo()  
Test run at 4/1/2024, 11:51:55 PM  
testConstrutorComParametro()  
Test run at 4/1/2024, 11:51:52 PM  
testConstrutorComParametro()  
Test run at 4/1/2024, 11:50:27 PM  
testConstrutorComParametro()  
Test run at 4/1/2024, 11:49:25 PM

#### 4) Teste do método subtrair: subtrair um número positivo

Método a ser Testado	subtrair(int)
Cenário de teste (entradas)	memória = 3, valor = 3
Resultado Esperado	0
Resultado Obtido	3

Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testSubtrairNumeroPositivo()
11 class CalculadoraTest {
45 // Teste do método subtrair: subtrair um número positivo (construtor com valor = 3)
46 @Test
47 void testSubtrairNumeroPositivo() throws Exception{
48     calc.subtrair(valor:3);
49     assertEquals(expected:0, calc.getMemoria()); Expected [0] but was [3]

```

Expected [0] but was [3] testSubtrairNumeroPositivo()

Expected	Actual
-0	+3

```
% TEST C 1 v 2
% TST TREE 2, com.example.calculadora.CalculadoraTest, t
r ]/[class:com.example.calculadora.CalculadoraTest]
% TST TREE 3, testSubtrairNumeroPositivo(com.example.ca
r airNumeroPositivo()), [engine:junit-jupiter]/[class:c
estSubtrairNumeroPositivo()]
% TESTS 3, testSubtrairNumeroPositivo(com.example.cal
% FAILED 3, testSubtrairNumeroPositivo(com.example.cal
% EXPECTS
0
% EXPECTE

```

Test run at 4/1/2024, 11:54:28 PM  
testSubtrairNumeroPositivo()  
Expected [0] but was [3]  
org.opentest4j.AssertionFailedError: expected: [0] but wa...  
Test run at 4/1/2024, 11:52:57 PM  
testSomarNumeroNegativo()  
Test run at 4/1/2024, 11:51:55 PM  
testConstrutorComParametro()  
Test run at 4/1/2024, 11:51:52 PM  
testConstrutorComParametro()

5) Teste do método multiplicar: multiplicar um número positivo.

Método a ser Testado	multiplicar(int)
Cenário de teste (entradas)	memória = 3, valor = 2
Resultado Esperado	6
Resultado Obtido	1

Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testMultiplicarNumeroPositivo()
11 class CalculadoraTest {
12     // Teste do método multiplicar: multiplicar um número positivo (construtor com valor = 3)
13     @Test
14     void testMultiplicarNumeroPositivo() throws Exception{
15         calc.multiplicar(valor:2);
16         assertEquals(expected:6, calc.getMemoria()); Expected [6] but was [1]
17     }
18 }
```

Expected [6] but was [1] testMultiplicarNumeroPositivo()

Expected	Actual
-6	+1

```
% TESTC 1 v 2
% TSTTTEE2, com.example.calculadora.CalculadoraTest, t
r]/ [class: com.example.calculadora.CalculadoraTest]
% TSTTTEE3, testMultiplicarNumeroPositivo(com.example
ultiplicarNumeroPositivo()), [engine:junit-jupiter]/ [c
thod:testMultiplicarNumeroPositivo()]
% TESTS 3, testMultiplicarNumeroPositivo(com.example.

% FAILED3, testMultiplicarNumeroPositivo(com.example.
% EXPECTS
6
% EXPECTE
```

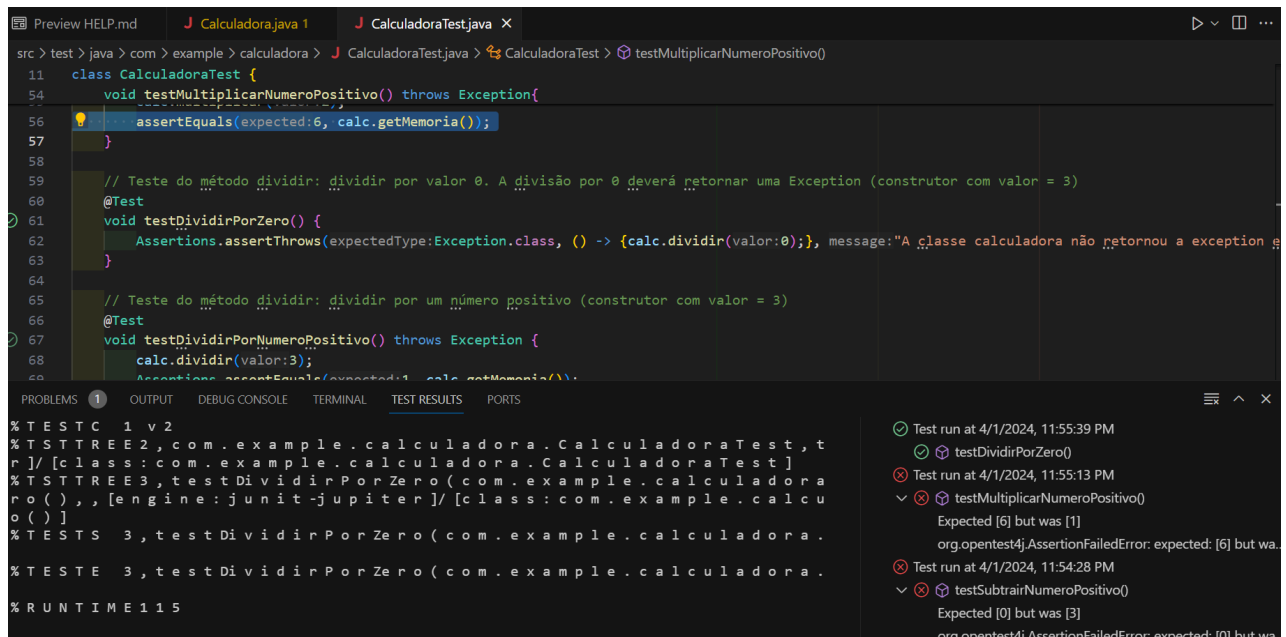
Test run at 4/1/2024, 11:55:13 PM  
testMultiplicarNumeroPositivo()  
Expected [6] but was [1]  
org.opentest4j.AssertionFailedError: expected: [6] but wa...  
Test run at 4/1/2024, 11:54:28 PM  
testSubtrairNumeroPositivo()  
Expected [0] but was [3]  
org.opentest4j.AssertionFailedError: expected: [0] but wa...  
Test run at 4/1/2024, 11:52:57 PM

- 6) Testes do método dividir: dividir por valor 0 e dividir por um valor positivo. A divisão por 0 deverá retornar uma Exception

a) Dividir por 0

Método a ser Testado	dividir(int)
Cenário de teste (entradas)	memória = 3, valor = 0
Resultado Esperado	Exception
Resultado Obtido	Exception

Captura da Tela do Resultado do Teste:



The screenshot shows an IDE with a Java file named `CalculadoraTest.java`. The code defines a class `CalculadoraTest` with two test methods: `testDividirPorZero()` and `testDividirPorNumeroPositivo()`. The `testDividirPorZero()` method is annotated with `@Test` and `Assertions.assertThrows` to expect an `Exception` when `dividir(0)` is called. The `testDividirPorNumeroPositivo()` method is also annotated with `@Test` and `throws Exception`. The test results panel at the bottom shows the following output:

```
% TESTC 1 v 2
% TSTTREE2, com.example.calculadora.CalculadoraTest, t
r ]/ [class: com.example.calculadora.CalculadoraTest]
% TSTTREE3, testDividirPorZero (com.example.calculadora
ro ( ), [engine: junit-jupiter] / [class: com.example.calcu
o ( ) ]
% TESTS 3, testDividirPorZero (com.example.calculadora.
% TESTE 3, testDividirPorZero (com.example.calculadora.
% RUNTIME 115
```

The test results panel also shows the following details:

- Test run at 4/1/2024, 11:55:39 PM
- testDividirPorZero()
- Test run at 4/1/2024, 11:55:13 PM
- testMultiplicarNumeroPositivo()
- Expected [6] but was [1]
- org.opentest4j.AssertionFailedError: expected: [6] but wa
- Test run at 4/1/2024, 11:54:28 PM
- testSubtrairNumeroPositivo()
- Expected [0] but was [3]
- org.opentest4j.AssertionFailedError: expected: [0] but wa

b) Dividir por valor positivo

Método a ser Testado	dividir(int)
Cenário de teste (entradas)	memória = 3, valor = 3
Resultado Esperado	1
Resultado Obtido	1

## Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testMultiplicarNumeroPositivo()
11 class CalculadoraTest {
60     @Test
61     void testDividirPorZero() {
62         Assertions.assertThrows(expectedType:Exception.class, () -> {calc.dividir(valor:0);}, message:"A classe calculadora não retornou a exception e
63     }
64
65     // Teste do método dividir: dividir por um número positivo (construtor com valor = 3)
66     @Test
67     void testDividirPorNumeroPositivo() throws Exception {
68         calc.dividir(valor:3);
69         Assertions.assertEquals(expected:1, calc.getMemoria());
70     }
71
72     // Teste do método exponenciação: exponenciar a memória por 1
73     @Test
74     void testExponenciarMemoriaPorUm() throws Exception {
75
% TESTC 1 v 2
% TSTTREE2, com.example.calculadora.CalculadoraTest, t
r]/[class:com.example.calculadora.CalculadoraTest]
% TSTTREE3, testDividirPorNumeroPositivo(com.example.c
vidirPorNumeroPositivo(),, [engine:junit-jupiter]/[cla
od:testDividirPorNumeroPositivo()]
% TESTS 3, testDividirPorNumeroPositivo(com.example.ca
% TESTE 3, testDividirPorNumeroPositivo(com.example.ca
% RUNTIME111

Test run at 4/1/2024, 11:56:11 PM
  ✓ testDividirPorNumeroPositivo()
  ✓ Test run at 4/1/2024, 11:55:39 PM
  ✓ testDividirPorZero()
  ✗ Test run at 4/1/2024, 11:55:13 PM
    ✗ testMultiplicarNumeroPositivo()
      Expected [6] but was [1]
      org.opentest4j.AssertionFailedError: expected: [6] but wa
  ✗ Test run at 4/1/2024, 11:54:28 PM
```

7) Testes da exponenciação: exponenciar a memória por 1 e por 10. Se a entrada for um valor maior que 10 deverá retornar uma Exception.

a) Exponenciar por 1

Método a ser Testado	exponenciar(int)
Cenário de teste (entradas)	memória = 3, valor = 1
Resultado Esperado	3
Resultado Obtido	1995565057

Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testExponenciarMemoriaPorUm()
11 class CalculadoraTest {
12     // Teste do método exponenciação: exponenciar a memória por 1
13     @Test
14     void testExponenciarMemoriaPorUm() throws Exception {
15         calc.exponenciar(valor:1);
16         assertEquals(expected:3, calc.getMemoria()); Expected [3] but was [1995565057]
17     }
18 }

Expected [3] but was [1995565057] testExponenciarMemoriaPorUm()

Expected
-3

Actual
+1995565057

% TEST C 1 v 2
% TSTTTEE2, com.example.calculadora.CalculadoraTest, t
r]/ [class:com.example.calculadora.CalculadoraTest]
% TSTTTEE3, testExponenciarMemoriaPorUm(com.example.c
onenciarMemoriaPorUm()), [engine:junit-jupiter]/ [class
:testExponenciarMemoriaPorUm()]
% TESTS 3, testExponenciarMemoriaPorUm(com.example.ca
% FAILED 3, testExponenciarMemoriaPorUm(com.example.ca
% EXPECTS
3
% EXPECTE

Test run at 4/1/2024, 11:56:38 PM
testExponenciarMemoriaPorUm()
Expected [3] but was [1995565057]
org.opentest4j.AssertionFailedError: expected: [3] but wa...

Test run at 4/1/2024, 11:56:11 PM
testDividirPorNumeroPositivo()

Test run at 4/1/2024, 11:55:39 PM
testDividirPorZero()

Test run at 4/1/2024, 11:55:13 PM
testMultiplicarPorNumeroPositivo()
```

b) Exponenciar por 10

Método a ser Testado	exponenciar(int)
Cenário de teste (entradas)	memória = 3, valor = 10
Resultado Esperado	59049
Resultado Obtido	1995565057



## Captura da Tela do Resultado do Teste:

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testExponenciarMemoriaPorDez()
11 class CalculadoraTest {
79 // Teste do método exponenciação: exponenciar a memória por 10
80 @Test
81 void testExponenciarMemoriaPorDez() throws Exception{
82     calc.exponenciar(valor:10);
83     assertEquals(expected:59049, calc.getMemoria()); ... Expected [59049] but was [1995565057]
```

Expected	Actual
-59049	+1995565057

```
% TESTC 1 v 2
% TSTTREE2, com.example.calculadora.CalculadoraTest, t
r]/[class:com.example.calculadora.CalculadoraTest]
% TSTTREE3, testExponenciarMemoriaPorDez(com.example.c
ponenciarMemoriaPorDez()), [engine:junit-jupiter]/[clas
od:testExponenciarMemoriaPorDez()]
% TESTS 3, testExponenciarMemoriaPorDez(com.example.ca
% FAILED3, testExponenciarMemoriaPorDez(com.example.ca
% EXPECTS
59049
% EXPECTE
```

Test run at 4/1/2024, 11:57:09 PM  
✗ testExponenciarMemoriaPorDez()  
Expected [59049] but was [1995565057]  
org.opentest4j.AssertionFailedError: expected: [59049] bu...

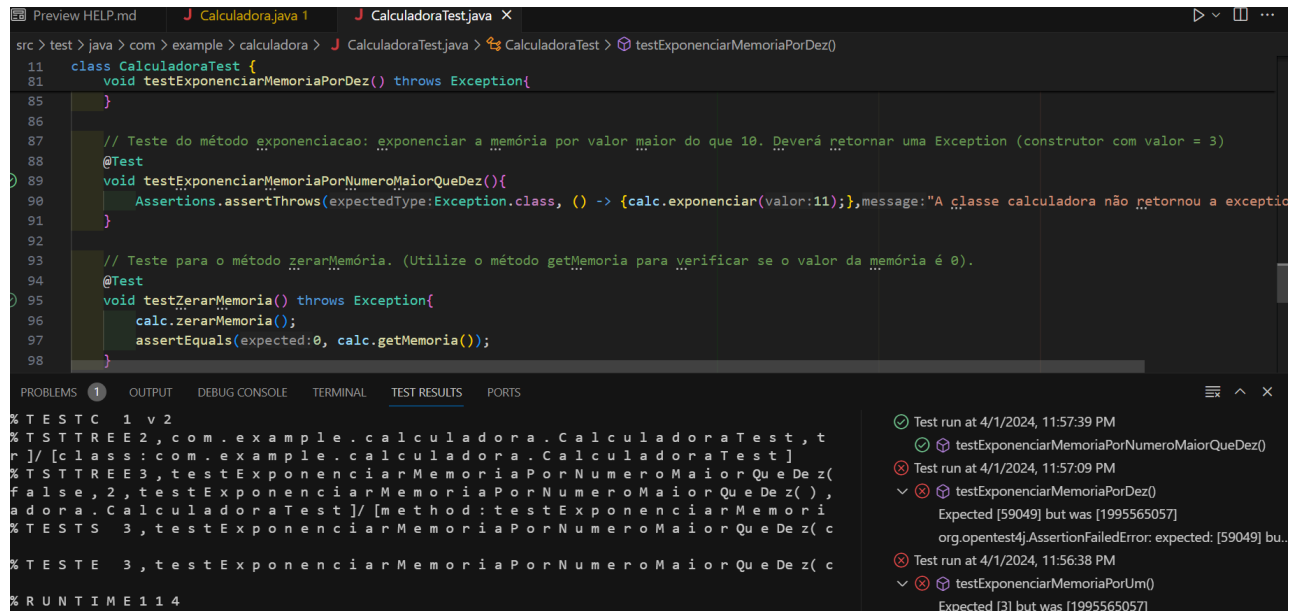
Test run at 4/1/2024, 11:56:38 PM  
✗ testExponenciarMemoriaPorUm()  
Expected [3] but was [1995565057]  
org.opentest4j.AssertionFailedError: expected: [3] but wa...

Test run at 4/1/2024, 11:56:11 PM  
✓ testDividirPorNumeroPositivo()

c) Exponenciar por um valor maior do que 10

Método a ser Testado	exponenciar(int)
Cenário de teste (entradas)	memória = 3, valor = 11
Resultado Esperado	Exception
Resultado Obtido	Exception

## Captura da Tela do Resultado do Teste:



The screenshot shows an IDE with two tabs: 'Calculadora.java' and 'CalculadoraTest.java'. The 'CalculadoraTest.java' tab is active, displaying the following code:

```
11 class CalculadoraTest {
81     void testExponenciarMemoriaPorDez() throws Exception{
85     }
86
87     // Teste do método exponenciacao: exponenciar a memória por valor maior do que 10. Deverá retornar uma Exception (construtor com valor = 3)
88     @Test
89     void testExponenciarMemoriaPorNumeroMaiorQueDez(){
90         Assertions.assertThrows(expectedType:Exception.class, () -> {calc.exponenciar(valor:11);},message:"A classe calculadora não retornou a exceptio
91     }
92
93     // Teste para o método zerarMemória. (Utilize o método getMemoria para verificar se o valor da memória é 0).
94     @Test
95     void testZerarMemoria() throws Exception{
96         calc.zerarMemoria();
97         assertEquals(expected:0, calc.getMemoria());
98     }
}
```

The bottom of the IDE shows the 'TEST RESULTS' tab. It displays the following test results:

- Test run at 4/1/2024, 11:57:39 PM
  - testExponenciarMemoriaPorNumeroMaiorQueDez()
- Test run at 4/1/2024, 11:57:09 PM
  - testExponenciarMemoriaPorDez()
    - Expected [59049] but was [1995565057]
    - org.opentest4j.AssertionFailedError: expected: [59049] bu...
- Test run at 4/1/2024, 11:56:38 PM
  - testExponenciarMemoriaPorUm()
    - Expected [3] but was [1995565057]

The left side of the IDE shows the command prompt output:

```
% TESTC 1 v 2
% TSTTREE2, com.example.calculadora.CalculadoraTest, t
r ]/[class:com.example.calculadora.CalculadoraTest]
% TSTTREE3, testExponenciarMemoriaPorNumeroMaiorQueDez(
false, 2, testExponenciarMemoriaPorNumeroMaiorQueDez(
adora.CalculadoraTest ]/[method:testExponenciarMemori
% TESTS 3, testExponenciarMemoriaPorNumeroMaiorQueDez(c
% TESTE 3, testExponenciarMemoriaPorNumeroMaiorQueDez(c
% RUNTIME 114
```

8) Teste para o método zerarMemória. (Utilizando o método getMemoria para verificar se o valor da memória é 0).

Método a ser Testado	zerarMemoria()
Cenário de teste (entradas)	memória = 3
Resultado Esperado	0
Resultado Obtido	0

Captura da Tela do Resultado do Teste:

The screenshot shows an IDE with a Java file named `CalculadoraTest.java`. The code defines a `CalculadoraTest` class with a `testZerarMemoria()` method. The test method calls `calc.zerarMemoria()` and then `assertEquals(0, calc.getMemoria())` to verify that the memory value is 0. The IDE also shows the test results in the terminal, indicating that the test passed.

```
src > test > java > com > example > calculadora > J CalculadoraTest.java > CalculadoraTest > testExponenciarMemoriaPorDez()
11 class CalculadoraTest {
12
13 // Teste para o método zerarMemória. (Utilize o método getMemoria para verificar se o valor da memória é 0).
14 @Test
15 void testZerarMemoria() throws Exception{
16     calc.zerarMemoria();
17     assertEquals(expected:0, calc.getMemoria());
18 }
19
20 @AfterEach
21 void finalizarCadaMetodoTeste(){
22     System.out.println(x:"Finalizando caso de teste");
23 }
24
25 @AfterAll
26 static void finalizarTeste(){
```

Test results summary:

- Test run at 4/1/2024, 11:58:00 PM
- testZerarMemoria() (Pass)
- Test run at 4/1/2024, 11:57:39 PM
- testExponenciarMemoriaPorNumeroMaiorQueDez() (Pass)
- Test run at 4/1/2024, 11:57:09 PM
- testExponenciarMemoriaPorDez() (Fail)
- Expected [59049] but was [1995565057]
- org.opentest4j.AssertionFailedError: expected: [59049] but was [1995565057]
- Test run at 4/1/2024, 11:56:38 PM