

Atividade A2: JUNIT

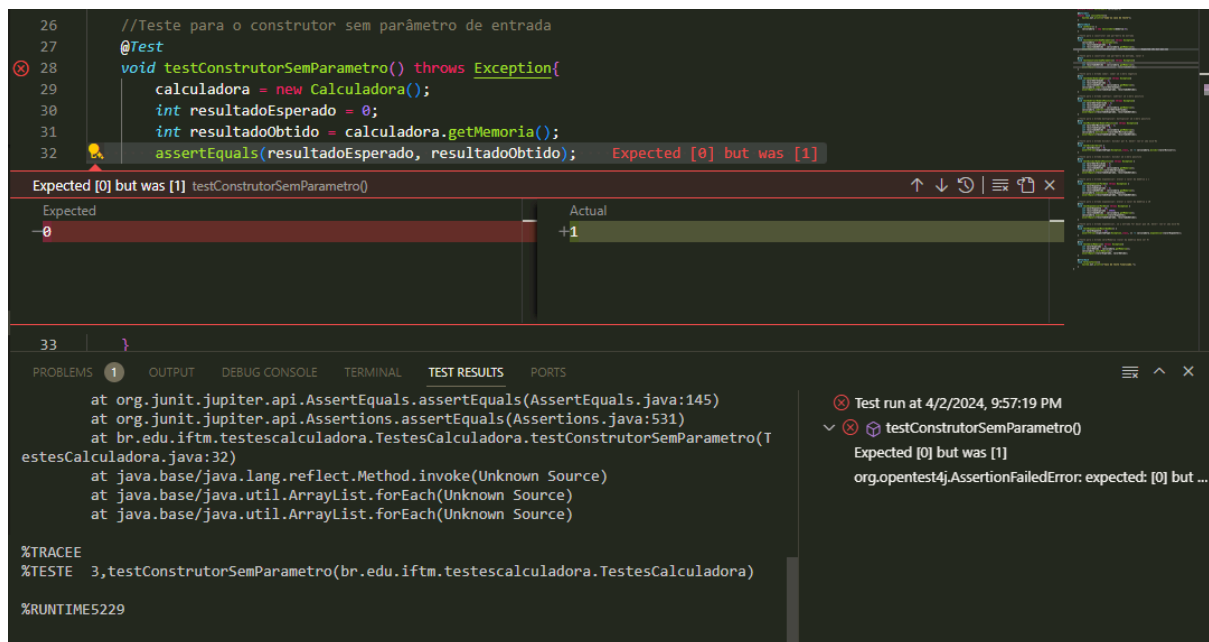
Implementação de testes para validar a classe Calculadora.

Aluna: Ana Clara Custodio

Relatório Casos de Teste

- 1) Implemente um teste para o construtor sem parâmetro de entrada (verificar se o valor da memória é 0).

Método a ser testado	Calculadora()
Cenário de teste (entradas)	Construtor sem parâmetro
Resultado esperado	0
Resultado obtido	1



```
26 //Teste para o construtor sem parâmetro de entrada
27 @Test
28 void testConstrutorSemParametro() throws Exception{
29     calculadora = new Calculadora();
30     int resultadoEsperado = 0;
31     int resultadoObtido = calculadora.getMemoria();
32     assertEquals(resultadoEsperado, resultadoObtido); Expected [0] but was [1]
33 }
```

Expected [0] but was [1] testConstrutorSemParametro()

Expected	Actual
-0	+1

Test run at 4/2/2024, 9:57:19 PM

testConstrutorSemParametro()

Expected [0] but was [1]

org.opentest4j.AssertionFailedError: expected: [0] but ...

at org.junit.jupiter.api.Assertions.assertEquals(Assertions.java:145)

at org.junit.jupiter.api.Assertions.assertEquals(Assertions.java:531)

at br.edu.iftm.testescalculadora.TestesCalculadora.testConstrutorSemParametro(TestesCalculadora.java:32)

at java.base/java.lang.reflect.Method.invoke(Unknown Source)

at java.base/java.util.ArrayList.forEach(Unknown Source)

at java.base/java.util.ArrayList.forEach(Unknown Source)

%TRACEE

%TESTE 3,testConstrutorSemParametro(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME5229

- 2) Implemente os testes para o construtor com parâmetro: recebendo o valor 3.

Método a ser testado	Calculadora(int)
Cenário de teste (entradas)	Construtor com parâmetro, memoria = 3
Resultado esperado	3
Resultado obtido	3

```

35 //Teste para o construtor com parâmetro de entrada, valor 3
36 @Test
37 void testConstrutorComParametro() throws Exception{
38     int resultadoEsperado = 3;
39     int resultadoObtido = calculadora.getMemoria();
40     assertEquals(resultadoEsperado, resultadoObtido);
41 }
42

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```

%TESTC 1 v2
%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculad
ora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testConstrutorComParametro(br.edu.iftm.testescalculadora.TestesCalculadora),f
alse,1,false,2,testConstrutorComParametro(),,[engine:junit-jupiter]/[class:br.edu.iftm.
testescalculadora.TestesCalculadora]/[method:testConstrutorComParametro()]
%TESTS 3,testConstrutorComParametro(br.edu.iftm.testescalculadora.TestesCalculadora)

%TESTE 3,testConstrutorComParametro(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME5565

```

Test run at 4/2/2024, 9:58:04 PM
testConstrutorComParametro()

3) Implemente os testes do método somar: somar um número negativo.

Método a ser testado	somar(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorASerSomado = -2
Resultado esperado	1
Resultado obtido	1

```

43 //Teste para o método somar: somar um número negativo
44 @Test
45 void testSomarNumeroNegativo() throws Exception{
46     int valorASerSomado = -2;
47     int resultadoEsperado = 1;
48     calculadora.somar(valorASerSomado);
49     int resultadoObtido = calculadora.getMemoria();
50     assertEquals(resultadoEsperado, resultadoObtido);
51 }

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```

%TESTC 1 v2
%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculad
ora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testSomarNumeroNegativo(br.edu.iftm.testescalculadora.TestesCalculadora),false
,1,false,2,testSomarNumeroNegativo(),,[engine:junit-jupiter]/[class:br.edu.iftm.testesca
lculadora.TestesCalculadora]/[method:testSomarNumeroNegativo()]
%TESTS 3,testSomarNumeroNegativo(br.edu.iftm.testescalculadora.TestesCalculadora)

%TESTE 3,testSomarNumeroNegativo(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME4456

```

Test run at 4/2/2024, 10:00:42 PM
testSomarNumeroNegativo()

4) Implemente os testes do método subtrair: subtrair um número positivo.

Método a ser testado	subtrair(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorASerSubtraído = 2
Resultado esperado	1
Resultado obtido	3

```

53 //Teste para o método subtrair: subtrair um número positivo
54 @Test
55 void testSubtrairNumeroPositivo() throws Exception{
56     int valorASerSubtraido = 2;
57     int resultadoEsperado = 1;
58     calculadora.subtrair(valorASerSubtraido);
59     int resultadoObtido = calculadora.getMemoria();
60     assertEquals(resultadoEsperado, resultadoObtido); Expected [1] but was [3]
61 }

```

Expected [1] but was [3] testSubtrairNumeroPositivo()

Expected	Actual
-1	+3

61 }

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

at br.edu.iftm.testescalculadora.TestesCalculadora.testSubtrairNumeroPositivo(TestesCalculadora.java:60)
at java.base/java.lang.reflect.Method.invoke(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)

%TRACEE
%TESTE 3,testSubtrairNumeroPositivo(br.edu.iftm.testescalculadora.TestesCalculadora)
%RUNTIME4617

Test run at 4/2/2024, 10:03:29 PM
testSubtrairNumeroPositivo()
Expected [1] but was [3]
org.opentest4j.AssertionFailedError: expected: [1]...

5) Implemente os testes do método multiplicar: multiplicar um número positivo.

Método a ser testado	multiplicar(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorASerMultiplicado = 3
Resultado esperado	9
Resultado obtido	1

```

63 //Teste para o método multiplicar: multiplicar um número positivo
64 @Test
65 void testMultiplicarNumeroPositivo() throws Exception{
66     int valorASerMultiplicado = 3;
67     int resultadoEsperado = 9;
68     calculadora.multiplicar(valorASerMultiplicado);
69     int resultadoObtido = calculadora.getMemoria();
70     assertEquals(resultadoEsperado, resultadoObtido); Expected [9] but was [1]
71 }

```

Expected [9] but was [1] testMultiplicarNumeroPositivo()

Expected	Actual
-9	+1

71 }

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

at br.edu.iftm.testescalculadora.TestesCalculadora.testMultiplicarNumeroPositivo(TestesCalculadora.java:70)
at java.base/java.lang.reflect.Method.invoke(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)

%TRACEE
%TESTE 3,testMultiplicarNumeroPositivo(br.edu.iftm.testescalculadora.TestesCalculadora)
%RUNTIME5007

Test run at 4/2/2024, 10:04:27 PM
testMultiplicarNumeroPositivo()
Expected [9] but was [1]
org.opentest4j.AssertionFailedError: expected: [9]...

6) Implemente os testes do método dividir: dividir por valor 0 e dividir por um valor positivo. A divisão por zero deverá retornar uma exception.

a) Divisão por 0:

Método a ser testado	dividir(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorDivisor = 0
Resultado esperado	Exception
Resultado obtido	Exception

```

73 //Teste para o método dividir: dividir por 0, deverá lançar uma exceção
74 @Test
75 void testDividirZero() {
76     int valorDivisor = 0;
77     assertThrows(Exception.class, () -> calculadora.dividir(valorDivisor));
78 }
79

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```

%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculadora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testDividirZero(br.edu.iftm.testescalculadora.TestesCalculadora),false,1,false,2,testDividirZero(),,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]/[method:testDividirZero()]
%TESTS 3,testDividirZero(br.edu.iftm.testescalculadora.TestesCalculadora)
%TESTE 3,testDividirZero(br.edu.iftm.testescalculadora.TestesCalculadora)
%RUNTIME5977

```

Test run at 4/2/2024, 10:05:40 PM
testDividirZero()

b) Divisão por um número positivo:

Método a ser testado	dividir(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorASerDividido = 3
Resultado esperado	1
Resultado obtido	1

```

80 //Teste para o método dividir: dividir um número positivo
81 @Test
82 void testDividirNumeroPositivo() throws Exception {
83     int valorASerDividido = 3;
84     int resultadoEsperado = 1;
85     calculadora.dividir(valorASerDividido);
86     int resultadoObtido = calculadora.getMemoria();
87     assertEquals(resultadoEsperado, resultadoObtido);
88 }
89

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```

%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculadora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testDividirNumeroPositivo(br.edu.iftm.testescalculadora.TestesCalculadora),false,1,false,2,testDividirNumeroPositivo(),,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]/[method:testDividirNumeroPositivo()]
%TESTS 3,testDividirNumeroPositivo(br.edu.iftm.testescalculadora.TestesCalculadora)
%TESTE 3,testDividirNumeroPositivo(br.edu.iftm.testescalculadora.TestesCalculadora)
%RUNTIME3607

```

Test run at 4/2/2024, 10:06:48 PM
testDividirNumeroPositivo()

- 7) Implemente os testes do método exponenciação: exponenciar a memória por 1 e por 10. Se a entrada for um valor maior que 10 deverá retornar uma exception.

a) Memória elevada a 1:

Método a ser testado	exponenciar(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorExpoente = 1
Resultado esperado	3
Resultado obtido	1995565057

```
90 //Teste para o método exponenciar: elevar o valor da memória a 1
91 @Test
92 void testExponenciarPorUm() throws Exception {
93     int valorExpoente = 1;
94     int resultadoEsperado = 3;
95     calculadora.exponenciar(valorExpoente);
96     int resultadoObtido = calculadora.getMemoria();
97     assertEquals(resultadoEsperado, resultadoObtido); // Expected [3] but was [1995565057]
```

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

Expected	Actual
-3	+1995565057

98 }

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

ra.java:97)

- at java.base/java.lang.reflect.Method.invoke(Unknown Source)
- at java.base/java.util.ArrayList.forEach(Unknown Source)
- at java.base/java.util.ArrayList.forEach(Unknown Source)

%TRACEE

%TESTE 3,testExponenciarPorUm(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME3836

Test run at 4/2/2024, 10:07:37 PM

- testExponenciarPorUm()

Expected [3] but was [1995565057]

org.opentest4j.AssertionFailedError: expecte..

b) Memória elevada a 10:

Método a ser testado	exponenciar(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorExpoente = 10
Resultado esperado	59049
Resultado obtido	1995565057

```
100 //Teste para o método exponenciar: elevar o valor da memória a 10
101 @Test
102 void testExponenciarPorDez() throws Exception {
103     int valorExpoente = 10;
104     int resultadoEsperado = 59049;
105     calculadora.exponenciar(valorExpoente);
106     int resultadoObtido = calculadora.getMemoria();
107     assertEquals(resultadoEsperado, resultadoObtido); Expected [59049] but was [1995565057]

```

Expected	Actual
-59049	+1995565057

```
108 }

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```
at org.junit.jupiter.api.Assertions.assertEquals(Assertions.java:531)
at br.edu.iftm.testescalculadora.TestesCalculadora.testExponenciarPorDez(TestesCalculadora.java:107)
at java.base/java.lang.reflect.Method.invoke(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)
at java.base/java.util.ArrayList.forEach(Unknown Source)

%TRACEE
%TESTE 3,testExponenciarPorDez(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME4267

```

Test run at 4...
testExp...
Expected [59...
org.opentest...

c) Memória elevada a um valor maior que 10:

Método a ser testado	exponenciar(int)
Cenário de teste (entradas)	Método com parâmetro, memoria = 3, valorExpoente = 12
Resultado esperado	Exception
Resultado obtido	Exception

```
110 //Teste para o método exponenciar, se a entrada for maior que 10, deverá lançar uma exceção
111 @Test
112 void testExponenciarMaiorQueDez() {
113     int valorExpoente = 12;
114     assertThrows(Exception.class, () -> calculadora.exponenciar(valorExpoente));
115 }
116

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

```
%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculadora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testExponenciarMaiorQueDez(br.edu.iftm.testescalculadora.TestesCalculadora),false,1,failure,2,testExponenciarMaiorQueDez(),,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]/[method:testExponenciarMaiorQueDez()]
%TESTS 3,testExponenciarMaiorQueDez(br.edu.iftm.testescalculadora.TestesCalculadora)

%TESTE 3,testExponenciarMaiorQueDez(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME4113

```

Test run at 4/2/2024, 10:09:06 PM
testExponenciarMaiorQueDez()

8) Implemente os testes para o método zerarMemória (verificar se o valor da memória voltou a ser 0).

Método a ser testado	zerarMemoria()
Cenário de teste (entradas)	Método sem parâmetro, memoria = 3
Resultado esperado	0
Resultado obtido	0

117 //Teste para o método zerarMemoria (valor da memória deve ser 0)
118 @Test
119 void testZerarMemoria() throws Exception{
120 int valorEsperado = 0;
121 calculadora.zerarMemoria();
122 int valorObtido = calculadora.getMemoria();
123 assertEquals(valorEsperado, valorObtido);
124 }

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

%TSTTREE2,br.edu.iftm.testescalculadora.TestesCalculadora,true,1,false,1,TestesCalculadora,,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]
%TSTTREE3,testZerarMemoria(br.edu.iftm.testescalculadora.TestesCalculadora),false,1,false,2,testZerarMemoria(),,[engine:junit-jupiter]/[class:br.edu.iftm.testescalculadora.TestesCalculadora]/[method:testZerarMemoria()]
%TESTS 3,testZerarMemoria(br.edu.iftm.testescalculadora.TestesCalculadora)

%TESTE 3,testZerarMemoria(br.edu.iftm.testescalculadora.TestesCalculadora)

%RUNTIME4173

Test run at 4/2/2024, 10:10:08 PM
testZerarMemoria()