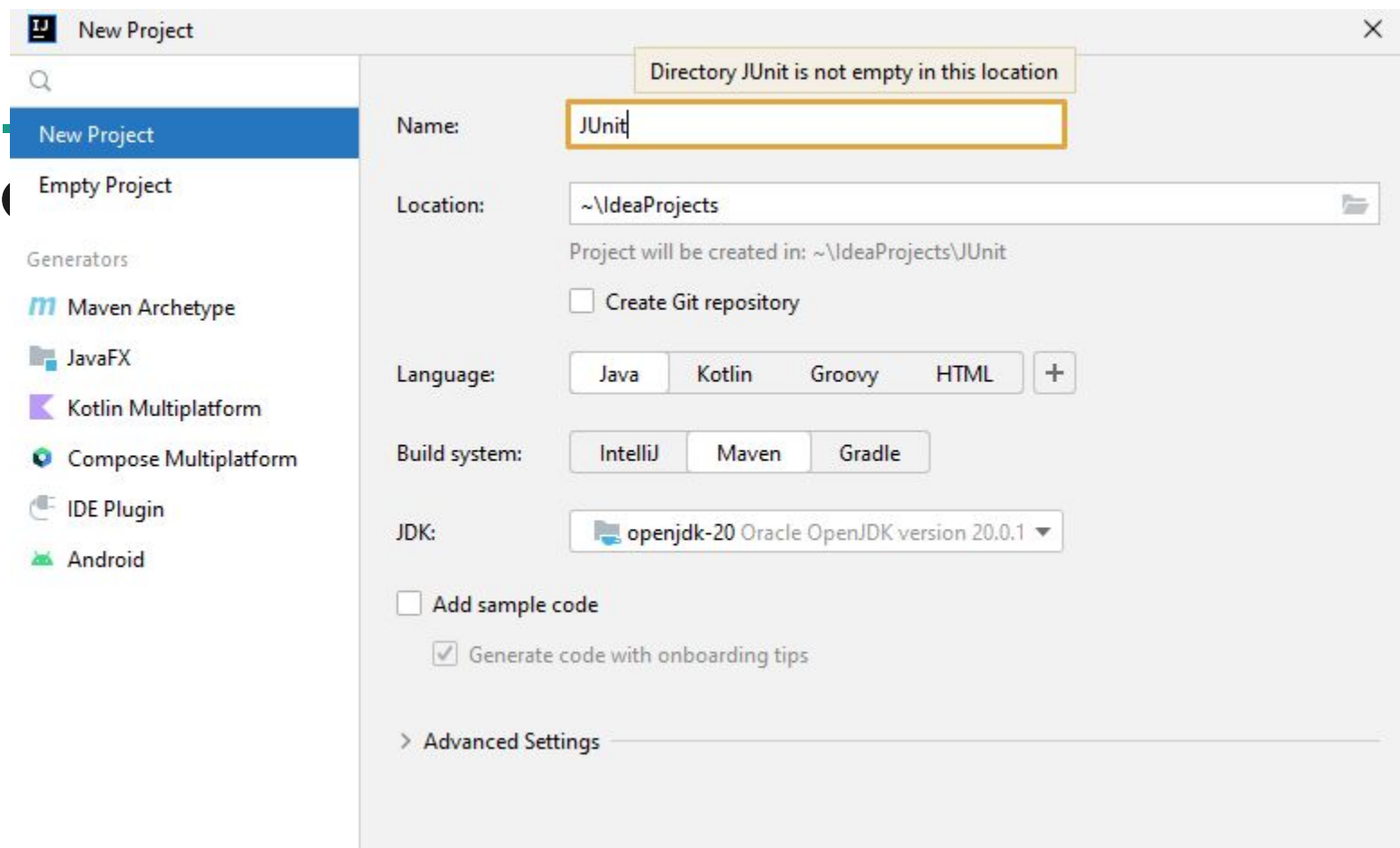




Teste Unitário

Junit

Herysson R. Figueiredo
herysson.figueiredo@ufn.edu.br





Dependências

Localize o pom.xml e adicione a dependência ao lado.

```
<dependencies>
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter</artifactId>
    <version>5.10.0</version>
  </dependency>
</dependencies>
```



Classe

Crie uma classe que realize a soma de 2 valores inteiros e retorne o resultado

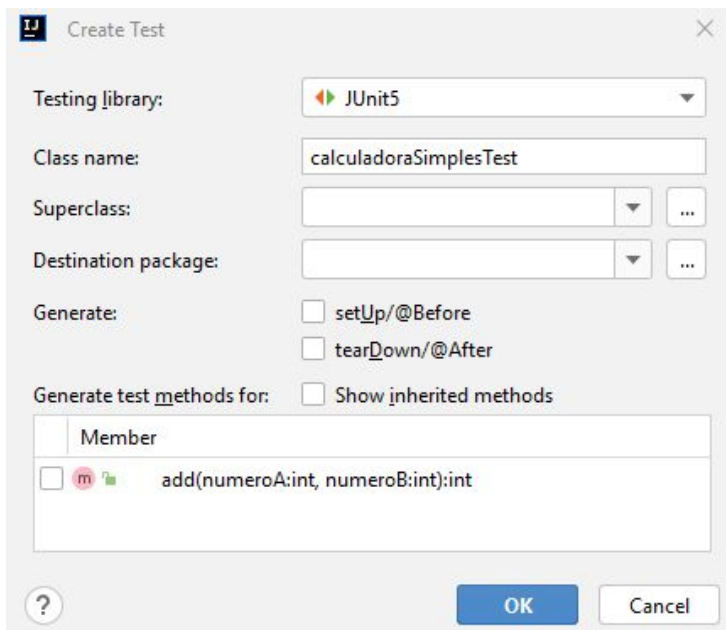
```
1 1 usage  
1 public class CalculadoraSimples {  
2  
3 1 usage  
3 public int add (int numeroA, int numeroB){  
4 1 usage  
4     return numeroA+numeroB;  
5 }  
6 }
```

Classe de Teste

Pressione **ctrl+shift+t** para criação de um teste



Classe de Teste



Create Test

Testing library: JUnit5

Class name: calculadoraSimplesTest

Superclass: ...

Destination package: ...

Generate:

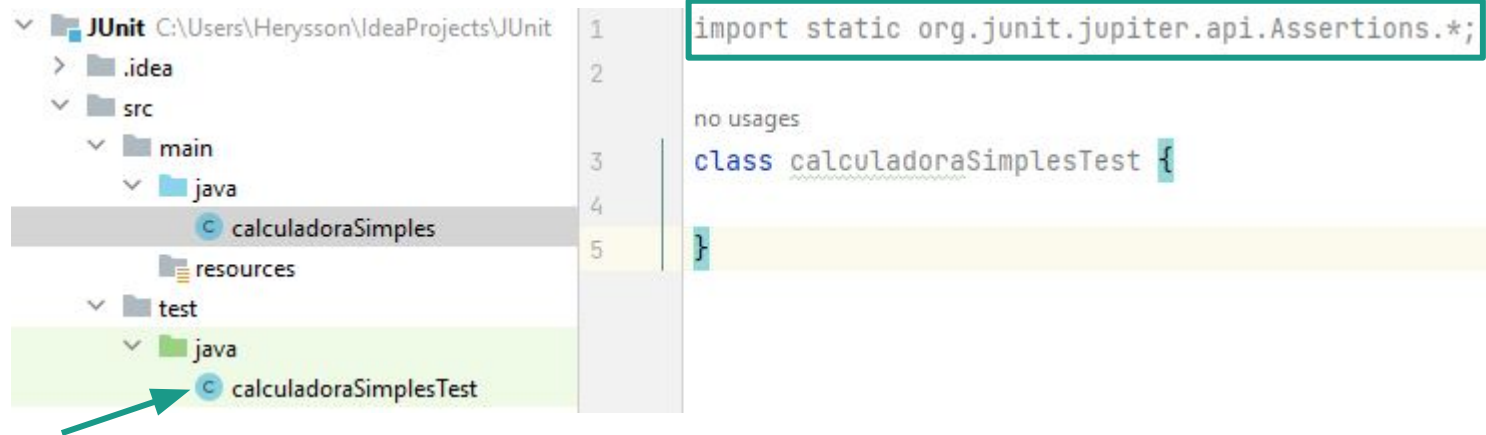
- ☐ setUp/@Before
- ☐ tearDown/@After

Generate test methods for: ☐ Show inherited methods

	Member
<input type="checkbox"/>	m add(numeroA:int, numeroB:int):int

? OK Cancel

Classe de Teste



The image displays a screenshot of an IDE interface, likely IntelliJ IDEA, showing a project structure and a code editor.

Project Structure (Left Panel):

- JUnit C:\Users\Herysson\IdeaProjects\JUnit
 - .idea
 - src
 - main
 - java
 - calculadoraSimples
 - resources
 - test
 - java
 - calculadoraSimplesTest

A green arrow points to the `calculadoraSimplesTest` class in the `test/java` package.

Code Editor (Right Panel):

The code editor shows the content of the `calculadoraSimplesTest` class. The first line, `import static org.junit.jupiter.api.Assertions.*;`, is highlighted with a green box. The class definition is `class calculadoraSimplesTest {`.

```
1 import static org.junit.jupiter.api.Assertions.*;
2
3 no usages
4 class calculadoraSimplesTest {
5 }
```

Classe de Teste

```
1  import org.junit.jupiter.api.Test;
2
3  import static org.junit.jupiter.api.Assertions.*;
4
5  class CalculadoraSimplesTest {
6      //cada teste deve testar somente uma unica coisa em um unico cenário
7      //o nome do teste deve descrever o cenário e o resultado
8      @Test
9      void doisMaisDoisIgualQuatro(){
10         //CalculadoraSimples calculadora = new CalculadoraSimples();
11         var calculadora = new CalculadoraSimples();
12         assertEquals( expected: 4, calculadora.add( numeroA: 2, numeroB: 2));
13     }
14 }
```


Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     // Run 'doisMaisDoisIgualQ...' Ctrl+Shift+F10
10    // Debug 'doisMaisDoisIgualQ...'
11    // Run 'doisMaisDoisIgualQ...' with Coverage
12    // Modify Run Configuration...
13    // Calculadora = new CalculadoraSimples();
14    // Calculadora.add(numeroA: 2, numeroB: 2);
15 }
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro() {
10         CalculadoraSimples calculadora = new CalculadoraSimples();
11         assertEquals(4, calculadora.somar(2, 2));
12     }
13 }
14
```

Run 'doisMaisDoisIgualQuatro()' Ctrl+Shift+F10

Run: CalculadoraSimplesTest.doisMaisDoisIgualQuatro

Tests passed: 1 of 1 test - 13 ms

CalculadoraSimplesTest 13 ms

doisMaisDoisIgualQuatro() 13 ms

C:\Users\Herysson\.jdk\openjdk-20.0.1\bin\java.exe ...

Process finished with exit code 0



Classe

Alterando a classe CalculadoraSimples

```
1 public class CalculadoraSimples {  
2  
3     1 usage  
4     public int add (int numeroA, int numeroB){  
5         return numeroA-numeroB;  
6     }  
}
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro() {
10         CalculadoraSimples calculadora = new CalculadoraSimples();
11         assertEquals(4, calculadora.soma(2, 2));
12     }
13 }
14
```

Run 'doisMaisDoisIgualQuatro()' Ctrl+Shift+F10

Debug 'doisMaisDoisIgualQuatro()' Run: CalculadoraSimplesTest.doisMaisDoisIgualQuatro

Run 'doisMaisDoisIgualQuatro()' Modify Run

Tests failed: 1 of 1 test – 25 ms

Test Name	Duration
CalculadoraSimplesTest	25 ms
doisMaisDoisIgualQuatro()	25 ms

C:\Users\Herysson\.jdk\openjdk-20.0.1\bin\java.exe ...

org.opentest4j.AssertionFailedError:
Expected :4
Actual :0
<Click to see differences>



Classe

Alterando a classe CalculadoraSimples

```
1 public class CalculadoraSimples {  
2  
3     2 usages  
    public int add (int numeroA, int numeroB){  
4        return numeroA*numeroB;  
5    }  
6 }  
7 |
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro(){
10         var calculadora = new CalculadoraSimples();
11         assertTrue( condition: calculadora.add( numeroA: 2, numeroB: 2)==4);
12     }
13
14     @Test
15     void tresMaisSeteIgualDez(){
16         var calculadora = new CalculadoraSimples();
17         assertEquals( expected: 10, calculadora.add( numeroA: 3, numeroB: 7));
18     }
19 }
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5
6
7
8
9
10 var calculadora = new CalculadoraSimples();
11 assertTrue( condition: calculadora.add( numeroA: 2, numeroB: 2)==4);
12
13 }
14 @Test
15 void tresMaisSeteIgualDez(){
16     var calculadora = new CalculadoraSimples();
17     assertEquals( expected: 10, calculadora.add( numeroA: 3, numeroB: 7));
18 }
19 }
```

Run 'CalculadoraSimplesTest' Ctrl+Shift+F10

Debug 'CalculadoraSimplesTest'

Run 'CalculadoraSimplesTest' with Coverage

Modify Run Configuration...

*coisa em um unico cenário
e o resultado*

Classe de Teste

```
1  import org.junit.jupiter.api.Test;
2
3  import static org.junit.jupiter.api.Assertions.*;
4
5  class CalculadoraSimplesTest {
6      //cada teste deve testar somente uma unica coisa em um unico cenário
7      //o nome do teste deve descrever o cenário e o resultado
8      @Test
9      void doisMaisDoisIgualQuatro(){
10         var calculadora = new CalculadoraSimples();
11         assertTrue( condition: calculadora.add( numeroA: 2, numeroB: 2)==4);
12     }
13
14     @Test
15     void tresMaisSeteIgualDez(){
16         var calculadora = new CalculadoraSimples();
17         assertEquals( expected: 10, calculadora.add( numeroA: 3, numeroB: 7));
18     }
19 }
```


Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro(){
10         var calculadora = new CalculadoraSimples();
```

Run: CalculadoraSimplesTest

Tests failed: 1, passed: 1 of 2 tests – 23 ms

Test Name	Status	Duration
doisMaisDoisIgualQuatro()	Passed	17 ms
tresMaisSeteIgualDez()	Failed	6 ms

org.opentest4j.AssertionFailedError:
Expected :10
Actual :21
[<Click to see difference>](#)



Classe

Alterando a classe CalculadoraSimples

```
1  public class CalculadoraSimples {  
2  
3      public int add (int numeroA, int numeroB){  
4          return numeroA+numeroB;  
5      }  
6  }  
7
```

2 usages

2 usages

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro(){
10         var calculadora = new CalculadoraSimples();
11         assertTrue( condition: calculadora.add( numeroA: 2, numeroB: 2)==4);
12     }
13
14     @Test
15     void tresMaisSeteIgualDez(){
16         var calculadora = new CalculadoraSimples();
17         assertEquals( expected: 10, calculadora.add( numeroA: 3, numeroB: 7));
18     }
19 }
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class CalculadoraSimplesTest {
6     //cada teste deve testar somente uma unica coisa em um unico cenário
7     //o nome do teste deve descrever o cenário e o resultado
8     @Test
9     void doisMaisDoisIgualQuatro() {
10         var calculadora = new CalculadoraSimples();
11         assertTrue(condition: calculadora.add(2, 2) == 4);
12     }
13
14     // ...
15
16     // ...
17
18 }
19 }
```

Run: CalculadoraSimplesTest x

✓ Tests passed: 2 of 2 tests – 16 ms

Test Name	Duration
✓ CalculadoraSimplesTest	16 ms
✓ doisMaisDoisIgualQuatro()	15 ms
✓ tresMaisSeteIgualDez()	1 ms

C:\Users\Herysson\.jdk\openjdk-20.0.1\bin\java.exe ...

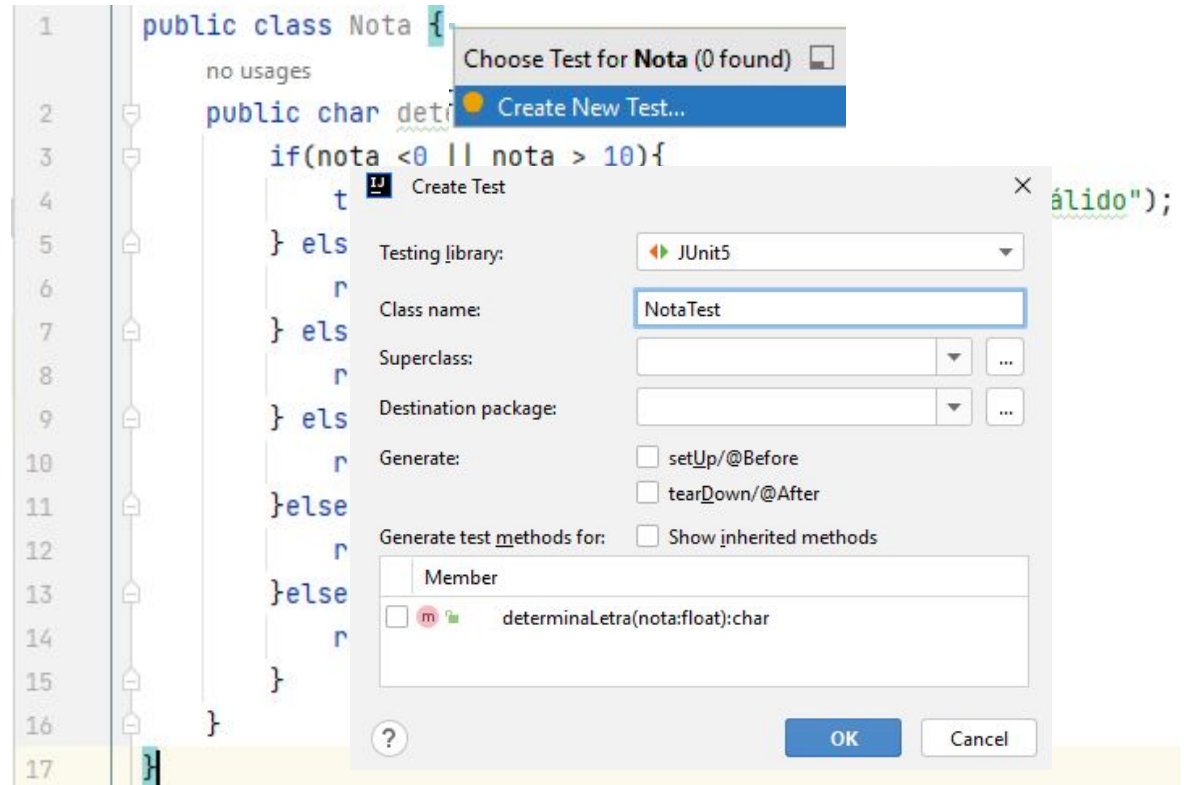
Process finished with exit code 0

Classe : Nota

```
1 public class Nota {  
    no usages  
2     public char determinaLetra(float nota){  
3         if(nota < 0 || nota > 10){  
4             throw new IllegalArgumentException("Valor inválido");  
5         } else if (nota < 60){  
6             return 'F';  
7         } else if (nota < 70) {  
8             return 'D';  
9         } else if (nota < 80) {  
10            return 'C';  
11        } else if (nota < 90){  
12            return 'B';  
13        } else {  
14            return 'A';  
15        }  
16    }  
17 }
```

Classe : Nota

Ctrl+Shift+t



Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class NotaTest {
6     @Test
7     void cinquentaENoveRetornaF(){
```

Run: NotaTest x

Tests failed: 1 of 1 test – 17 ms

Test Name	Duration
NotaTest	17 ms
cinquentaENoveRetornaF()	17 ms

java.lang.IllegalArgumentException: Valor inválido

at Nota.determinaLetra(Nota.java:4)
at NotaTest.cinquentaENoveRetornaF(NotaTest.java:9) <1 internal line>
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1511)

Classe : Nota

```
1 public class Nota {  
    no usages  
2     public char determinaLetra(float nota){  
3         if(nota < 0 || nota > 10){  
4             throw new IllegalArgumentException("Valor inválido");  
5         } else if (nota < 60){  
6             return 'F';  
7         } else if (nota < 70) {  
8             return 'D';  
9         } else if (nota < 80) {  
10            return 'C';  
11        } else if (nota < 90){  
12            return 'B';  
13        } else {  
14            return 'A';  
15        }  
16    }  
17 }
```


Classe : Nota

```
1 public class Nota {  
    1 usage  
2     public char determinaLetra(float nota){  
3         if(nota < 0 || nota > 100){  
4             throw new IllegalArgumentException("Valor inválido");  
5         } else if (nota < 60){  
6             return 'F';  
7         } else if (nota < 70) {  
8             return 'D';  
9         } else if (nota < 80) {  
10            return 'C';  
11        } else if (nota < 90){  
12            return 'B';  
13        } else {  
14            return 'A';  
15        }  
16    }  
17 }
```

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class NotaTest {
6     @Test
7     void cinquentaENoveRetornaF(){
```

Run: NotaTest x

✓ Tests passed: 1 of 1 test – 21 ms

Test Name	Duration
✓ NotaTest	21 ms
✓ cinquentaENoveRetornaF()	21 ms

C:\Users\Herysson\.jdk\openjdk-20.0.1\bin\java.exe ...

Process finished with exit code 0

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class NotaTest {
6
7     // ...
8     // ...
9     // ...
10 }
11
12 }
```

Run 'NotaTest' Ctrl+Shift+F10
Debug 'NotaTest'
Run 'NotaTest' with Coverage
Modify Run Configuration...

ornaF(){
a();
ted: 'F', nota.determinaLetra(nota: 59.8F));

Classe de Teste

```
1 import org.junit.jupiter.api.Test;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5 class NotaTest {
6
7
8
9
10 }
11
12 }
```

Run 'NotaTest' Coverage: NotaTest x

Debug 'NotaTest'

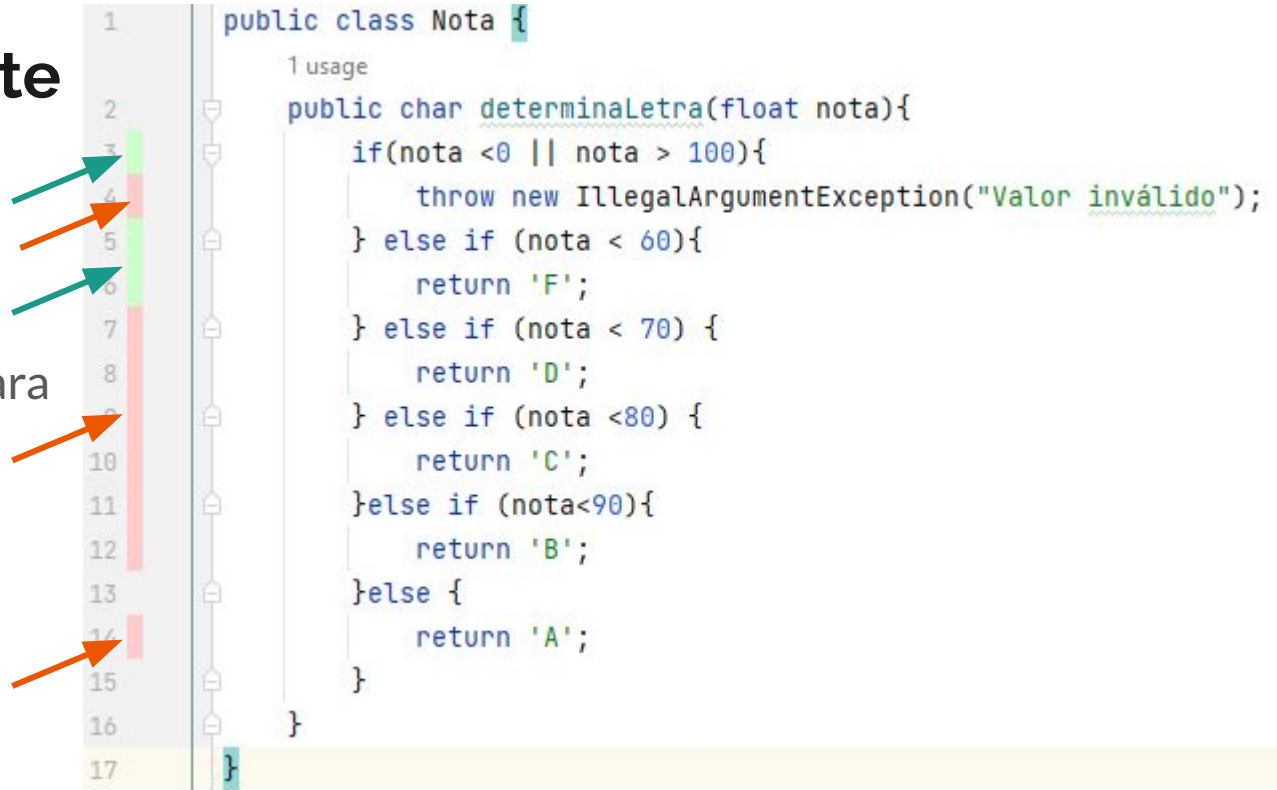
Run 'NotaTest'

Modify Run

Element	Class, %	Method, %	Line, %
all	50% (1/2)	50% (1/2)	30% (4/13)
CalculadoraSimples	0% (0/1)	0% (0/1)	0% (0/1)
Nota	100% (1/1)	100% (1/1)	33% (4/12)

Classe de Teste

Construa teste para
100% de cobertura.





Referências

BRAGA, Pedro Henrique Cacique. Teste de Software. Pearson Education do Brasil. São Paulo. 2016. Disponível na Biblioteca Virtual. DELAMARO, Márcio;

MALDONADO, José Carlos, JINO, Mario. Introdução ao teste de software. Elsevier. Rio de Janeiro. 2007.

PRESSMAN, Roger S. Engenharia de software. 5. Ed. Rio de Janeiro: McGraw Hill, 2002.