

Programação Orientada a Objetos

Laboratório

Testes de Software com JUnit

Prof. Dirson Santos de Campos
dirson_campos@ufg.br

07/12/2023



Instalação do JUnit 5.0 IDE Eclipse


Programação Orientada a Objetos

- **Ferramenta para Testes no IDE Eclipse**



 JUnit 4


The 5th major version of the programmer-friendly testing framework for Java and the JVM

 User Guide

 Javadoc

 Code & Issues

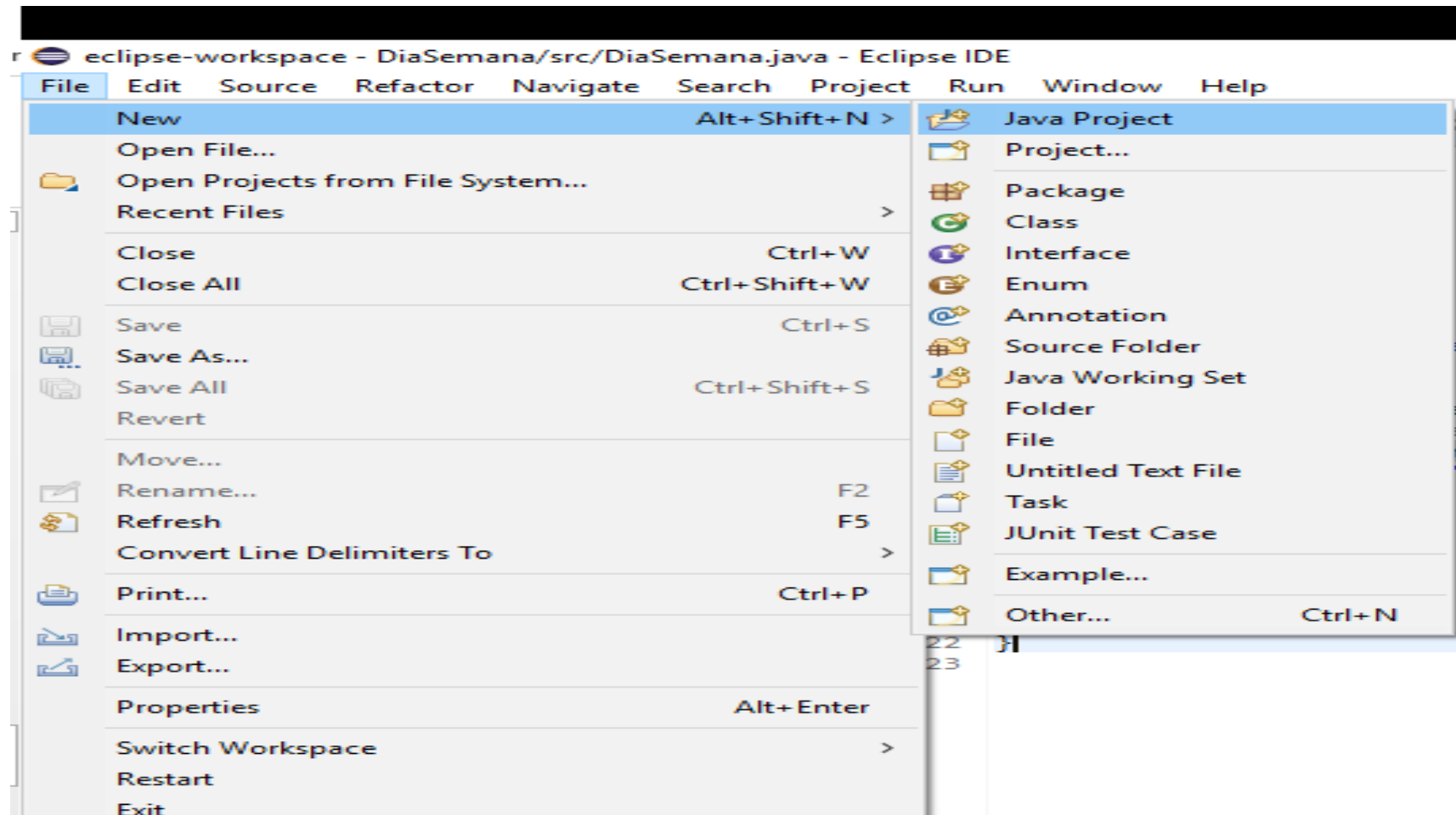
 Q & A

 Support JUnit

- **Fonte: <https://junit.org/junit5/>**

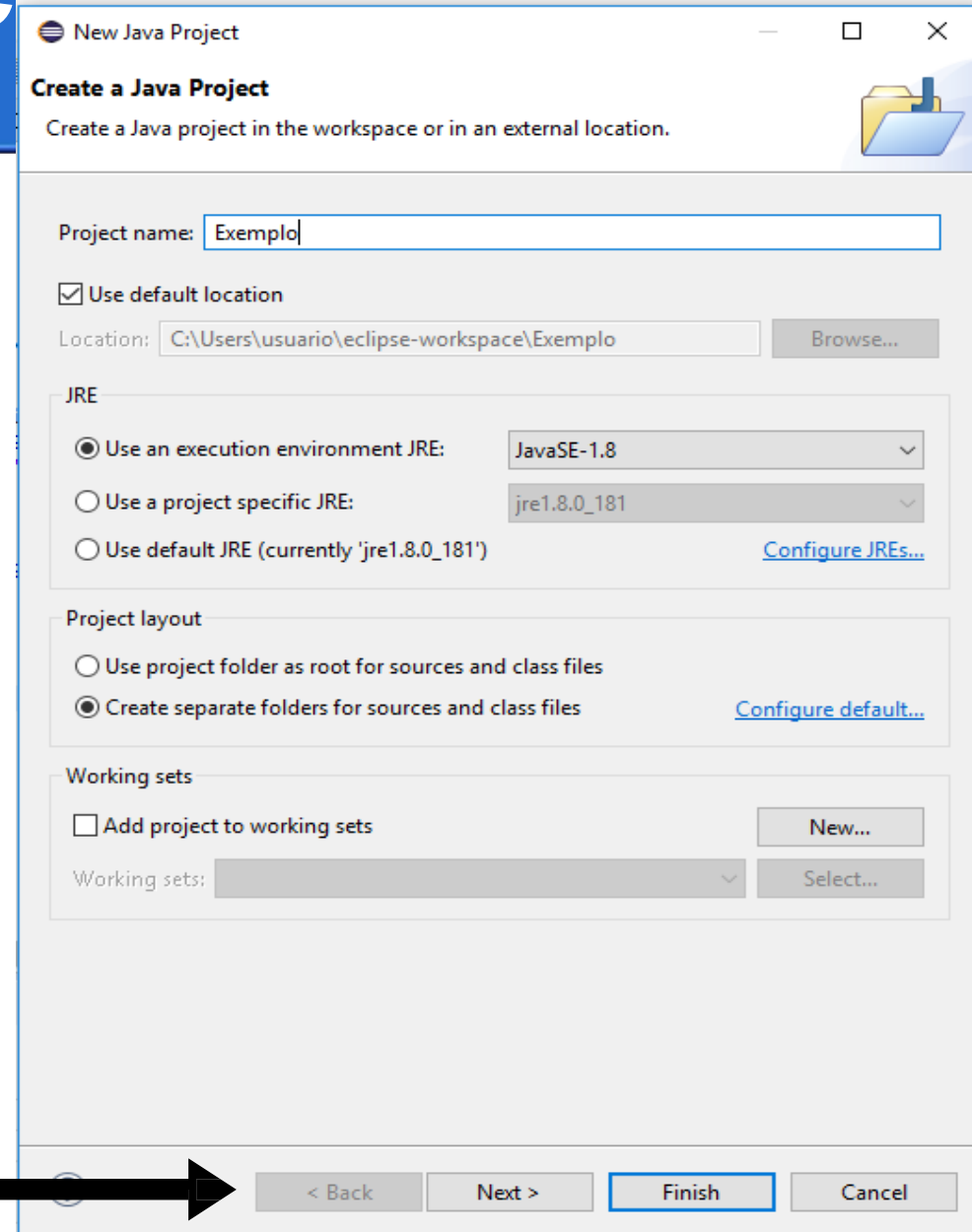
JUnit no Eclipse

- Crie um novo projeto na IDE Eclipse



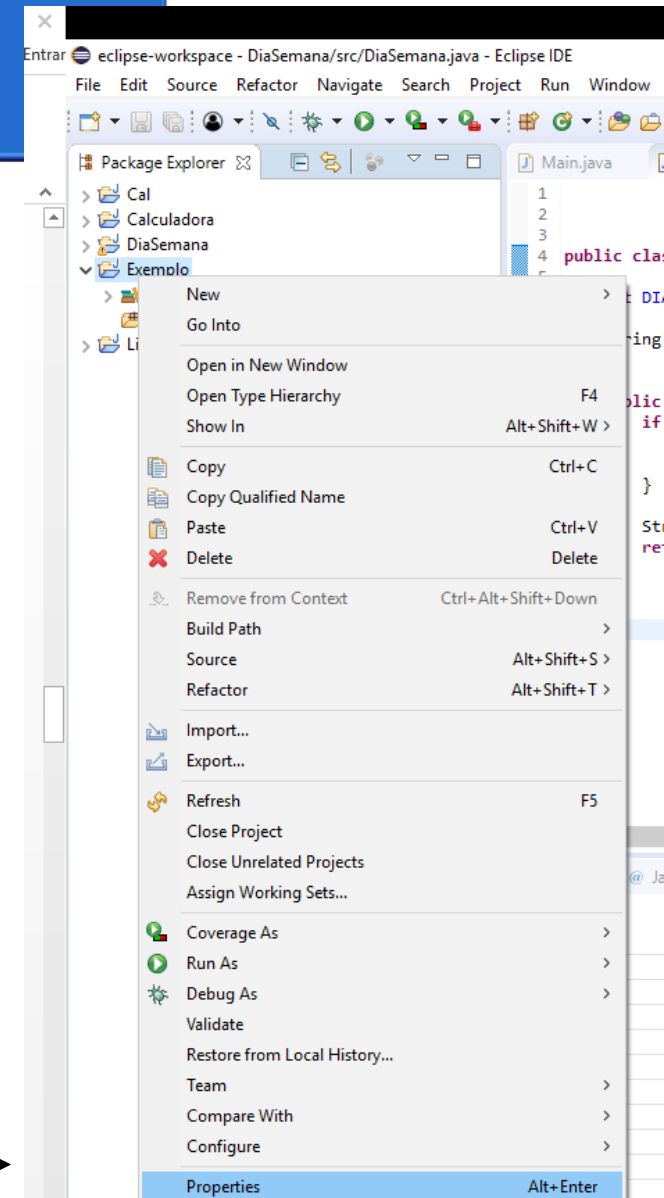
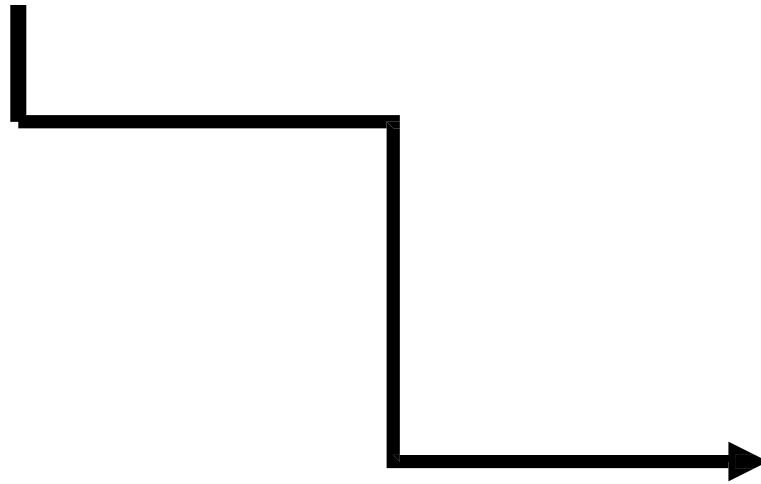
JUnit no Eclipse

- Adicione o nome ao projeto e clique em Finalizar



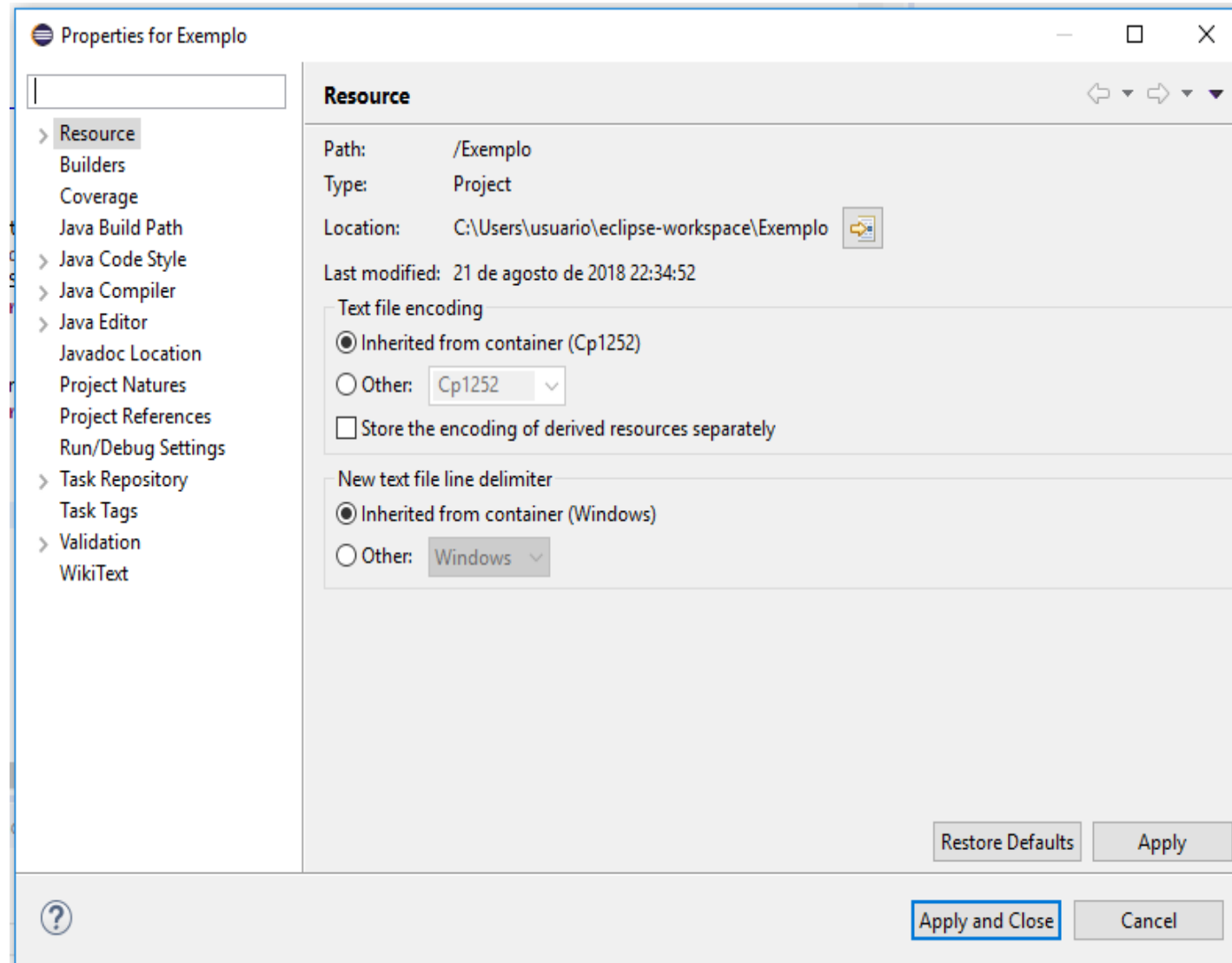
JUnit no Eclipse

- Após a criação:
 - Vá até a lista de pacotes
 - Selecione o projeto;
 - Clique com o botão direito;
 - No menu referente ao projeto, escolha propriedades



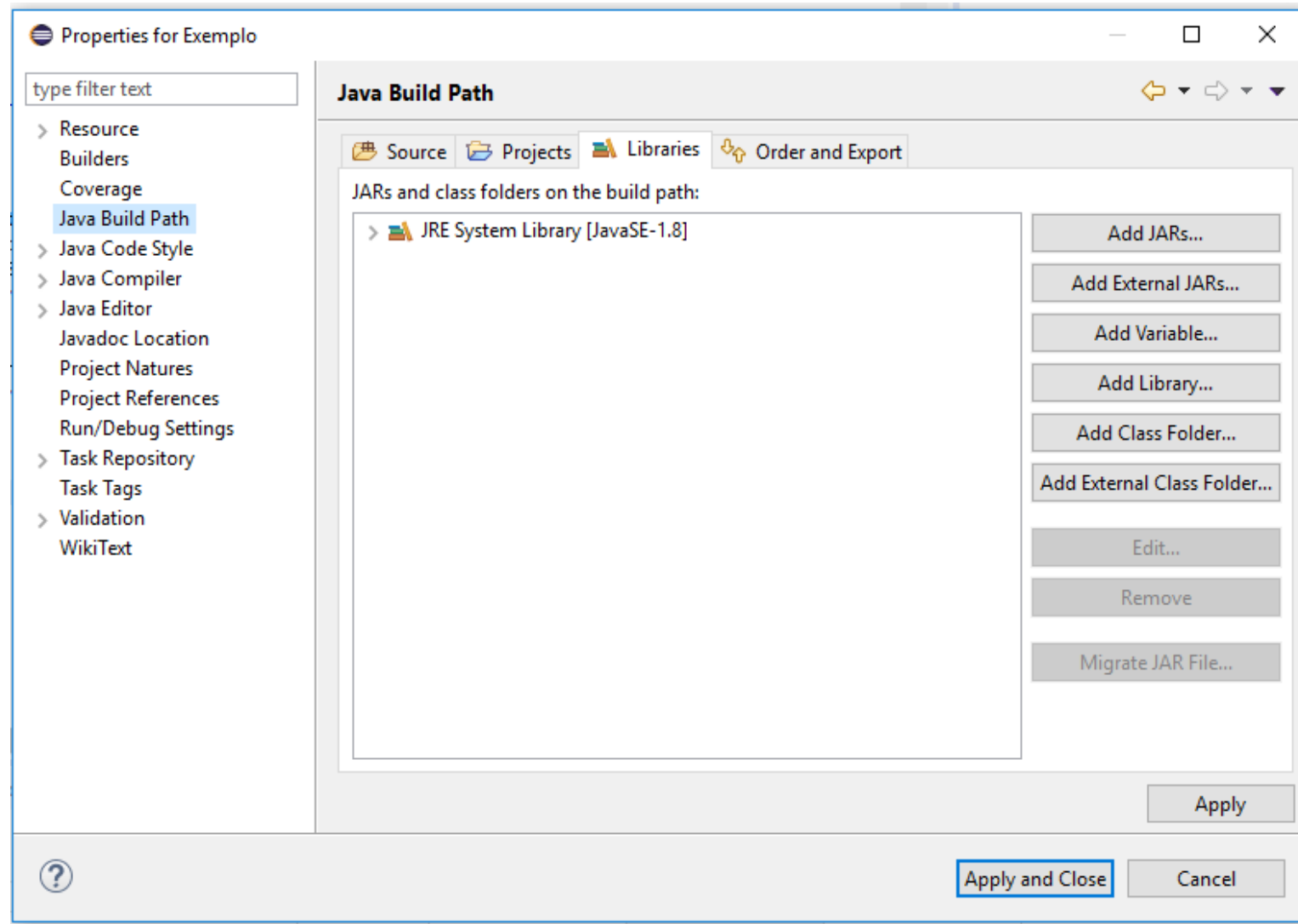
JUnit no Eclipse

- Na janela que será aberta, escolha a opção “Java Build Path”.



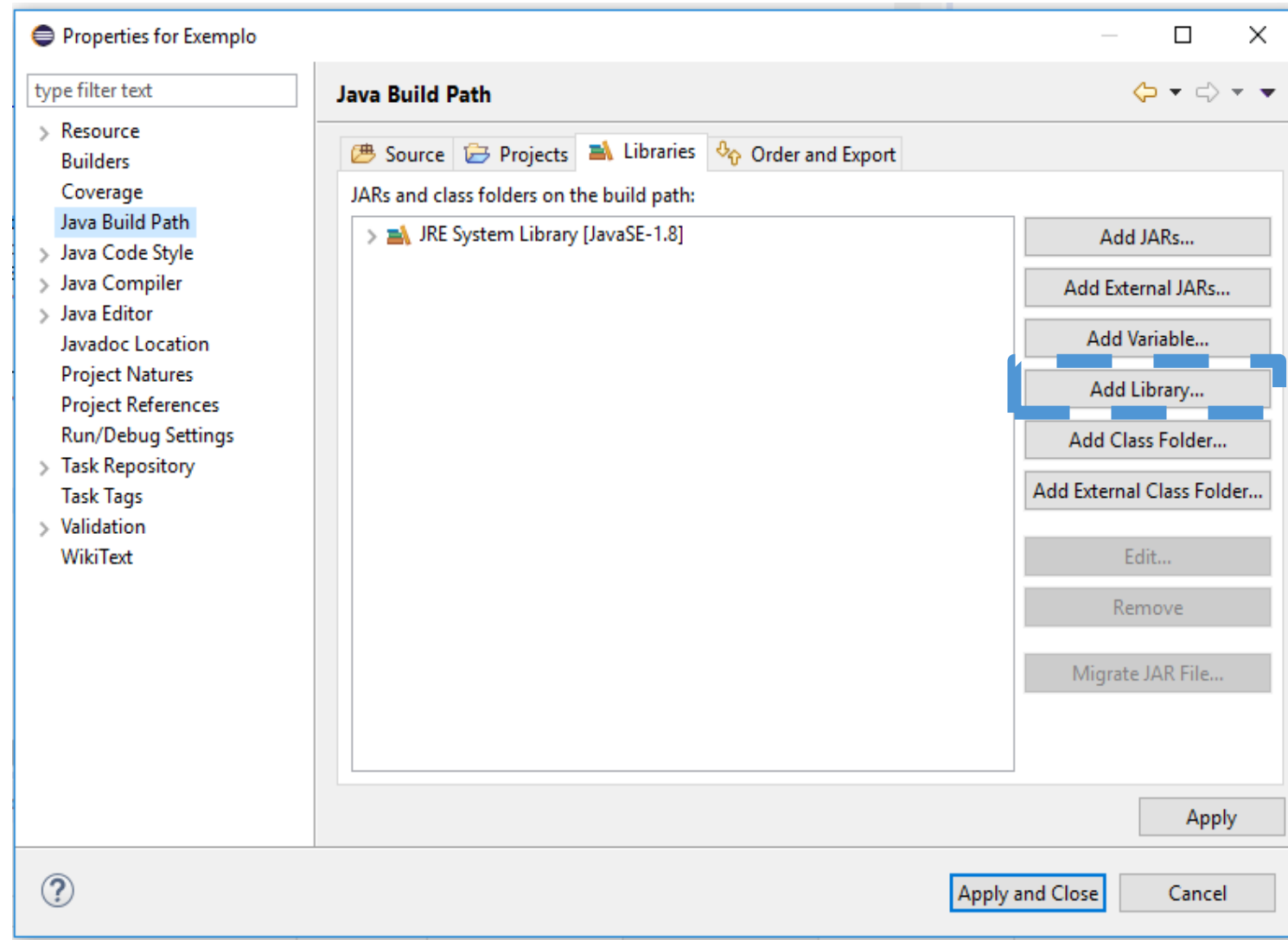
JUnit no Eclipse

- Observe as bibliotecas disponíveis para o projeto



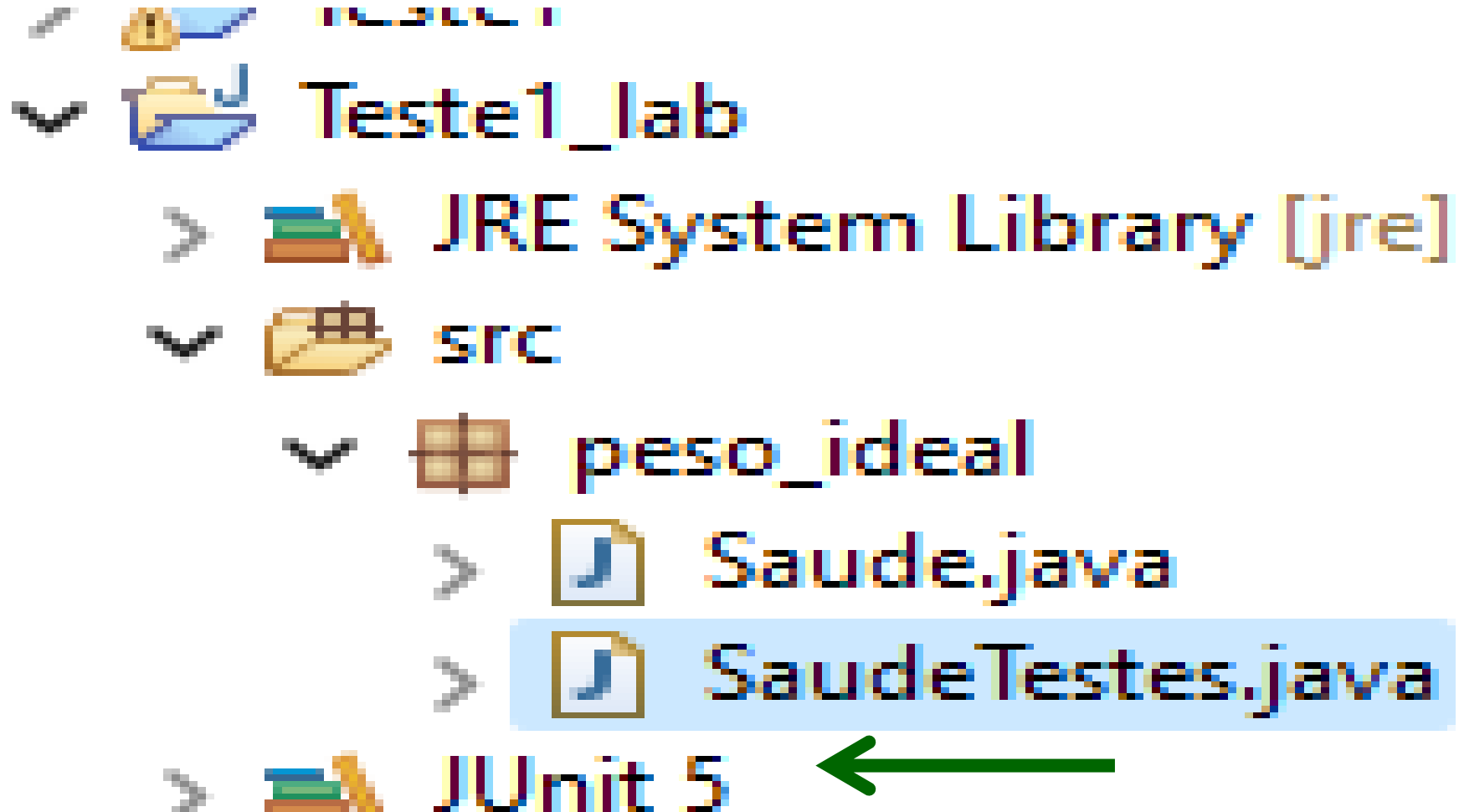
JUnit no Eclipse

- Clique no botão adicionar biblioteca



JUnit no Eclipse

- *A ferramenta Junit deve aparecer no IDE Eclipse do Projeto*





Instalação do JUnit 5.0 IDE Visual Studio Code

❑ Criando um projeto Java no VS Code

Ctrl + Shift + P

Java:Create Project

❑ Java Testing no VS Code

<https://code.visualstudio.com/docs/java/java-testing>

Programação Orientada a Objetos

. *Ferramenta para Testes no IDE VS Code*

The screenshot displays the Visual Studio Code interface with the 'Extension Pack for Java' by Microsoft selected. The left sidebar shows a list of extensions under the search term 'java'. The main panel shows the details of the 'Extension Pack for Java' extension, including its version (v0.25.11), download count (19,997,812), and a 4-star rating (59 reviews). Below the main extension details, there is a section for 'Pacote de Extensões (6)' which lists other extensions included in the pack: IntelliCode, Language Support for Java(TM) by Red Hat, and others. The right sidebar shows a list of categories including Programming Languages, Snippets, Linters, Debuggers, Formatters, and Extension Packs.

EXTENSÕES: MARKETP...

Extension Pac... 19.9M ★ 4
Popular extensions for Java ...
Microsoft [Instalar](#)

Debugger for... 22.8M ★ 4
A lightweight Java debugge...
Microsoft [Instalar](#)

Maven for Java 21.4M ★ 4
Manage Maven projects, ex...
Microsoft [Instalar](#)

Test Runner f... 20.5M ★ 4
Run and debug JUnit or Test...
Microsoft [Instalar](#)

Project Mana... 20.2M ★ 4
Manage Java projects in Vis...
Microsoft [Instalar](#)

Language S 25.2M ★ 3.5

Extensão: Extension Pack for Java - Django-3-by-Example - Visual Studio Code

Extensão: Extension Pack for Java ✕

Extension Pack for Java v0.25.11 [Visualizar](#)
Microsoft [microsoft.com](#) | 19.997.812 | ★★★★★ (59)
Popular extensions for Java development that provides Java IntelliSense, de

[Instalar](#) ⚙️

DETALHES CONTRIBUIÇÕES DE RECURSO LOG DE MUDANÇAS

Pacote de Extensões (6)

IntelliCode
AI-assisted development
Microsoft [Instalar](#)

Language Support for Java(TM) by Red Hat
Java Linting, Intellisense, formatting, refactor...
Red Hat [Instalar](#)

Categorias

- Programming Languages
- Snippets
- Linters
- Debuggers
- Formatters
- Extension Packs

Programação Orientada a Objetos

. *Ferramenta para Testes no IDE VS Code*

The screenshot displays the Visual Studio Code interface with the 'Extension Pack for Java' installed. The top bar shows the extension name and version (v0.25.11). The left sidebar lists the installed extensions: 'Extension Pack for Java', 'Debugger for Java', 'Maven for Java', 'Test Runner for Java', and 'Project Manager for Java'. The main panel shows the details of the 'Extension Pack for Java', including its description, version, and a list of sub-extensions. A white arrow points to the 'Test Runner for Java' sub-extension in the 'Pacote de Extensões (6)' section.

Extensão: Extension Pack for Java - Django-3-by-Example - Visual Studio Code

EXTENSÕES: MARKETPLACE

java

Extension Pack for Java v0.25.11 Visualizar
Microsoft | 19.997.812 | ★★★★★ (59)
Popular extensions for Java development that provides Java IntelliSense.

Desabilitar | Desinstalar | Alternar para a Versão de Pré-Lançamento

Esta extensão foi habilitada globalmente.

DETALHES | CONTRIBUIÇÕES DE RECURSO | LOG DE MUDANÇAS

Pacote de Extensões (6)

- Manage Maven projects, execute goals, gen... | Microsoft
- Test Runner for Java** | Microsoft
Run and debug JUnit or TestNG test cases.

Categorias

- Programming Languages
- Snippets
- Linters
- Debuggers
- Formatters
- Extension Packs

Programação Orientada a Objetos

. *Ferramenta para Testes no IDE Vs Code*

Testing Java with Visual Studio Code

Edit

Testing Java in Visual Studio Code is enabled by the [Test Runner for Java](#) extension. It's a lightweight extension to run and debug Java test cases.

Overview

The extension supports the following test frameworks:

- [JUnit 4](#) (v4.8.0+)
- [JUnit 5](#) (v5.1.0+)
- [TestNG](#) (v6.9.13.3+)



The [Test Runner for Java](#) works with the [Language Support for Java™](#) by Red Hat and [Debugger for Java](#) extensions to provide the following features:

Programação Orientada a Objetos

- **Anotações *JUnit***

- **@Test** – Identifica método que contem teste
- **@After** - Identifica método para ser executado após cada método de teste
- **@Before** - Identifica método para ser executado antes cada método de teste
- **@AfterClass** - Identifica método estático para ser executado após a execução de todos os métodos de teste da classe
- **@BeforeClass** - Identifica método estático para ser executado antes da execução de todos os métodos de teste da classe

Programação Orientada a Objetos

- ***JUnit*** - Testes Unitários

- Resultados Possíveis



- ***Sucesso***



- ***Falha***



- ***exceção***

Programação Orientada a Objetos

- **Exemplo 1**
 - Classe a ser testada : Saude
 - Métodos a serem testados :
 - **calculaIMC e condicaoFisica**

Programação Orientada a Objetos

Exemplo 1

```
package peso_ideal;

public class Saude {
    int idade; double altura; double peso;

    public Saude(int idade, double altura, double peso) {
        this.idade = idade;
        this.altura = altura;
        this.peso = peso;
    }

    public double calculaIMC() {
        return peso / (altura * altura);
    }

    public String condiçãoFísica() {
        if ( calculaIMC() < 18.5 )
            return "Abaixo do peso adequado";
        else if ( calculaIMC() <= 24.9 )
            return "Peso adequado";
        return "Acima do peso adequado";
    }
}
```

Programação Orientada a Objetos

Exemplo 1

```
package peso_ideal;

import org.junit.jupiter.api.Test;
import static org.junit.jupiter.api.Assertions.*;

public class SaudeTestes {
    @Test
    public void testeIMC() {
        Saude saude = new Saude(20, 1.64, 62);
        assertEquals( saude.calculaIMC(), 23, 0.5 );
    }
    @Test
    public void testeCondição() {
        Saude saúde = new Saude(20, 1.64, 62);
        assertTrue( saúde.condiçãoFísica().equals("Peso adequado") );
    }
}
```

Programação Orientada a Objetos

Exemplo1

- Simulando um erro no código-fonte

Package Explorer | Navigator | JUnit

Finished after 0.066 seconds

Runs: 2/2 | Errors: 0 | Failures: 1

▼ SaudeTest [Runner: JUnit 5] (0.003 s)

- testIMC() (0.000 s)
- testCondicao() (0.003 s)

Failure Trace

org.opentest4j.AssertionFailedError: expected: <true> but was: <false>
at SaudeTest.testCondicao(SaudeTest.java:10)
at java.util.ArrayList.forEach(ArrayList.java:1257)
at java.util.ArrayList.forEach(ArrayList.java:1257)

```
1 import static org.junit.jupiter.api.Assertions.*;
4
5 class SaudeTest {
6
7     @Test
8     void testCondicao() {
9         Saude saude = new Saude(20,1.64,62);
10        assertTrue(saude.condicaoFisica().equals("Peso Normal"));
11    }
12
13    @Test
14    void testIMC() {
15        Saude saude = new Saude(20,1.64,62);
16        assertEquals(saude.calculaIMC(),23,0.5);
17    }
18
19 }
20
```

Programação Orientada a Objetos

Exemplo 1 – Testando com JUnit

Verde: Código Executado
Amarelo: Ponto de decisão
Vermelho: Código não executado

```
1 package peso_ideal;
2
3 public class Saude {
4     int idade; double altura; double peso;
5
6     public Saude(int idade, double altura, double peso) {
7         this.idade = idade;
8         this.altura = altura;
9         this.peso = peso;
10    }
11
12    public double calculaIMC() {
13        return peso / (altura * altura);
14    }
15
16    public String condiçãoFísica() {
17        if ( calculaIMC() < 18.5 )
18            return "Abaixo do peso adequado";
19        else if ( calculaIMC() <= 24.9 )
20            return "Peso adequado";
21        return "Acima do peso adequado";
22    }
23 }
```

Coverage Report:

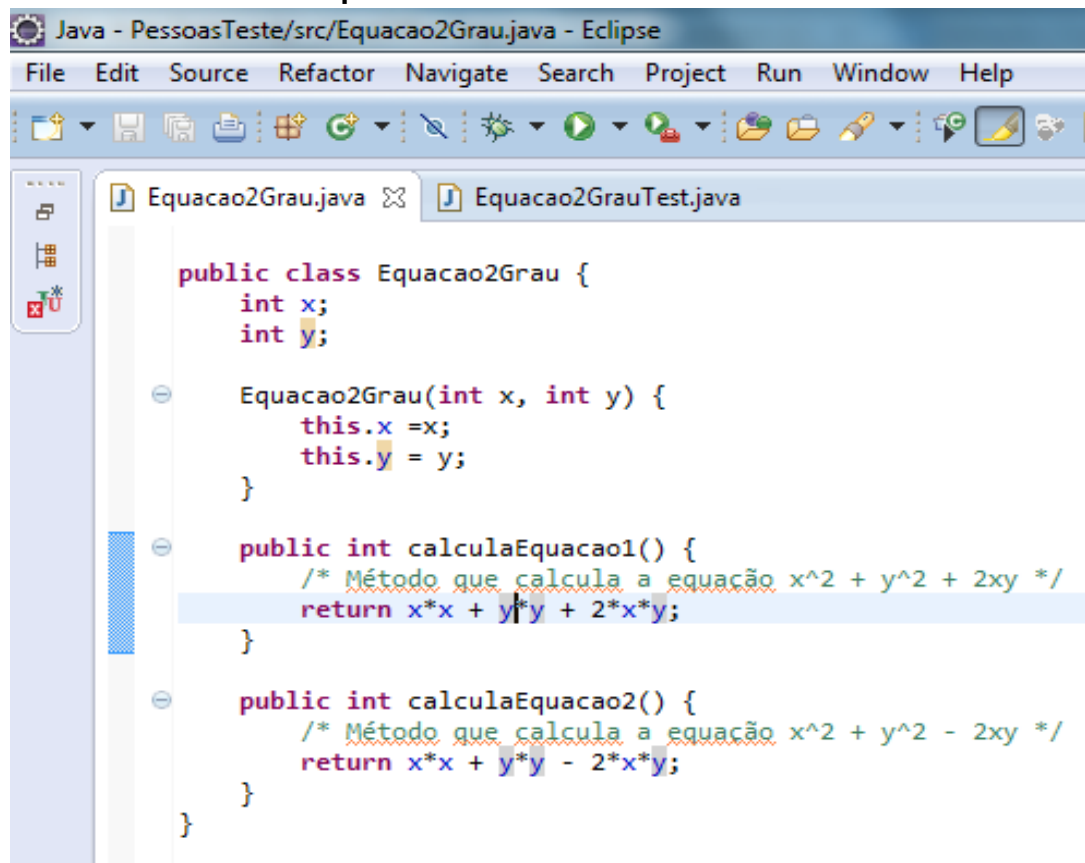
Element	Coverage	Cov
Teste1_lab	93,9 %	

O Junit fornece um percentual a cobertura dos casos de testes por execução de linhas do código fonte

Programação Orientada a Objetos

• Exemplo

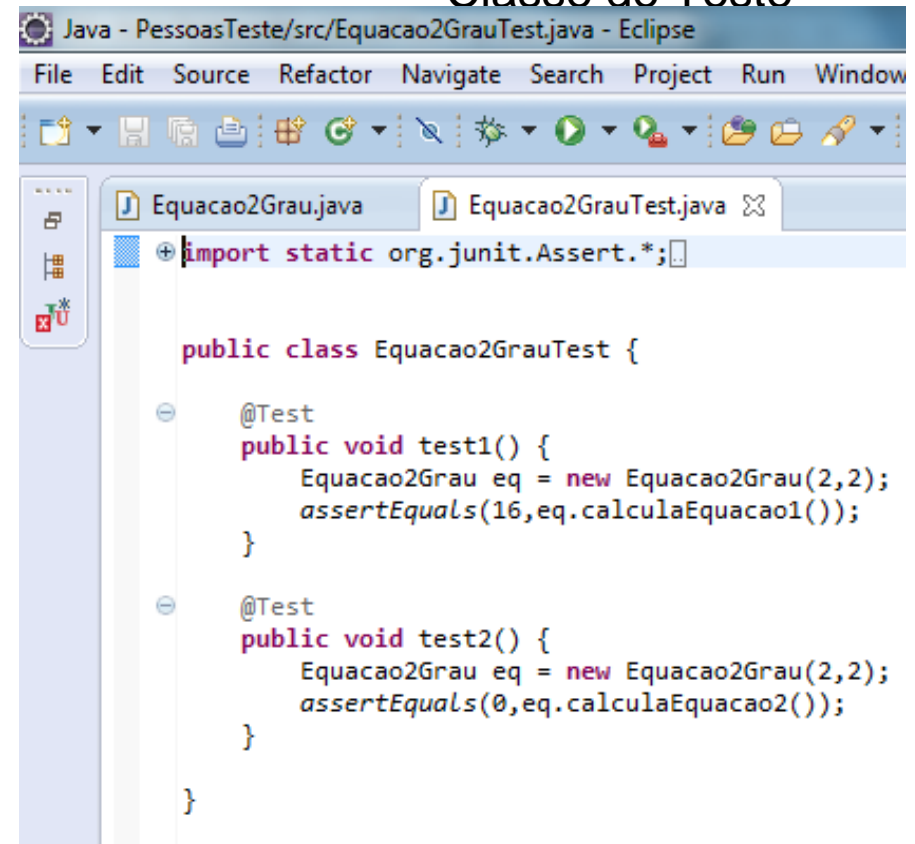
• Classe Equacao2Grau



The screenshot shows the Eclipse IDE with the file Equacao2Grau.java open. The code defines a public class Equacao2Grau with two integer attributes, x and y. It includes a constructor that initializes these attributes and two public methods: calculaEquacao1() and calculaEquacao2(). Both methods calculate quadratic expressions based on x and y. The first method calculates $x^2 + y^2 + 2xy$ and the second calculates $x^2 + y^2 - 2xy$. Comments in Portuguese describe the methods.

```
public class Equacao2Grau {  
    int x;  
    int y;  
  
    Equacao2Grau(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
  
    public int calculaEquacao1() {  
        /* Método que calcula a equação  $x^2 + y^2 + 2xy$  */  
        return x*x + y*y + 2*x*y;  
    }  
  
    public int calculaEquacao2() {  
        /* Método que calcula a equação  $x^2 + y^2 - 2xy$  */  
        return x*x + y*y - 2*x*y;  
    }  
}
```

Classe de Teste

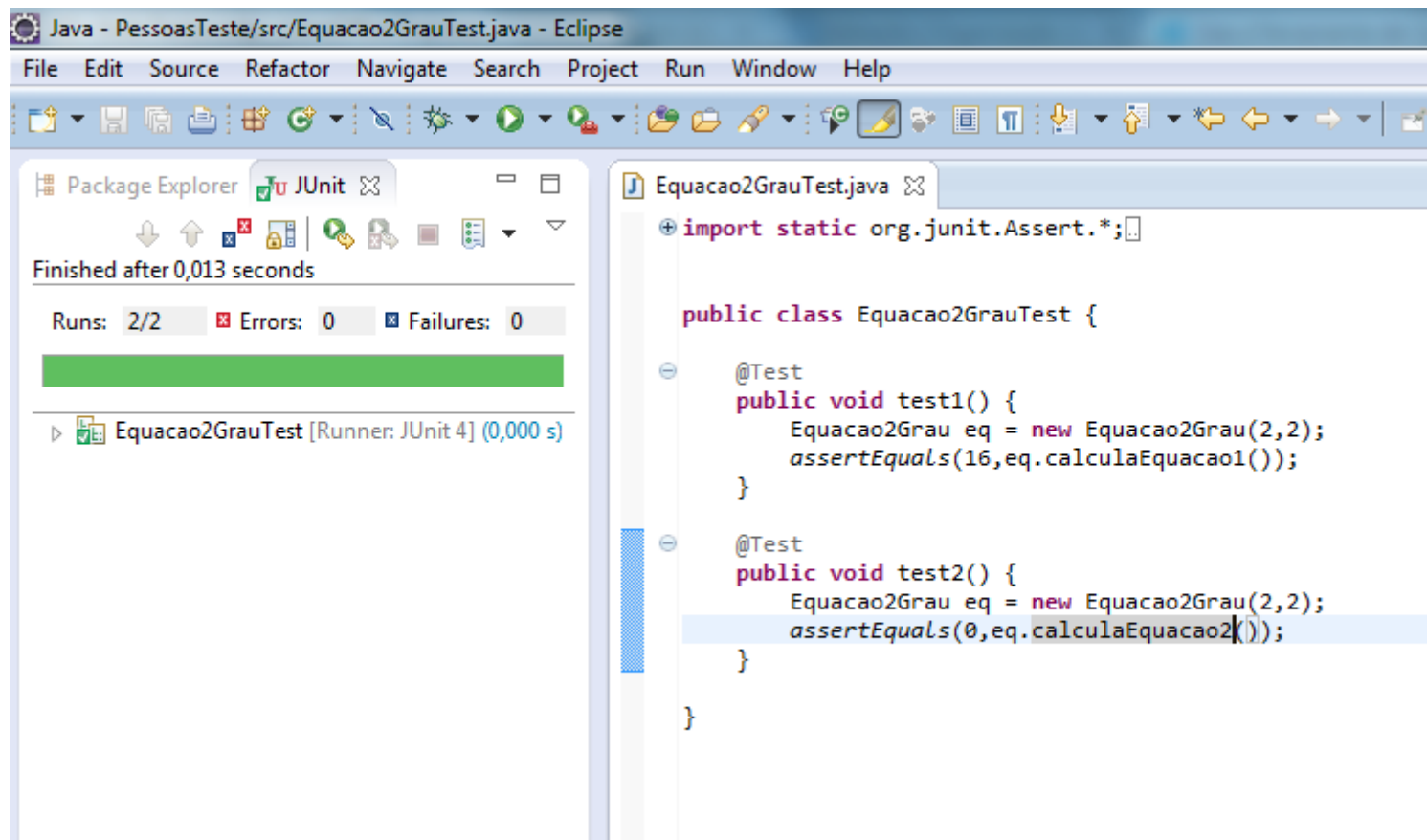


The screenshot shows the Eclipse IDE with the file Equacao2GrauTest.java open. The code imports org.junit.Assert.* and defines a public class Equacao2GrauTest. It contains two test methods: test1() and test2(). test1() creates an instance of Equacao2Grau with parameters (2, 2) and asserts that the result of calculaEquacao1() is 16. test2() creates an instance of Equacao2Grau with parameters (2, 2) and asserts that the result of calculaEquacao2() is 0.

```
import static org.junit.Assert.*;  
  
public class Equacao2GrauTest {  
  
    @Test  
    public void test1() {  
        Equacao2Grau eq = new Equacao2Grau(2,2);  
        assertEquals(16,eq.calculaEquacao1());  
    }  
  
    @Test  
    public void test2() {  
        Equacao2Grau eq = new Equacao2Grau(2,2);  
        assertEquals(0,eq.calculaEquacao2());  
    }  
}
```

Programação Orientada a Objetos

- Exemplo
- Rodando com sucesso



Programação Orientada a Objetos

- Exemplo

- Rodando com falha

Defeito no código

The screenshot displays the Eclipse IDE with two Java files: `Equacao2Gru.java` and `Equacao2GruTest.java`.

Equacao2Gru.java:

```
public class Equacao2Gru {
    int x;
    int y;

    Equacao2Gru(int x, int y) {
        this.x = x;
        this.y = y;
    }

    public int calculaEquacao1() {
        /* Método que calcula a equação x^2 + y^2 + 2xy */
        return x*x + 4*y + 2*x*y;
    }

    public int calculaEquacao2() {
        /* Método que calcula a equação x^2 + y^2 - 2xy */
        return x*x + y*y - 2*x*y;
    }
}
```

Equacao2GruTest.java:

```
import static org.junit.Assert.*;

public class Equacao2GruTest {

    @Test
    public void test1() {
        Equacao2Gru eq = new Equacao2Gru(2,2);
        assertEquals(16, eq.calculaEquacao1());
    }

    @Test
    public void test2() {
        Equacao2Gru eq = new Equacao2Gru(2,2);
        assertEquals(0, eq.calculaEquacao2());
    }
}
```





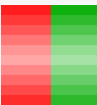
The JUnit test runner shows the following results:

- Finished after 0,014 seconds
- Runs: 2/2
- Errors: 0
- Failures: 1

The failure trace indicates the error:

```
java.lang.AssertionError: expected:<16> but was:<20>
at Equacao2GruTest.test1(Equacao2GruTest.java:11)
```

Programação Orientada a Objetos

 Coverage X				
				
Element	 Coverage	Covered In...	Missed Instructions	Total Instructions
>  Teste1_lab	 47,8 %	66	72	138
O Junit fornece informações sobre a cobertura dos casos de testes				