



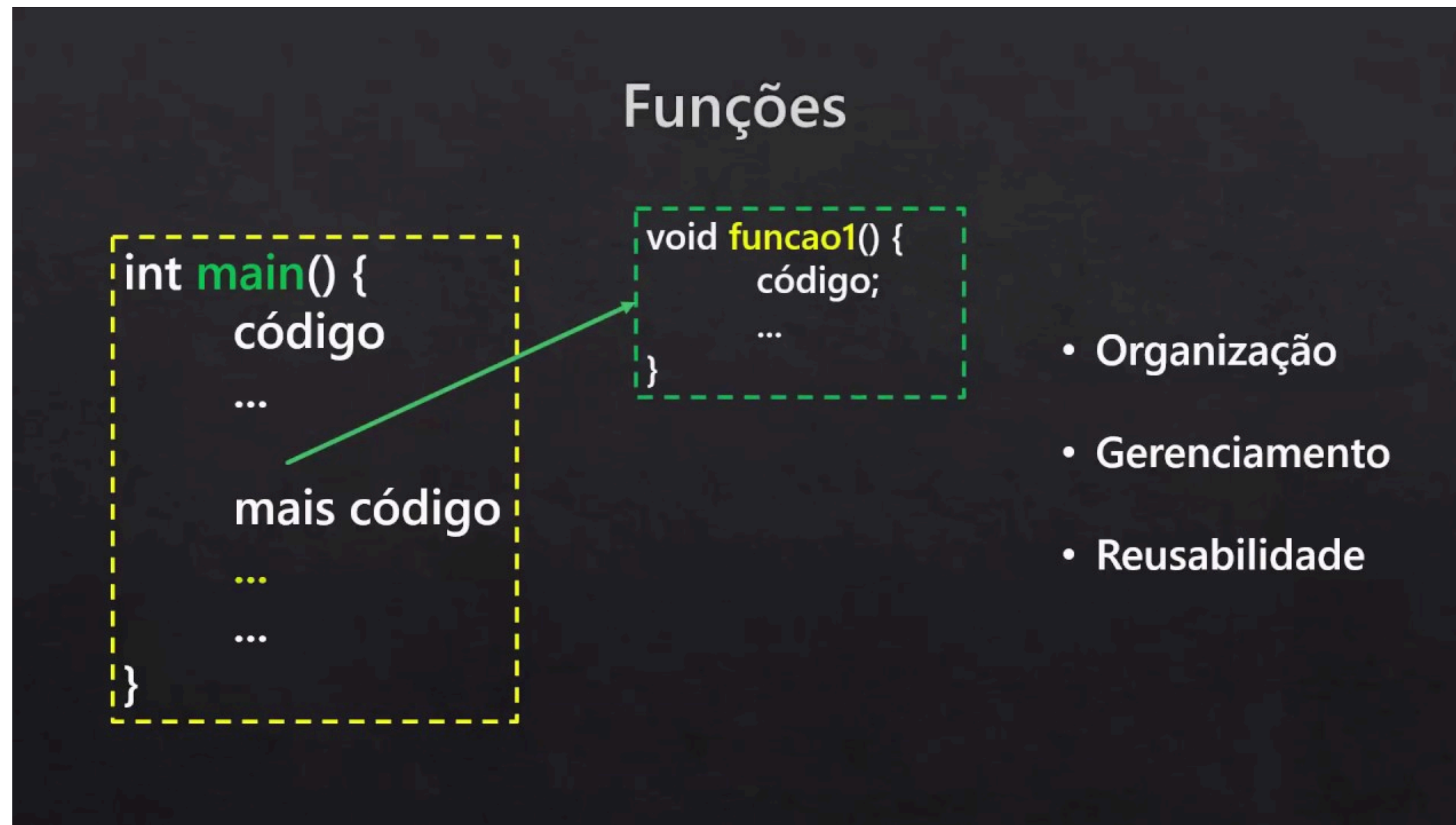
UNINASSAU

FUNÇÕES

Análise e Desenvolvimento de Sistemas
Lógica e Programação Algorítmica
Profº Lindenberg Andrade

O QUE É UMA FUNÇÃO EM PROGRAMAÇÃO?

Em programação, uma função é um bloco de código que executa uma tarefa específica e pode ser reutilizada várias vezes. Elas permitem que o código seja organizado, modularizado e mais fácil de ler e manter.



POR QUE USAR FUNÇÕES?

1. Modularização do código

- As funções permitem dividir um programa em partes menores e independentes.
- Cada função realiza uma tarefa específica, facilitando o entendimento do programa como um todo.

POR QUE USAR FUNÇÕES?

2. Reutilização de código

- Funções podem ser chamadas várias vezes sem precisar reescrever o mesmo código.
- Isso economiza tempo e reduz erros.

POR QUE USAR FUNÇÕES?

3. Clareza e legibilidade

- Um código bem estruturado em funções é mais fácil de ler e entender.
- É mais simples saber "o que está acontecendo" em cada parte do programa.

POR QUE USAR FUNÇÕES?

4. Facilidade de manutenção

- Se algo precisar ser corrigido ou alterado, basta modificar a função em um só lugar.
- Isso diminui a chance de bugs e facilita atualizações.

POR QUE USAR FUNÇÕES?

5. Testabilidade

- Funções isoladas são mais fáceis de testar individualmente (testes unitários).
- Permite validar partes específicas do código sem depender do programa inteiro.

FORMATO DE UMA FUNÇÃO EM JAVA

```
[modificadores] tipoDeRetorno nomeDaFuncao([parâmetros]) {  
    // corpo da função  
    [return valor]; // se não for void  
}
```


FORMATO DE UMA FUNÇÃO EM JAVA

Componente	Descrição	Exemplo
modificadores	Define a visibilidade e o contexto da função	public, private, static
tipoDeRetorno	Especifica o tipo de dado que a função retorna	int, void, String
nomeDaFuncao	Nome dado à função	somar, mostrarMensagem
parâmetros	Dados de entrada da função	(int a, int b)
corpo da função	Bloco de código que define o que a função faz	{ return a + b }
return (opcional)	Palavrinha usada para retornar um valor	return a + b;

RETORNO DO TIPO **VOID**?

Em Java, **void** é um tipo de retorno especial usado quando a função não retorna nenhum valor. Ou seja, a função executa uma ação, mas não entrega nenhum resultado para quem a chamou.

QUANDO USAR **VOID**?

Use **void** quando:

- Você não precisa de um valor de retorno.
- A função serve apenas para executar um procedimento, como imprimir algo na tela, alterar um valor ou gravar em um arquivo.

EXEMPLO FUNÇÃO COM VOID

```
public static void exibirMensagem() {  
    System.out.println("Bem-vindo ao sistema!");  
}
```

```
exibirMensagem(); // Executa a ação, mas não retorna nada
```

EXEMPLO FUNÇÃO COM VOID

Tipo de retorno	Descrição	Exemplo
void	Não retorna valor	System.out.println(...)
int, double	Retorna um número inteiro/decimal	return 42;
String	Retorna uma sequência de texto	return \"Olá\";

**PROGRAMANDO
NO ECLIPSE IDE**

O que é o IDE Eclipse?

O Eclipse é um Ambiente de Desenvolvimento Integrado (IDE) livre e de código aberto que é amplamente utilizado para desenvolver aplicações Java. É uma ferramenta poderosa que fornece uma variedade de recursos para escrever, depurar, testar e gerenciar projetos Java.



Java Development Kit (JDK)

O Java Development Kit (JDK) é um pacote de software fornecido pela Oracle (ou outros distribuidores, como o OpenJDK) que contém tudo o que você precisa para desenvolver programas em Java.

Instalar o JDK

- Baixar o JDK mais recente (Java Development Kit):
- <https://www.oracle.com/java/technologies/javase-downloads.html>
- ou usar o OpenJDK: <https://adoptium.net/>





Baixar o JDK mais recente (Java Development Kit)

ORACLE

Products Industries Resources Customers Partners Developers Company

🔍



 View Accounts

Java downloads

Tools and resources

Java archive

JDK 24

JDK 21

GraalVM for JDK 24

GraalVM for JDK 21

Java SE Development Kit 24.0.1 downloads

JDK 24 binaries are free to use in production and free to redistribute, at no cost, under the [Oracle No-Fee Terms and Conditions \(NFTC\)](#).

JDK 24 will receive updates under these terms, until September 2025, when it will be superseded by JDK 25.

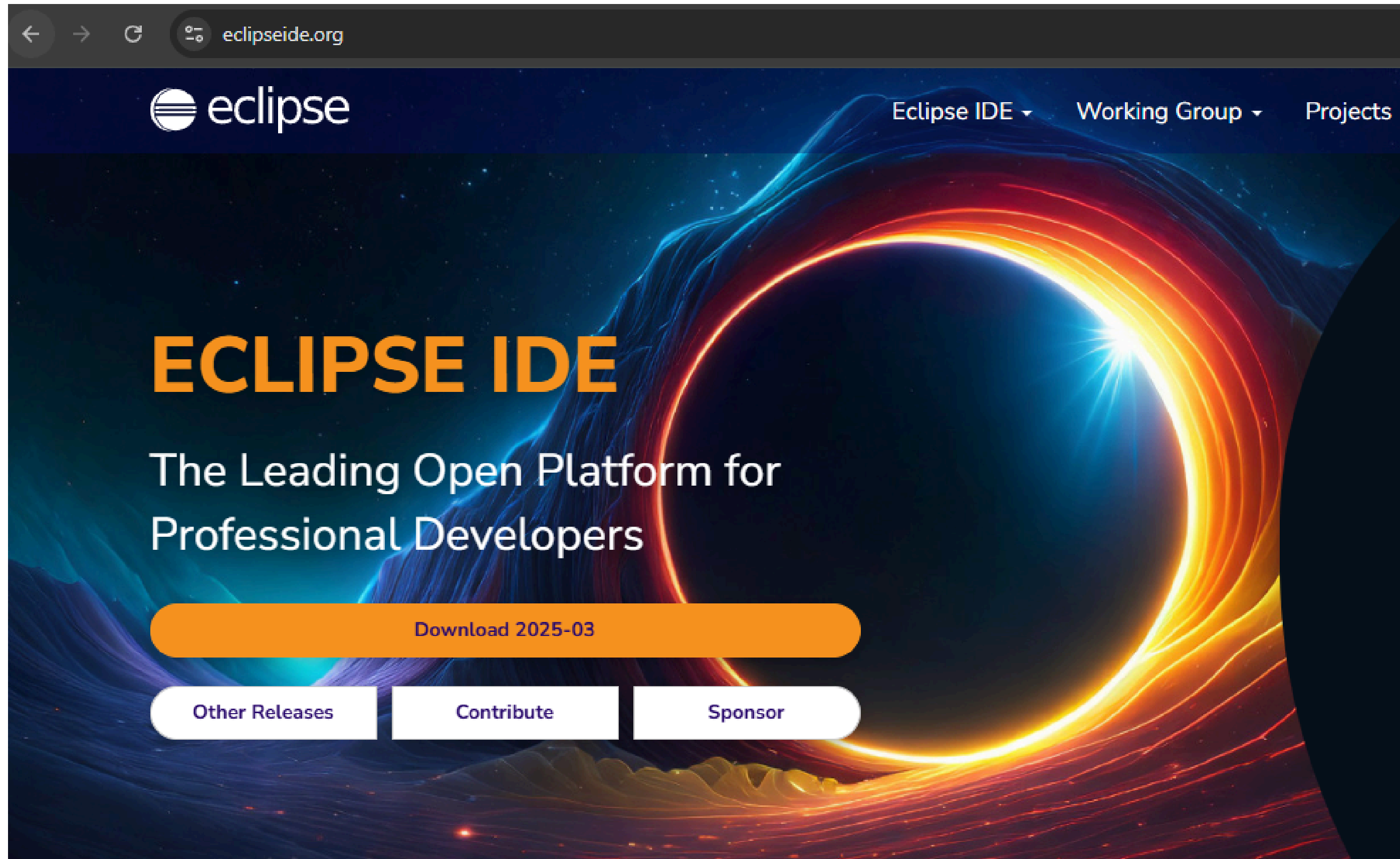
Linux

macOS

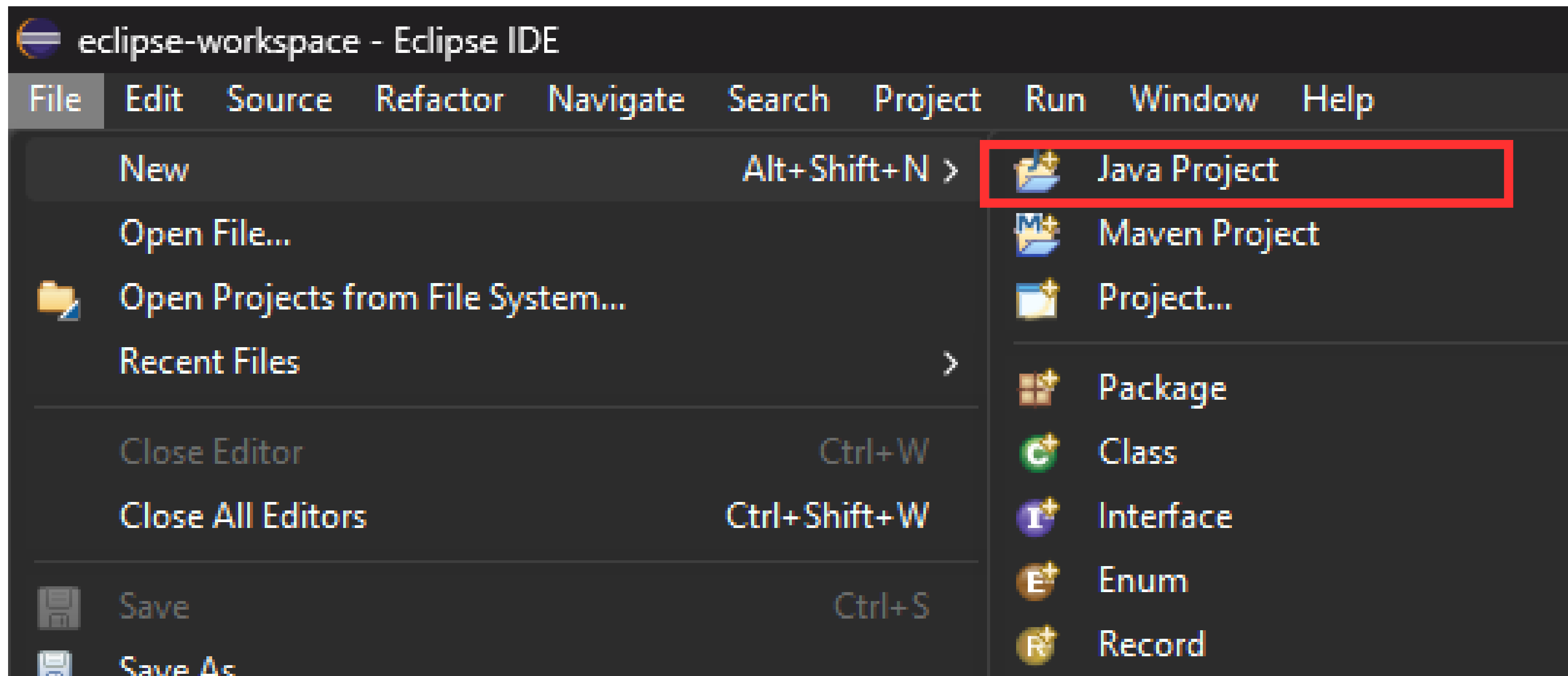
Windows

Product/file description	File size	Download
x64 Compressed Archive	229.51 MB	https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.zip (sha256)
x64 Installer	205.85 MB	https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.exe (sha256)
x64 MSI Installer	204.60 MB	https://download.oracle.com/java/24/latest/jdk-24_windows-x64_bin.msi (sha256)

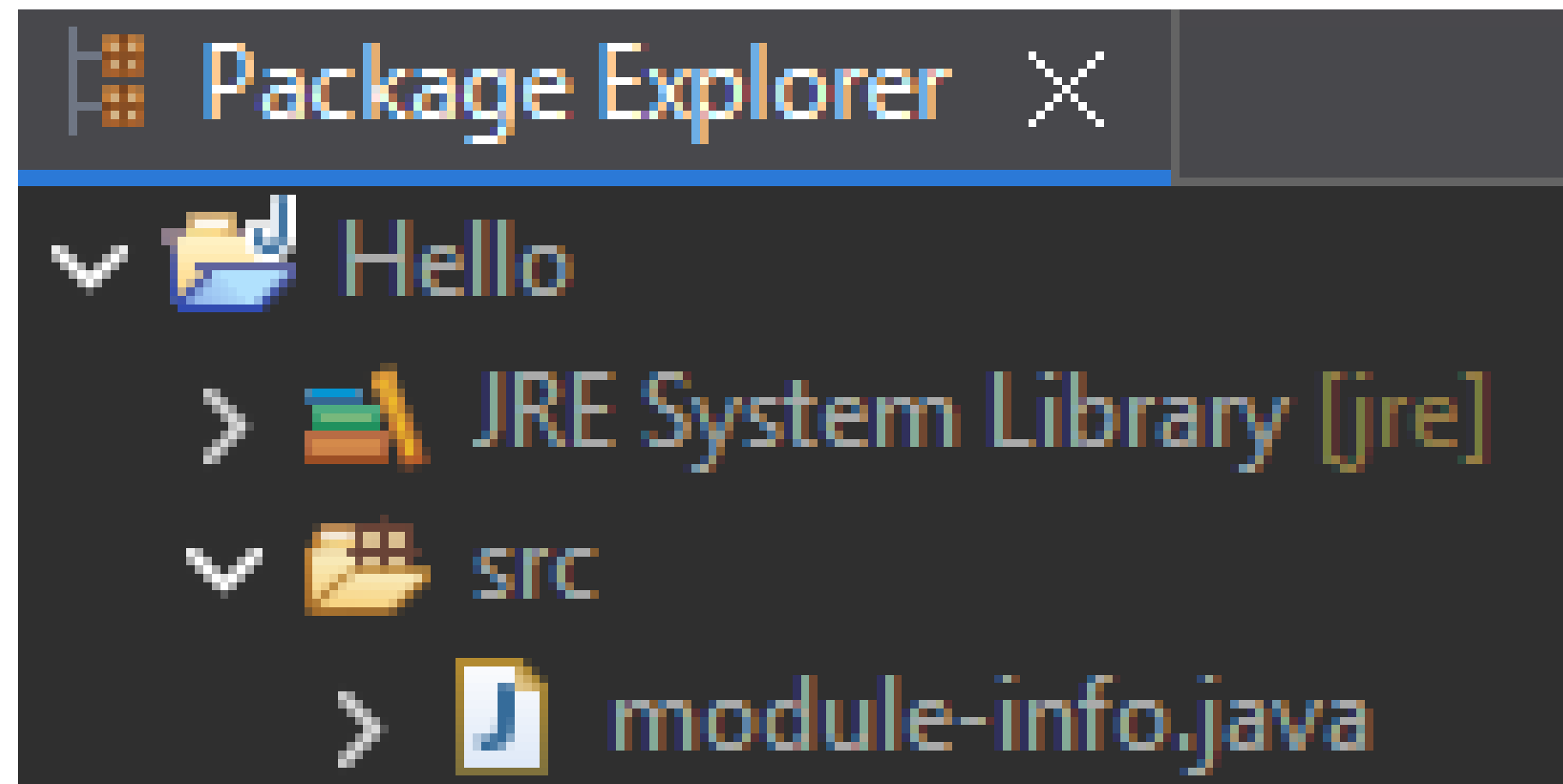
Baixando o Eclipse IDE em <https://eclipseide.org/>



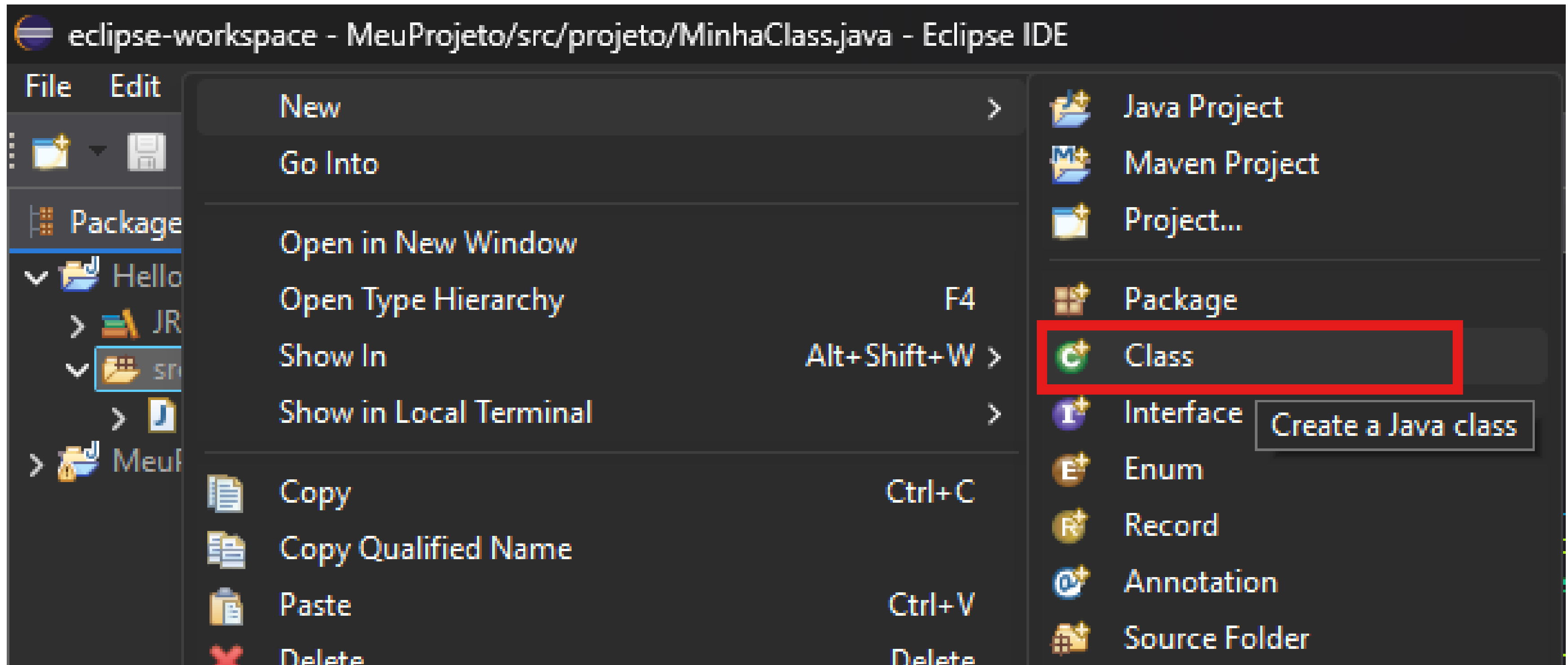
Criando o primeiro Projeto no Eclipse IDE



Criando o primeiro Projeto no Eclipse IDE



Criando o primeiro Projeto no Eclipse IDE



Criando o primeiro Projeto no Eclipse IDE

New Java Class

Java Class
Create a new Java class.

Source folder: Hello/src Browse...

Package: hello Browse...

☐ Enclosing type: Browse...

Name: Hello

Modifiers:
☒ public ☐ package ☐ private ☐ protected
☐ abstract ☐ final ☐ static
☒ none ☐ sealed ☐ non-sealed ☐ final

Superclass: java.lang.Object Browse...

Interfaces: Add...
Remove

Which method stubs would you like to create?

☒ public static void main(String[] args)
☐ Constructors from superclass
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

? Finish Cancel

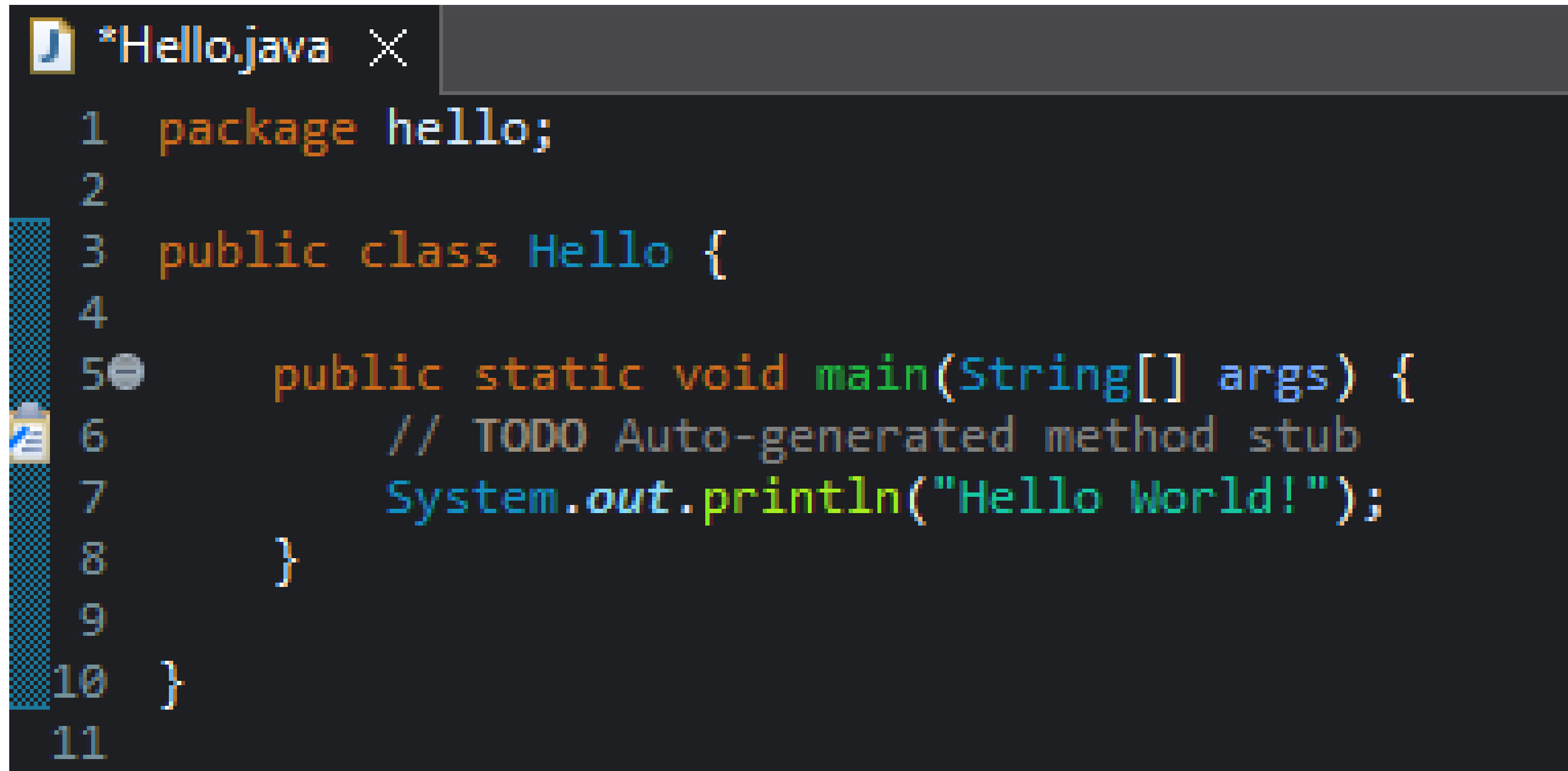
Which method stubs would you like to create?

☒ public static void main(String[] args)

☐ Constructors from superclass

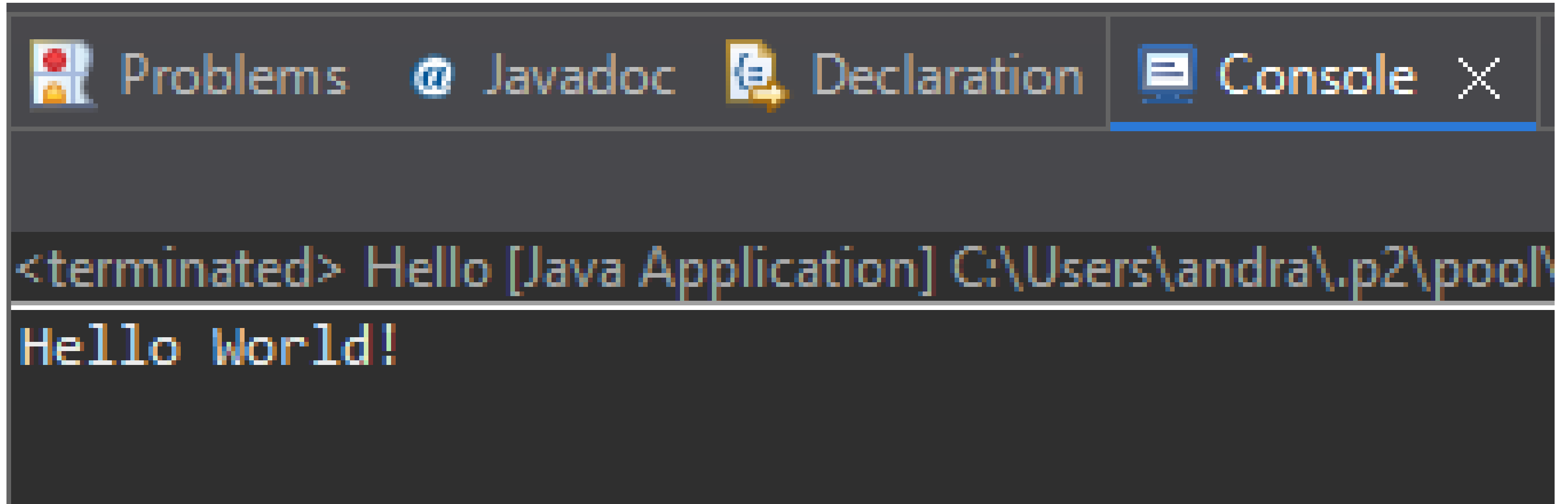
☒ Inherited abstract methods

Criando o primeiro Projeto no Eclipse IDE



```
*Hello.java X
1  package hello;
2
3  public class Hello {
4
5      public static void main(String[] args) {
6          // TODO Auto-generated method stub
7          System.out.println("Hello World!");
8      }
9
10 }
11
```

Criando o primeiro Projeto no Eclipse IDE



The screenshot shows the Eclipse IDE's Console window. The title bar at the top includes icons for Problems, Javadoc, Declaration, and Console, with the Console tab selected. The console output displays the text: `<terminated> Hello [Java Application] C:\Users\andra\.p2\pool\` followed by `Hello World!` on a new line.

```
<terminated> Hello [Java Application] C:\Users\andra\.p2\pool\  
Hello World!
```


TIPOS DE FUNÇÕES

Função sem retorno e sem parâmetros

```
*saudacao.java X
1 package saudacao;
2
3 public class saudacao {
4
5     public static void main(String[] args) {
6         saudacao1(); // chamada da função
7     }
8
9     public static void saudacao1() {
10        System.out.println("Olá, seja bem-vindo ao programa!");
11    }
12 }
```

Problems Javadoc Declaration Console X

<terminated> saudacao [Java Application] C:\Users\andra\.p2\p
Olá, seja bem-vindo ao programa!

Função com parâmetros e sem retorno

```
1 package mensagem;  
2  
3 public class mensagem {  
4  
5     public static void main(String[] args) {  
6         exibirMensagem("Carlos");  
7  
8     }  
9  
10    public static void exibirMensagem(String nome) {  
11        System.out.println("Olá, " + nome + "!");  
12    }  
13  
14  
15 }
```

Problems @ Javadoc Declaration Console X

<terminated> mensagem [Java Application] C:\Users\andra\.p2
Olá, Carlos!

Função com retorno e sem parâmetros

```
1 package obterM;  
2  
3 public class obterM {  
4  
5     public static void main(String[] args) {  
6         String msg = obterMensagemPadrao();  
7         System.out.println(msg);  
8  
9     }  
10  
11     public static String obterMensagemPadrao() {  
12         return "Mensagem padrão do sistema.";  
13     }  
14 }
```

Problems @ Javadoc Declaration Console X

<terminated> obterM [Java Application] C:\Users\andra\.p2\pc
Mensagem padrão do sistema.

Função com retorno e com parâmetros

```
1 package soma;
2
3 public class soma {
4
5     public static void main(String[] args) {
6         int resultado = somar(10, 5);
7         System.out.println("Resultado: " + resultado);
8
9     }
10
11     public static int somar(int a, int b) {
12         return a + b;
13     }
14
15 }
```

Problems @ Javadoc Declaration Console X

<terminated> soma [Java Application] C:\Users\andra\.p2\pool\...
Resultado: 15

Função que chama outra função

```
1 package dobro;
2
3 public class dobro {
4
5     public static void main(String[] args) {
6         mostrarDobro(8);
7
8     }
9
10    public static int calcularDobro(int x) {
11        return x * 2;
12    }
13
14    public static void mostrarDobro(int valor) {
15        int resultado = calcularDobro(valor);
16        System.out.println("O dobro é: " + resultado);
17    }
18
19 }
20
```

Problems Javadoc Declaration Console X

<terminated> dobro [Java Application] C:\Users\andra\.p2\poo

O dobro é: 16

EXEMPLOS

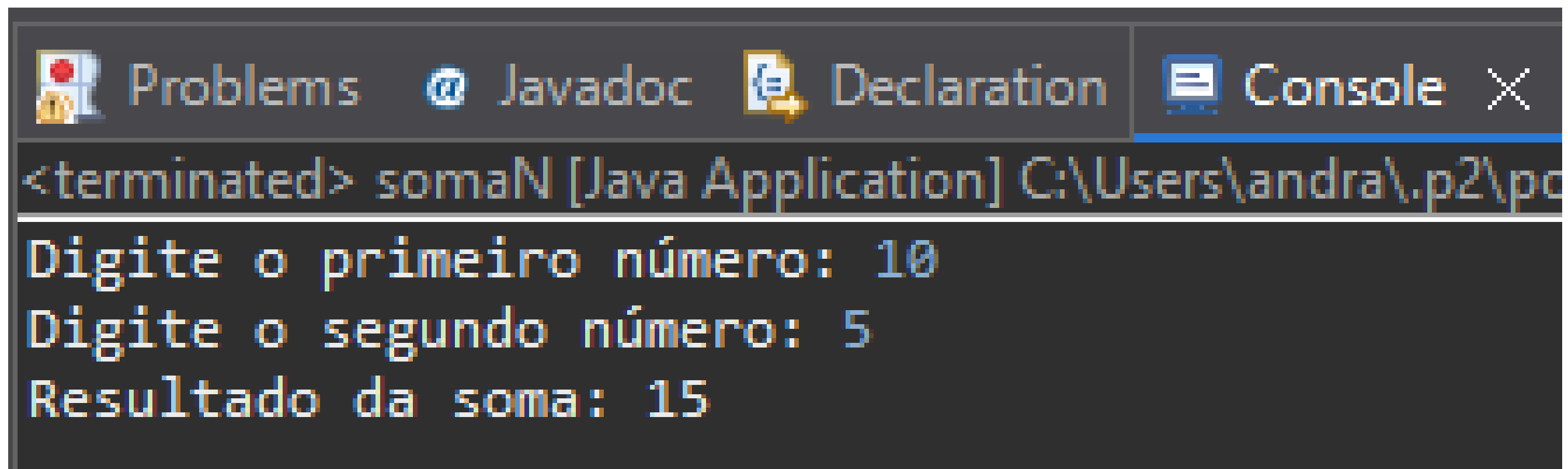
EXEMPLO 1

1) Crie uma função em Java que faça a soma de dois número

EXEMPLO 1

```
1 package somaN;
2 import java.util.Scanner;
3 public class somaN {
4
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7
8         System.out.print("Digite o primeiro número: ");
9         int num1 = scanner.nextInt();
10
11         System.out.print("Digite o segundo número: ");
12         int num2 = scanner.nextInt();
13
14         int resultado = somar(num1, num2);
15         System.out.println("Resultado da soma: " + resultado);
16
17         scanner.close();
18     }
19
20     public static int somar(int a, int b) {
21         return a + b;
22     }
23 }
```

EXEMPLO 1



The screenshot shows an IDE window with four tabs: 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active and displays the following text:

```
<terminated> somaN [Java Application] C:\Users\andra\.p2\po  
Digite o primeiro número: 10  
Digite o segundo número: 5  
Resultado da soma: 15
```

EXEMPLO 2:

2) Crie uma função Calculadora de IMC
(Índice de Massa Corporal)

EXEMPLO 2:

Índice de Massa Corporal

$$\mathbf{IMC} = \frac{\text{Peso (Kg)}}{\text{Altura (m)}^2}$$

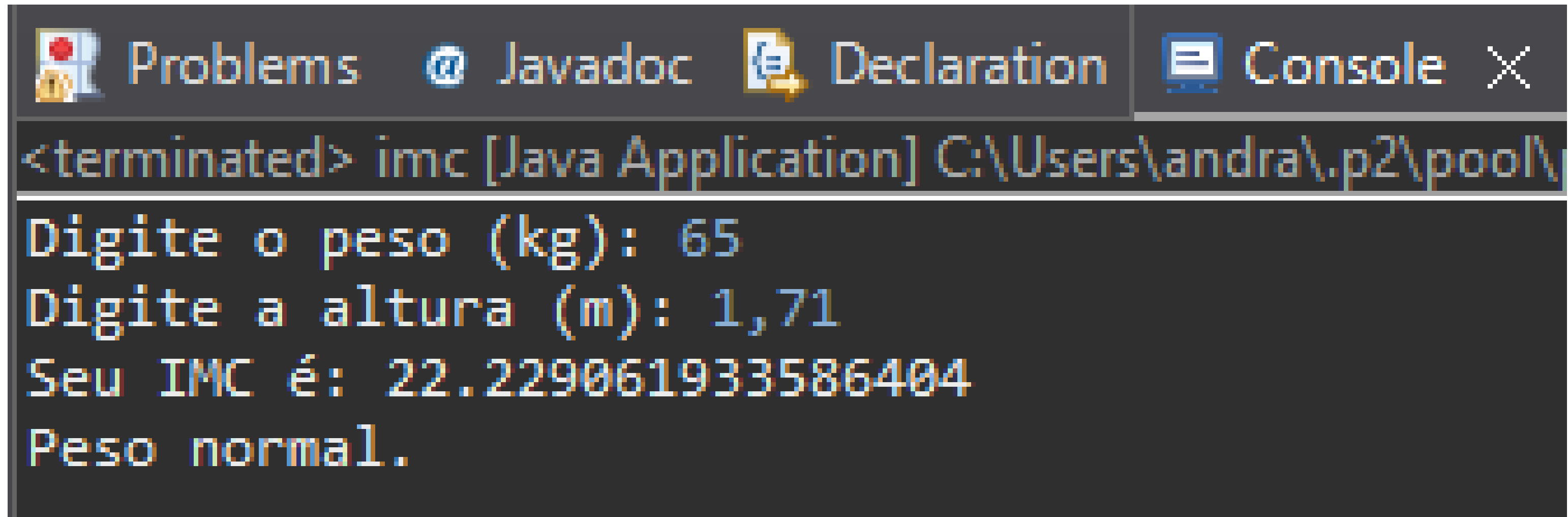
EXEMPLO 2: Parte 1 do código

```
1 package imc;
2
3 import java.util.Scanner;
4
5 public class imc {
6
7     public static void main(String[] args) {
8         Scanner scanner = new Scanner(System.in);
9
10        System.out.print("Digite o peso (kg): ");
11        double peso = scanner.nextDouble();
12
13        System.out.print("Digite a altura (m): ");
14        double altura = scanner.nextDouble();
15
16        double imc = calcularIMC(peso, altura);
17        System.out.println("Seu IMC é: " + imc);
18        interpretarIMC(imc);
19
20        scanner.close();
21    }
22
```

EXEMPLO 2: Parte 2 do código

```
22
23 public static double calcularIMC(double peso, double altura) {
24     return peso / (altura * altura);
25 }
26
27
28 public static void interpretarIMC(double imc) {
29     if (imc < 18.5) {
30         System.out.println("Abaixo do peso.");
31     } else if (imc < 25) {
32         System.out.println("Peso normal.");
33     } else if (imc < 30) {
34         System.out.println("Sobrepeso.");
35     } else {
36         System.out.println("Obesidade.");
37     }
38 }
39
40 }
```

EXEMPLO 2: Resultado no Terminal/Console



The screenshot shows an IDE window with four tabs: 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active, displaying the output of a Java application named 'imc'. The application prompts the user for weight and height, calculates the BMI, and provides a classification. The user input is 65 kg and 1.71 m, resulting in a BMI of 22.229061933586404 and a classification of 'Peso normal'.

```
<terminated> imc [Java Application] C:\Users\andra\.p2\pool\
Digite o peso (kg): 65
Digite a altura (m): 1,71
Seu IMC é: 22.229061933586404
Peso normal.
```

EXERCÍCIOS

EXERCÍCIO 1: Soma de Três Números

1) Escreva uma função **somar(int a, int b, int c)** que retorna a soma de três números inteiros.

EXERCÍCIO 2: Maior entre Dois Números

2) Crie uma função **maior(int a, int b)** que retorna o maior dos dois valores.

EXERCÍCIO 3: Verificação de Número Par

3) Crie a função **ehPar(int x)** que retorna true se o número for par e false caso contrário.

EXERCÍCIO 4: Conversão de Temperatura

4) Crie uma função

celsiusParaFahrenheit(double c) que converte a temperatura de Celsius para Fahrenheit usando a fórmula:

$$F = (C * 9/5) + 32$$

EXERCÍCIO 5: Cálculo de Fatorial

5) Crie uma função **fatorial(int n)** que calcula o fatorial de um número inteiro (usando recursão ou laço).

EXERCÍCIO 6: Verificar se é Palíndromo

6) Escreva uma função **ehPalindromo(String palavra)** que verifica se uma palavra é igual quando lida de trás para frente.

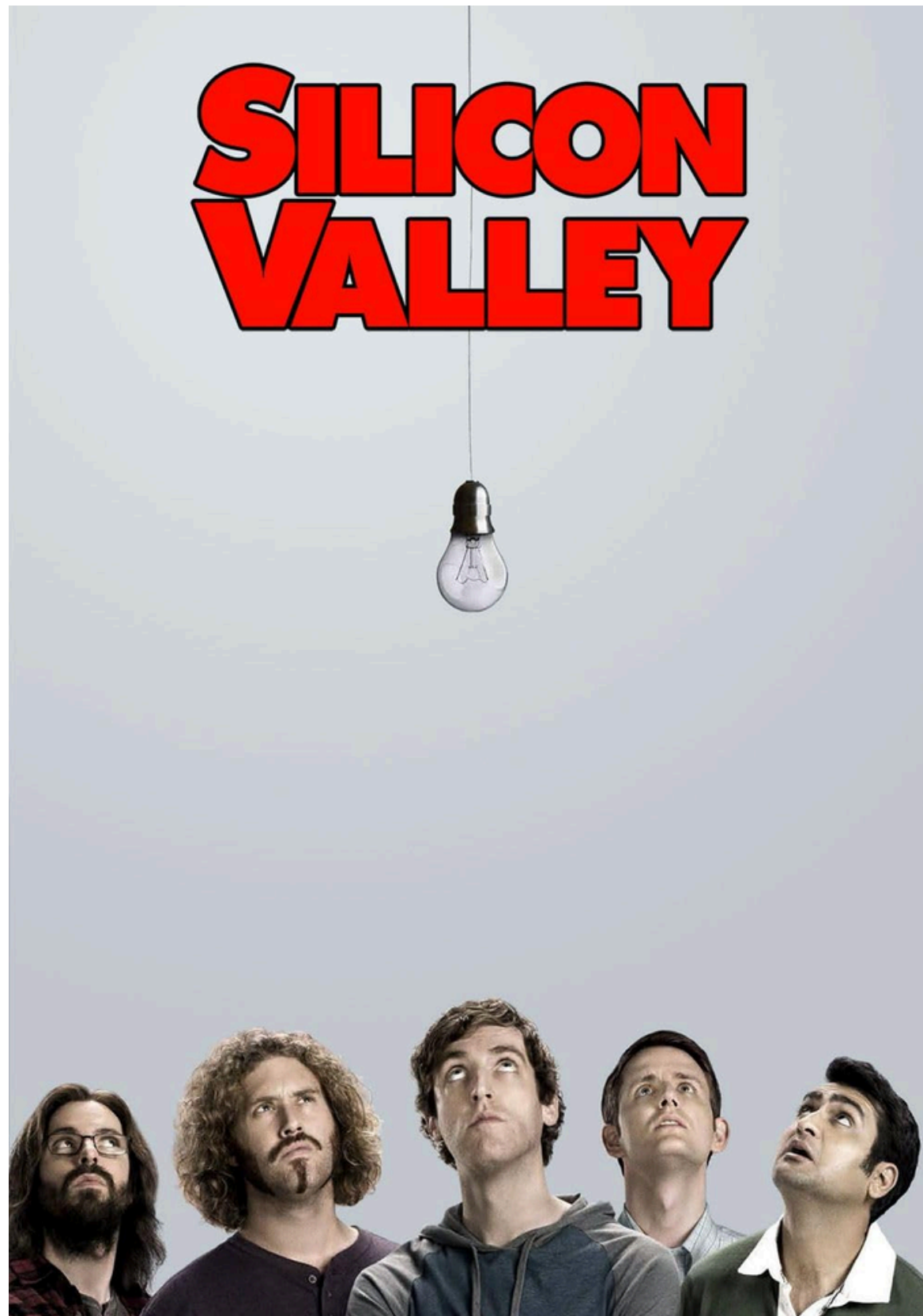
EXERCÍCIO 6: Verificar se é Palíndromo

PALÍNDROMO

É toda palavra ou frase que pode ser lida de trás para frente e que, independente da direção, mantém o seu sentido.



Dica de Filme: Silicon Valley (2014)



Ambientada no Vale do Silício, região da Califórnia fértil em inovações tecnológicas e científicas, a série mostra um grupo de desenvolvedores que cria novo um programa com o objetivo de impressionar um bilionário excêntrico do ramo tecnológico.

REFERÊNCIAS

- SOUZA, Marco Antonio Furlan de. **Algoritmos e lógica de programação**. São Paulo: Cengage, 2004.
- LOPES, Anita. **Introdução a Programação**. Rio de Janeiro, Elsevier, 2002.

Dúvidas?



Profº Lindenberg Andrade
E-mail: linndemberg1@gmail.com



Additional contacts via QR code