

INTRODUÇÃO AO JAVA

Prof. Ricardo Mesquita

Sumário

- Breve Histórico
- Características da Linguagem Java
- O Ambiente Java
- Ambiente de Desenvolvimento
- Packages
- Classpath
- O Programa Java
- Sobre a Orientação a Objetos...
- Objetos
- Classes, Objetos e Métodos

Sobre a Orientação a Objetos...

- O ser humano se relaciona com o mundo através do conceito de **objetos**.
- Estamos sempre **identificando** qualquer objeto ao nosso redor.
- Para isso lhe **damos nomes**, e de acordo com suas características lhes **classificamos em grupos**, ou seja, **classes**.

Sobre a Orientação a Objetos...

- Objetos do mundo real possuem duas características: **estado** e **comportamento**.
- *Exemplos:*
 - Cachorros → **estado**: nome, cor, raça
comportamento: latir, correr
 - Bicicletas → **estado**: marcha atual, velocidade atual
comportamento: trocar marcha, aplicar freios

Sobre a Orientação a Objetos...

- Identificar o **estado** e o **comportamento** de objetos do mundo real é o primeiro passo para começar a pensar em programação OO.
- Observe um objeto e pergunte:
 - Quais os possíveis estados que esse objeto pode estar?
 - Quais os possíveis comportamentos que ele pode executar?

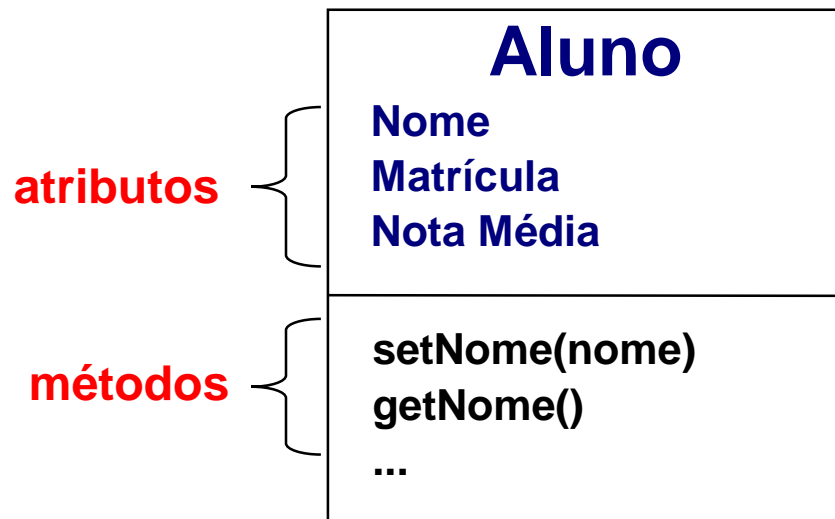
Sobre a Orientação a Objetos...

- A *unidade fundamental* de programação em orientação a objetos (POO) é a *classe*.
- Classes contém:
 - *Atributos*: determinam o *estado* do objeto;
 - *Métodos*: semelhantes a procedimentos em linguagens convencionais, são utilizados para *manipular os atributos*.

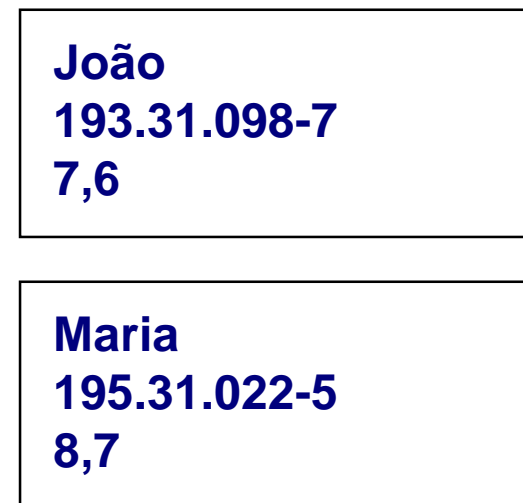
Sobre a Orientação a Objetos...

- As classes provêm a *estrutura para a construção de objetos* – estes são ditos *instâncias* das classes!

Classe



Instâncias



A Linguagem Java



Características da Linguagem Java

- É uma *linguagem simples* de fácil aprendizado.
- É uma linguagem *puramente orientada a objetos*.
 - A abordagem de OO permite o desenvolvimento de sistemas de uma forma “mais natural”.

Características da Linguagem Java

- Java foi projetada para trabalhar em um ambiente de redes
- Java *não é* uma linguagem para programação distribuída; apenas *oferece bibliotecas para facilitar o processo de comunicação*.
 - Por exemplo, java RMI

Características da Linguagem Java

- Java é uma *linguagem interpretada*, logo ela nunca será tão rápida quanto as linguagens compiladas.
- Java chega a ser 20 vezes mais lento que C!
- Se serve de compiladores *just in time* (JIT), *que interpretam os bytecodes para um código nativo* durante a execução.

Características da Linguagem Java

- Java possui as seguintes características que contribuem para torná-la mais *robusta e segura*:
 - É *fortemente tipada*;
 - Não possui aritmética de ponteiros;
 - Possui mecanismo de *coleta de lixo*;
 - Possui *verificação rigorosa* em tempo de compilação;
 - Possui *mecanismos para verificação em tempo de execução*;
 - Possui *gerenciador de segurança*.

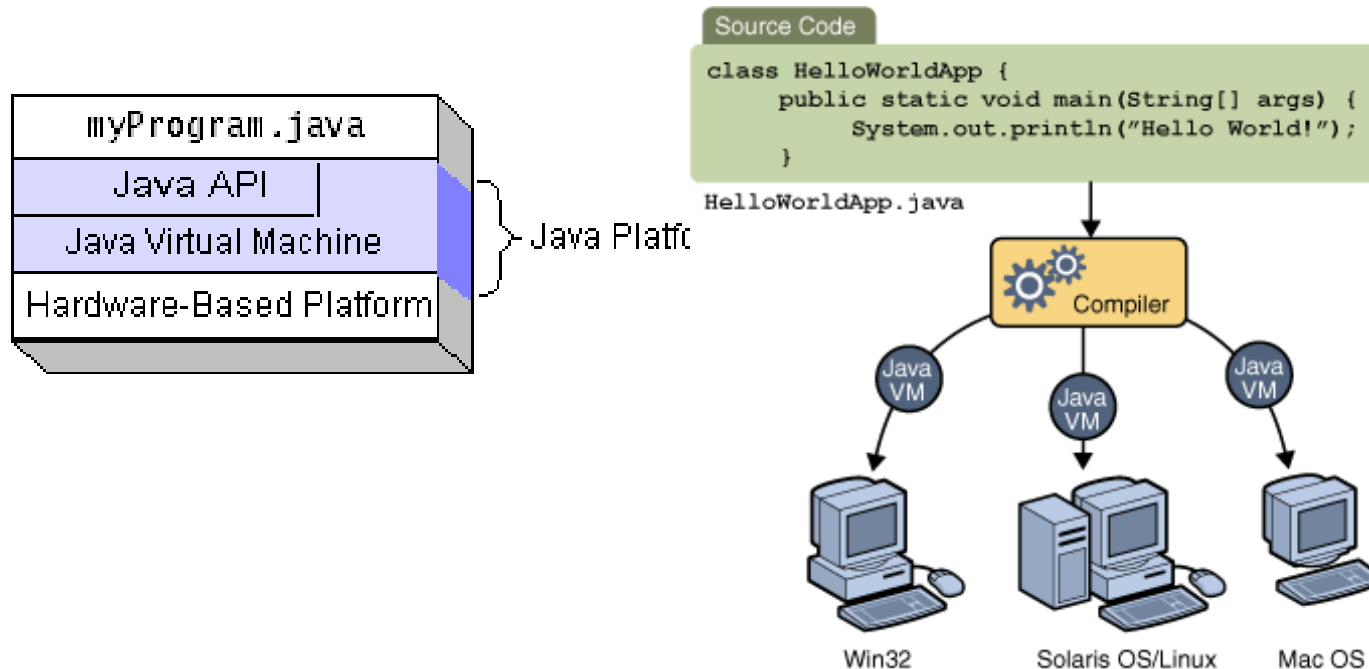
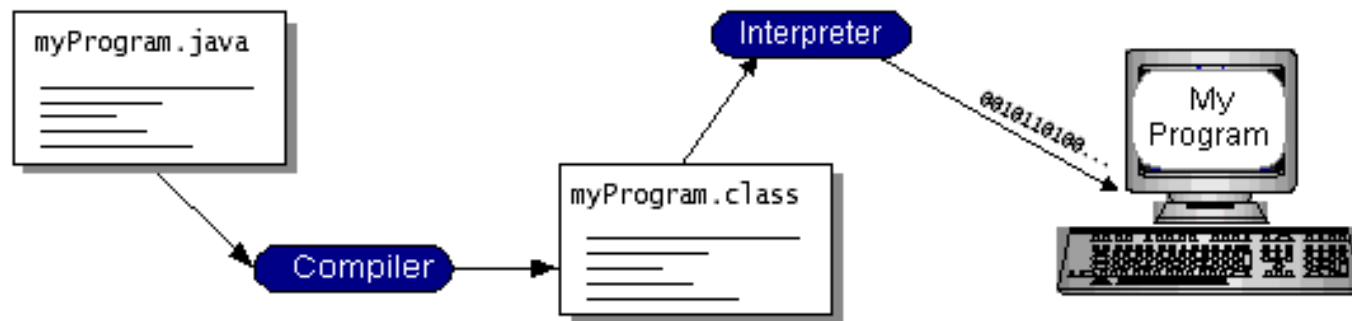
Características da Linguagem Java

- **Segurança:** Java possui mecanismos de segurança que podem evitar qualquer operação no sistema de arquivos da máquina alvo, minimizando problemas.
- **Bytecodes** executam em qualquer máquina que possua uma JVM, permitindo que o código em Java possa ser escrito *independente da plataforma*.
- A característica de ser *neutra em relação à arquitetura* permite uma grande *portabilidade*.

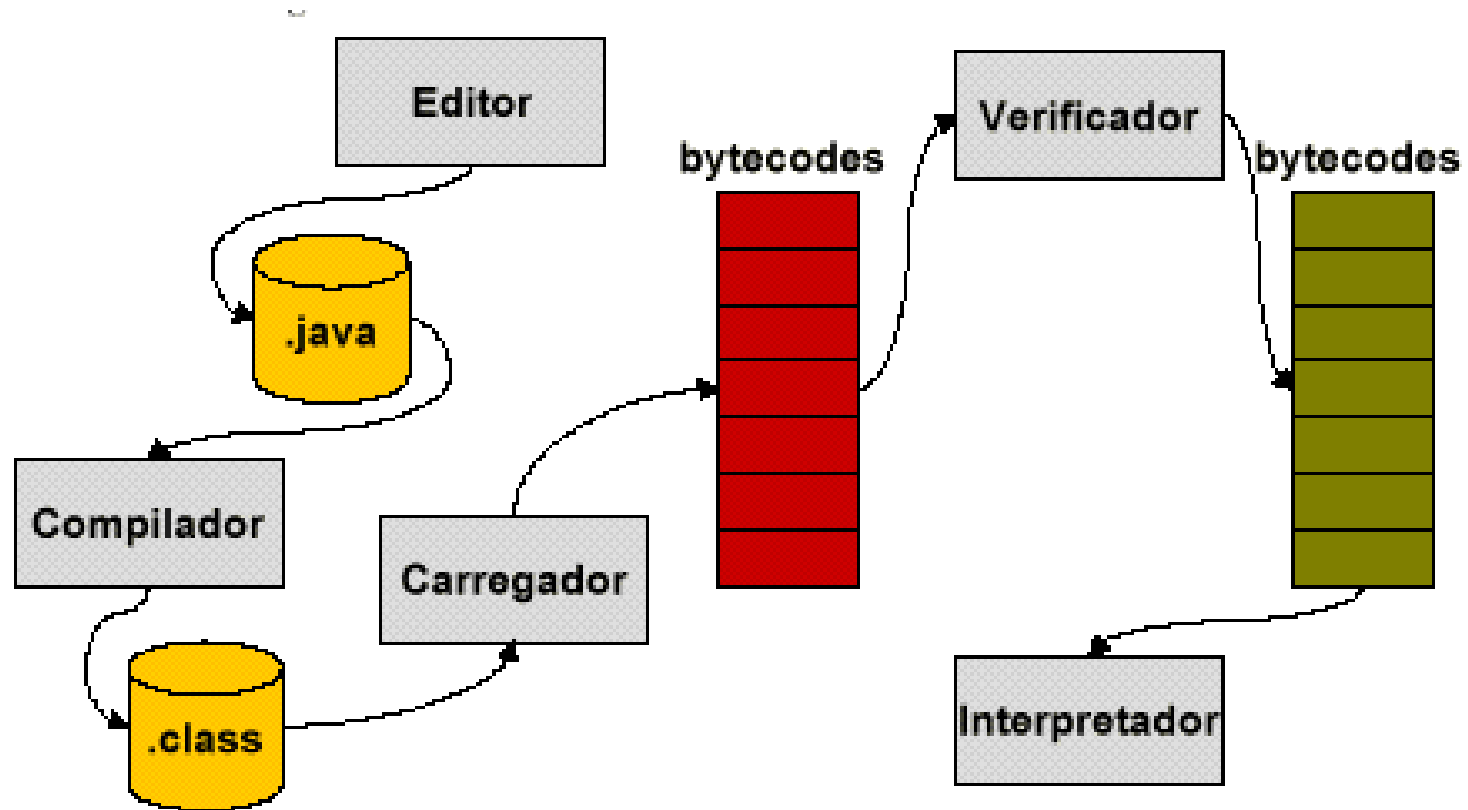
Características da Linguagem Java

- Java possui mecanismos para a resolução de referências em tempo de execução, o que permite flexibilidade nas aplicações.
- Java provê *suporte para múltiplas threads* (processos leves) *em execução*, que podem *tratar diferentes tarefas concorrentemente*.

Interpretada, Neutra, Portável



O Ambiente Java



Ambiente de Desenvolvimento

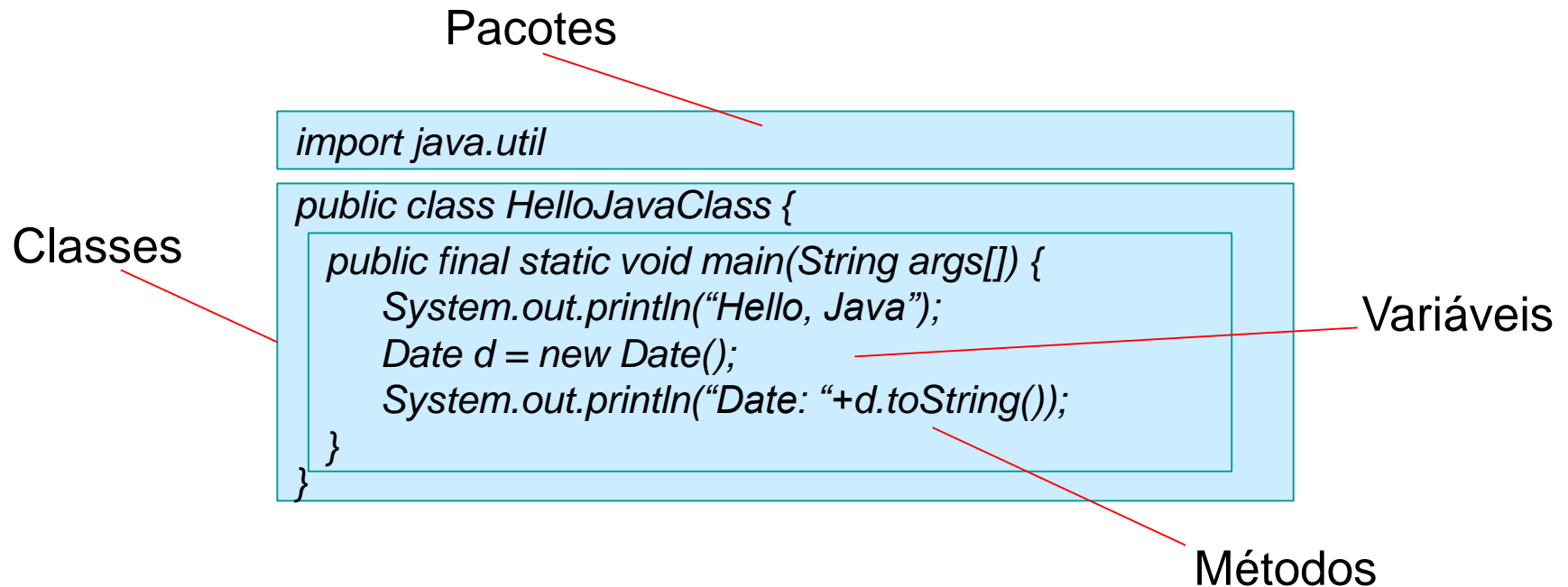
- **Algumas ferramentas do Java SDK:**
 - o *compilador* Java (javac)
 - o *interpretador* de aplicações Java (java)
 - o *interpretador de applets* Java (appletviewer)

e ainda:

- javadoc (um *gerador de documentação* para programas Java)
- jar (o *manipulador de arquivos comprimidos* no formato Java Archive)
- jdb (um *depurador de programas* Java), entre outras ferramentas.

O Programa Java

- Todos os programas em Java possuem quatro elementos básicos:



Objetos

- O que são objetos?
 - São *instâncias de uma classe*.
 - Sob o ponto de vista da programação orientada a objetos, um objeto não é muito diferente de uma variável normal.
- Um *programa orientado a objetos* é composto por um conjunto de objetos que interagem entre si.

Objetos

- Objetos de software são conceitualmente similares a objetos do mundo real: eles consistem do *estado* e o *comportamento* relacionado.
- Um objeto armazena seu estado em *campos* (variáveis) e expõe seu comportamento através de *métodos* (funções).

Objetos

- *Encapsulamento*: princípio de projeto pelo qual cada componente de um programa deve **agregar toda a informação relevante para sua manipulação** como uma unidade (uma cápsula).
- *Ocultação da Informação*: princípio pelo qual *cada componente deve manter oculta sob sua guarda uma decisão de projeto única*. Para a utilização desse componente, apenas o mínimo necessário para sua operação deve ser revelado (tornado público)

O Básico...


- Um código em Java é pré-compilado e posteriormente interpretado pela JVM (Java Virtual Machine)
- Um exemplo simples:

```
1. public class Bemvindo1
2. {
3.     public static void main( String[ ] args )
4.     {
5.         System.out.println("Bem-vindo ao Java!");
6.     } //fim do método main
7. } //fim da classe Bemvindo1
```

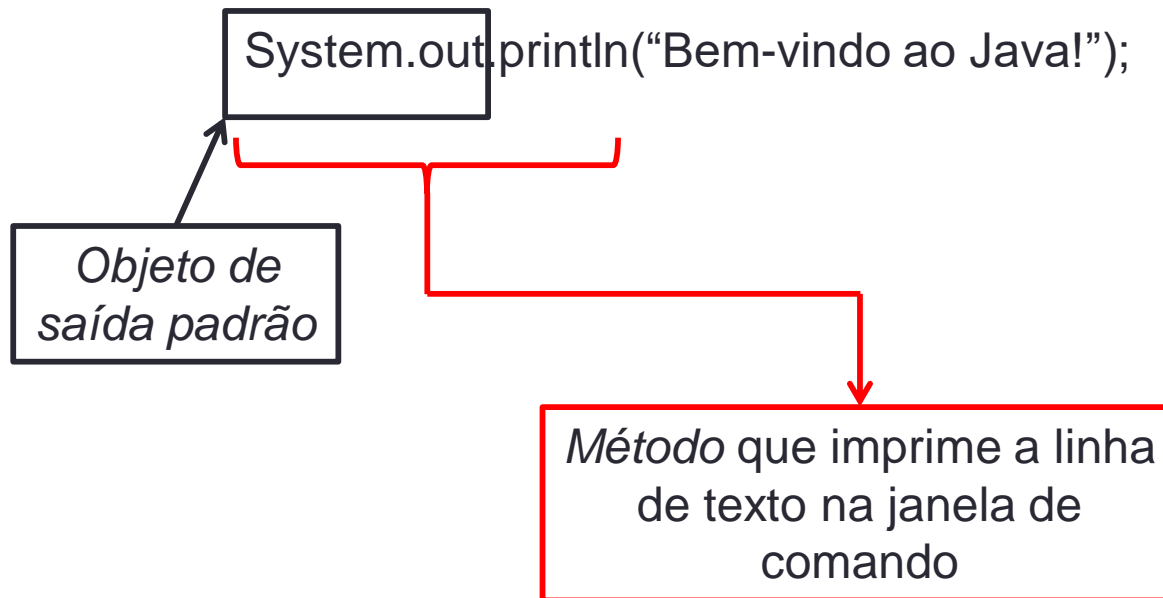
O Básico...

A classe public Bemvindo1 deve ser salva em um arquivo de nome **Welcome1.java**

```
1. public class Bemvindo1
2. {
3.     public static void main( String[ ] args )
4.     {
5.         System.out.println("Bem-vindo ao Java!");
6.     } //fim do método main
7. } //fim da classe Bemvindo1
```



O Básico...



Boas Práticas

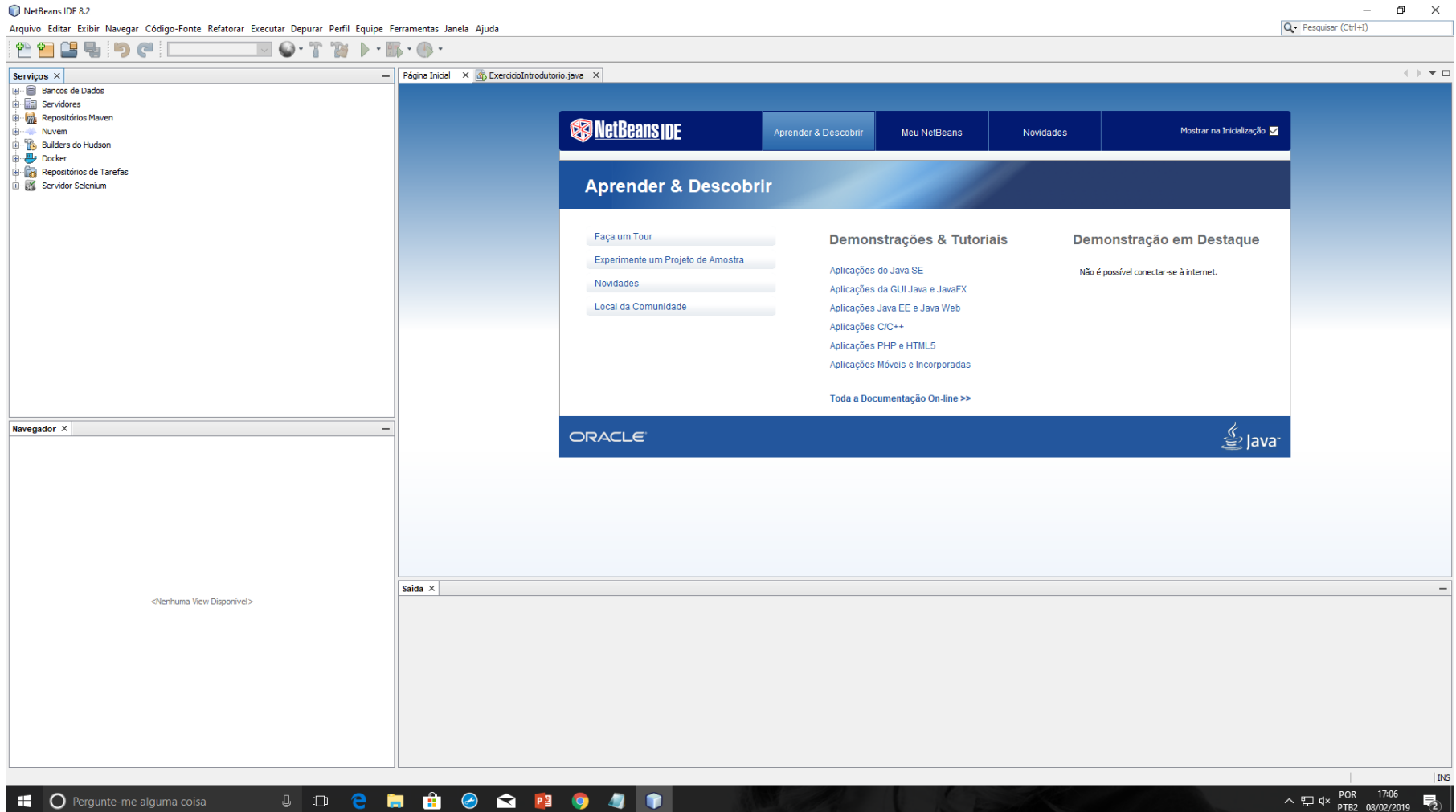
- Use comentários em seu programa
- Faça um cabeçalho que inclua:
 - Propósito do programa
 - Autor do programa
 - Data da última alteração
- Utilize linhas em branco para melhorar a legibilidade de seu programa
- *Versione.*

Exercício Introdutório

- Escreva um programa em Java que solicite ao usuário dois números inteiros e verifique se os mesmos são múltiplos entre si.

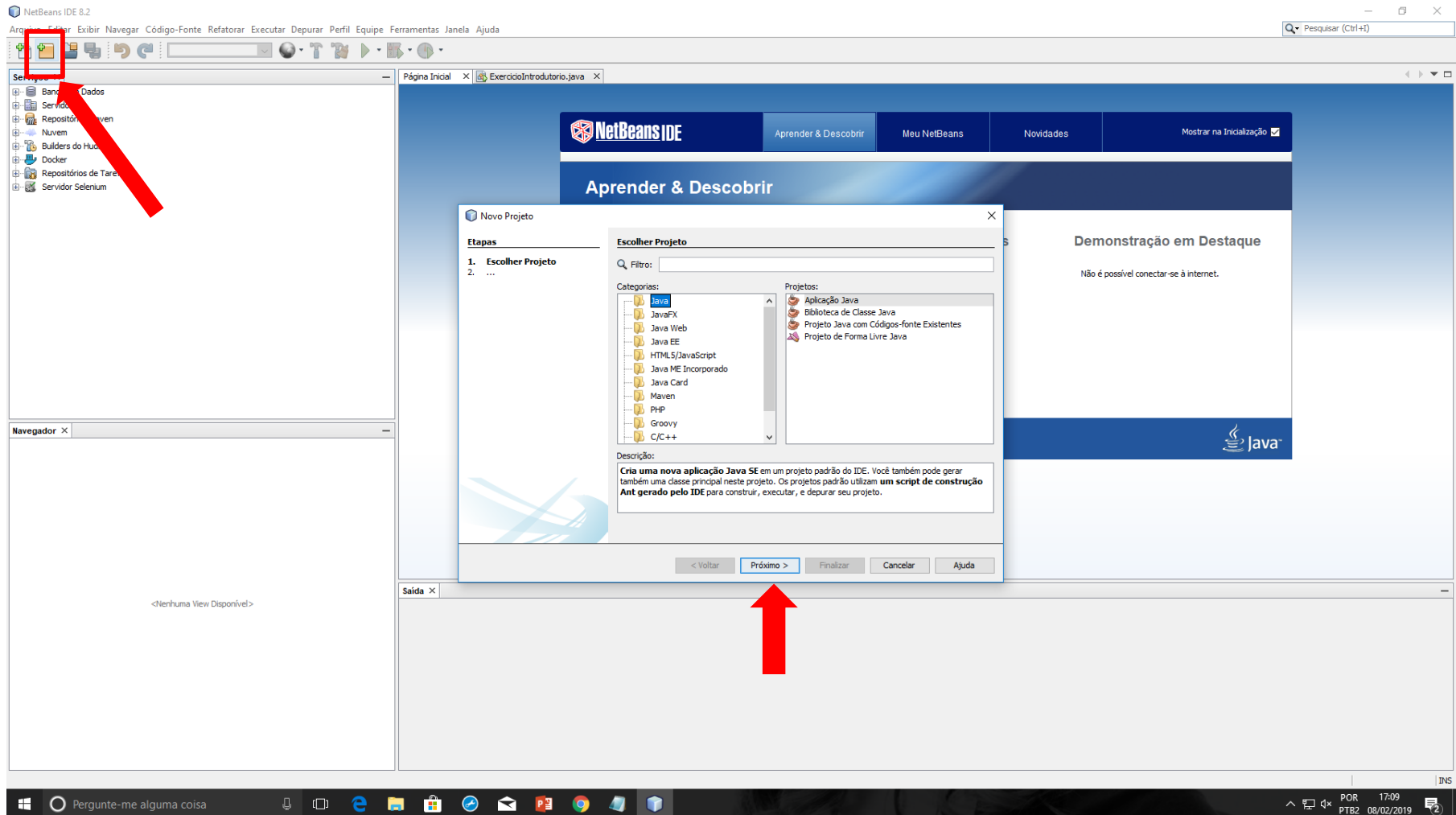
Resolução do exercício:

1. Abra o NetBeans



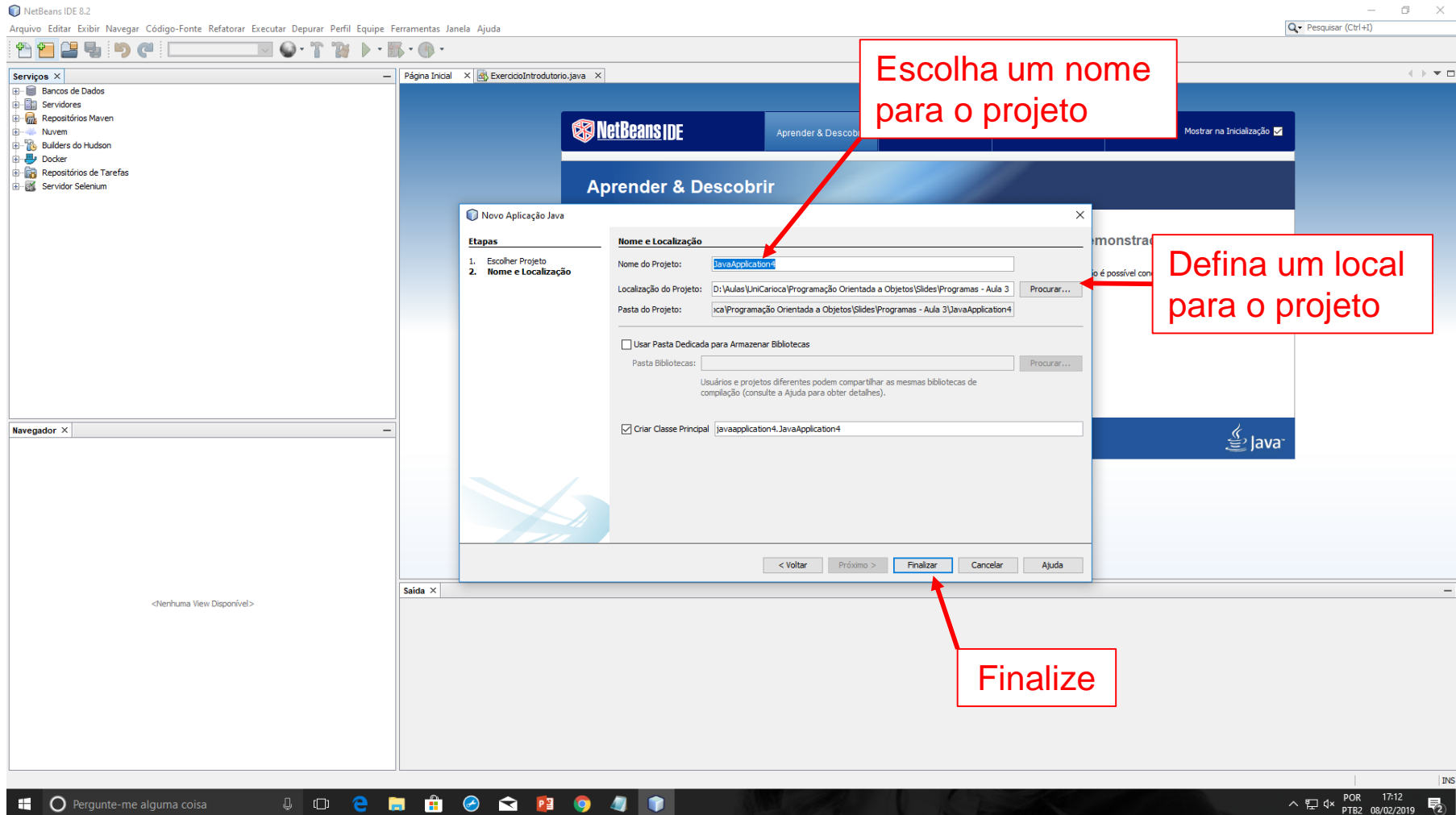
Resolução do exercício:

2. Crie um novo projeto



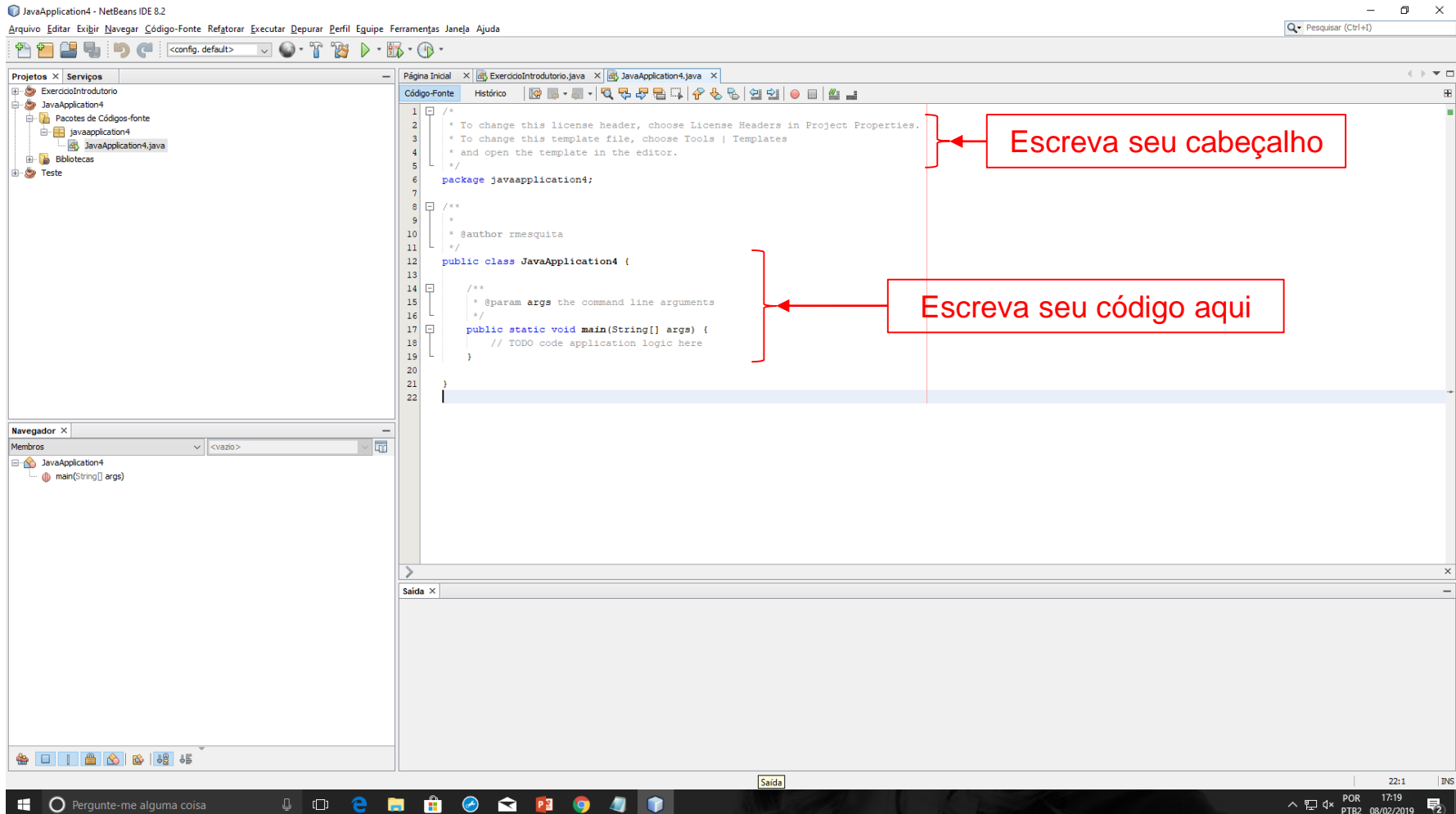
Resolução do exercício:

3. Dê um nome para o projeto



Resolução do exercício:

4. Escreva o código na tela de edição



Resolução do exercício:

Codificando...

- Será preciso fazer uma leitura a partir do teclado, então, use: `import java.util.Scanner;`
- Observe que a **classe** tem o mesmo nome do **arquivo**

```
import java.util.Scanner;
```

```
public class ExercicioIntrodutorio {
```

```
...
```

Resolução do exercício:

- Em seguida, vamos definir o método **main**

```
import java.util.Scanner;
```

```
public class ExercicioIntrodutorio {  
    public static void main( String[] args )  
    {  
        .  
        .  
        .  
    }  
}
```

- Antes de mais nada, vamos precisar de um objeto da classe Scanner:

```
Scanner input = new Scanner( System.in );
```

- E declaramos as variáveis...

```
int n1; // primeiro valor a ser digitado  
int n2; // segundo valor a ser digitado
```


Resolução do exercício:

- Agora, acrescentamos as operações necessárias:

```
System.out.print( "Digite o primeiro inteiro: " ); // prompt
n1 = input.nextInt(); // lê o primeiro valor digitado pelo usuário

System.out.print( "Digite o segundo inteiro: " ); // prompt
n2 = input.nextInt(); // lê o segundo valor digitado pelo usuário

if ((n1 % n2) == 0 || (n2 % n1) == 0){
    System.out.printf( "%d e %d são múltiplos entre si", n1, n2 );
} else
    System.out.printf( "%d e %d não são múltiplos entre si", n1, n2 );
```

Resolução do exercício:

- Ficou assim:

```
import java.util.Scanner;

public class ExercicioIntrodutorio {

    public static void main( String[] args )
    {
        Scanner input = new Scanner( System.in );

        int n1; // primeiro valor a ser digitado
        int n2; // segundo valor a ser digitado

        System.out.print( "Digite o primeiro inteiro: " ); // prompt
        n1 = input.nextInt(); // lê o primeiro valor digitado pelo usuário

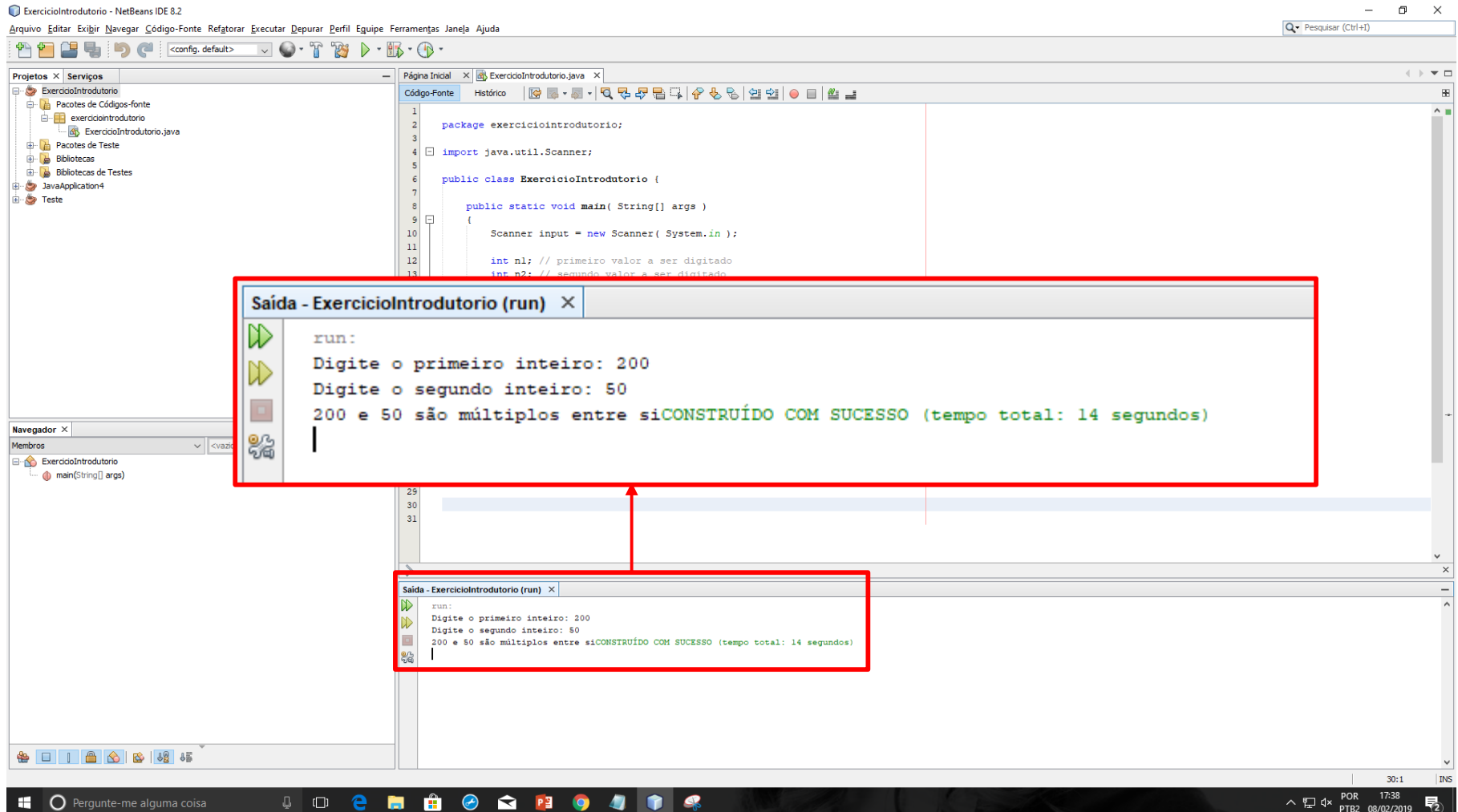
        System.out.print( "Digite o segundo inteiro: " ); // prompt
        n2 = input.nextInt(); // lê o segundo valor digitado pelo usuário

        if ( ((n1 % n2) == 0) || ((n2 % n1) == 0) ){
            System.out.printf( "%d e %d são múltiplos entre si", n1, n2 );
        } else
            System.out.printf( "%d e %d não são múltiplos entre si", n1, n2 );

    } // fim do método main
} // fim da classe ExercicioIntrodutorio
```

Resolução do exercício:

- Para compilar: f9; Execução: f6



Exercício proposto

- Crie um aplicativo “*Calculadora IMC*” que leia o peso do usuário em quilogramas e a altura em metros) e, então, calcule e exiba o *índice de massa corporal* dele. Onde:

$$IMC = \frac{peso}{altura}$$

- Além disso, que exiba as seguintes informações do *Department of Health and Human Services/National Institutes of Health*, assim o usuário pode avaliar o seu IMC:
 - Abaixo do peso: $IMC < 18.5$
 - Normal: $18.5 \leq IMC \leq 24.9$
 - Sobrepeso: $25 \leq IMC \leq 29.9$
 - Obeso: $IMC \geq 30$

Para a próxima aula...

- Vamos iniciar as práticas!