

Programação orientada a objetos em Java

Profª. Heloisa Moura

Programação orientada a objetos em Java

O que vamos ver:

- Atributos (de instância, de classe, locais) Variáveis;
- Passagem por valor/referência;
- Métodos estáticos;
- Relacionamento entre classes

Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributo de instância

Uma classe pode conter campos que são declarados fora de métodos, chamados de atributos de instância. São usados pelos objetos para armazenar seus estados.

Existem antes dos métodos serem chamados, durante sua execução e após terminarem sua tarefa. Seus valores são específicos de cada instância(objeto) e não são compartilhados entre as instâncias. Podem usar qualquer nível de acesso – public, private, protected ou default, e serem manipulados por todos os métodos da classe.

Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributo de instância

Exemplo:

```
private String nomeTime;
```


Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributo de classe

Os atributos de classe são campos que pertencem à classe em si, e não ao objeto instanciado. Eles são compartilhados por todos os objetos da classe. Dessa forma, quando você tem um atributo de classe, ao instanciar objetos, os objetos também terão aquele atributo, só que com uma diferença significativa em relação ao atributo de instância, o valor armazenado em um atributo de classe é o mesmo para todos os objetos e qualquer objeto poderá alterar esse valor, que será visível por todos os outros.

Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributo de classe

Exemplo:

```
private static String nomeSelecao;
```


Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributos locais – escopo de atributos

Os atributos locais, são aqueles declarados dentro de métodos ou em blocos de código. Os parâmetros do método também são locais a ele. Atributos locais servem somente no escopo daquele método ou bloco de código onde foi declarado.

Exemplo:

```
public int calculaValor(int valor) {  
    valor = valor + 10;  
    return valor;  
}
```


Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Atributos locais – escopo de atributos

Qual a saída/output na tela?

```
EscopoAtributosLocais escopo = new EscopoAtributosLocais();  
    escopo.setValor(10);  
    System.out.println(escopo.getValor());  
    System.out.println(escopo.calculaValor(20));  
    System.out.println(escopo.getValor());
```


Programação orientada a objetos em Java

Atributos (instância, classe, locais) Variáveis

Exemplo na pratica


```
public class EscopoAtributosLocais {
```

```
    private int valor;
```

Escopo: classe

```
    public void maiUmOutroTeste(int num){
```

Escopo: método
(parâmetros)

```
        int total = 0;
```

Escopo: método
(variáveis locais do método)

```
        try {
```

```
            int outroNum = 0;
```

Escopo: bloco
(variáveis locais declarada dentro
do bloco try, catch, if, else, while,
for, finally, etc)

```
            total = num/outroNum;
```

```
        } catch (Exception e){  
            e.printStackTrace();
```

```
        } finally {
```

```
            total++;
```

```
        }
```

```
        num++;
```

```
    }
```


Programação orientada a objetos em Java

Passagem por valor/referência

Passagem por valor por referência

Como é o comportamento do Java quando passamos parâmetros para os métodos?

Exemplo 1

Programação orientada a objetos em Java

Passagem por valor/referência

Passagem por valor por referência

```
private static void testePassagemValorReferencia(int valor, Aluno
aluno) {
    int novovalor = valor + 10;
    valor = novovalor;
    aluno = new Aluno("Jose 2", " 12365 ", "HTML");
}
```


Programação orientada a objetos em Java

Passagem por valor/referência

No main

```
Aluno aluno = new Aluno("Jose 1", "1234", "Java");  
int valor = 10;  
System.out.println(aluno);  
System.out.println(valor);  
  
testePassagemValorReferencia(valor, aluno);  
System.out.println(aluno);  
System.out.println(valor);
```

Qual a saída dos atributos valor e aluno?

Programação orientada a objetos em Java

Passagem por valor/referência

Exemplo 2

```
private static void testePassagemValorReferencia2(int valor, Aluno
aluno) {
    int novovalor = valor + 10;
    valor = novovalor;
    aluno.setNome("Jose" + valor);
}
```


Programação orientada a objetos em Java

Passagem por valor/referência

No main

```
Aluno aluno = new Aluno("Jose 1", "1234", "Java");
```

```
int valor = 10;
```

```
System.out.println(aluno);
```

```
System.out.println(valor);
```

```
testePassagemValorReferencia2(valor, aluno);
```

```
System.out.println(aluno);
```

```
System.out.println(valor);
```

E agora? Qual a saída dos atributos valor e aluno?

Programação orientada a objetos em Java

Passagem por valor/referência

Como funciona a passagem de parâmetros? Exemplo 1

```
Aluno aluno = new Aluno("Jose 1", "1234", "Java");
```

```
int valor = 10;
```

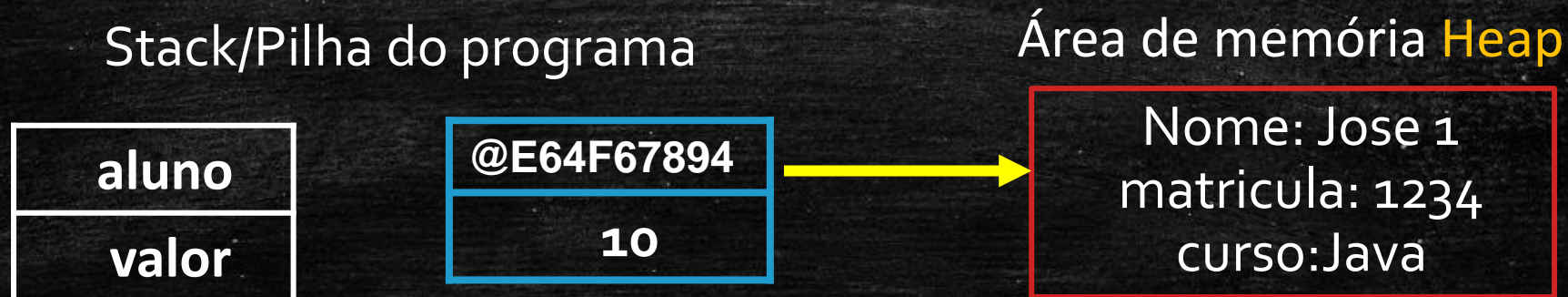
```
System.out.println(aluno);
```

```
System.out.println(valor);
```

```
testePassagemValorReferencia(valor, aluno);
```

```
System.out.println(aluno);
```

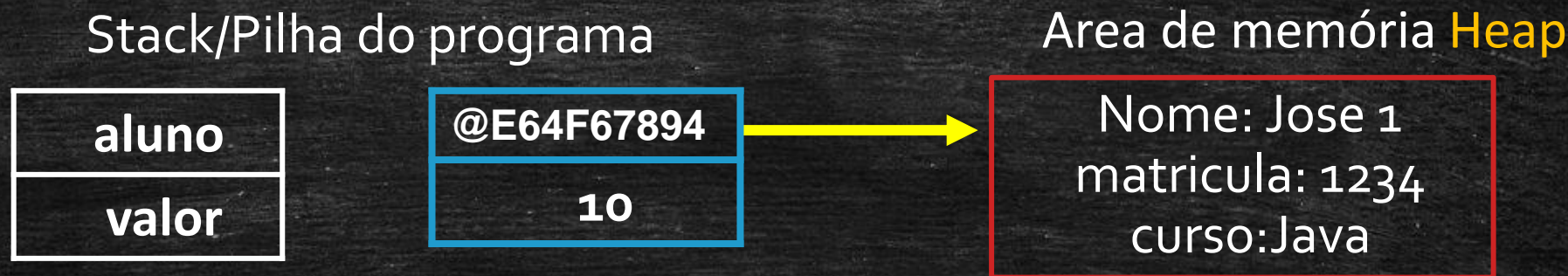
```
System.out.println(valor);
```



Programação orientada a objetos em Java

Passagem por valor/referência

Como funciona a passagem de parâmetros? Exemplo 2



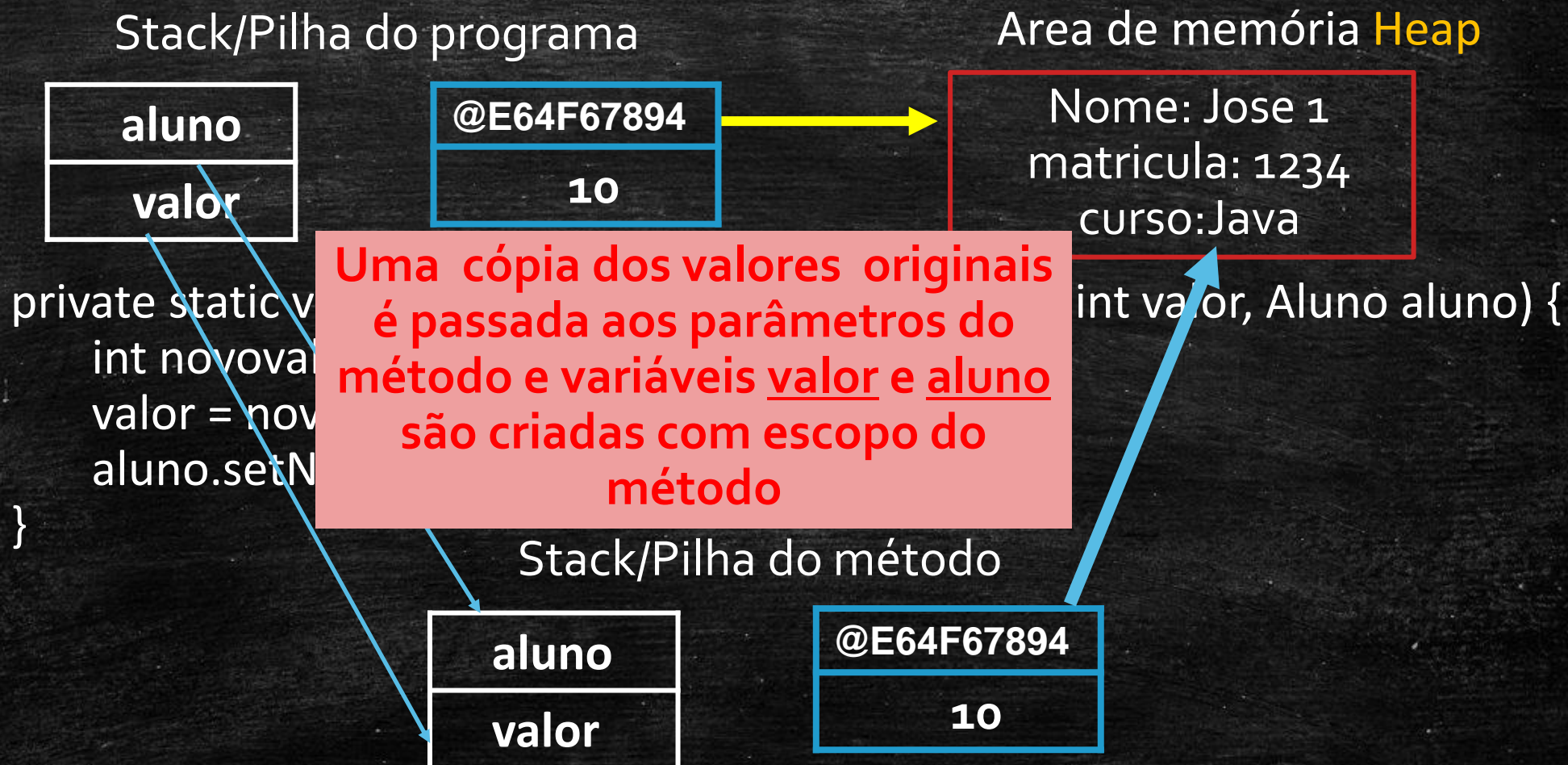
```
private static void testePassagemValorReferencia2(int valor, Aluno aluno) {  
    int novovalor = valor + 10;  
    valor = novovalor;  
    aluno.setNome("Jose" + valor);  
}
```



Programação orientada a objetos em Java

Passagem por valor/referência

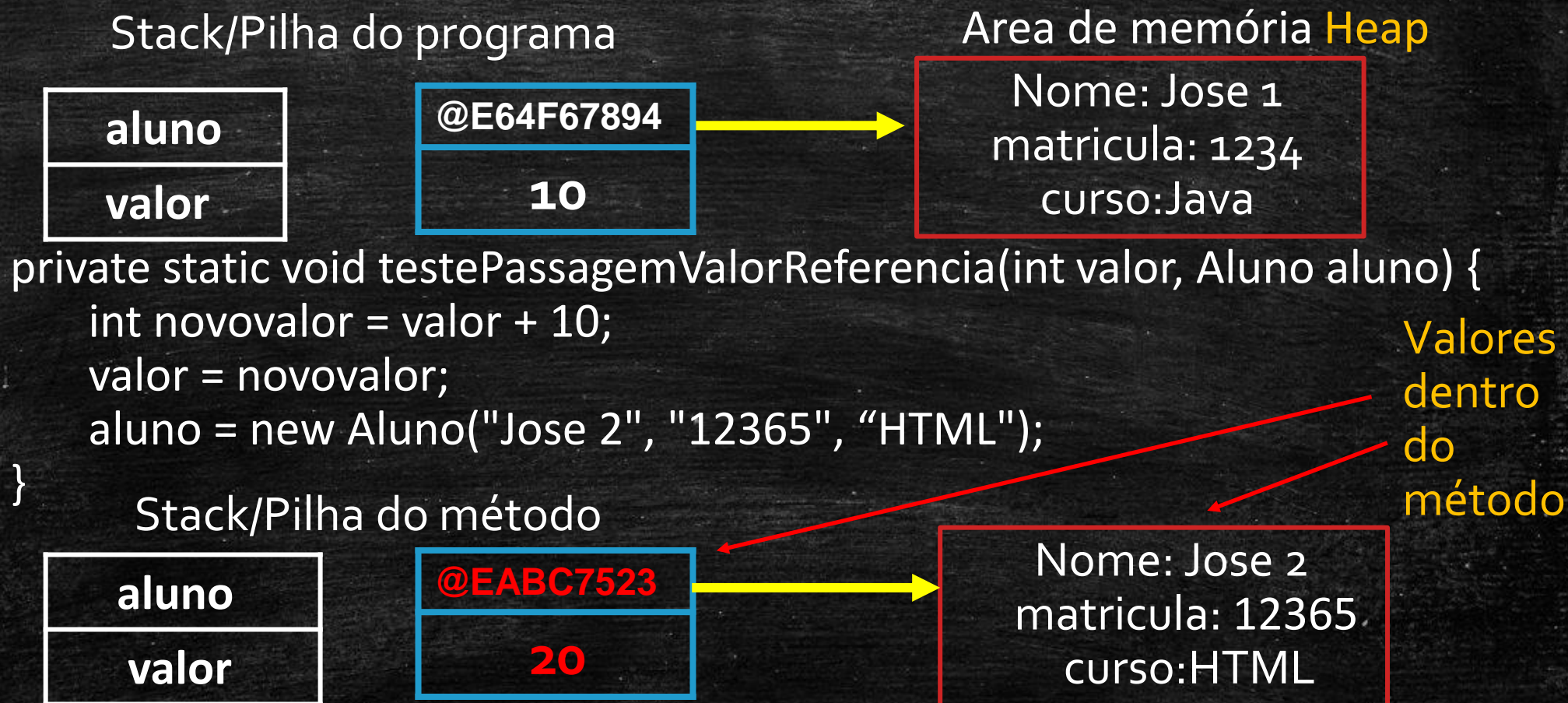
Como funciona a passagem de parâmetros? Exemplo 2



Programação orientada a objetos em Java

Passagem por valor/referência

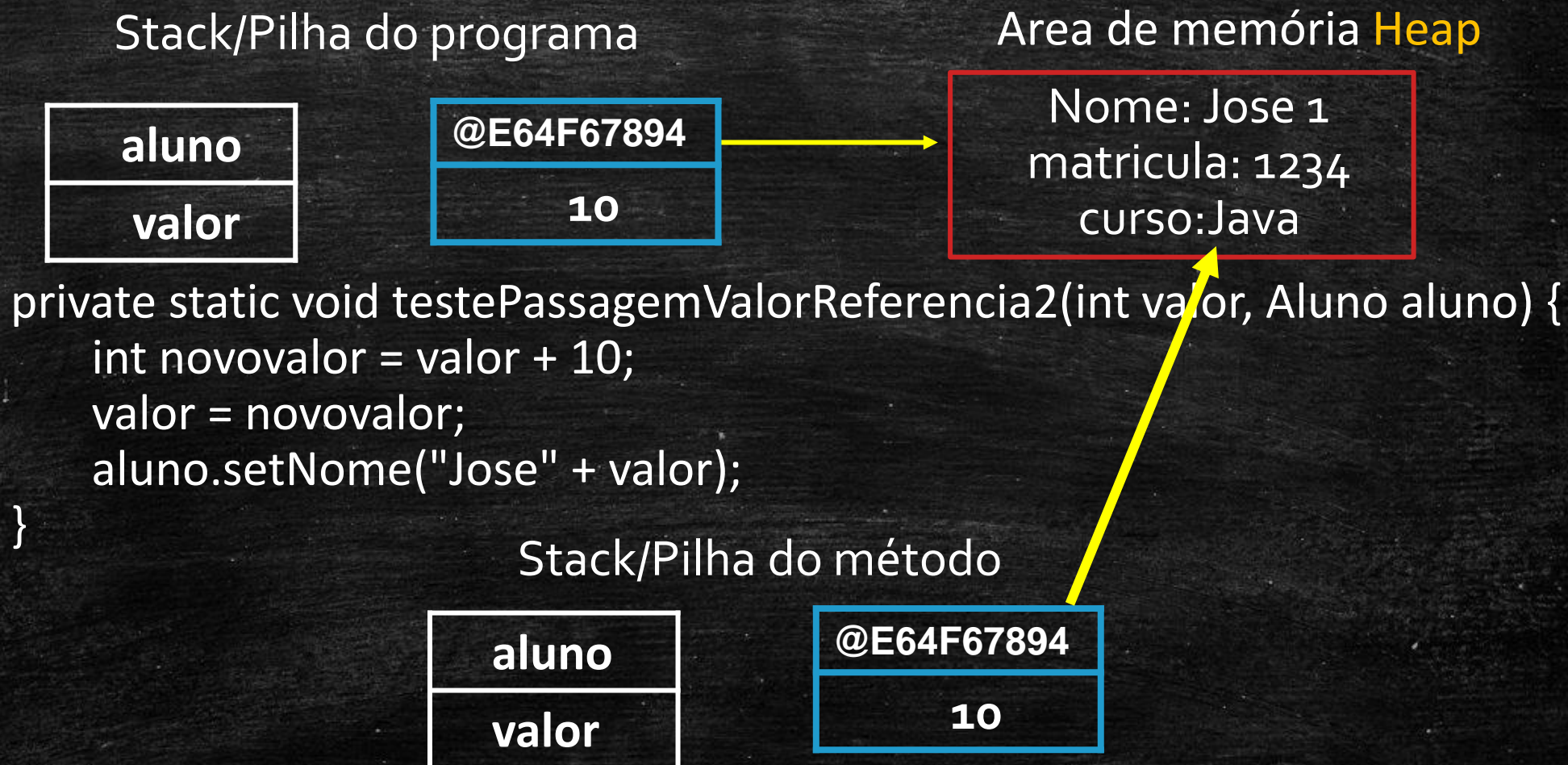
Como funciona a passagem de parâmetros? Exemplo 1



Programação orientada a objetos em Java

Passagem por valor/referência

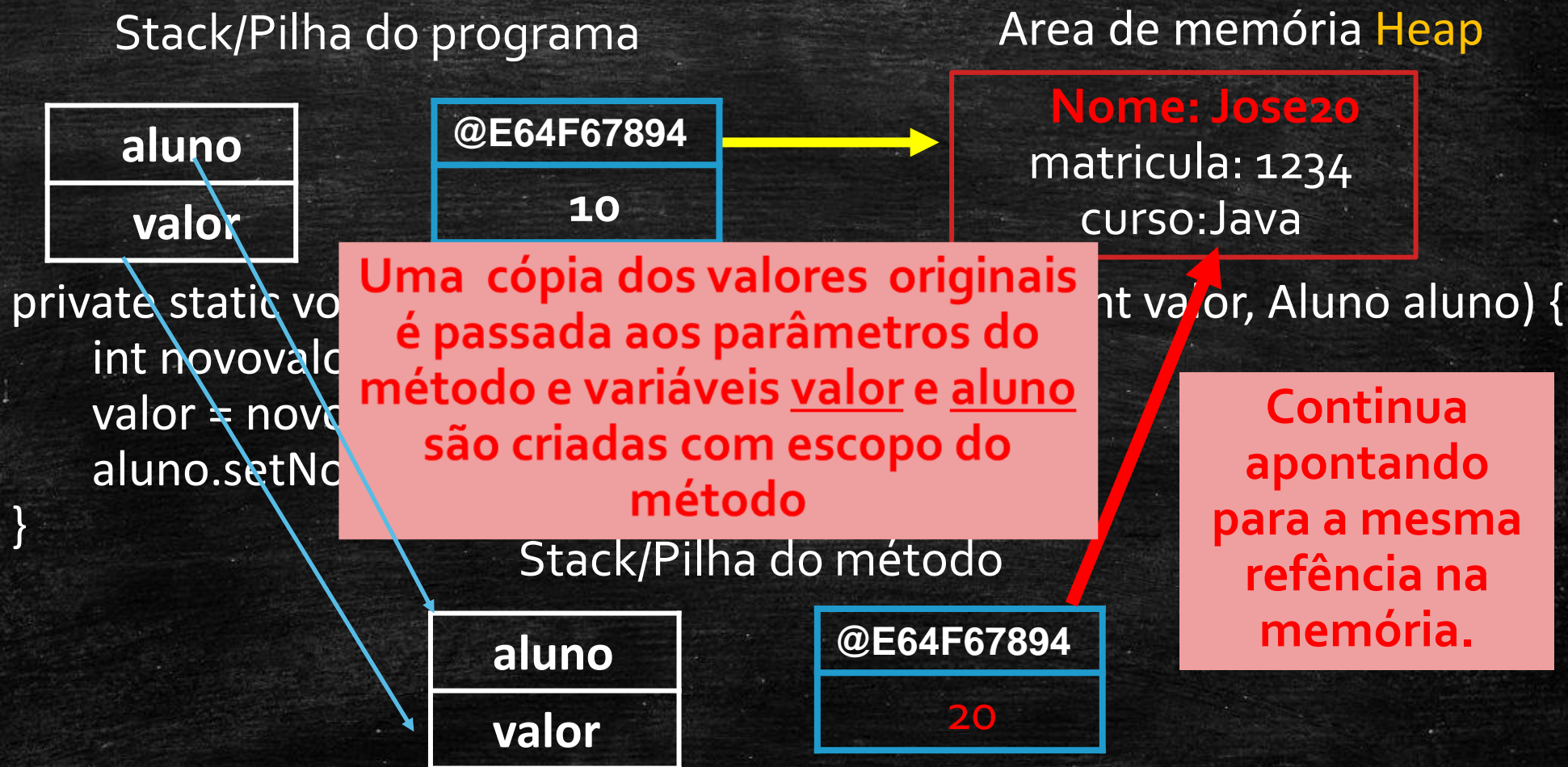
Como funciona a passagem de parâmetros? Exemplo 2



Programação orientada a objetos em Java

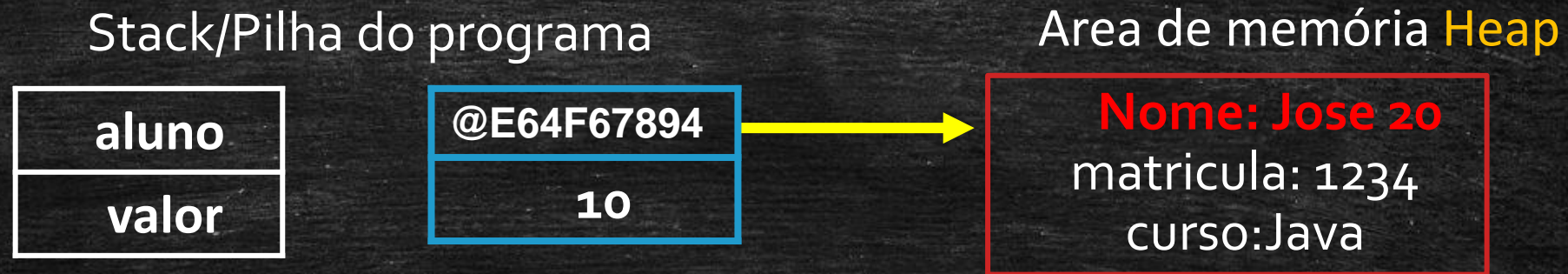
Passagem por valor/referência

Como funciona a passagem de parâmetros? Exemplo 2



Programação orientada a objetos em Java

Passagem por valor/referência



Em resumo...

Tipos primitivos: passagem por **valor**

Tipos classe, enum, array: passagem por **referência**

Programação orientada a objetos em Java

Métodos estáticos

Métodos estáticos

A linguagem Java nos permite ter atributos, blocos e métodos do tipo static. Assim como um atributo static, um método do tipo static, não faz parte da instância de uma classe e sim da classe. Quando criamos uma classe com membros estáticos, esses serão os primeiros códigos a serem carregados, mesmo que não haja nenhuma instância desta classe.

O método estático mais conhecido é sem dúvida nenhuma o main(), utilizado para inicializar uma aplicação desenvolvida em Java. Dentro de um método ou de um bloco de inicialização estático, só podemos acessar variáveis ou outros métodos estáticos, e para acessarmos membros não estáticos precisamos então criar uma instância da classe.

Programação orientada a objetos em Java

Métodos estáticos

Métodos estáticos

A classe Math do pacote java.lang é um exemplo de classe que possui seus membros (atributos e métodos) declarados como static.

Em que situação criamos métodos estáticos?

Quando não precisamos ter uma instância de uma classe, para utilizar os membros dela.

Quando não temos muita lógica de programação dentro dos métodos. Essa classe passar a ser uma classe utilitária.

Exemplo na prática

MinhaCalculadora

Programação orientada a objetos em Java

Relacionamento entre classes

Relacionamento entre classes

Objetos do mundo real relacionam-se uns com os outros de diversas formas:

Um objeto motor **é parte de** um objeto carro.

Um objeto turma **tem vários** objetos alunos.

Um objeto botão **tem um** objeto tratador de eventos.

Tipos de relacionamentos entre objetos de software:

Agregação: estabelecem um vínculo entre objetos.

Composição: relacionamento do tipo todo/parte.

Uso: um objeto usa a funcionalidade de outro sem estabelecer vínculo duradouro (referências).

Programação orientada a objetos em Java

Relacionamento entre classes

Relacionamento entre classes

Agregação: Forma de composição em que o objeto composto apenas usa ou tem conhecimento da existência do(s) objeto(s) componente(s). Os objetos componentes podem existir sem o agregado e vice-versa.

Composição: Forma de associação em que o objeto composto é responsável pela existência dos componentes. O componente não tem sentido fora da composição.

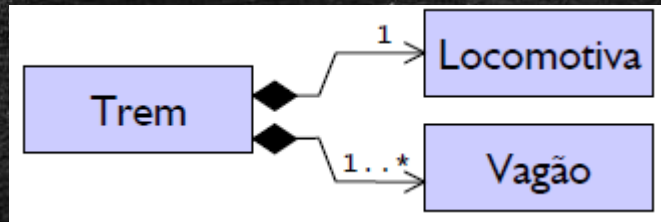
Uma classe pode ter referências a objetos de outras classes como membros. Isso é chamado de **composição** e, as vezes, é referido como um **relacionamento tem um**.

Programação orientada a objetos em Java

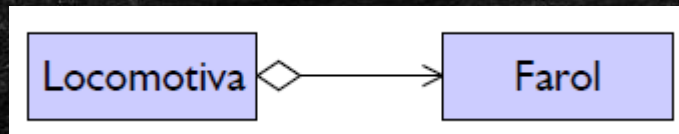
Relacionamento entre classes

Relacionamento entre classes - Exemplo:

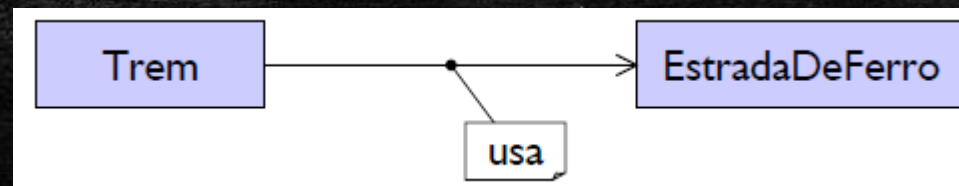
Composição: um trem é formado por locomotiva e vagões.



Agregação: uma locomotiva tem um farol (mas não vai deixar de ser uma locomotiva se não o tiver)



Associação: um trem usa uma estrada de ferro (não faz parte do trem, mas ele depende dela)



Programação orientada a objetos em Java

Relacionamento entre classes

Relacionamento entre classes

Outro Exemplo:

Uma conta corrente é formada por várias transações de crédito e débito → **composição**

Um cadastro de clientes é formado por vários clientes → **agregação**

Um cliente tem uma conta-corrente → **agregação**

Um documento possui um conjunto de parágrafos → **composição**

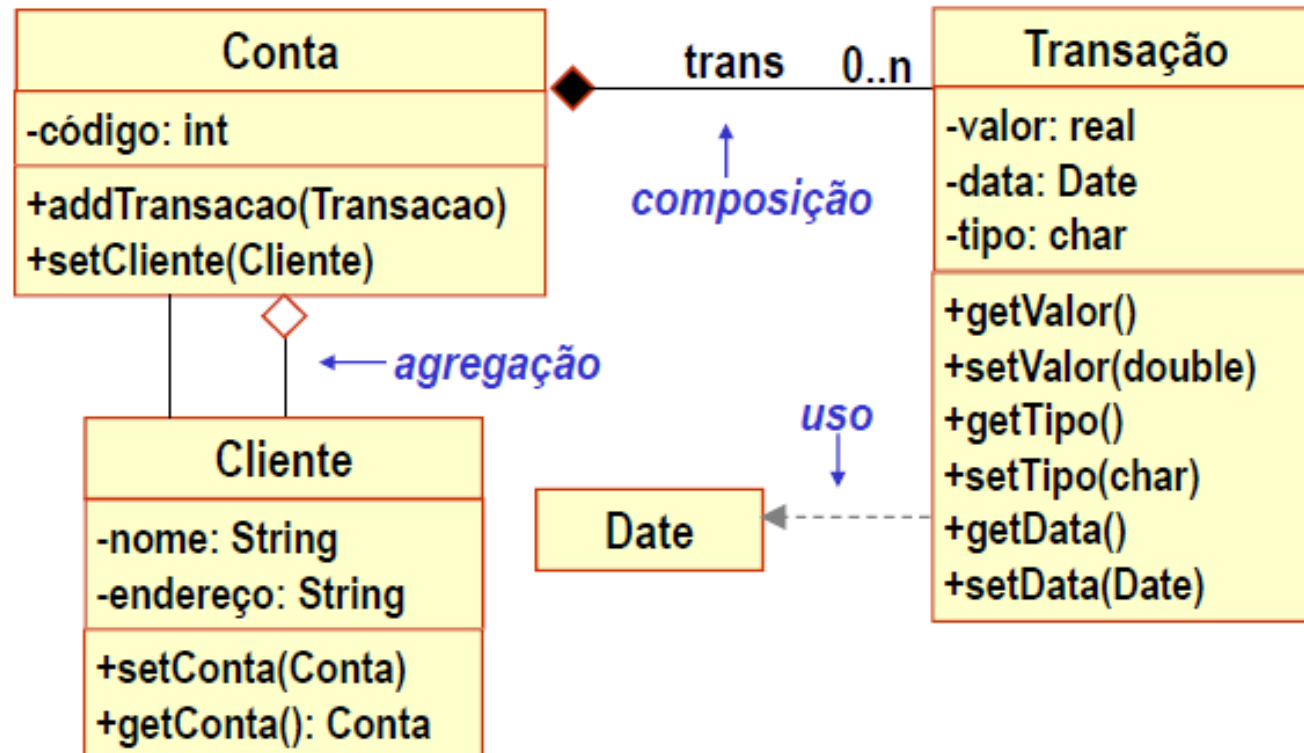
Uma turma é um conjunto de alunos → **composição**

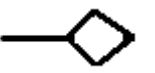
Programação orientada a objetos em Java


Relacionamento entre classes

Relacionamento entre classes

Notação UML para definir composições, agregações e uso:



•  Se aberto, tem-se **Agregação** Simples.

•  Se escuro, têm-se **Agregação por Composição**, ou simplesmente **Composição**.

Programação orientada a objetos em Java

Relacionamento entre classes

Relacionamento entre classes

Implementação em Java:

```
public class Conta {  
    private int codigo;  
    private Cliente cliente;  
    private Transacao[] trans;  
    private int qTransacoes;  
    //...  
}
```

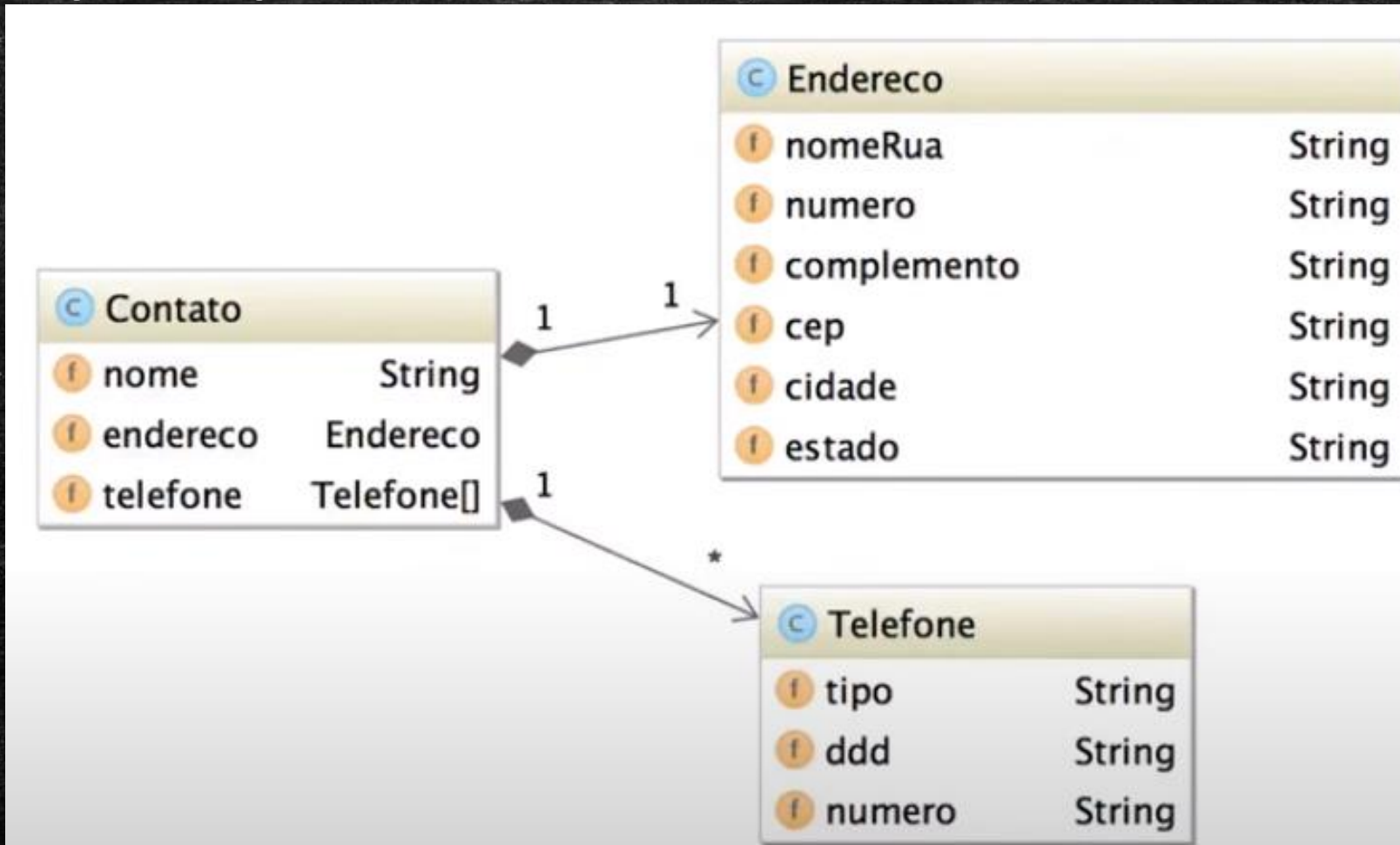
```
public class Cliente {  
    private String nome;  
    private Conta conta;  
    private String endereço;  
    //...  
}
```

```
public class Transacao {  
    private double valor;  
    private char tipo;  
    private Date data;  
    //...  
}
```


Programação orientada a objetos em Java

Relacionamento entre classes

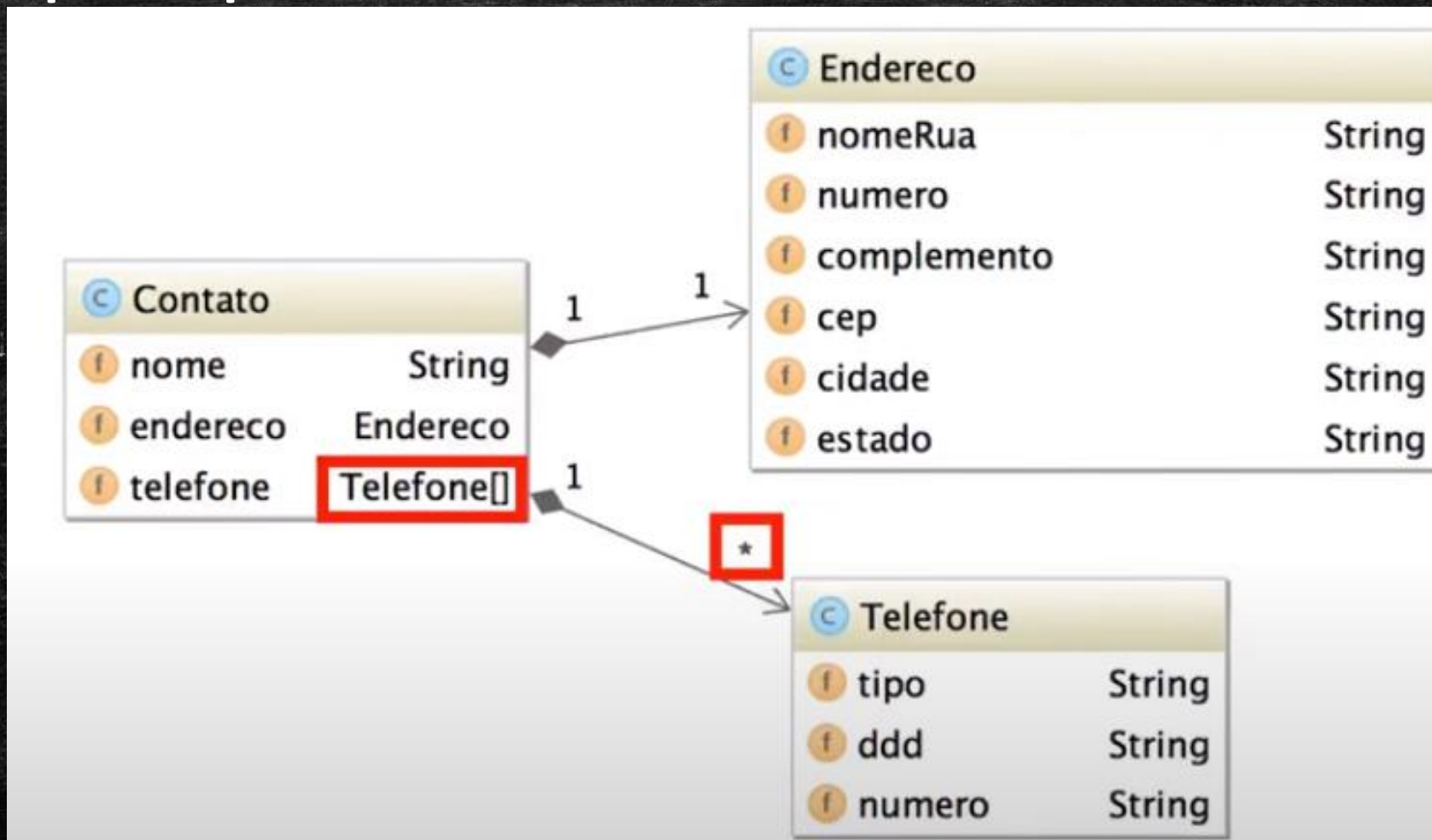
Exemplo na pratica: Relacionamento tem um



Programação orientada a objetos em Java

Relacionamento entre classes

Exemplo na pratica: Relacionamento tem muitos



Programação orientada a objetos em Java

Exercícios

Exercícios