

# POOI

# PROGRAMAÇÃO ORIENTADA A OBJETOS I

*Ramon Lummertz*



Introdução a POO

Classe e Objeto

Método

Herança

Encapsulamento

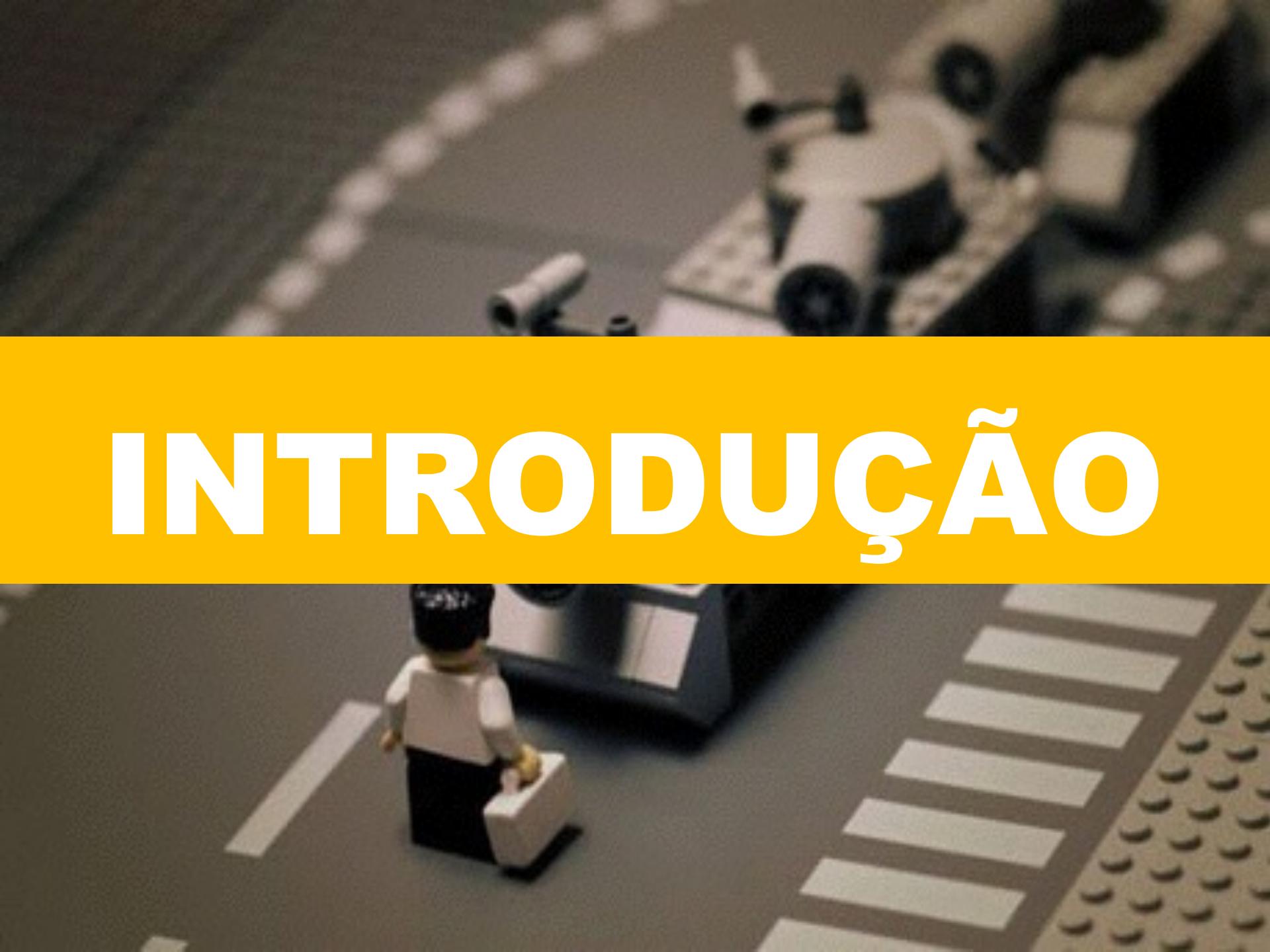
Abstração

Polimorfismo

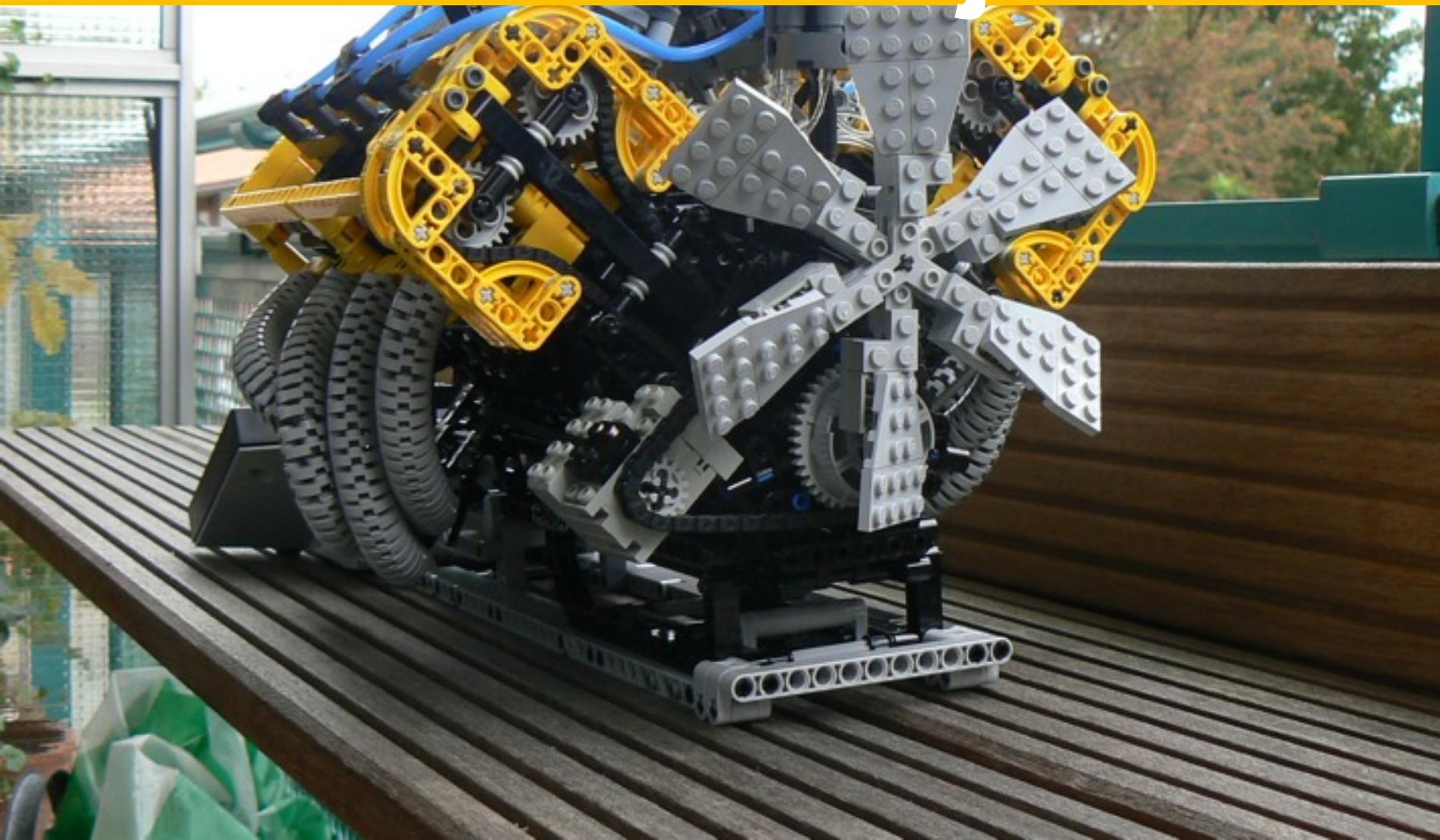
Interface

Coesão / Acoplamento

# INTRODUÇÃO



# Programação Orientada a Objetos?



# Programação Orientada a Objetos?

É UM PARADIGMA  
DE PROGRAMAÇÃO



# Programação Orientada a Objetos?



UM ESTILO  
DE PROGRAMAÇÃO

# Programação Orientada a Objetos?

Lógico

Prolog – Popler –  
QLISP

Imperativo/  
Procedural

Pascal – C - Basic

Funcional/Descritiva

ML – F# - Haskell

E muito mais... Orientada a Teste, Escalar, Restritiva,  
Genérica....

EXISTEM  
OUTRAS

# Programação Orientada a Objetos?

## DIFICULDADES

- Complexidade no aprendizado em comparação com a programação estruturada
- Seus conceitos são de difícil compreensão

## BENEFÍCIOS

- Mais fácil descrever o mundo real através dos objetos
- O encapsulamento facilita a manutenção do código
- Maior facilidade para reutilização de código

# Programação Orientada a Objetos?

## RESUMO

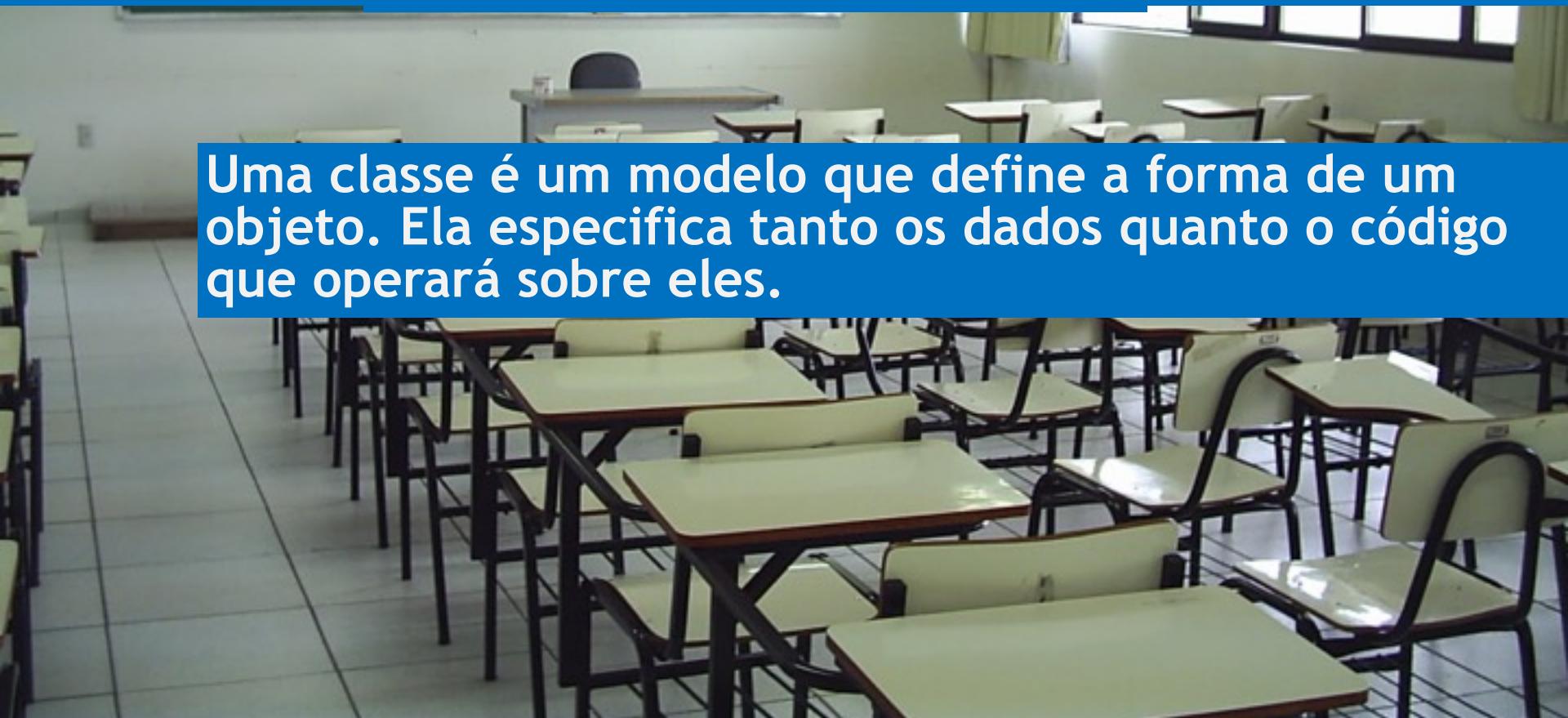
Um **modelo de programação** ou **paradigma de programação** é um conjunto de princípios, ideias, conceitos e abstrações utilizado para o desenvolvimento de uma aplicação.

Orientação à objetos é uma maneira de programar que ajuda na organização e resolve muitos problemas enfrentados pela programação procedural.

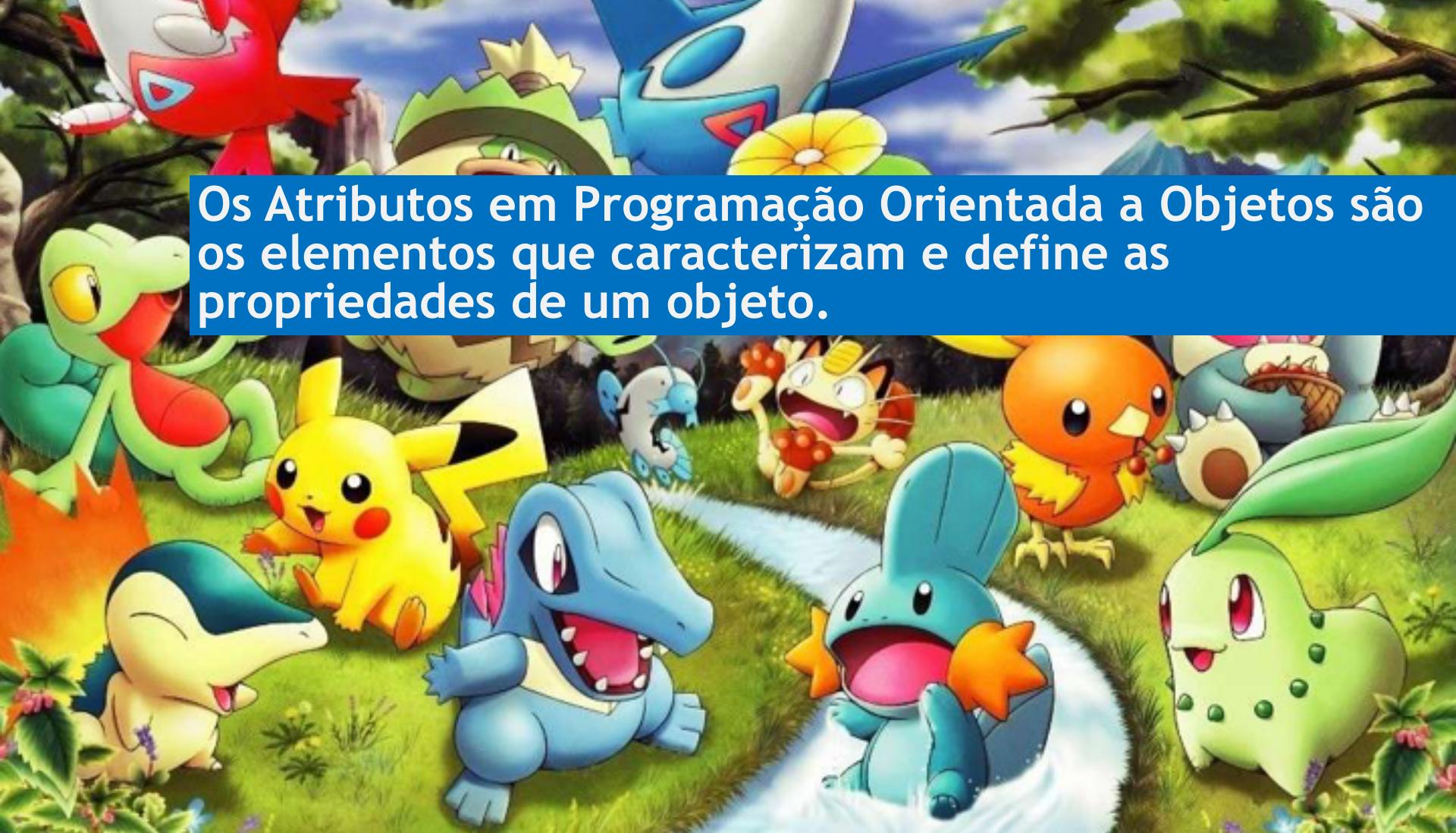
POO é organizar o mundo real como uma coleção de objetos que incorporam estrutura de dados e um conjunto de operações que manipulam estes dados e trocam mensagens entre si. [http://www.macoratti.net/oo\\_conc2.htm](http://www.macoratti.net/oo_conc2.htm)

# CLASSE

Uma classe é um modelo que define a forma de um objeto. Ela especifica tanto os dados quanto o código que operará sobre eles.



# Classe: Atributos



Os Atributos em Programação Orientada a Objetos são os elementos que caracterizam e define as propriedades de um objeto.

# Classe: Métodos

Os métodos são componentes de uma classe, assim como os atributos, entretanto os métodos definem ações que o objeto pode ter/realizar.

# Ações



# Classe: Visibilidade

## + Public

Quem tem acesso à classe tem acesso também a qualquer membro com visibilidade public, é raro ter atributos públicos, mas é comum ter métodos públicos.

<http://www.dsc.ufcg.edu.br/~jacques/cursos/p2/html/oo/visibilidade.htm>

## - Private

O membro private não é acessível fora da classe, a intenção é que apenas quem escreve a classe possa usar esses membros.

<http://www.dsc.ufcg.edu.br/~jacques/cursos/p2/html/oo/visibilidade.htm>

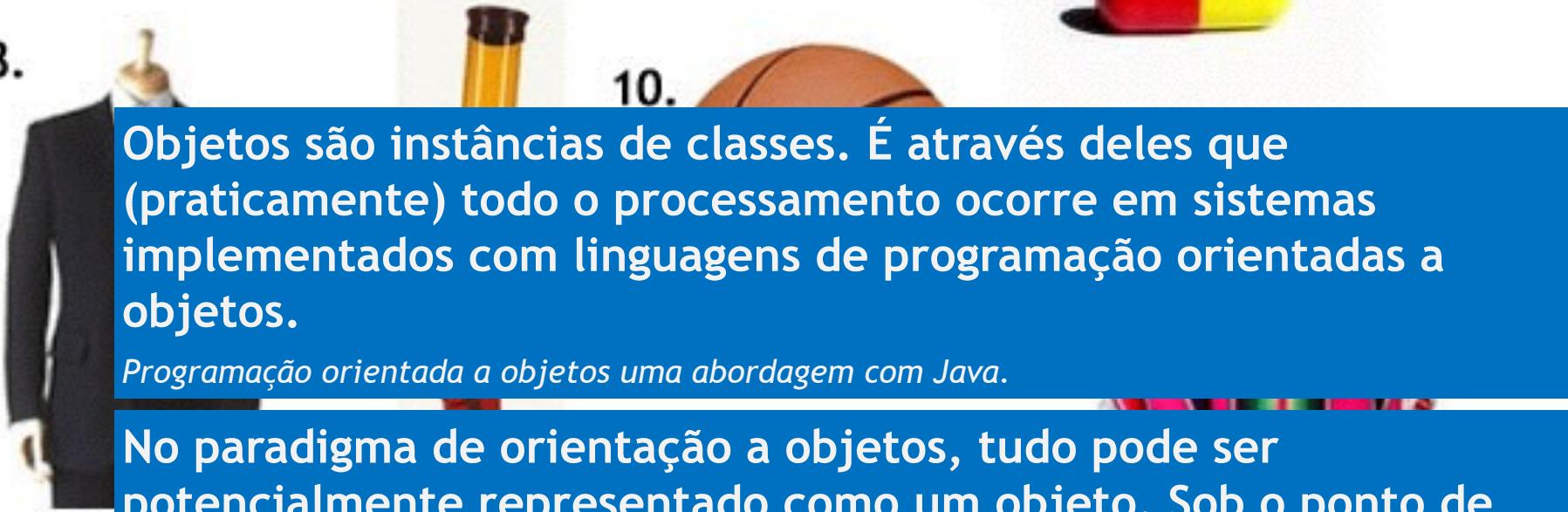
## # Protected

O membro protected é acessível à classe e a suas subclasses, a intenção é dar acesso aos programadores que estenderão sua classe.

<http://www.dsc.ufcg.edu.br/~jacques/cursos/p2/html/oo/visibilidade.htm>

# Classe: Objeto

8.



10.

No paradigma de orientação a objetos, tudo pode ser potencialmente representado como um objeto. Sob o ponto de vista da programação orientada a objetos, um objeto não é muito diferente de uma variável normal.

13.

Programação orientada a objetos uma abordagem com Java.

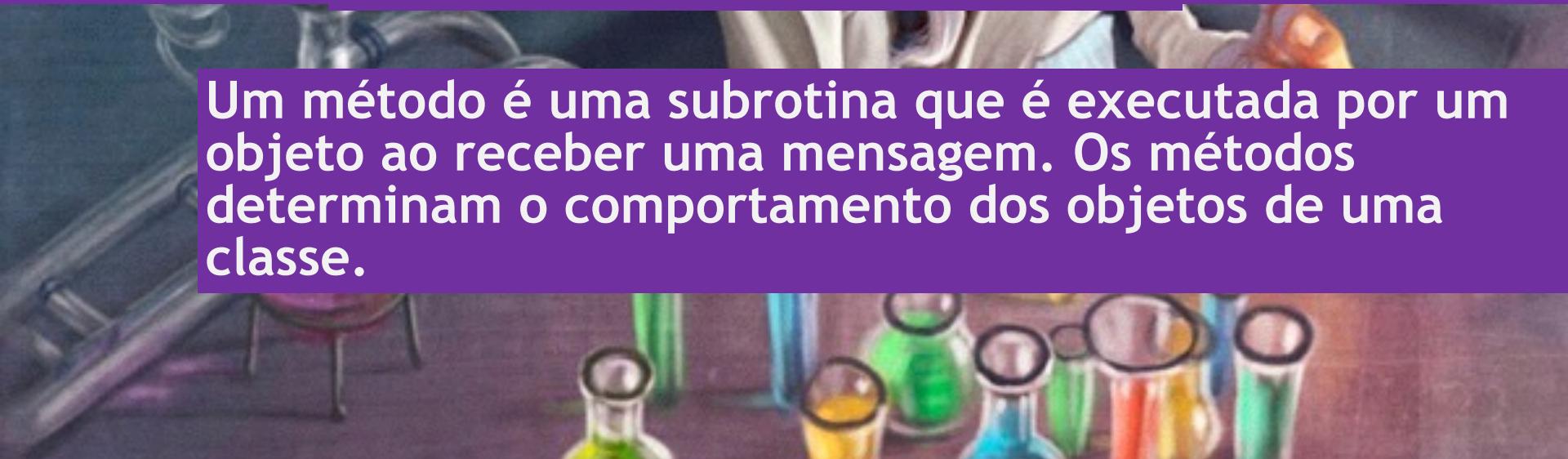
# Classe: Construtores



Um construtor é uma estrutura especial da classe, que tem como objetivo definir a configuração inicial de um objeto. É utilizado quando existem atributos da classe que são essenciais para o funcionamento do objeto, porém, são atributos de instância, e, assim, variam de acordo com cada objeto.

A cartoon illustration of a scientist with wild grey hair and two sets of glasses. He is wearing a lab coat and holding a green globe. In the background, there are laboratory glassware like flasks and beakers.

# MÉTODO

A photograph of a laboratory setup. In the foreground, there's a purple Erlenmeyer flask on a stand. Behind it, several test tubes filled with different colored liquids (green, yellow, blue, red) are arranged in a rack. A person's hands are visible, holding one of the test tubes.

Um método é uma subrotina que é executada por um objeto ao receber uma mensagem. Os métodos determinam o comportamento dos objetos de uma classe.

# Método: Assinatura

Cada método é especificado por uma assinatura, composta por um identificador para o método (o nome do método), o tipo para o valor de retorno e sua lista de argumentos, sendo cada argumento identificado por seu tipo e nome.

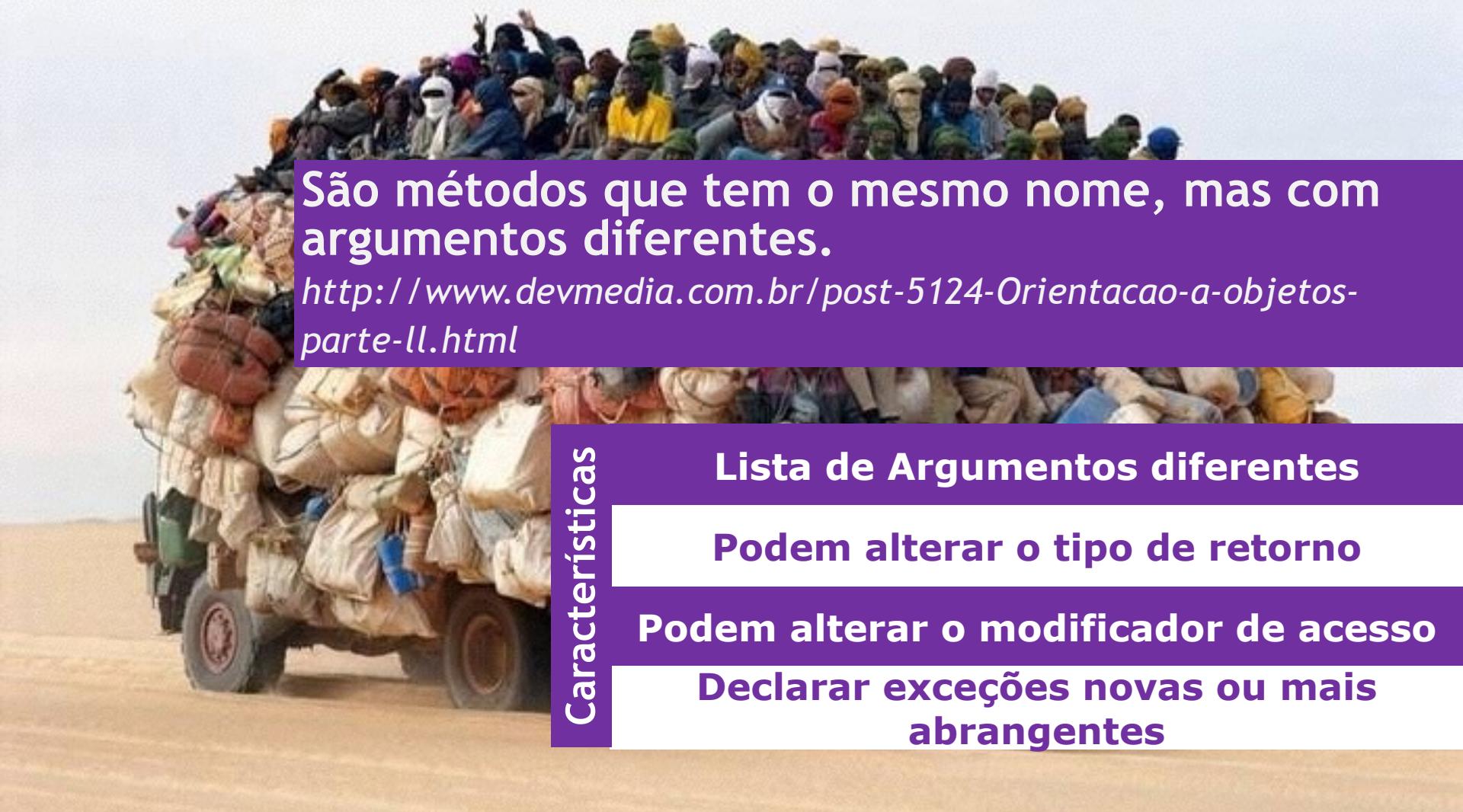
*Programação orientada a objetos uma abordagem com Java.*

```
[modificador] tipo nome (argumentos)  
{  
    corpo do método  
}
```

Através do mecanismo de sobrecarga (overloading), dois métodos de uma classe podem ter o mesmo nome, desde que suas assinaturas de argumentos sejam diferentes.

*Programação orientada a objetos uma abordagem com Java.*

# Método: Sobrecarga



São métodos que tem o mesmo nome, mas com argumentos diferentes.

<http://www.devmedia.com.br/post-5124-Orientacao-a-objetos-parte-II.html>

Características

**Lista de Argumentos diferentes**

**Podem alterar o tipo de retorno**

**Podem alterar o modificador de acesso**

**Declarar exceções novas ou mais abrangentes**

# Método: Mensagens

Um programa orientado a objetos é composto por um conjunto de objetos que interagem através de “trocas de mensagens”. Na prática, essa troca de mensagem traduz-se na aplicação de métodos a objetos.

 **ULBRA** *Programação orientada a objetos uma abordagem com Java.*

Neste domingo iniciam as comemorações dos 43 anos da ULBRA. Confira a programação e participe das atividades que... [fb.me/7r5ph8JQN](http://fb.me/7r5ph8JQN)

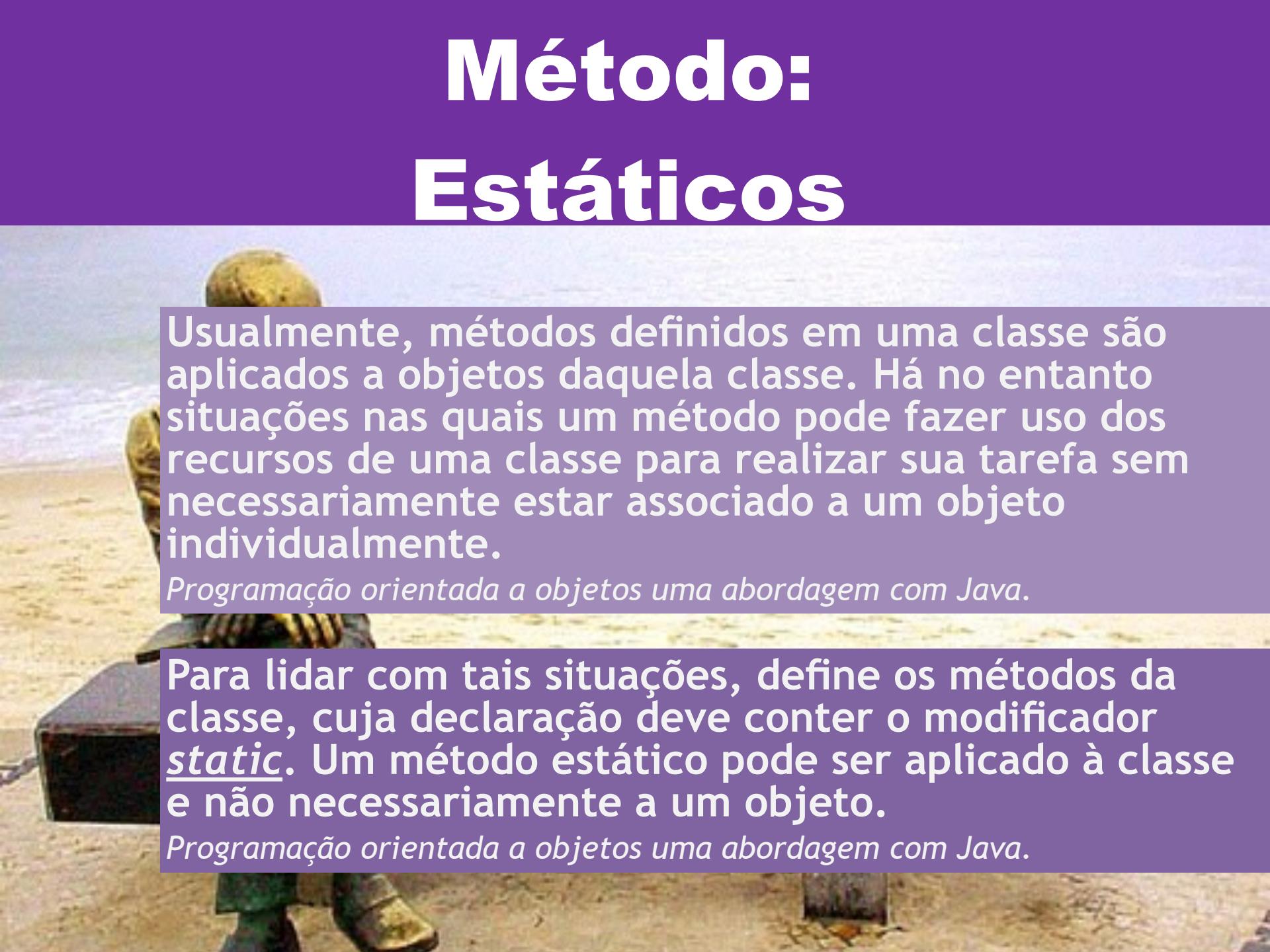
[View translation](#)

RETWEET

1



# Método: Estáticos



Usualmente, métodos definidos em uma classe são aplicados a objetos daquela classe. Há no entanto situações nas quais um método pode fazer uso dos recursos de uma classe para realizar sua tarefa sem necessariamente estar associado a um objeto individualmente.

*Programação orientada a objetos uma abordagem com Java.*

Para lidar com tais situações, define os métodos da classe, cuja declaração deve conter o modificador **static**. Um método estático pode ser aplicado à classe e não necessariamente a um objeto.

*Programação orientada a objetos uma abordagem com Java.*



# HERANÇA

Herança é um mecanismo que permite que características comuns a diversas classes sejam fatoradas em uma classe pai, ou superclasse. A partir de uma classe base, outras classes podem ser especificadas.

*Programação orientada a objetos uma abordagem com Java.*

# Herança: Sobreposição

A sobreposição refere-se à redefinição de métodos na hierarquia da herança, de forma que estes métodos implementam definições diferentes (mais especializadas) nos subtipos.



# ENCAPSULAMENTO

É o princípio pelo qual cada componente de um programa deve agregar toda a informação relevante para sua manipulação como uma unidade (uma cápsula). Aliado ao conceito de ocultamento de informação.

*Programação orientada a objetos uma abordagem com Java.*

# ABSTRAÇÃO

Abstração é o processo de extrair as características essenciais de um objeto real. A abstração é necessária para se ter um modelo fiel da realidade sobre a qual se possa operar.

O conjunto de características resultante da abstração forma um tipo de dado abstrato com informações sobre seu estado e comportamento.



# POLIMORFISMO

O polimorfismo permite que um mesmo método ou classe pode apresentar várias formas, de acordo com o seu contexto.



# INTERFACE

É um conjunto de métodos que um objeto deve suportar, mas contendo apenas a especificação da funcionalidade que uma classe deve conter, sem determinar como essa funcionalidade deve ser implementadas.

*Programação orientada a objetos uma abordagem com Java.*

# COESÃO / ACOPLAMENTO

Acoplamento é o grau que uma classe depende, interage com outra classe. Coesão refere a finalidade que a classe foi criada.

Quanto menos uma classe depende da outra, menor o acoplamento, e maior a especialização dela, isso significa alta coesão.

A close-up photograph of a person's fingers pressing a key on a computer keyboard. The background is a plain, light color. A solid red rectangular overlay covers the middle portion of the image, containing the word "DEMO" in large, white, sans-serif capital letters.

**DEMO**