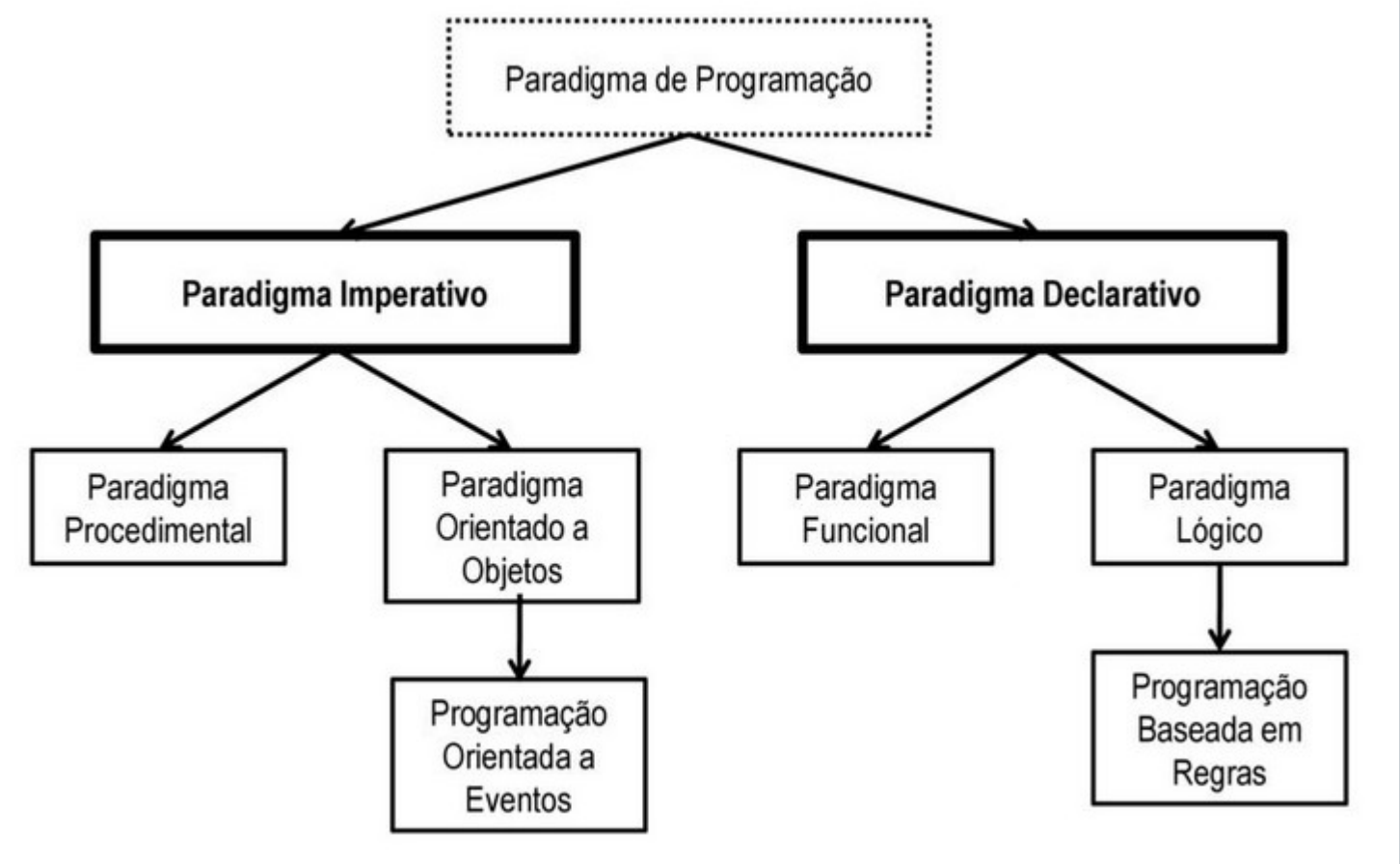
Linguagem de Programação II

Prof.Antonio Carlos Sobieranski

DEC7532 | ENC | DEC | CTS

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

Orientação à Objetos



Diferentemente dos paradigmas orientados a procedimentos, onde o principal foco são os sub-programas e bibliotecas, funções e suas passagens de parâmetros, o principal enfoque da OO são os dados sob a ótica de um objeto.

Orientação à Objetos

1. Contextualização

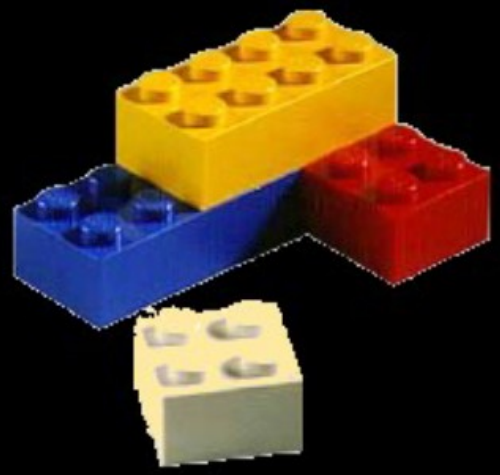
- Histórico
  - Fim da década de 1960 – A O.O. surgiu no Simula
  - 1967 – Simula introduziu os conceitos de classe e herança
  - 1983 – O termo “Programação Orientada a Objeto” (POO) é introduzido com a linguagem Smalltalk
  - Final dos anos 1980 - “Paradigma de Orientação a Objetos”: abordagem poderosa e prática para o desenvolvimento de software
  - 1986 – As Linguagens Híbridas: C++, Object-Pascal
  - Java, de fato, popularizou a OO

Orientação à Objetos

1. Contextualização

Definições da OO

- Paradigma Orientação a Objetos
  - Organização do mundo real em objetos
  - Esses objetos incorporam estados e comportamentos pertinentes as suas ações



Orientação à Objetos

1. Contextualização

Idéia básica expressa por:

- Percepção do mundo como uma coleção de objetos interagindo entre si;
- Filosoficamente
  "uma nova maneira de ver as coisas e pensar o mundo"

Mapear objetos do mundo no computador:

- Objetos comunicam-se através de mensagens (retornos)

Paradigma de OO

- Objetos + Classificação + Herança + Comunicação

Orientação à Objetos

2. Conceitos básicos

- Objeto e Classe
- Atributo e Método
- Instância
- Herança
- Polimorfismo
- Relacionamento

Orientação à Objetos

2. Conceitos básicos

2.1. Objeto e Classe

- Objeto (unicidade)
  - Coisas tangíveis (estruturas concretas)
  - Incidentes (eventos, ocorrências)
  - Interações (transações, contratos)
- Objeto ➔ composto por ESTADO, COMPORTAMENTO, IDENTIDADE

Orientação à Objetos

2. Conceitos básicos

2.1. Objeto e Classe

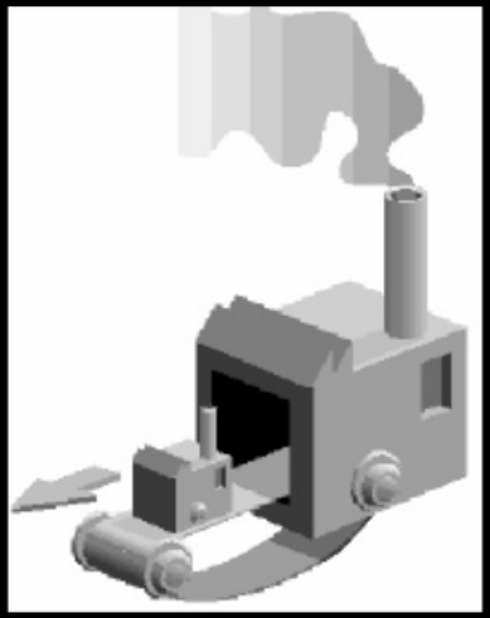
- Classes
  - Descreve uma coleção de objetos agrupados por alguma característica essencial em comum:
  - → propriedades semelhantes
  - → comportamentos semelhantes
  - → relacionamentos comuns a outros objetos

Orientação à Objetos

2. Conceitos básicos

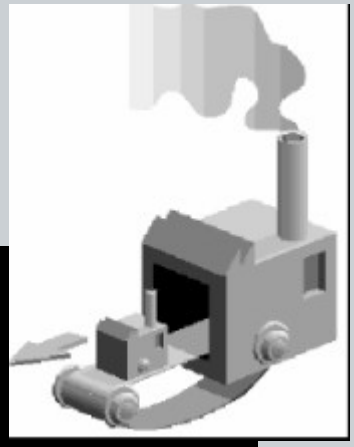
2.1. Objeto e Classe

- Classes
  - “Fábricas de instâncias”



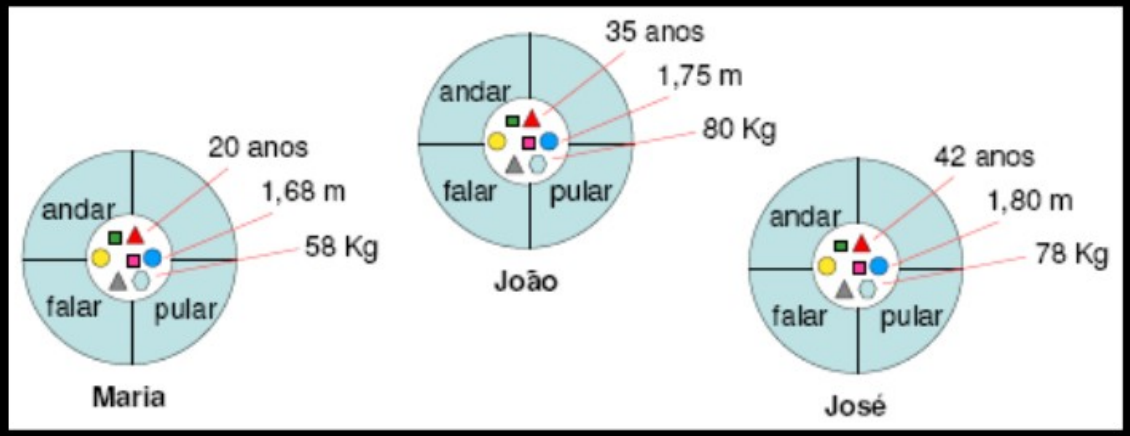
Orientação à Objetos

2. Conceitos básicos



2.1. Objeto e Classe

- Classes
  - "Fábricas de instâncias"



Orientação à Objetos

2. Conceitos básicos

2.2. Atributos e Métodos

Na linguagem C++:

private:

public:

protected:

Classe Funcionário
Nome Endereço Telefone CPF Salário_base
Funcionário () Insere_Dados () Altera_Dados () Retorna_Dados ()

Orientação à Objetos

2. Conceitos básicos

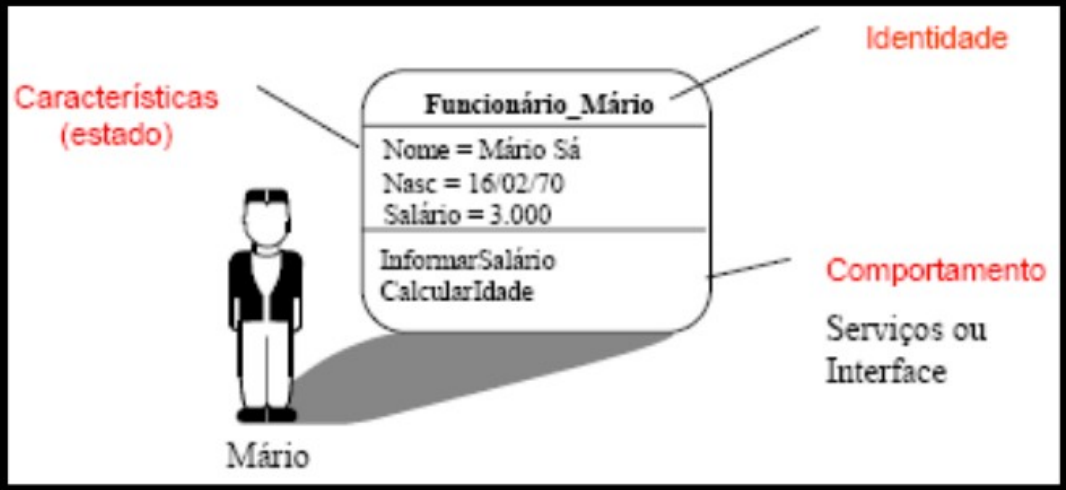
ENCAPSULAMENTO

- Conceito
  - Esconde a maneira de como as OPERACOES são implementadas
  - O acesso aos métodos e atributos ocorre através de uma espécie de INTERFACE (struct)
  - Restringir o escopo de informações
  - Se os dados estiverem encapsulados, só poderão ser acessados através de métodos. (da própria classe)

Orientação à Objetos

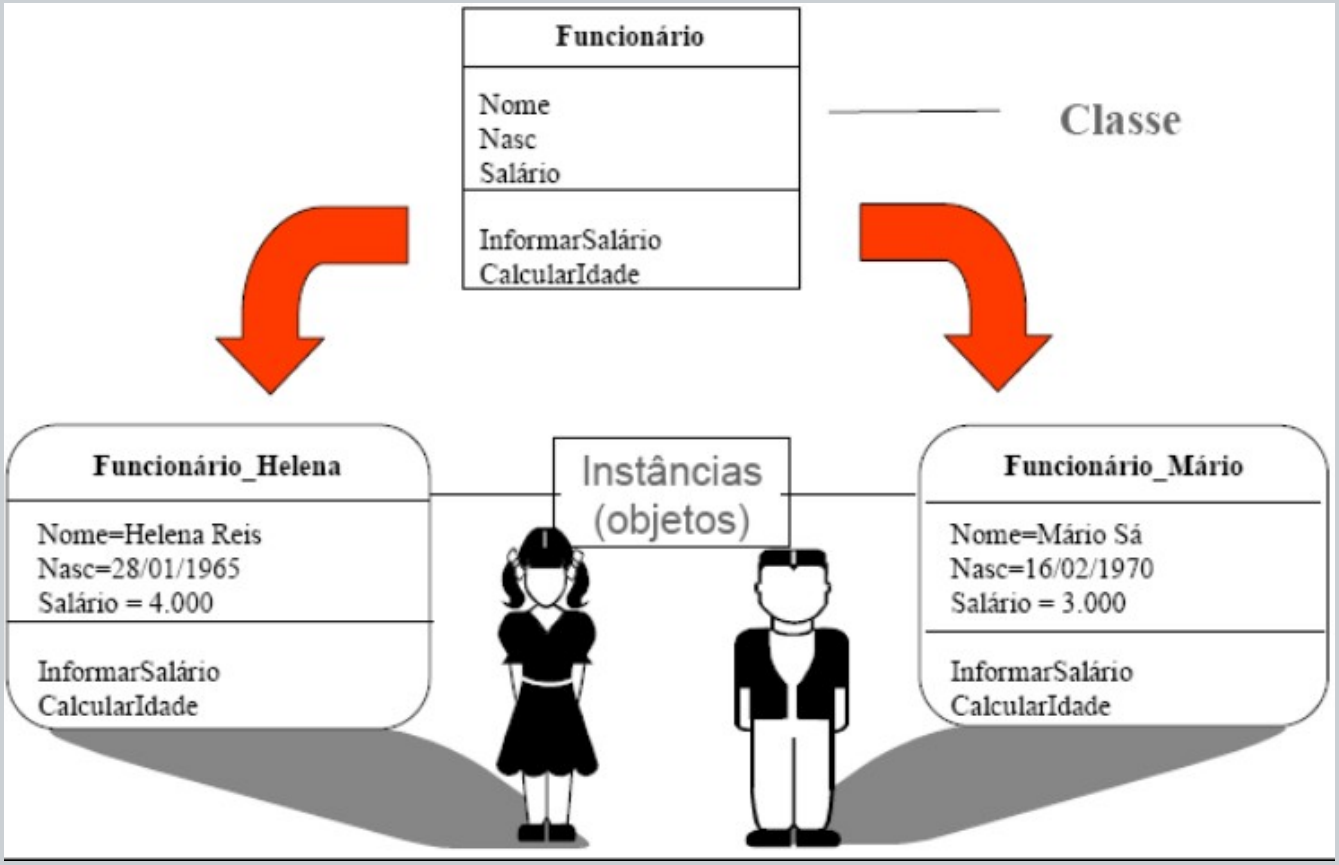
2. Conceitos básicos

2.2. Atributos e Métodos



Orientação à Objetos

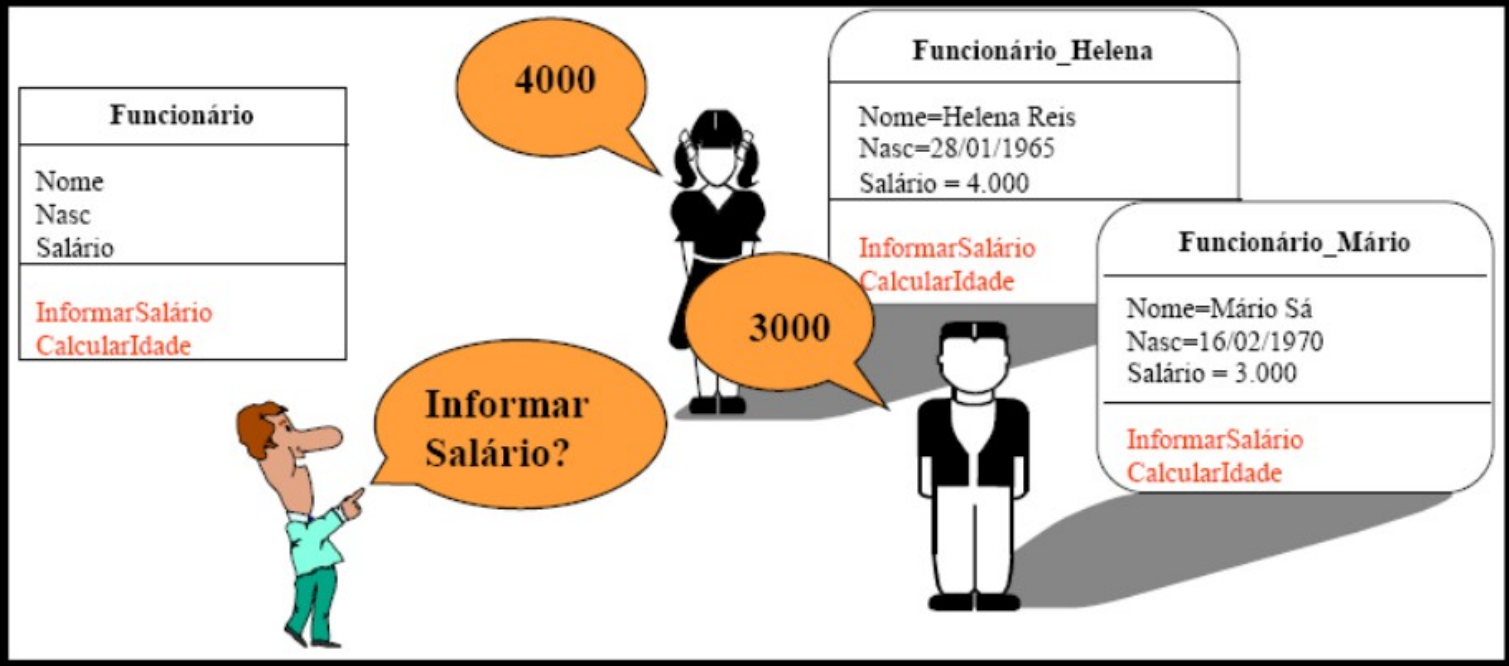
2. Conceitos básicos



Orientação à Objetos

2. Conceitos básicos

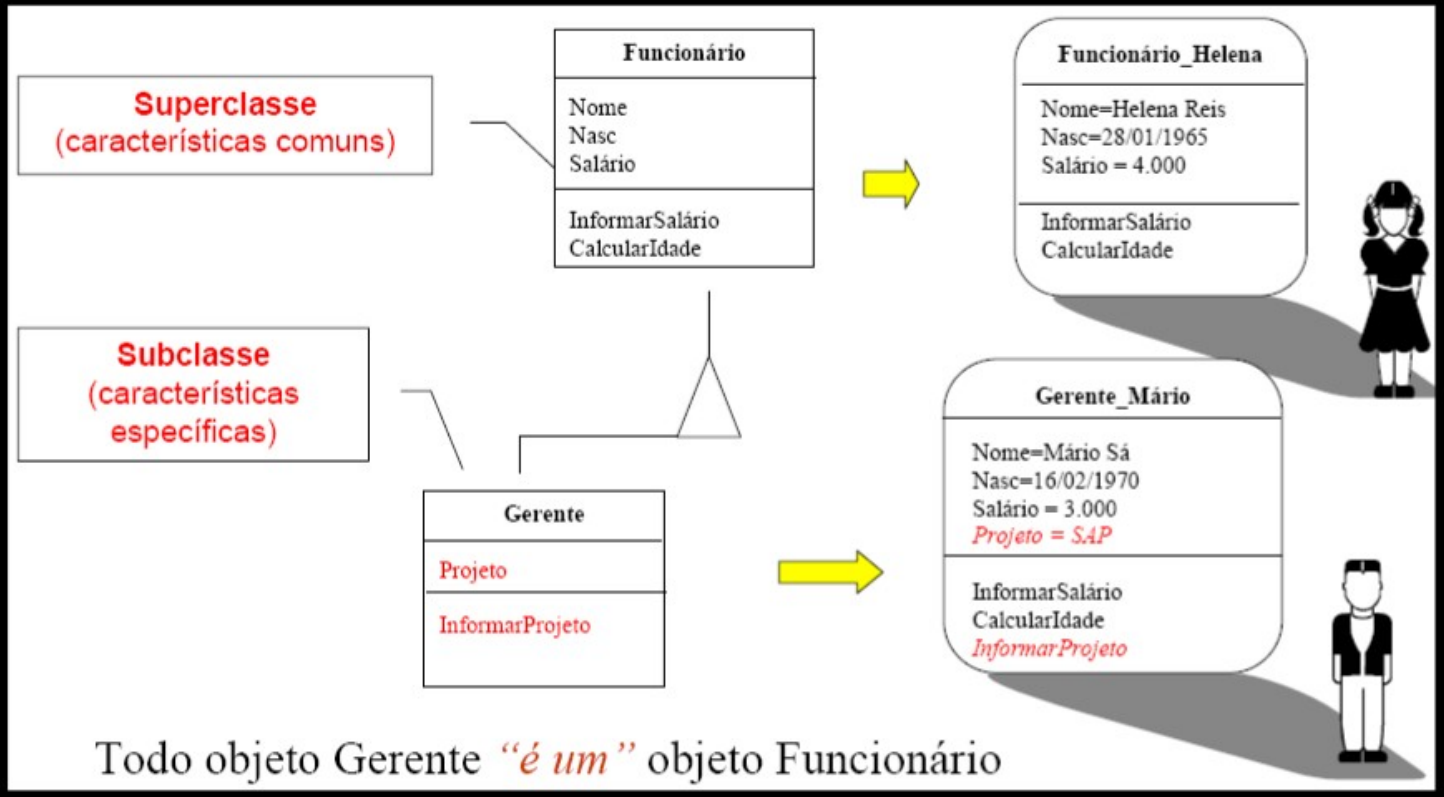
2.3. Instância (chamando métodos)



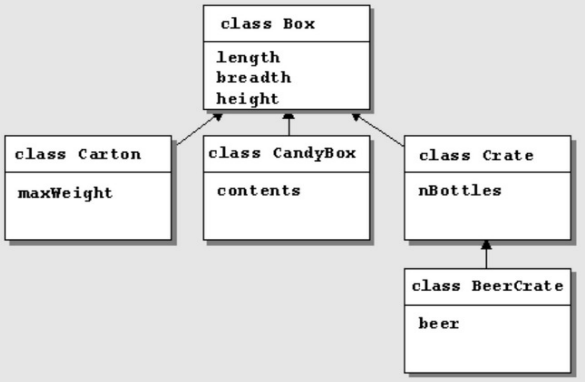
Orientação à Objetos

2. Conceitos básicos

2.4. Herança

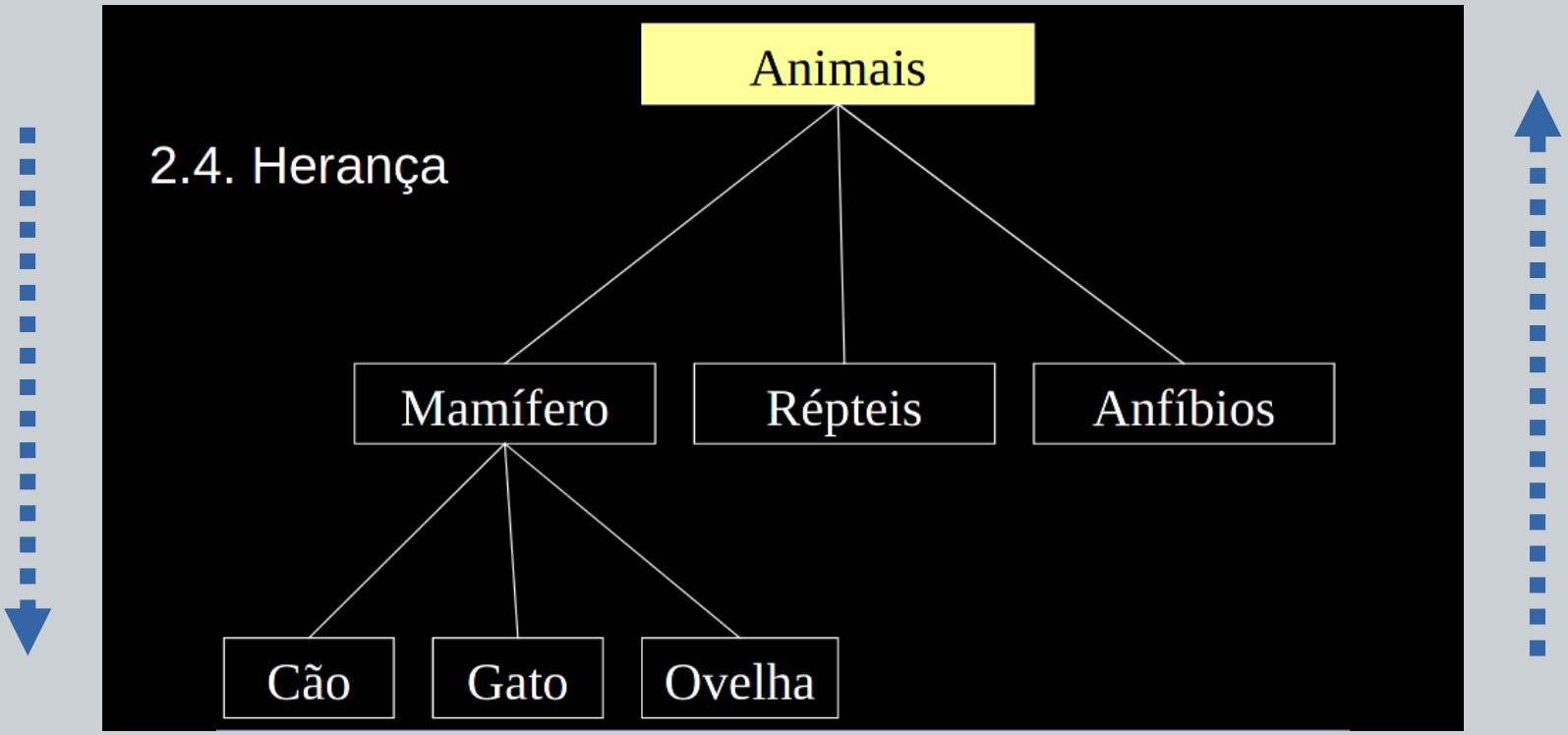


Orientação à Objetos



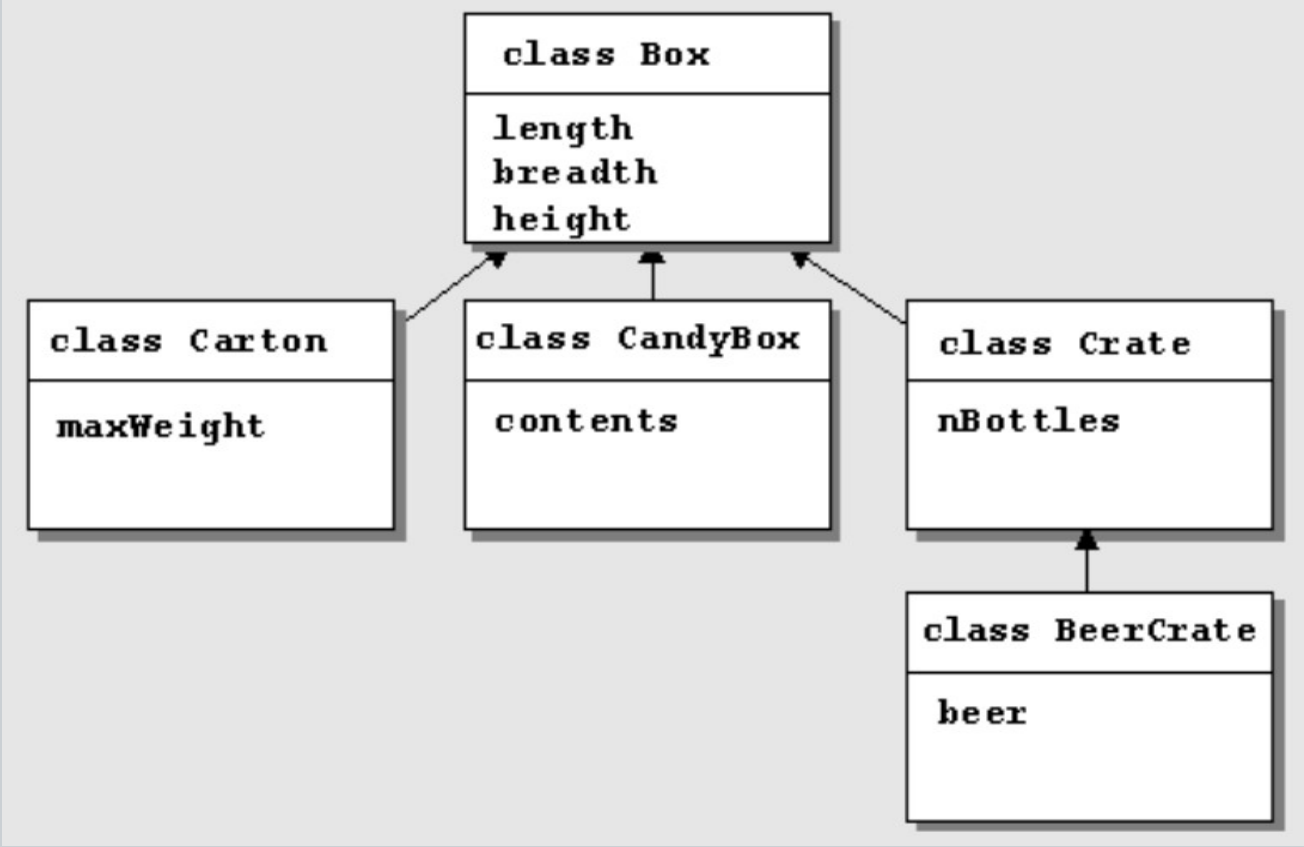
2. Conceitos básicos

Especialização X Generalização



Orientação à Objetos

2. Conceitos básicos

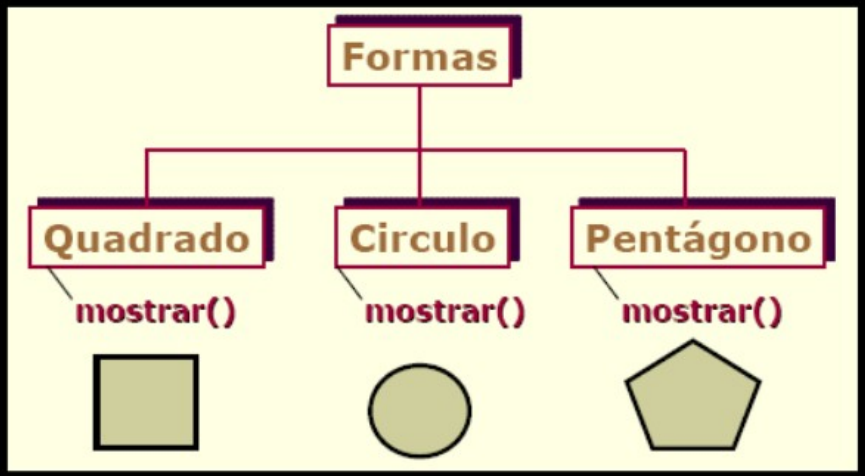


Orientação à Objetos

2. Conceitos básicos

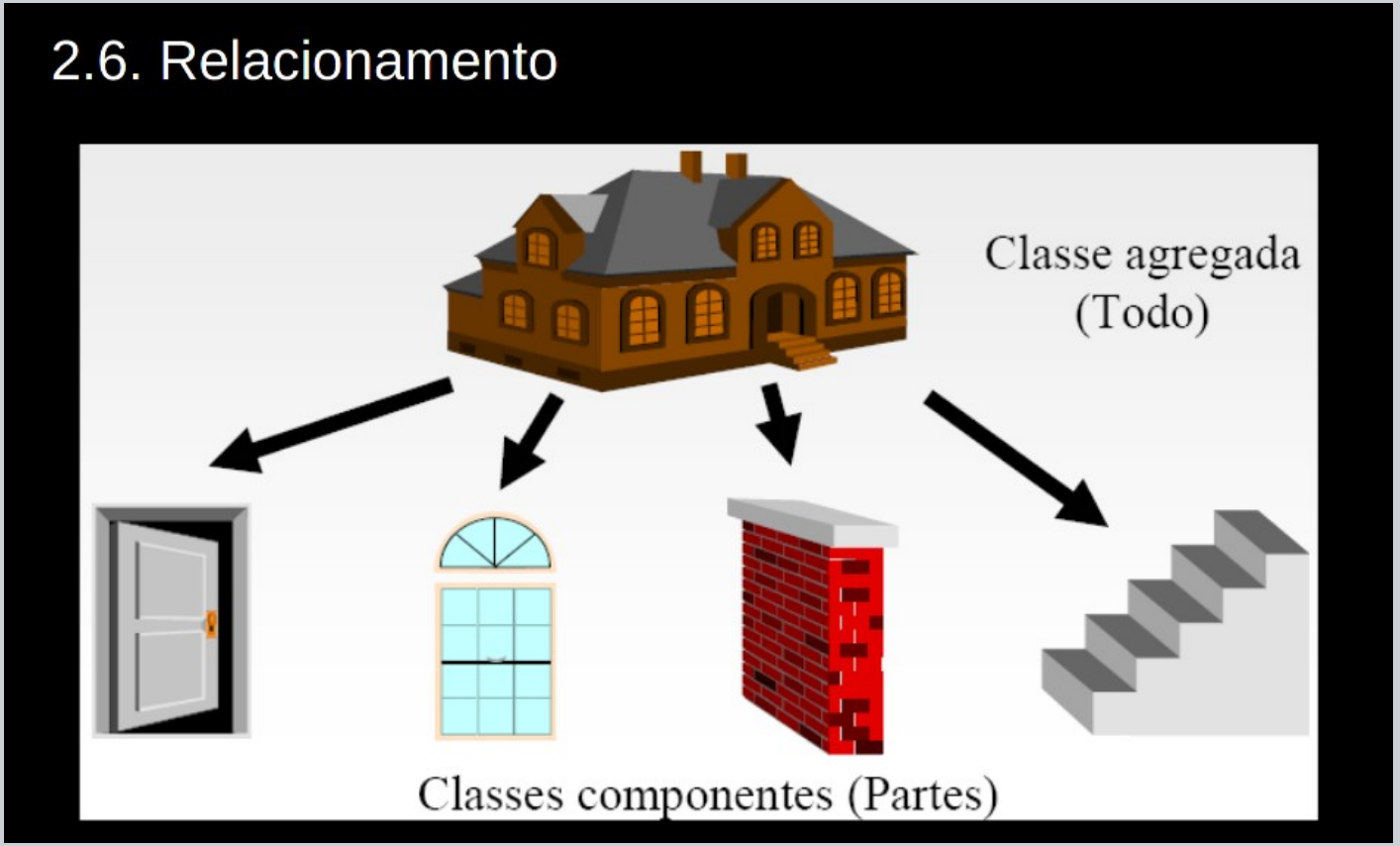
2.5. Polimorfismo

Poli → *"varias"* + morpho → *"forma"*



Orientação à Objetos

2. Conceitos básicos



Orientação à Objetos

Conclusões

- Utilização de uma mesma notação durante o desenvolvimento
- Redução no tempo de manutenção
- Maior reutilização de código

Orientação à Objetos

Conclusões

- Redução da complexidade de desenvolvimento (refinamento em níveis de abstração)
- Modelo semântico mais intuitivo com a realidade
- Encapsulamento de atributos e métodos
- Aumento de qualidade e produtividade

Orientação à Objetos

Recomendação Leitura

23. CLASSES AND OBJECTS

The main purpose of C++ programming is to add object orientation to the C programming language and classes are the central feature of C++ that supports object-oriented programming and are often called user-defined types.

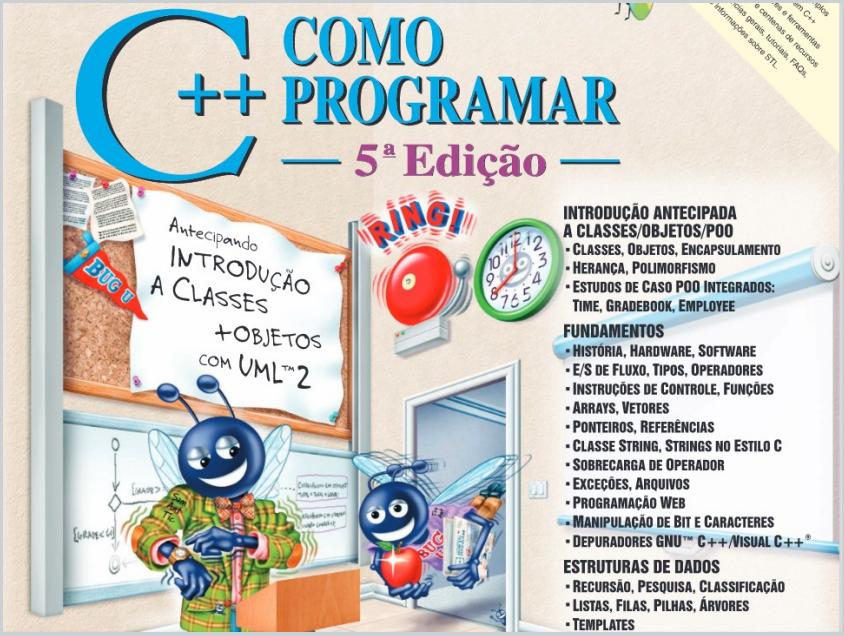
A class is used to specify the form of an object and it combines data representation and methods for manipulating that data into one neat package. The data and functions within a class are called members of the class.

C++ Class Definitions

When you define a class, you define a blueprint for a data type. This doesn't actually define any data, but it does define what the class name means, that is, what an object of the class will consist of and what operations can be performed on such an object.

A class definition starts with the keyword **class** followed by the class name; and the class body, enclosed by a pair of curly braces. A class definition must be followed either by a semicolon or a list of declarations. For example, we define the Box data type using the keyword **class** as follows:

```
class Box
{
   public:
      double length;   // Length of a box
      double breadth;  // Breadth of a box
      double height;   // Height of a box
```



Contato

Prof.Antonio Carlos Sobieranski – DEC | A316

E-mail: a.sobieranski@ufsc.br

Inst: @antonio.sobieranski

UNIVERSIDADE FEDERAL
DE SANTA CATARINA