TÉCNICAS DE PROGRAMAÇÃO I

JEANE A. MENEGUELI

Objetivos de Aprendizagem:

- Utilizar linguagem de programação, difundida no mercado, para codificação aplicando os conceitos de orientação a objetos.
- Abstração, encapsulamento, herança, polimorfismo. Relacionamento entre classes.
- Compreender e programar Tratamento de exceções.
- Criar Interfaces gráficas com usuário.
- Aplicar conceitos da Arquitetura Model-View-Controller.
- Conhecer frameworks de desenvolvimento front-end e back-end.
- Aplicar versionamento e documentação da aplicação

Conteúdo

- Conceitos de orientação a objetos: Classes, Objeto, Encapsulamento, Herança, Polimorfismo.
- Princípios de padrões de projeto.
- Declaração de Classes e Objetos.
- Classe Abstrata.
- Métodos.
- Sobrecarga de Métodos.
- Conceitos de Herança múltipla.
- Modificadores de acesso.
- Construtores.
- Manipulação de Exceções.
- Conceitos e aplicações de arquitetura em Camadas.
- Uso de Interface Gráfica.
- Teste de Software.

Conteúdo

- Conceitos de orientação a objetos: Classes, Objeto, Encapsulamento, Herança, Polimorfismo.
- Declaração de Classes e Objetos.
- Classe Abstrata.
- Princípios de padrões de projeto.
- Métodos.
- Sobrecarga de Métodos.
- Conceitos de Herança múltipla.
- Modificadores de acesso.
- Construtores.
- Manipulação de Exceções.
- Conceitos e aplicações de arquitetura em Camadas.
- Uso de Interface Gráfica.
- Teste de Software.

Instrumentos de avaliação

- Avaliação Formativa:
  - Exercícios para prática
  - Análise e Resolução de Problemas acompanhado de rubrica de avaliação
- Avaliação Somativa:
  - Provas
  - Projetos
  - Avaliação em pares
  - Desafios de Programação
  - Trabalhos Interdisciplinares.

Bibliografia Básica

- FURGERI, S. Programação orientada a objetos: Conceitos e técnicas. São Paulo: Erica. 2015.
- NASCIMENTO JR. O.S. Introdução à Orientação a Objetos com C++ e Python: Uma abordagem prática. São Paulo: Novatec, 2017
- SIERRA, K. BATES, B. Use a Cabeça! Java. 2 ed. São Paulo: O'Rilly, 2005.

Bibliografia Complementar

- BHARGAVA, A. Y. Entendendo Algoritmos: Um guia ilustrado para programadores e outros curiosos. São Paulo: Novatec,2019.
- KOPEC, D. Problemas Clássicos de Ciência da Computação com Python. São Paulo: Novatec,2019.
- MARTIN, Robert C. Código Limpo: Habilidades Práticas do Agile Software. Rio de Janeiro: Alta Books, 2012.
- RAMALHO, L. Python Fluente: Programação Clara, Concisa e Eficaz. São Paulo: Novatec,2015.
- SCHILDT, H. Java para Iniciantes: Crie, Compile e Execute Programas Java Rapidamente. 6 ed. Porto Alegre: Bookman: 2015.
- SILVERMAN, R. E. Git: guia prático. São Paulo: Novatec, 2019

CONCEITOS DE ORIENTAÇÃO A OBJETO

CLASSES, OBJETO, ENCAPSULAMENTO, HERANÇA, POLIMORFISMO

CONCEITOS DE ORIENTAÇÃO A OBJETO

A queda nos preços dos equipamentos de informática na década de 1970 motivou diversas empresas de pequeno e médio porte a informatizarem seus processos operacionais. Nessa época, os conhecimentos técnicos relacionados ao desenvolvimento de software não eram suficientes para resolver alguns problemas de desenvolvimento de sistemas.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Essa necessidade tornou-se evidente com a crescente demanda do público consumidor de software.
- Os novos usuários de sistemas não eram especialistas em computação, e sim profissionais de outras áreas.
- Foi esse cenário que motivou o surgimento da Orientação a Objetos (OO).

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Portanto, a Orientação a Objetos surgiu da necessidade de simular a realidade, criando abstrações na tentativa de representar as características relevantes dos objetos envolvidos no sistema que se deseja desenvolver.
- A compreensão do software tornou-se mais fácil, pois a representação em objetos é um processo natural.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Com o uso da Orientação a Objetos, a engenharia de software conseguiu avançar na habilidade de modelar e projetar softwares, que representam os problemas do mundo real no mundo computacional.
- O desenvolvedor aproveita os aspectos mais importantes do mundo real para realizar as representações no mundo computacional.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- A modelagem conceitual descreve as informações que o sistema irá gerenciar.

MACRO - ETAPAS DE DESENVOLVIMENTO DE SISTEMAS:

- MODELO CONCEITUAL
- MODELO LÓGICO
- MODELEO FÍSICO

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Dessa forma, constata-se que a tarefa mais importante de um processo de desenvolvimento de software é *realizar a análise do domínio da aplicação e a modelagem dos objetos*.
- A análise do domínio da aplicação corresponde às informações do ambiente em que a aplicação está inserida.
- Por exemplo, para projetar um sistema de biblioteca, é necessário que o desenvolvedor compreenda todas as **regras de negócio** relacionadas ao funcionamento da biblioteca (controle de livros, empréstimo, devolução etc.).

CONCEITOS DE ORIENTAÇÃO A OBJETO

- A compreensão do domínio da aplicação é pré-requisito para um bom projeto de software.
- Já a modelagem são fenômenos que ocorrem sobre estes, independentemente da forma como serão implementados posteriormente.
- Nesse sentido, o processo de modelagem dos objetos envolve dois mecanismos:
    - Abstração.
    - Representação.

CONCEITOS DE ORIENTAÇÃO A OBJETO

ABSTRAÇÃO

- Mecanismo utilizado na análise de um domínio da aplicação, em que se observa a realidade e dela se abstraem entidades e ações que são consideradas essenciais para uma aplicação, excluindo todos os aspectos julgados irrelevantes.

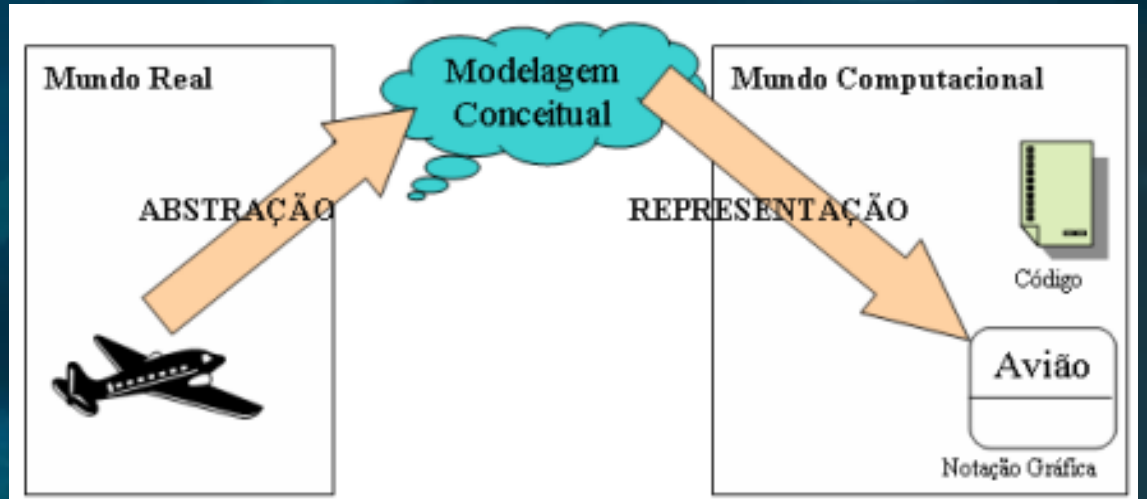


Figura 1 *Modelagem Conceitual.*

- A representação é o processo de traduzir essas informações essenciais no mundo computacional. Na modelagem orientada a objetos, destacamos as representações em formato gráfico (UML) e em formato de código (linguagem de programação Orientada a Objetos).

CONCEITOS DE ORIENTAÇÃO A OBJETO

ABSTRAÇÃO

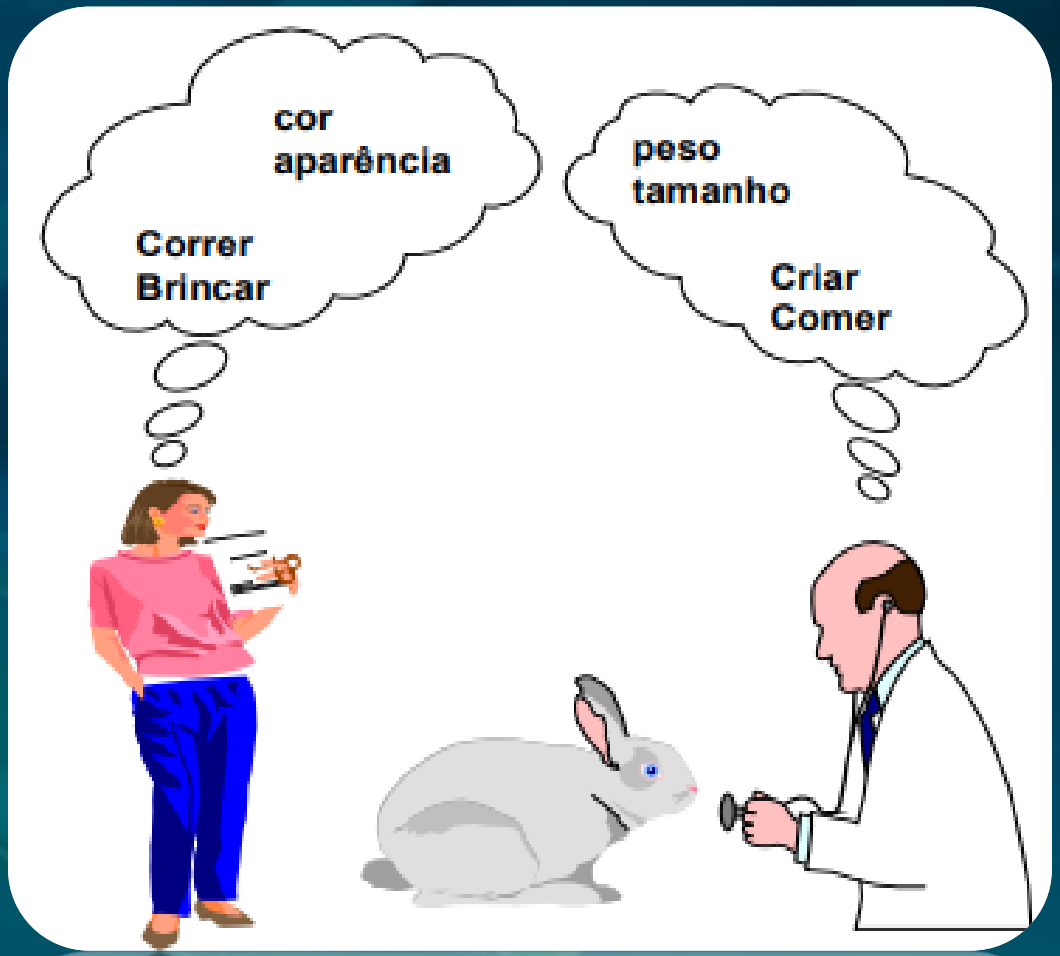
- A ideia básica da Orientação a Objetos é perceber o mundo como uma coleção de objetos que interagem entre si.
- Na modelagem de sistemas orientada a objetos, um objeto é uma entidade que possui:

a) características;

b) comportamento;

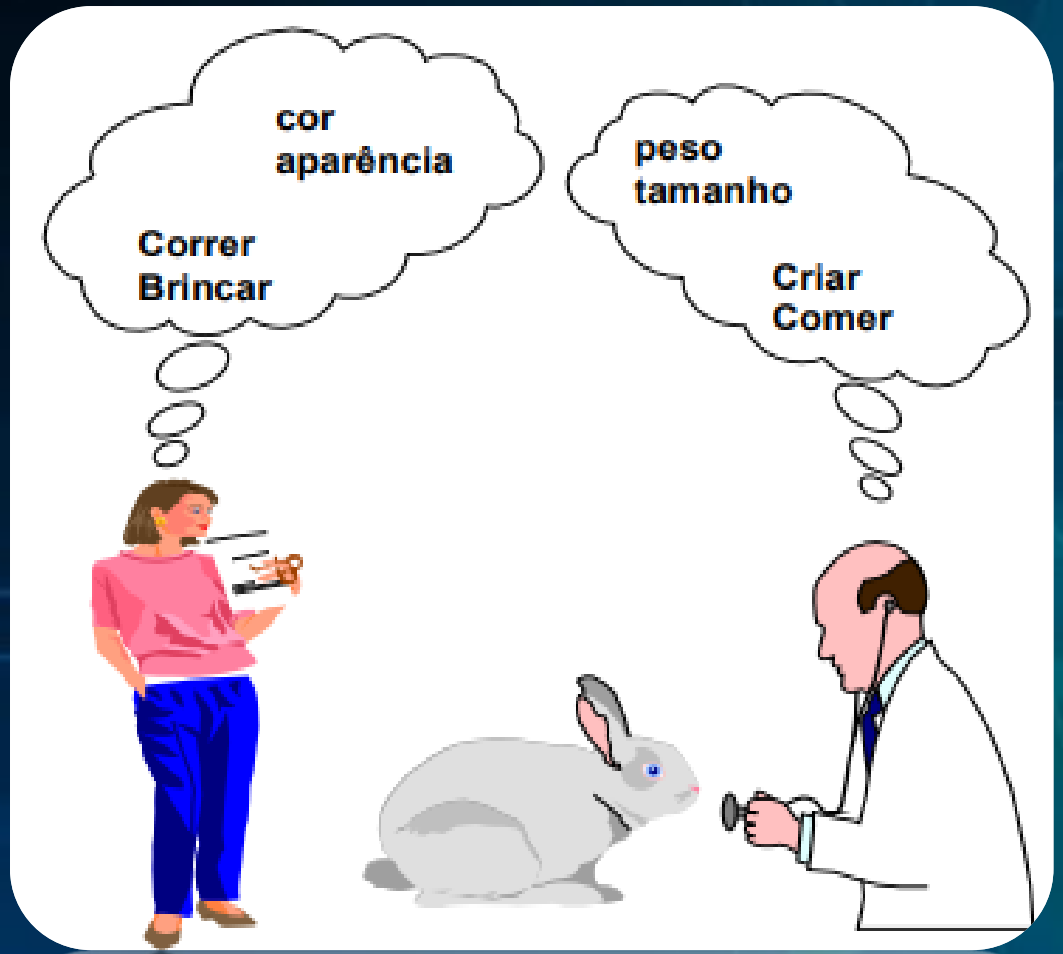
c) estado;

d) identidade única.



CONCEITOS DE ORIENTAÇÃO A OBJETO

ABSTRAÇÃO



- Observe que uma pessoa poderia olhar um coelho como um animal de estimação e descrevê-lo com as *características* "cor" e "*aparência*", e com os *comportamentos* "correr" e "brincar".
- Um médico veterinário, ao olhar o mesmo coelho, iria se preocupar com as *características* "peso" e "tamanho", bem como com os *comportamentos* "criar" e "comer".
- Note que ***um mesmo objeto pode possuir diferentes características, pois depende da abstração ou visão da pessoa que o analisa.***

CONCEITOS DE ORIENTAÇÃO A OBJETO

ABSTRAÇÃO

- Assim, na análise realizada pelo ***veterinário***, o animal coelho foi representado no objeto ***Coelho***, e tem como características "peso" e "tamanho", e como comportamentos "criar" e "comer"

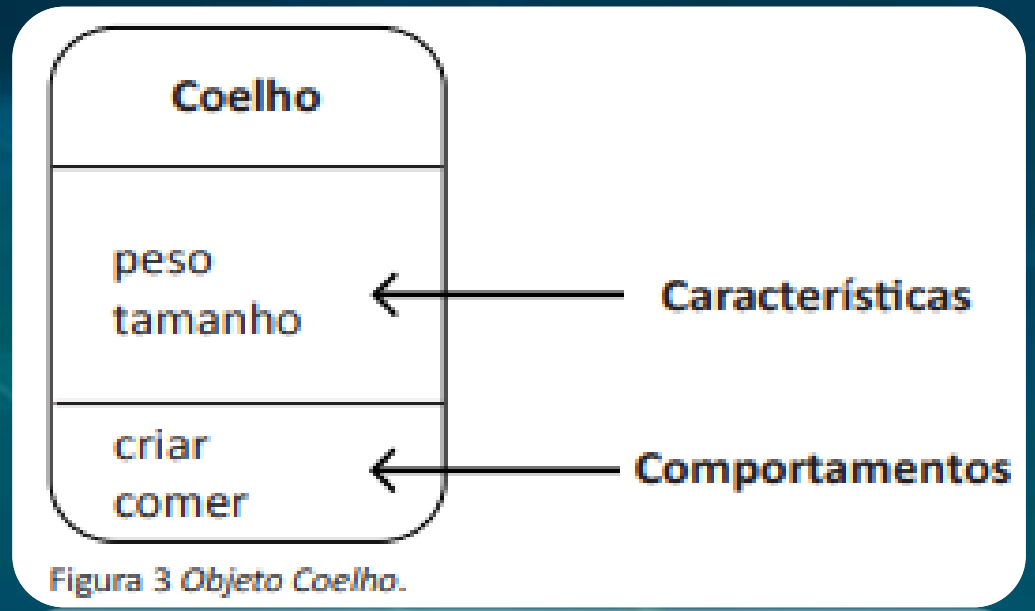


Figura 3 *Objeto Coelho.*

CONCEITOS DE ORIENTAÇÃO A OBJETO

ABSTRAÇÃO

- É importante considerar, ainda, o Estado de um objeto, que corresponde ao conjunto de valores associados às características do objeto. Dessa forma, considere o objeto Coelho e os dois estados:

Figura 4 *Estados.*

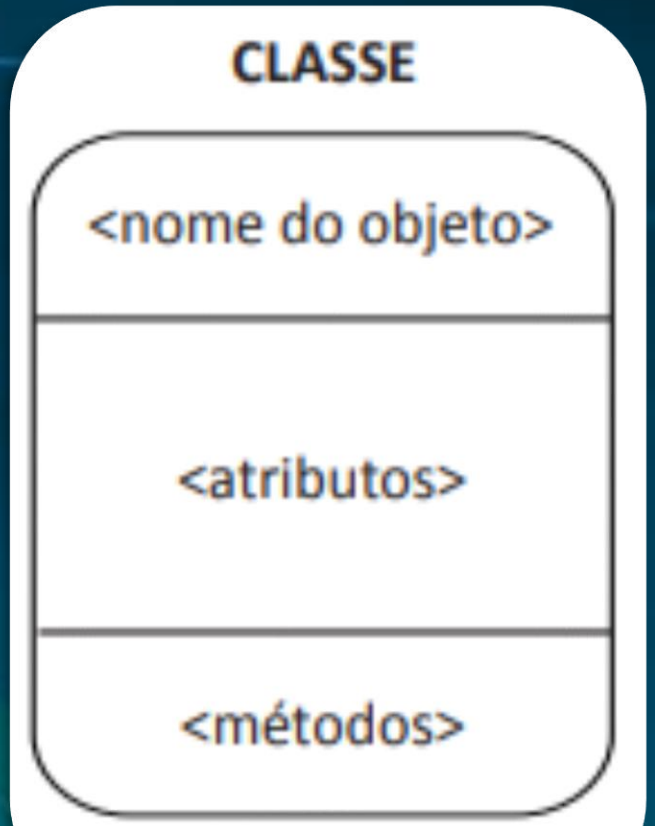
- O Estado do Objeto coelho no dia de "hoje" e em "3 meses antes".
- Perceba que o estado pode mudar com o passar do tempo.
- Assim, cada objeto tem uma identidade única, que o diferencia dos demais.
- Em um sistema orientado a objetos, por exemplo, cada objeto Coelho terá uma identificação única que o diferenciará de quaisquer outros coelhos que venham a existir no sistema.

CONCEITOS DE ORIENTAÇÃO A OBJETO

CLASSE E OBJETO

- Quando estamos modelando um sistema usando a Orientação a Objetos, notamos que existem vários objetos com características e comportamentos similares.

- A análise desses objetos semelhantes pode gerar, conforme a visão da abstração, uma representação única chamada **Classe**, a qual, na Orientação a Objetos, incorpora essa operação por meio da abstração dos atributos (características) e dos métodos (comportamentos) que caracterizam objetos semelhantes.



ura 5 Estrutura de Classe (UML)

CONCEITOS DE ORIENTAÇÃO A OBJETO

CLASSE E OBJETO

- Enquanto um Objeto é uma abstração de uma entidade do mundo real, por meio das características e comportamentos, a classe é a abstração de um conjunto de objetos similares do mundo real, que descreve a estrutura de dados e o comportamento de objetos similares.

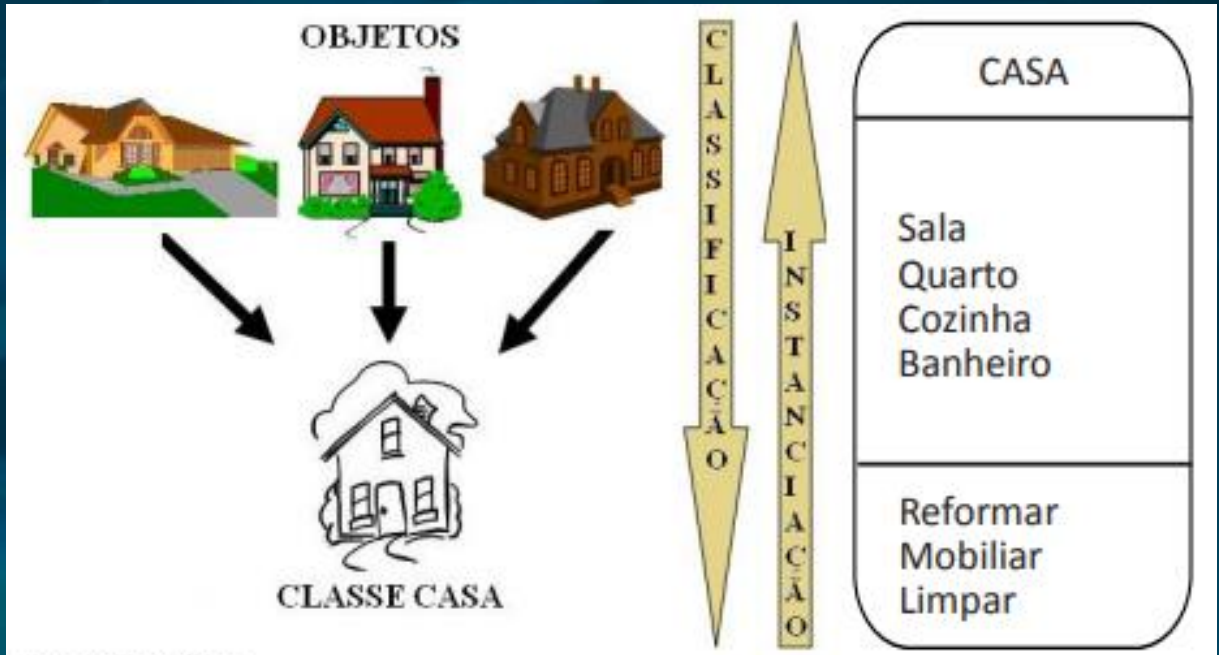


Figura 6 *Classe Casa*.

- Uma classe representa, portanto, um conjunto de objetos que possui características semelhantes (atributos), os mesmos comportamentos (métodos), os mesmos relacionamentos com outros objetos e a mesma semântica.

CONCEITOS DE ORIENTAÇÃO A OBJETO

CLASSE E OBJETO

- Na Figura 6, definimos a Classe Casa como a representação de vários objetos que possuem características (sala, quarto, cozinha e banheiro) e comportamentos (reformar, mobiliar e limpar) semelhantes.
- A ação de criar classes por meio da abstração de objetos similares é denominada Classificação.

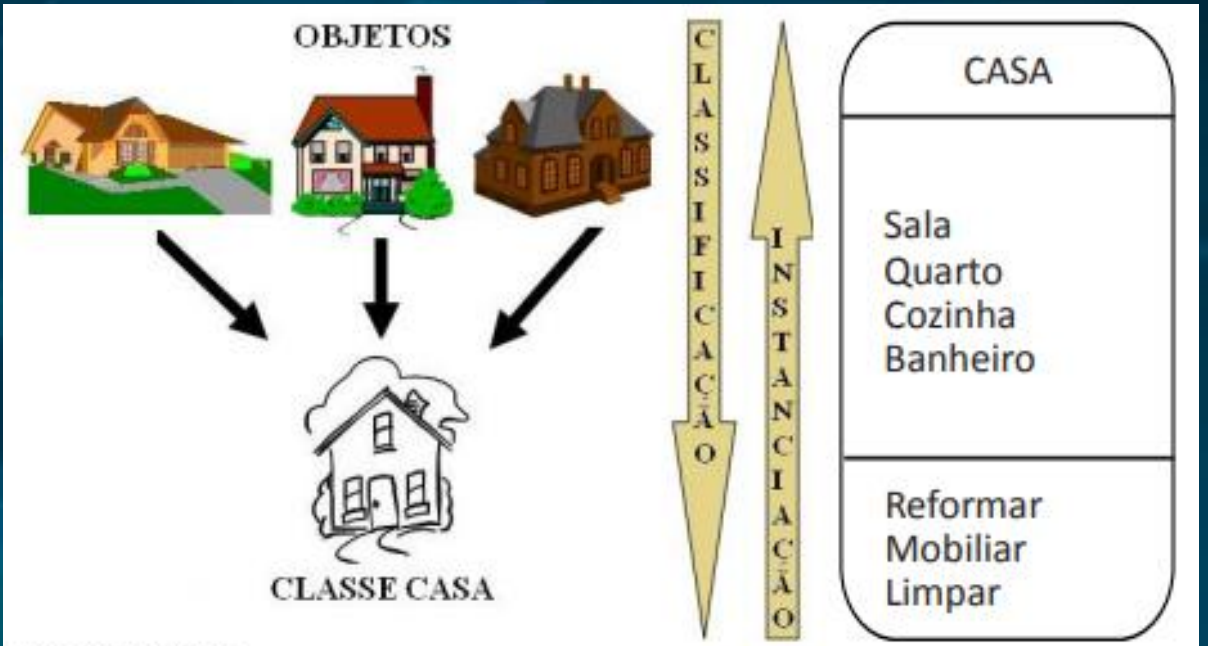


Figura 6 *Classe Casa.*

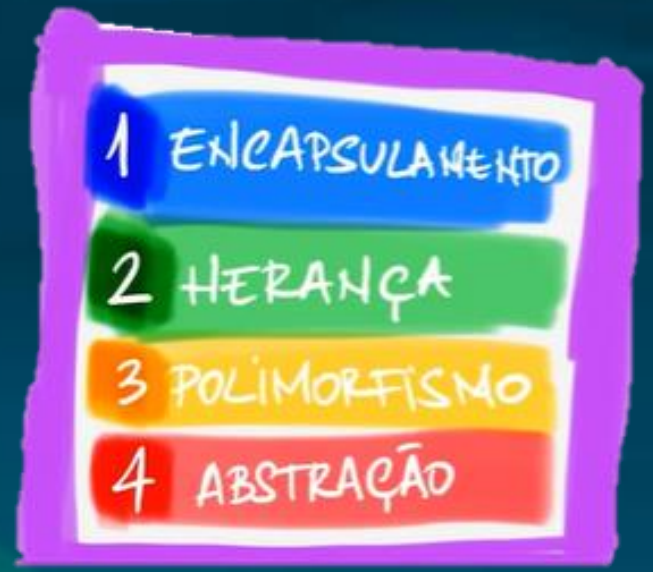
- A ação de criar objetos com base em uma classe é denominada Instanciação.
- É importante ressaltar, ainda, que todo objeto é uma instância de uma classe.
- Em um sistema Orientado a Objetos, não há limites relacionados à quantidade de objetos que podem ser instanciados ou criados a partir de uma classe.

Conceitos de orientação a objetos

ENCAPSULAMENTO, HERANÇA, POLIMORFISMO

CONCEITOS DE ORIENTAÇÃO A OBJETO

- O desenvolvimento de software é extremamente amplo.
- Nesse mercado, existem diversas linguagens de programação, que seguem diferentes paradigmas.
- Um desses paradigmas é a Orientação a Objetos, que atualmente é o mais difundido entre todos.
- Isso acontece porque se trata de um padrão que tem evoluído muito, principalmente em questões voltadas para segurança e reaproveitamento de código, o que é muito importante no desenvolvimento de qualquer aplicação moderna.



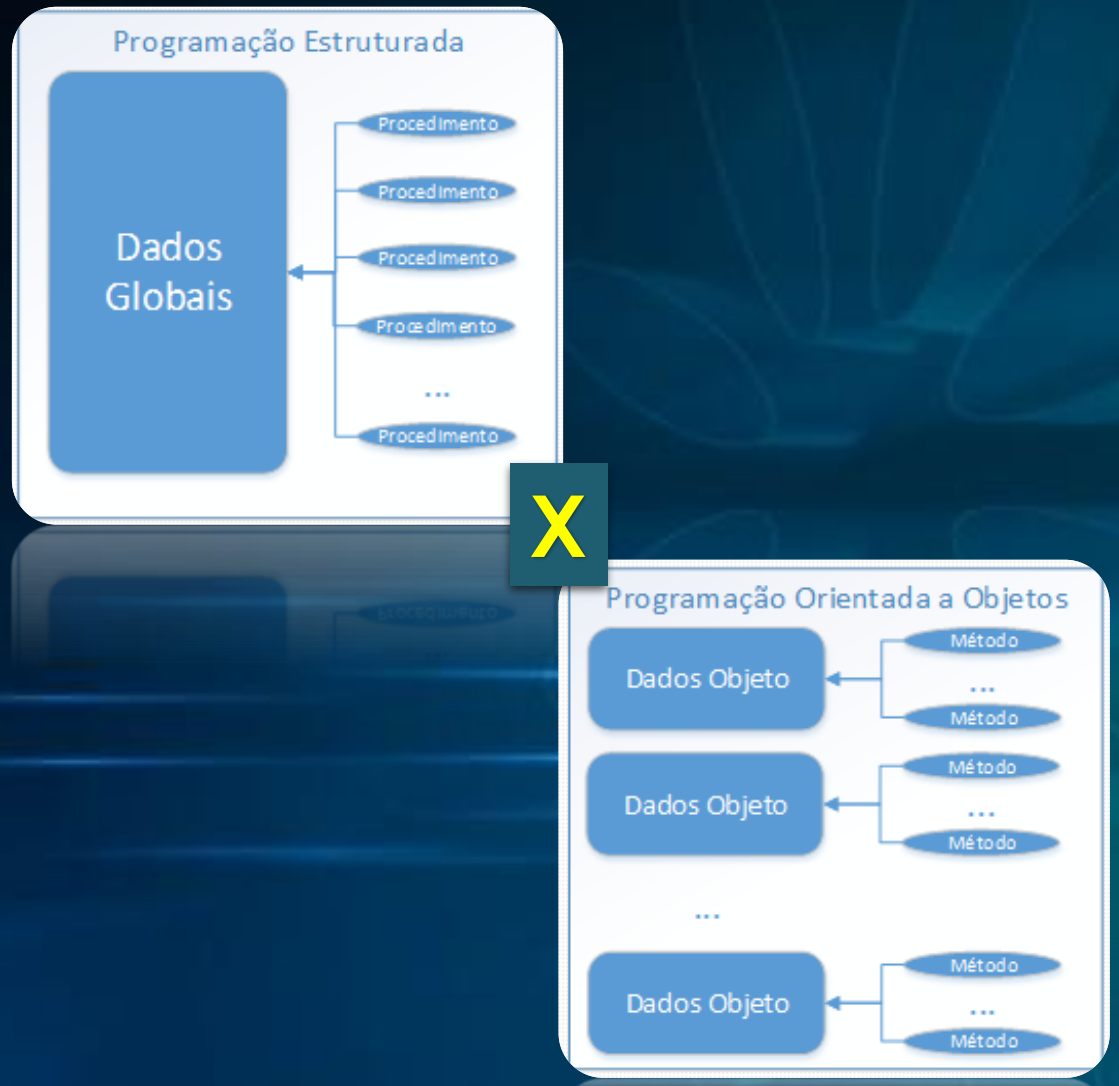
CONCEITOS DE ORIENTAÇÃO A OBJETO

- A Programação Orientada a Objetos (POO) diz respeito a um **padrão de desenvolvimento** que é seguido por muitas linguagens, como C# e Java.
- Vejamos as *diferenças entre a POO e a Programação Estruturada*, que era muito utilizada há alguns anos, principalmente com a linguagem C.

Programação Estruturada X Programação Orientada a Objetos

- Nas figuras vemos uma comparação muito clara entre a programação estruturada e a programação orientada a objetos no que diz respeito aos dados: no paradigma estruturado, temos procedimentos (ou funções) que são aplicados globalmente em nossa aplicação. No caso da orientação a objetos, temos métodos que são aplicados aos dados de cada objeto. Essencialmente, os procedimentos e métodos são iguais, sendo diferenciados apenas pelo seu escopo.

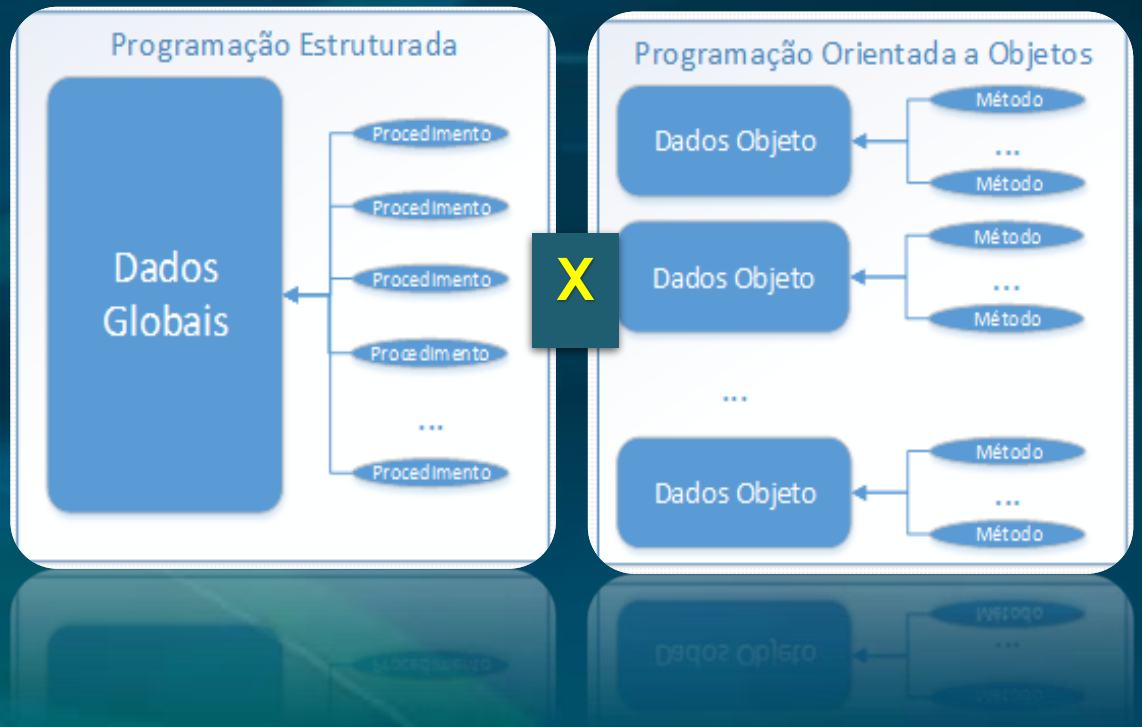
Programação Estruturada X Programação Orientada a Objetos



- A linguagem C é a principal representante da programação estruturada, trata-se de uma linguagem considerada de baixo nível
- A sua principal utilização, devido ao baixo nível, é em programação para ***sistemas embarcados*** ou outros em que o conhecimento do hardware se faz necessário para um bom programa.

Programação Estruturada X Programação Orientada a Objetos

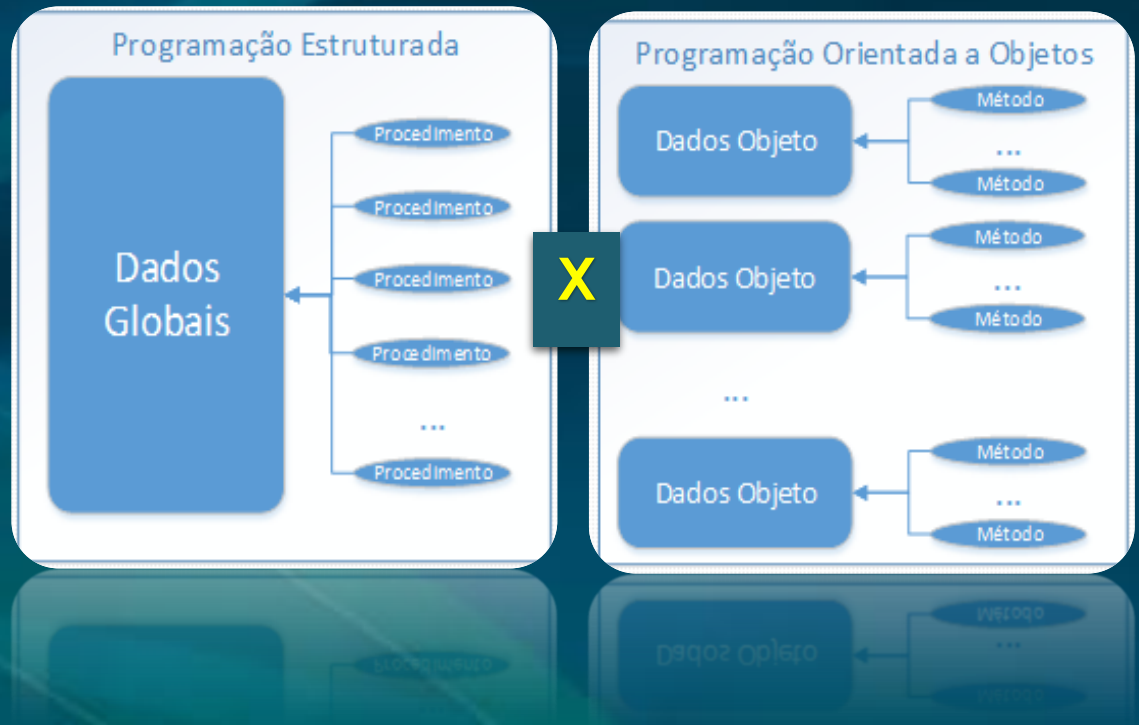
- a programação estruturada, quando bem feita, possui um desempenho superior ao que vemos na programação orientada a objetos.
- Isso ocorre pelo fato de ser um paradigma sequencial, em que cada linha de código é executada após a outra, sem muitos desvios, como vemos na POO.



- Além disso, o paradigma estruturado costuma permitir mais liberdades com o hardware, o que acaba auxiliando na questão desempenho.

Programação Estruturada X Programação Orientada a Objetos

- Porém, a programação orientada a objetos traz outros pontos que acabam sendo mais interessantes no contexto de aplicações modernas.
- Como o desempenho das aplicações não é uma das grandes preocupações na maioria das aplicações (devido ao poder de processamento dos computadores atuais), a programação orientada a objetos se tornou muito difundida.



- Essa difusão se dá muito pela questão da ***reutilização de código*** e pela **capacidade de representação** do sistema **muito mais perto do que veríamos no mundo real**.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- A Programação Orientada a Objetos (POO) é um padrão que se baseia em quatro pilares:
  - Encapsulamento
  - Herança
  - Polimorfismo
  - Abstração

CONCEITOS DE ORIENTAÇÃO A OBJETO

- A Programação Orientada a Objetos (POO) é um padrão que se baseia em quatro pilares:
  - Encapsulamento
  - Herança
  - Polimorfismo
  - Abstração

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Abstrair algo significa esconder os detalhes da implementação dentro de algo – às vezes um protótipo, às vezes em uma função.
- Portanto, quando você chama a função, não precisa entender exatamente o que ela está fazendo.

1. *Exemplo: funcionamento de um carro*
   - Quando acionamos ele para ligar, não precisamos saber quais passos ele faz para colocar o motor em funcionamento.
   - Quando acionamos o freio, não precisamos saber todos os mecanismos que são acionados para fazer o carro frear.
   - Apenas sabemos o que cada objeto ou função do carro produz como resultado.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- se você tivesse que entender cada função em uma base de código grande, você nunca codificaria nada, pois, levaria meses para terminar de ler e entender a lógica de tudo isso.
- Contudo, abstraindo certos detalhes, você é capaz de criar uma base de código reutilizável, simples de entender e facilmente alterável.

SEM ABSTRAÇÃO

- *Ter um botão escrito "Adicionar água fria à chaleira"*
- *Ter um botão escrito "Ferver a água"*
- *Ter um botão escrito "Adicionar uma cápsula de café"*
- *Ter um botão escrito "Passar a água pela cápsula de café"*
- *Além de vários outros botões para completar o processo*

X

COM ABSTRAÇÃO

- *Ter um botão escrito "Fazer café“*

CONCEITOS DE ORIENTAÇÃO A OBJETO

- A definição de encapsulamento é "a ação de colocar algo dentro ou como se estivesse em uma cápsula".
- *Remover o acesso a partes do seu código e tornar as coisas privadas* é exatamente o que o Encapsulamento faz (muitas vezes, as pessoas se referem a ele como "ocultação de dados").
- Encapsulamento significa que o *código de cada objeto deve controlar apenas seu próprio estado*.

CONCEITOS DE ORIENTAÇÃO A OBJETO

- *Se você não sabe **o que é o estado de um objeto**, vamos fazer a seguinte analogia:*
  - *Sabe aquele retrato de família, em que você era bebê ainda?*
  - *Ele é um registro do estado "instantâneo" em que você estava naquele exato momento.*
  - *De lá pra cá muita coisa mudou, e se hoje você tirar uma nova foto, seu estado já não é o mesmo que aquele.*
  - *Aquilo que você fez durante o tempo com sua vida, transformou você.*
  - *A mesma coisa ocorre com o objeto.*

CONCEITOS DE ORIENTAÇÃO A OBJETO

- ***O estado é o "instantâneo" atual do objeto***.
- Todas as chaves e métodos (funções) de um objeto são suas propriedades.
- Se você redefinir ou excluir uma chave, por exemplo, estará alterando o seu estado.
- Por isso, é importante limitar o acesso de quais partes do código podem ser acessadas.
- Caso não sejam necessárias, torne as coisas mais inacessíveis para não possibilitar efeitos colaterais no estado do objeto.

CONCEITOS DE ORIENTAÇÃO A OBJETO

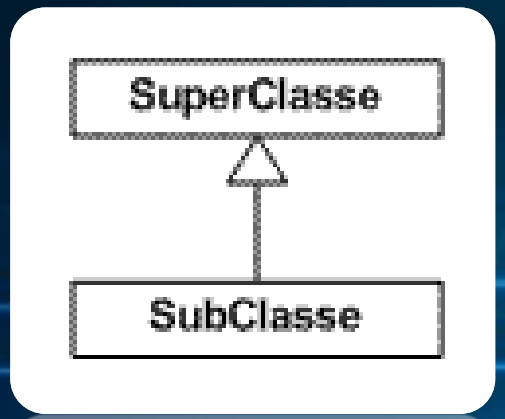
Por que devemos preferir a privacidade? Por que não ter tudo acessível globalmente?

- Muitos bits de código não relacionados se tornarão dependentes/acoplados uns dos outros por meio de variáveis globais.
- Você provavelmente substituirá as variáveis se o nome delas forem reutilizados, o que pode levar a erros ou comportamentos imprevisíveis.
- Você provavelmente terminará com um código espaguete – código que é difícil de raciocinar e entender o que está lendo e gravando suas variáveis, ou onde muda o estado de cada uma.

CONCEITOS DE ORIENTAÇÃO A OBJETO

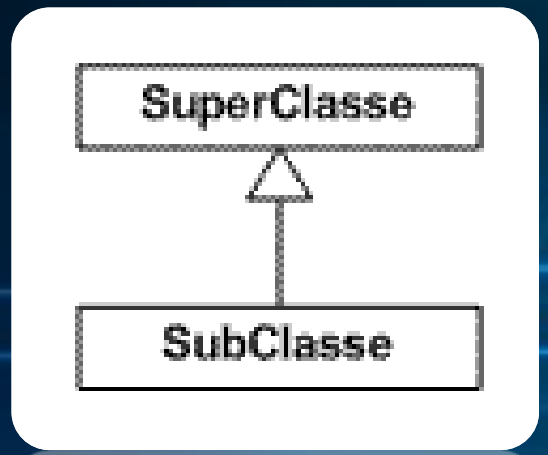
- O encapsulamento pode ser aplicado separando longas linhas de código em funções menores e separadas.
- O objetivo é sempre escondermos os dados em um lugar em que nada mais precise de acesso e expormos os dados de modo claro onde for necessário.
- O encapsulamento vincula seus dados a algo, seja uma classe, objeto, módulo ou função, e faz o possível para mantê-lo o mais privado possível.

CONCEITOS DE ORIENTAÇÃO A OBJETO



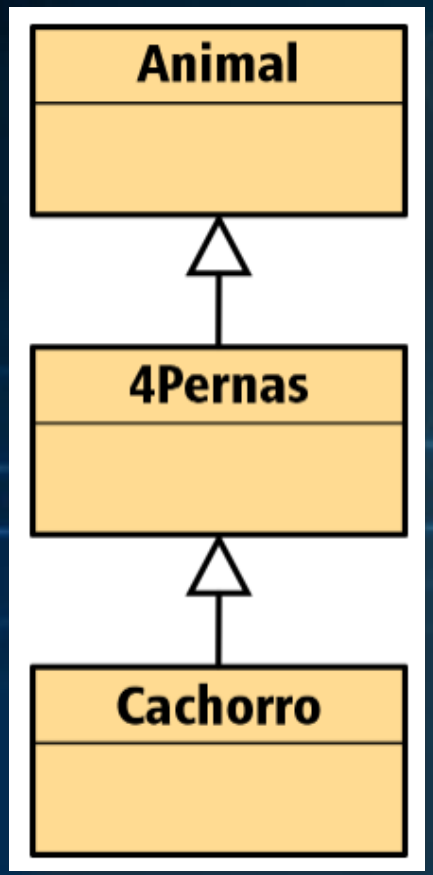
- A herança permite que um objeto adquira as propriedades e métodos de outro objeto.
- A reutilização é o principal benefício aqui.
- Sabemos que às vezes a mesma coisa precisa ser feita em vários lugares e sempre de forma igual, exceto em alguma pequena parte.
- Esse é um problema que a herança pode resolver.

CONCEITOS DE ORIENTAÇÃO A OBJETO



- Sempre que usamos herança, tentamos fazer com que o pai e o filho tenham alta coesão.
- Coesão é o quanto seu código está relacionado.
- Por isso, mantenha sua herança simples de entender e previsível. Não faça heranças completamente não relacionadas somente porque há um método ou uma propriedade de que você precisa. A herança não resolve bem esse problema específico.
- Ao usar herança, ela precisa ter a maior parte das funcionalidades.

CONCEITOS DE ORIENTAÇÃO A OBJETO



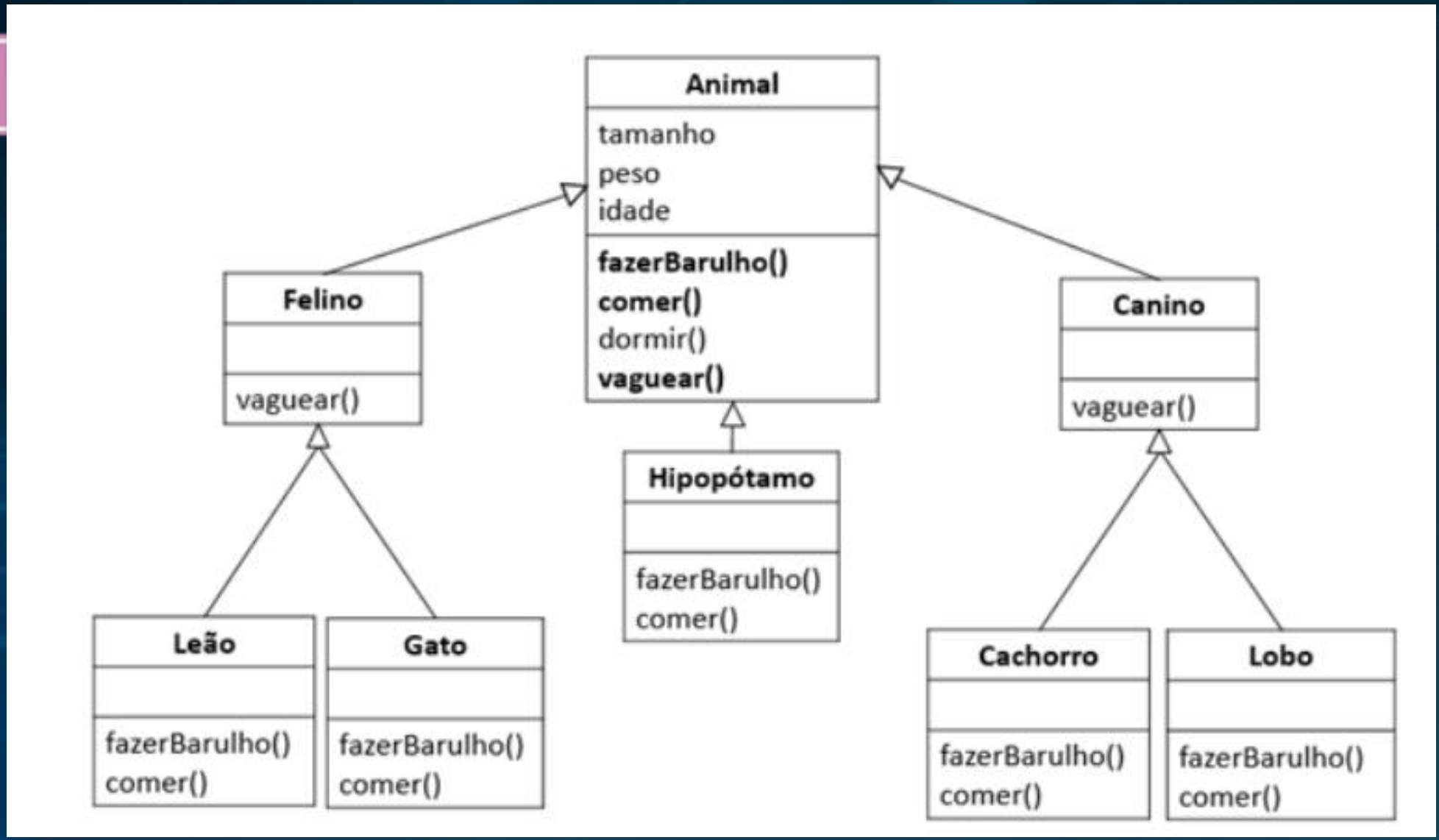
- A "cadeia de herança" é o termo usado para descrever esse fluxo de herança do protótipo do objeto base (aquele do qual todos os outros herdam) até o "final" da cadeia de herança (o último tipo que está herdando – Cachorro, no exemplo)

Princípio de substituição de Liskov:

- principal razão pela qual se falha.
- Se TipoFilho estiver removendo coisas do pai.
- Se TipoFilho remove métodos herdados do pai, isso gera diversos TypeError, onde haverá coisas que estarão indefinidas e que estamos esperando que não sejam.

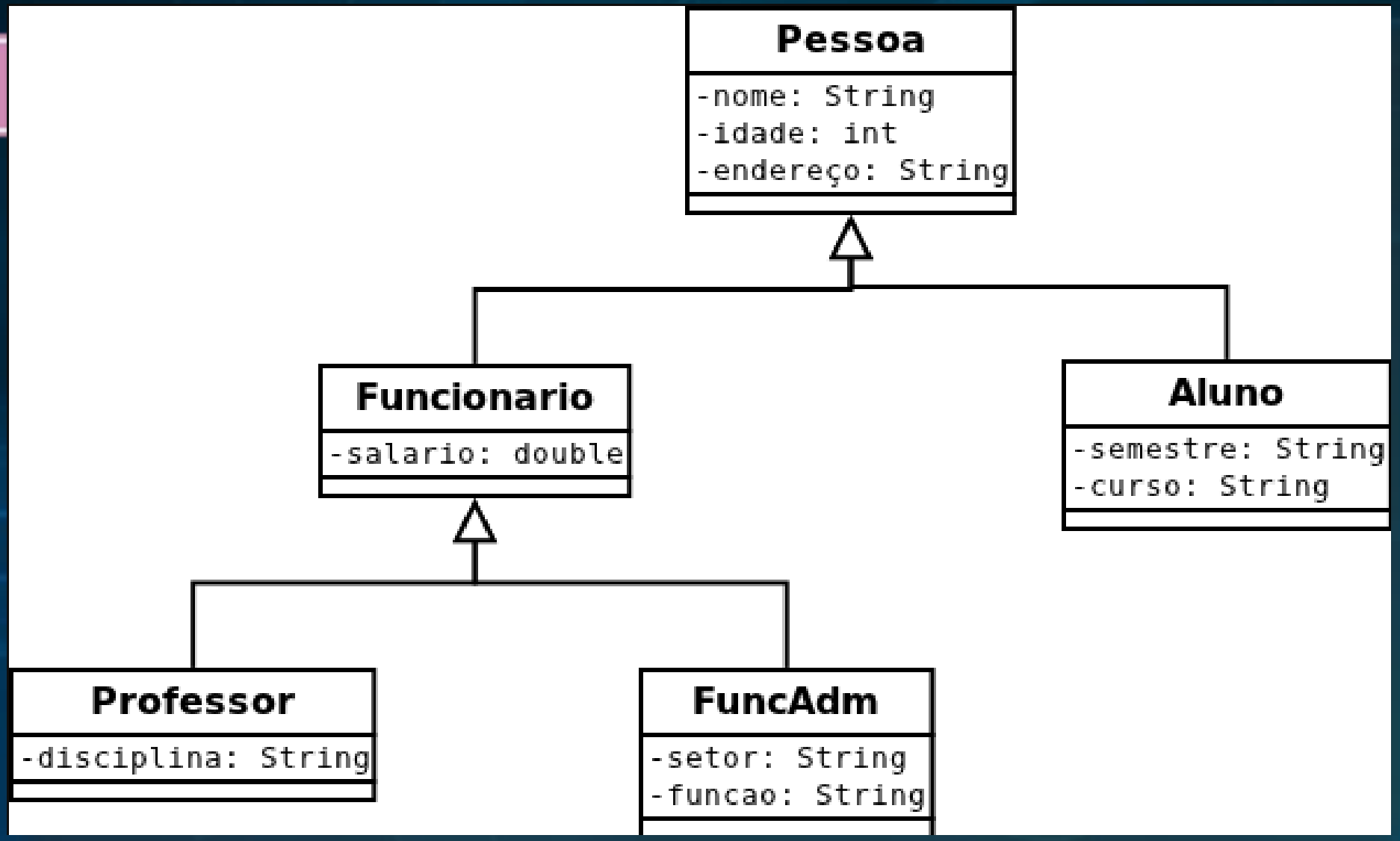
CONCEITOS DE ORIENTAÇÃO A OBJETO

3 Herança



CONCEITOS DE ORIENTAÇÃO A OBJETO

3 Herança



CONCEITOS DE ORIENTAÇÃO A OBJETO

- Polimorfismo significa "a condição de ocorrer de várias formas diferentes".
- Que tipos nas mesmas cadeias de herança sejam capazes de fazer coisas diferentes.
- Se você usou a herança corretamente, agora pode usar tanto os pais de maneira confiável como seus filhos.
- Quando dois tipos compartilham uma cadeia de herança, eles podem ser usados alternadamente sem erros ou declarações em seu código.
- Como se trata de um assunto que está intimamente conectado à herança, é importante entender os dois!

CONCEITOS DE ORIENTAÇÃO A OBJETO

- Na natureza, vemos animais que são capazes de alterar sua forma conforme a necessidade, e é dessa ideia que vem o polimorfismo na orientação a objetos.
- Como sabemos, os objetos filhos herdam as características e ações de seus "ancestrais".
- Entretanto, em alguns casos, é necessário que as ações para um mesmo método seja diferente.
- Em outras palavras, o polimorfismo consiste na ***alteração do funcionamento interno de um método herdado de um objeto pai***.

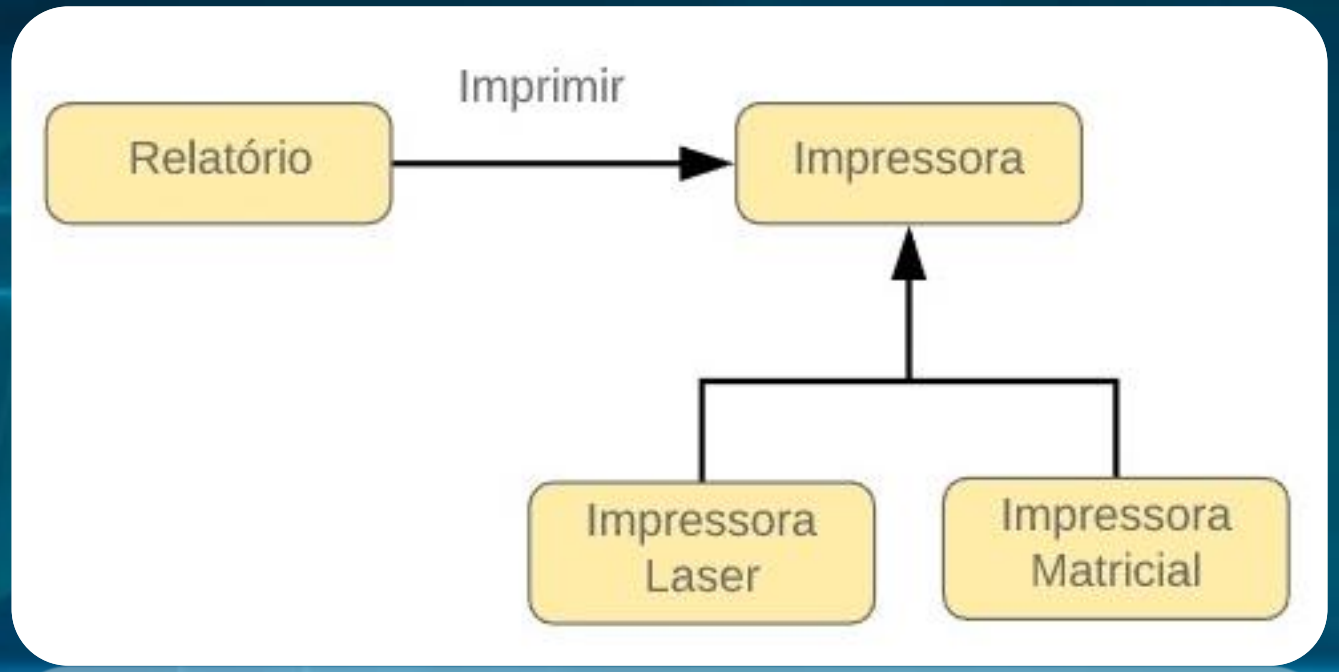
CONCEITOS DE ORIENTAÇÃO A OBJETO

- Exemplo
- Temos um objeto genérico "Eletrodoméstico".
- Esse objeto possui um método, ou ação, "Ligar()".
- Temos dois objetos:
  - "Televisão" e
  - "Geladeira",
- Que não irão ser ligados da mesma forma.
- Assim, precisamos, ***para cada uma das classes filhas, reescrever o método "Ligar()"***

CONCEITOS DE ORIENTAÇÃO A OBJETO

4 Polimorfismo

- *Duas subclasses de uma mesma classe podem ter implementações completamente diferentes de um mesmo método, o que leva os objetos a se comportarem de forma diferente, dependendo do seu tipo (classe).*



CONCEITOS DE ORIENTAÇÃO A OBJETO

Exercício de fixação:

1. Quais as vantagens da programação orientada a objetos?
2. Faça um quadro comparativo entre programação estruturada x orientada a objeto, com pontos positivos e negativos de cada paradigma.
3. O que são sistemas embarcados? Cite exemplos.