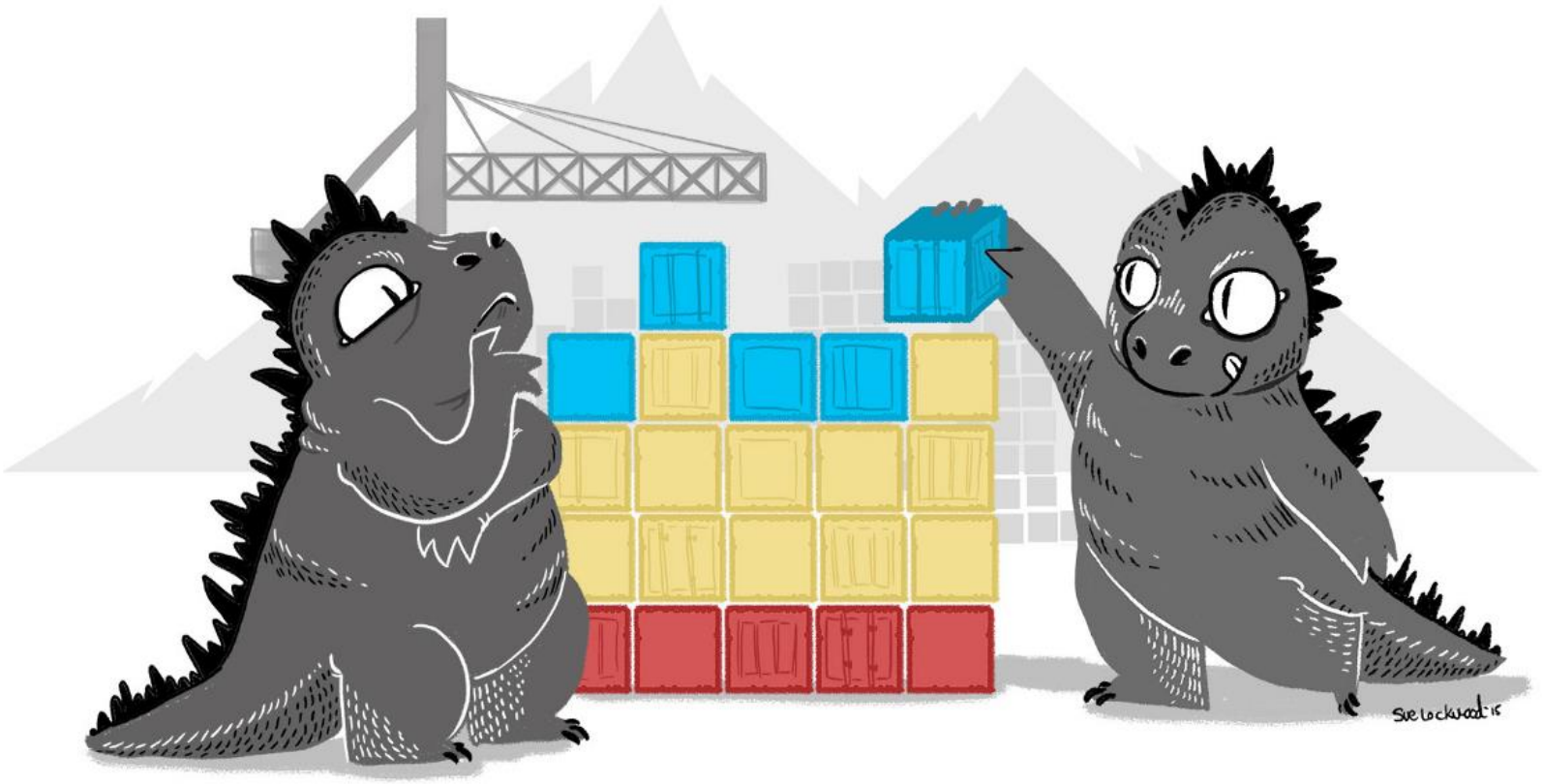


Curso: Análise e Desenvolvimento de Sistemas

Disciplina: Análise e Projeto Orientado a Objetos

Profª Drª Marcia Cassitas Hino

1º Semestre/2024

Universidade Tuiuti do Paraná

O que é um sistema?

- Sistema é um sistema é um conjunto de partes coordenadas para atingir um objetivo.

- Exemplos:
  - Sistema respiratório (respiração)
  - Sistema educativo (acesso à educação)
  - Sistema solar (gravidade)

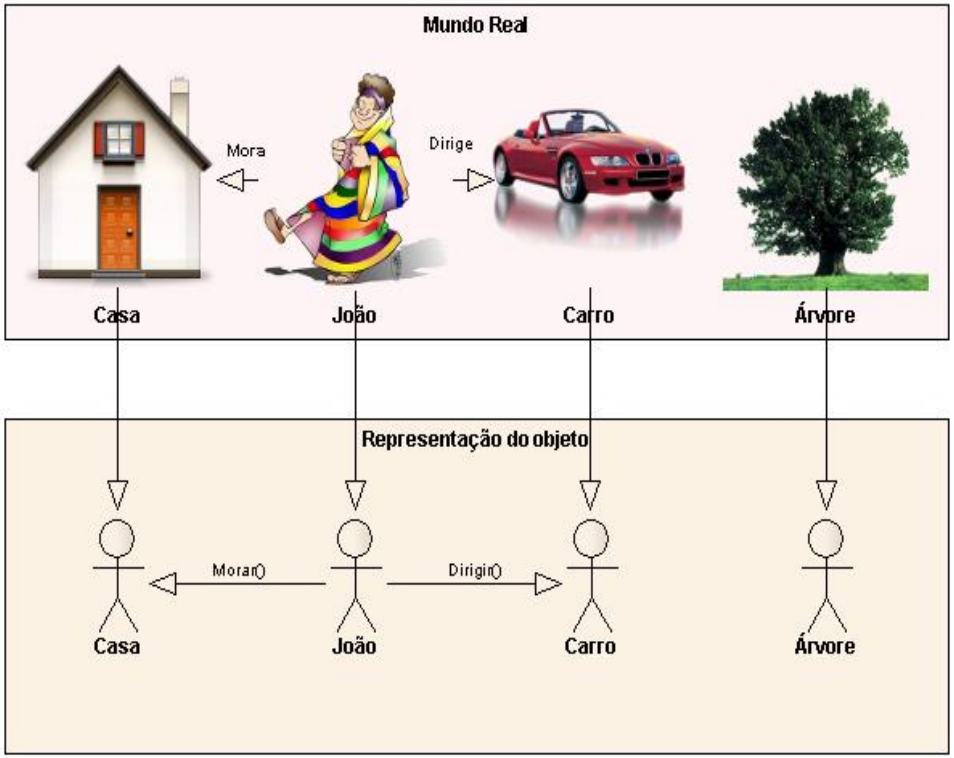
É possível construir uma casa sem definição da planta? Sem ver o todo? Sem entender como as partes se integram?

É possível construir um software sem definir sua arquitetura.

Paradigmas

- Um paradigma é um padrão, a forma como você soluciona um problemas.
- Paradigma de programação é um meio de se classificar as linguagens de programação baseado em suas funcionalidades.
- Entender os paradigmas auxilia o desenvolvedor a entender melhor e estruturar o sistema.
- Exemplos:
  - Programação estruturada
  - Programação orientada a objeto
  - Programação procedural
  - Programação funcional
  - Programação orientada a fluxos
  - Programação orientada a eventos

Orientação a Objetos



O termo “orientação a objetos” significa organizar o mundo real como uma **coleção de objetos** que incorporam estrutura de dados (**atributos**) e um conjunto de ações (**métodos**) que manipulam estes dados.

Orientação a Objetos

- Conceitos:
  - Objetos
  - Classes

- Características:
  - Encapsulamento
  - Abstração
  - Herança
  - Polimorfismo

Objeto

- Objetos são abstrações de entidades do mundo real (ou de algum sistema) que se auto gerenciam.
- Objetos são independentes e encapsulam suas representações de estado e de informações.
- A funcionalidade de um sistema é expressa em termos de serviços que os objetos prestam.

DEFINIÇÃO

- É uma entidade lógica que contém atributos e métodos
  - **Atributos**: são as características próprias dos objetos
    - Exemplo: nome da pessoa, cor do carro, valor da transação
  - **Métodos**: são blocos de instruções que manipulam os atributos
    - Exemplo: falar (pessoa), acelerar (carro), debitar (transação)

MUNDO REAL

Algo que pode ser percebido pelos sentidos.






MUNDO COMPUTACIONAL

Molde que passa a existir por meio de um conjunto de dados e métodos.

nome data de nascimento sexo profissão
andar() falar() ouvir() comer() dormir() vestir()

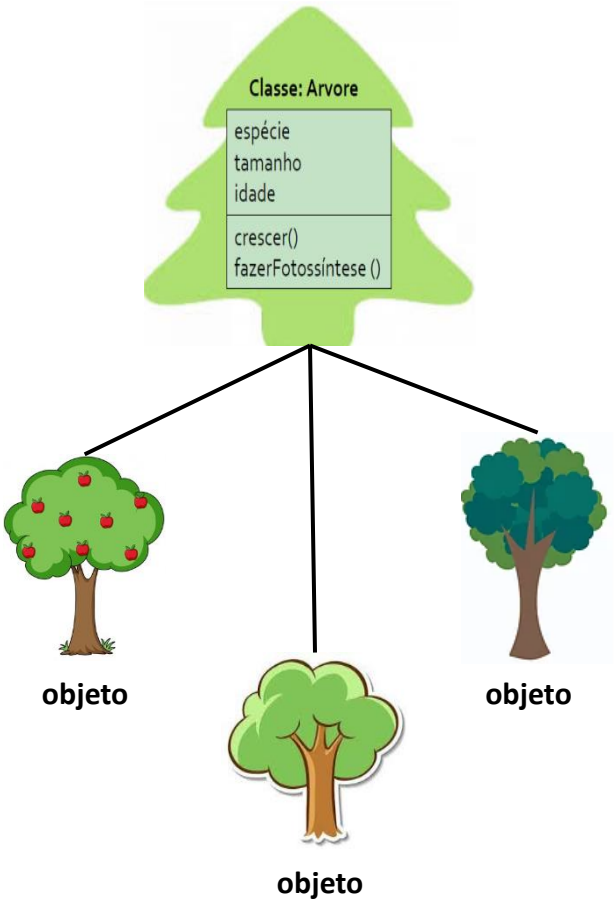
Um objeto possui:

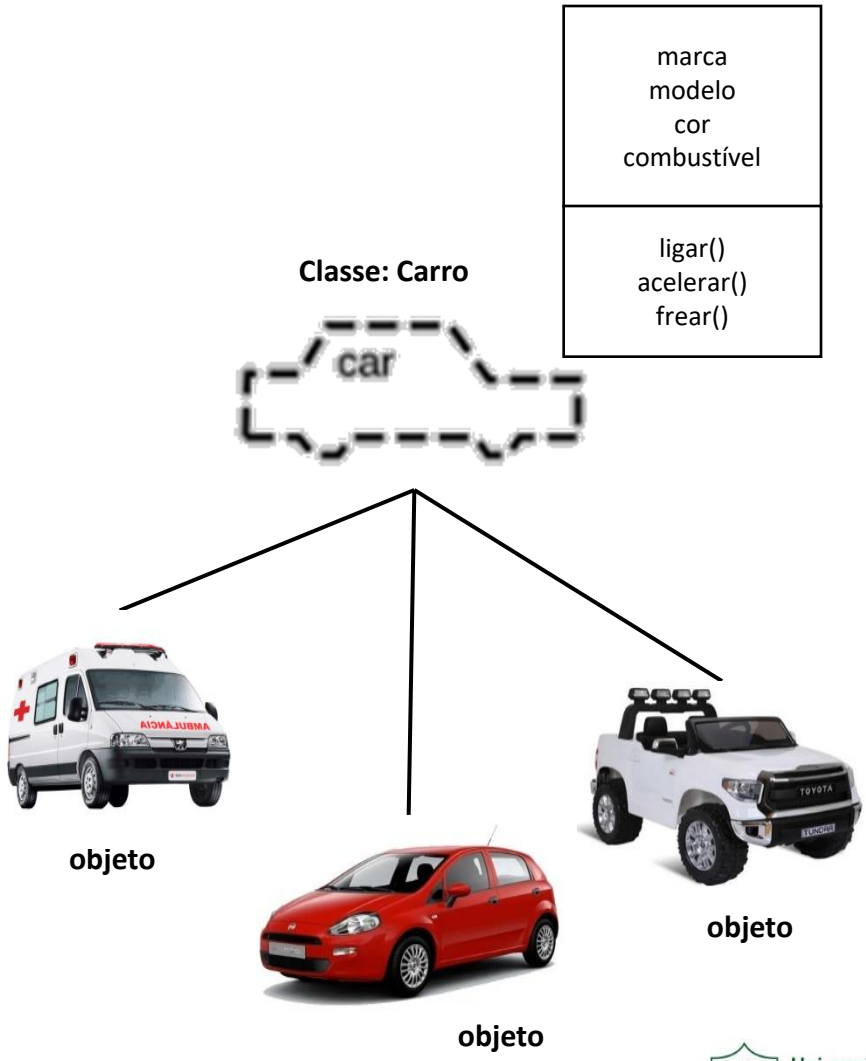
- Uma **identidade** única.
- Um **comportamento**
  Definido pelo conjunto de métodos definido na sua interface.
- Um **estado**
  Definido pelo conjunto de valores dos seus atributos em determinado instante.

Classe

- É uma **estrutura de dados** para declarar variáveis. Essa estrutura serve para definir objetos.
- Uma instância de uma classe é chamada de Objeto.
- Definir uma classe não cria um objeto, assim como um tipo de variável não é uma variável.

EXEMPLOS


Classe: Arvore
espécie
tamanho
idade
crescer()
fazerFotossíntese ()
objeto
objeto
objeto


marca
modelo
cor
combustível
ligar()
acelerar()
frear()
Classe: Carro
car
objeto
objeto
objeto

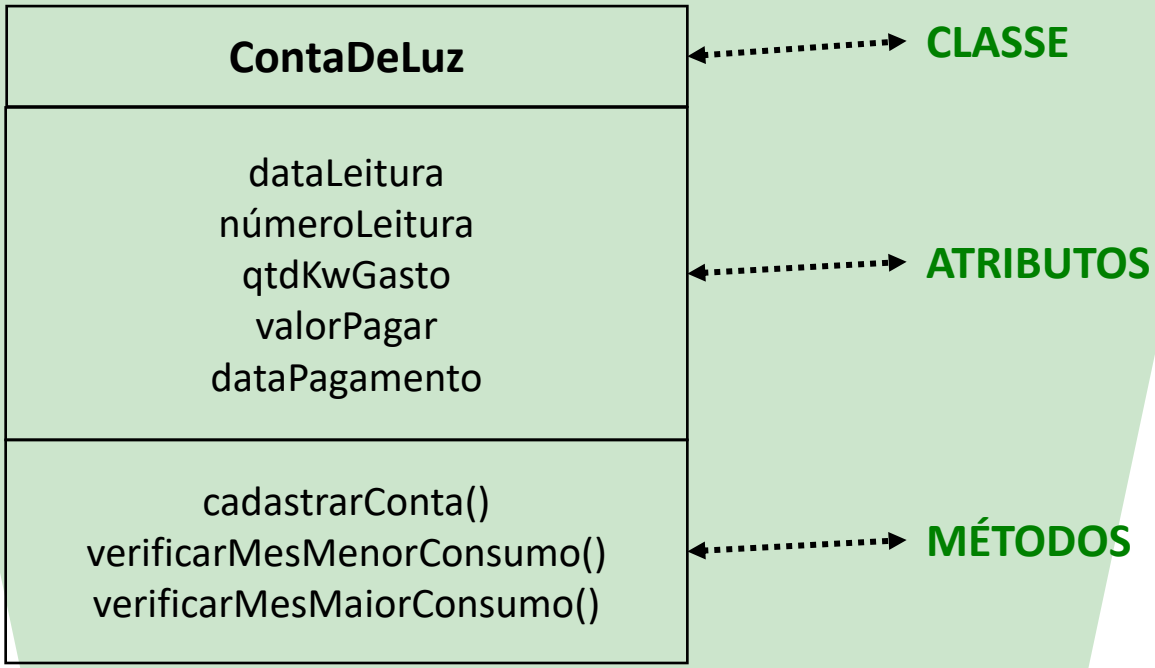
Importante entender que Classes não são Objetos, mas a estrutura de um Objeto.

EXERCÍCIO 1: Identifique as classes, atributos e métodos do cenário abaixo:

João controla todo o gasto mensal da sua conta de luz em uma planilha eletrônica.
Para cada conta de luz ele registra: data em que a leitura do relógio de luz foi realizada, número da leitura, quantidade de Kw gasto no mês, valor a pagar pela conta e data do pagamento.
Mensalmente se pesquisam o mês de menor consumo e o de maior consumo.

Universidade Tuiuti do Paraná

EXERCÍCIO 1: Resultado



Universidade Tuiuti do Paraná

Encapsulamento

- Um dado está encapsulado quando envolvido por código de forma que, só é visível na rotina onde foi criado.
- Não interessa saber como é o funcionamento interno da classe e sim sua função. É como uma “caixa preta”.
- Se refere ao agrupamento de dados com os métodos que operam nesses dados ou à restrição do acesso direto a alguns dos componentes de um objeto.

EXEMPLOS DA VIDA REAL

Quando você aperta o botão “*ligar*” do aparelho remoto da televisão, você não sabe o que acontece, como o sinal chega na televisão, que circuitos são ativados, mas a televisão “*liga*”.

Quando você aperta o botão “*play*” do rádio, você sabe o que acontece? Como o som é emitido?

Todas as ações necessárias estão encapsuladas no botão.

VISIBILIDADE

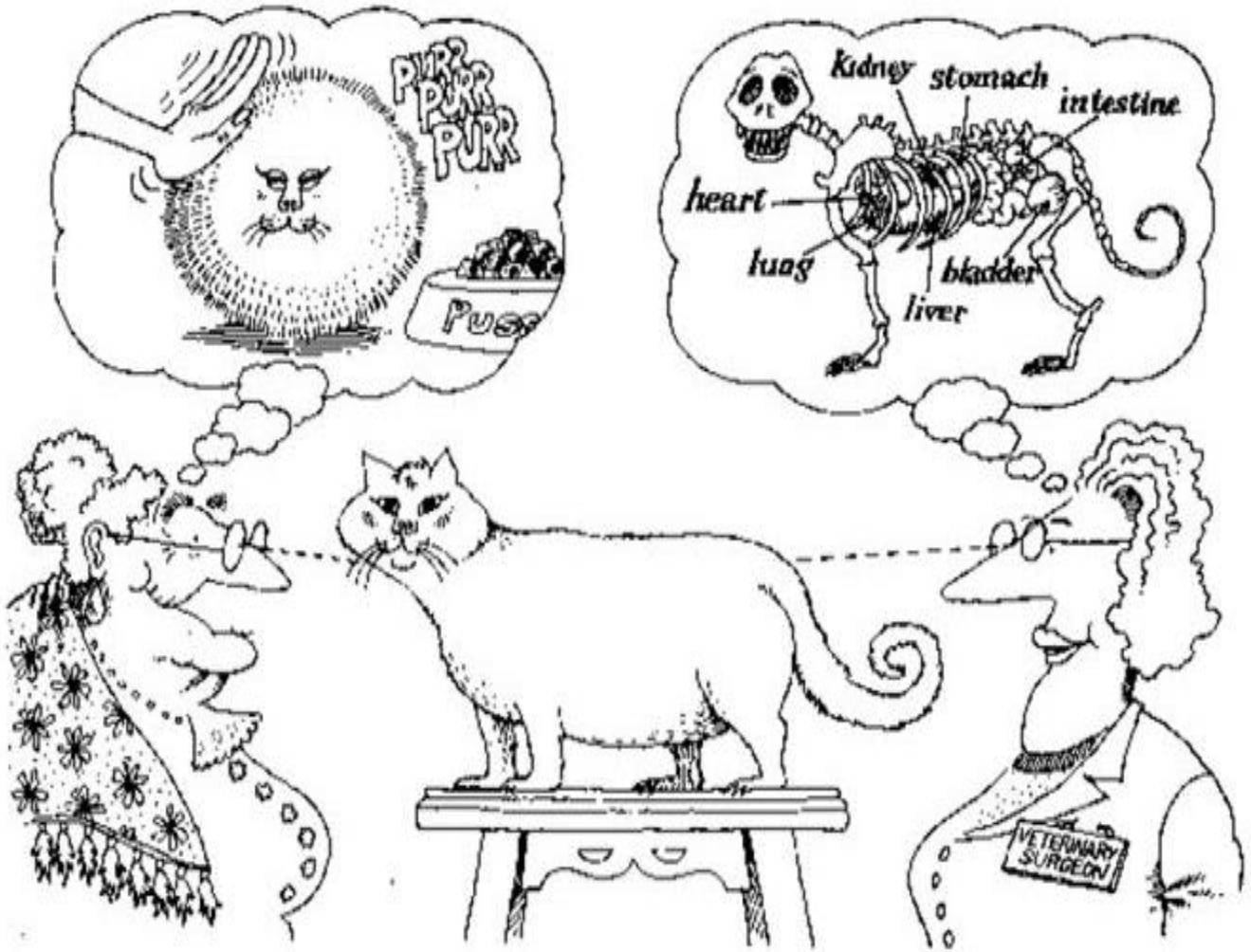
As opções de visibilidade que podem ser definidas em uma classe para atributos e serviços são:

- **+ public**: os elementos são acessíveis por todas as classes
- **#protected**: os elementos são acessíveis por subclasses, ou pela própria classe
- **–private**: os elementos são acessíveis somente pela própria classe

Abstração

- Focalizar o essencial, ignorando os pormenores
- Deve ser executada com algum objetivo para saber o que é importante e o que não é
- “A abstração é o processo de filtragem de detalhes sem importância do objeto, para que apenas as características apropriadas que o descrevem permaneçam” (*Peter Van*)
- “Extrair tudo o que for essencial e nada mais” (*Aaron Walsh*)

A abstração está nos olhos de quem vê.



Universidade Tuiuti do Paraná

tamanhoArea quantidadeArvores valorMercado
estimarValor() vernderLenha() calcularLucro()

alimento água ninho
comer() beber() dormir()

Normalmente abstraímos de objetos aquilo que nos interessa.

NÍVEIS DE ABSTRAÇÃO

- Floresta: visão da totalidade
  - Abstraindo os detalhes
  - O mais alto nível de abstração

- Árvore: nível de abstração inferior
  - Menos detalhes abstraídos

- Folha: o mais baixo nível de abstração
  - Foco no detalhe
  - Para evitar descrição densa, sem visão da globalidade

NÍVEIS DE ABSTRAÇÃO

- Em um projeto de software, temos a necessidade de trabalhar com diferentes níveis de abstração.

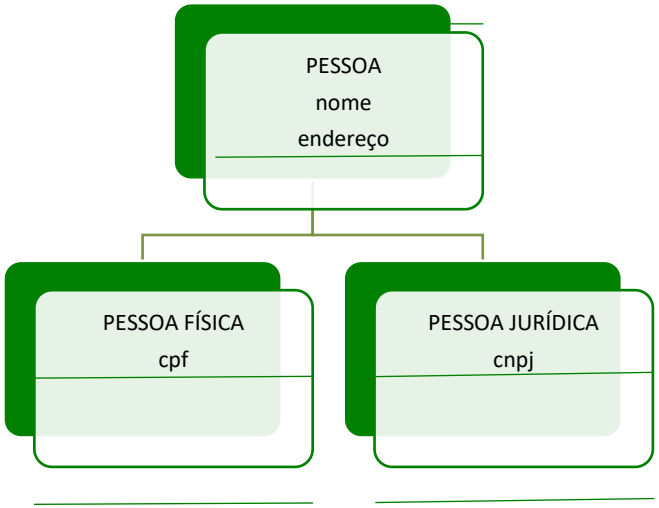


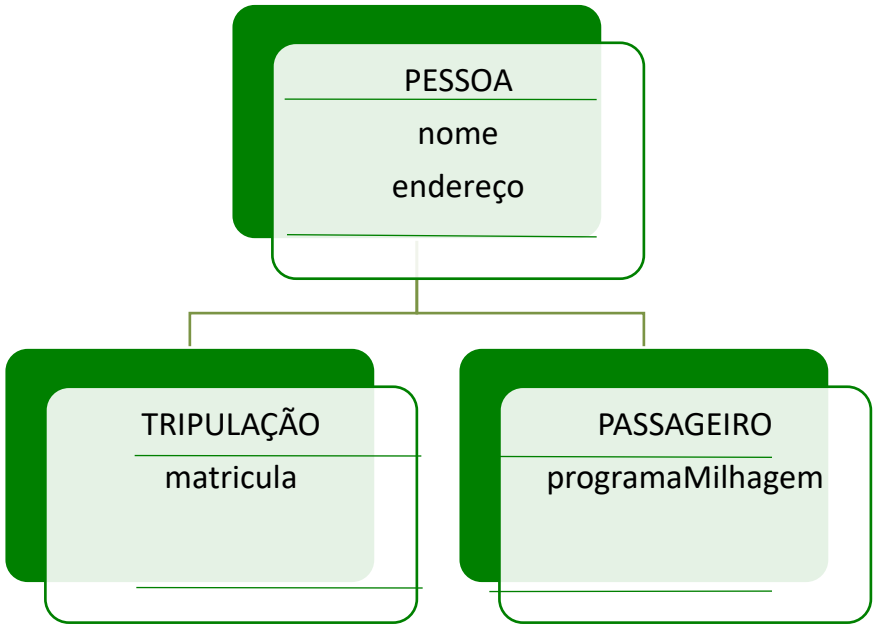
- A codificação é feita somente no nível de abstração mais baixo.

Herança

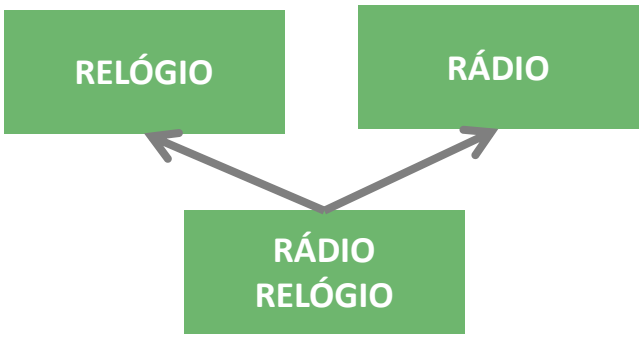
- Significa ser capaz incorporar os atributos e métodos de uma classe previamente definida.
- Tem por objetivo criar uma hierarquia de objetos do mundo real, estabelecendo um relacionamento de pai e filho
- Por meio da herança pode-se reutilizar ou alterar os métodos de classes existentes, bem como adicionar novos atributos a fim de adaptá-los a novas situações. Isso se chama especialização das classes
  - Herança simples: herda de um objeto
  - Herança múltipla: herda de vários objetos

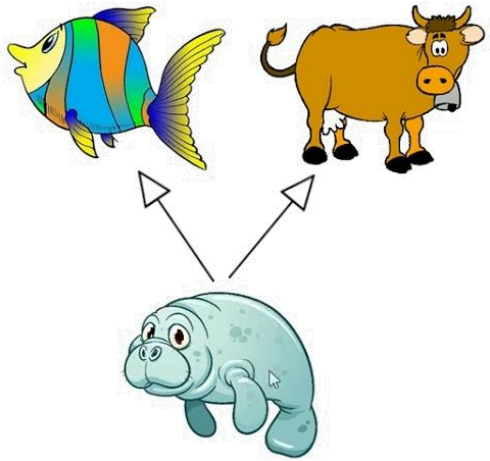
EXEMPLOS
HERANÇA SIMPLES

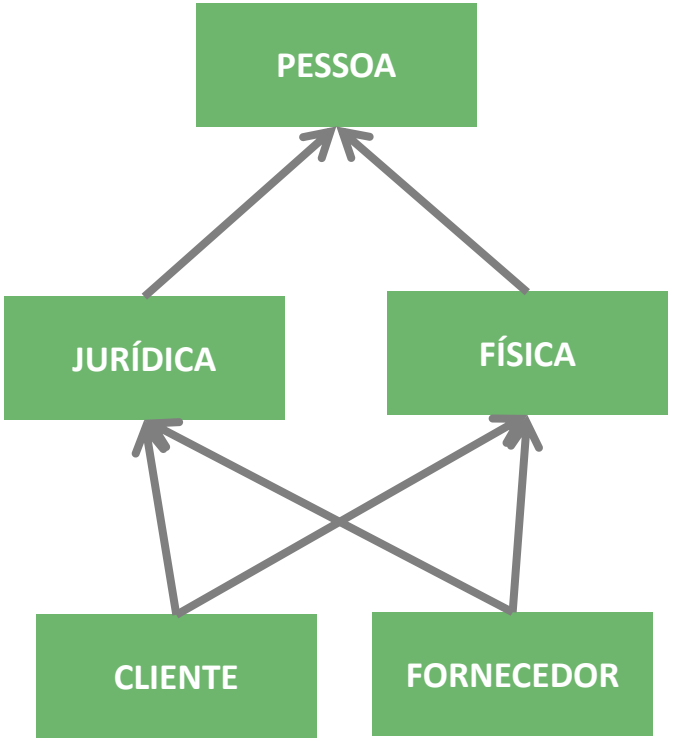




EXEMPLOS
HERANÇA MÚLTIPLA





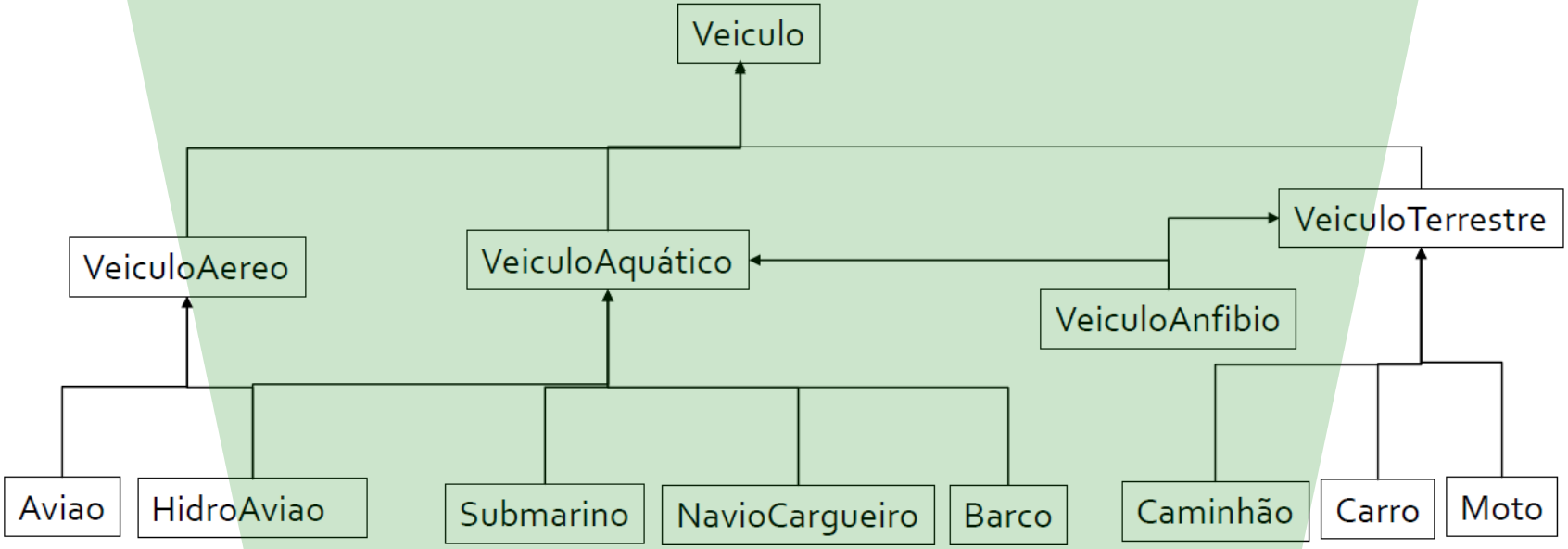


EXERCÍCIO 2: Organize hierarquicamente em um diagrama as seguintes classes:

1. VeiculoTerrestre
2. VeiculoAquático
3. Carro
4. Moto
5. Barco
6. NavioCargueiro
7. Caminhão
8. VeiculoAnfibio
9. Submarino
10. VeiculoAereo
11. Aviao
12. Veiculo
13. HidroAviao

Universidade Tuiuti do Paraná

EXERCÍCIO 2: Resultado



Universidade Tuiuti do Paraná

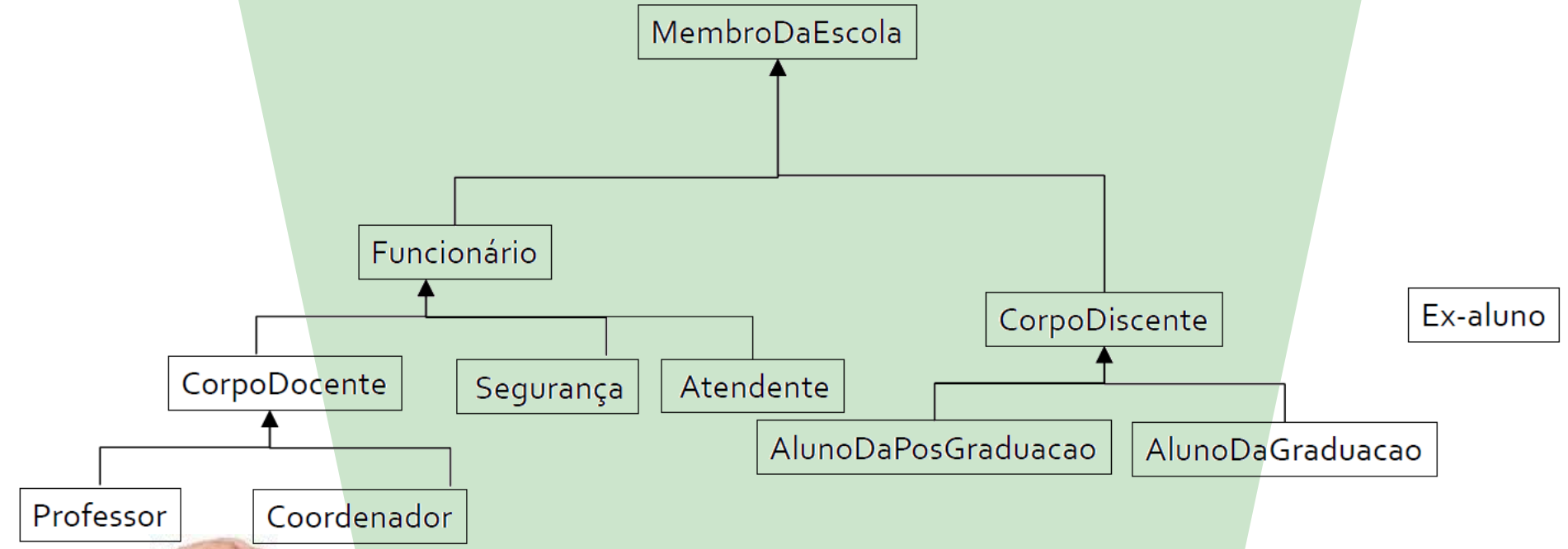
EXERCÍCIO 3: Organize hierarquicamente em um diagrama as seguintes classes:

1. MembroDaEscola
2. Professor
3. Coordenador
4. Funcionário
5. Ex aluno
6. CorpoDiscente (*)
7. AlunoDaGraduacao
8. AlunoDaPosGraduacao
9. Atendente
10. CorpoDocente
11. Segurança

(*) Discente é todo aluno regularmente matriculado, em regime de dependência ou trancado.

Universidade Tuiuti do Paraná

EXERCÍCIO 3: Resultado

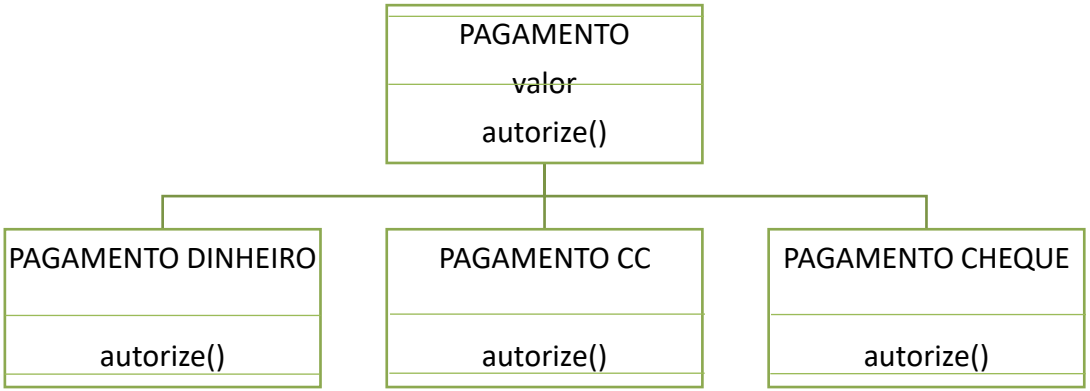


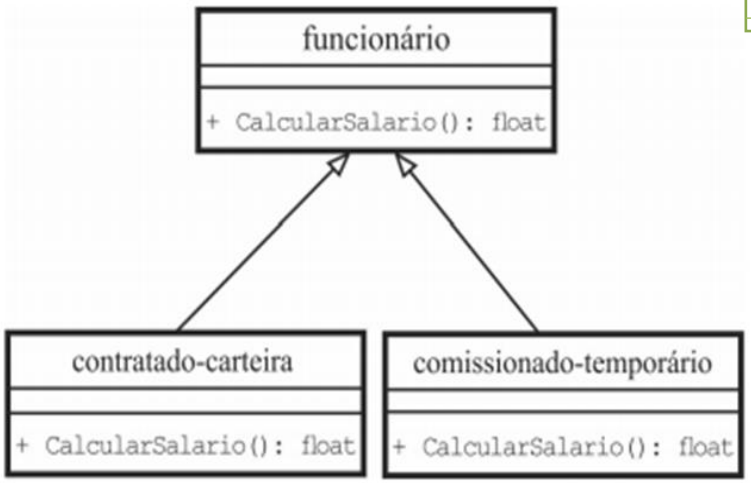
Universidade Tuiuti do Paraná

Polimorfismo

- É o princípio pelo qual classes derivadas de uma mesma superclasse podem invocar o “mesmo” método, porém com comportamentos distintos para cada uma das classes derivadas.
- Denota uma situação na qual um objeto pode se comportar de maneiras diferentes ao receber uma mensagem.
- Consiste em utilizar o mesmo nome do método dentro de uma classe, ou em uma hierarquia de classes, com comportamentos diferentes.
- É alcançado com auxílio do uso de herança nas classes e a reescrita de métodos das superclasses nas suas subclasses.

EXEMPLOS DE POLIMORFISMO





Universidade Tuiuti do Paraná

Fundamentos de Modelagem OO

- Descrição diagramática de algo a ser implementado em linguagem de programação
  - Modelagem prescritiva: antes do código
  - Modelagem descritiva: após o código

- Ato de criar uma representação de algo do mundo real por meio de modelos, simplificação da realidade:
  - Planta de uma casa representa mas não é a casa
  - Modelagem de sistemas: representações do sistema que estamos desenvolvendo

OBJETIVOS

- Especificar estrutura e/ ou o comportamento do sistema.
  - Softwares complexos demandam planejamento. Os de baixa complexidade podem ser construídos direto.
- Proporcionar guia para construção.
- Documentar tomadas de decisões.

VANTAGENS

- Descrição mais facilmente compreensível
  - Mais próxima da forma como as pessoas pensam
  - Não é natural “pensar” em linguagem de programação
- Proporciona diferentes pontos de vista
  - Descrição dos elementos que compõem um programa (estrutura)
  - Descrição do programa em execução (dinâmica)
  - Possibilidade de visão global
  - Possibilidade de atenção a detalhes.

O QUE TEMOS QUE APRENDER PARA MODELAR EM OO?

- Conhecer os conceitos referentes a modelagem
  - Saber porque modelar
  - Conhecer os paradigmas de OO
  - Conhecer os requisitos de uma modelagem completa
- Conhecer uma linguagem de modelagem
  - Ex: UML e todos seus diagramas
- Saber que passos seguir
  - Saber usar a linguagem de modelagem adotada
- Avaliar o que for produzido
  - Avaliar consistência do todo
  - Parâmetros de qualidade

Obrigada!