Programação Orientada a Objetos

Aula 5 – Herança Polimorfismo

Dra. Ana Patrícia F. Magalhães Mascarenhas

anapatriciamagalhaes@gmail.com

apmagalhaes@uneb.br

Plano de Aula

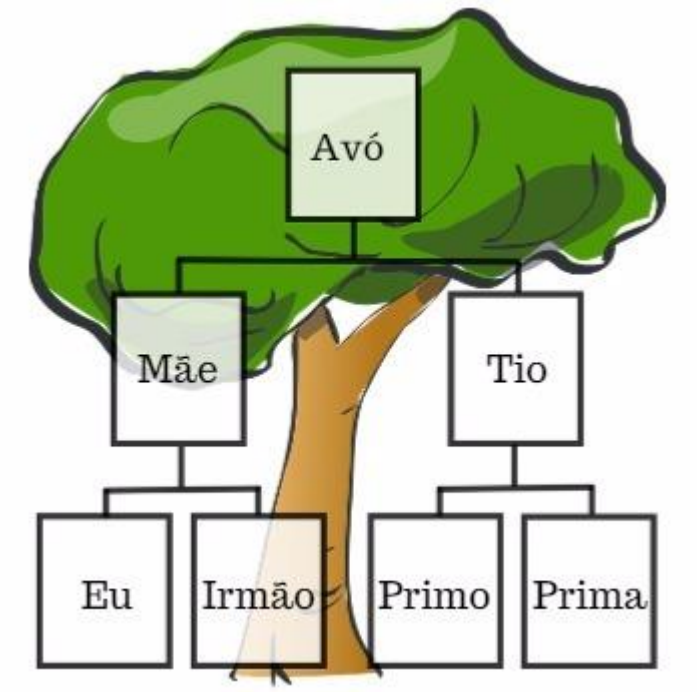
- Objetivo
  - Definir os conceitos de herança
  - Interpretar quando a herança é adequada ou não
  - Discutir os múltiplos níveis de herança e herança múltipla
  - Solucionar problemas utilizando herança entre classes

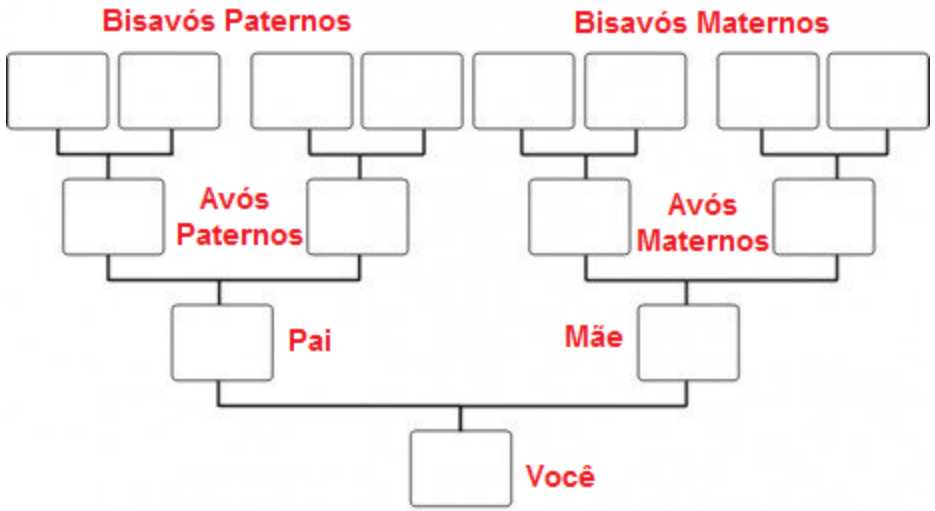
- Bibliografia básica
  - SEPE, Adriano e Roque Maitino Neto**. Programação orientada a objetos.** Londrina: Editora e Distribuidora Educacional AS, 2017. 176p.

Herança

No mundo real pessoas herdam características de seus ancestrais.

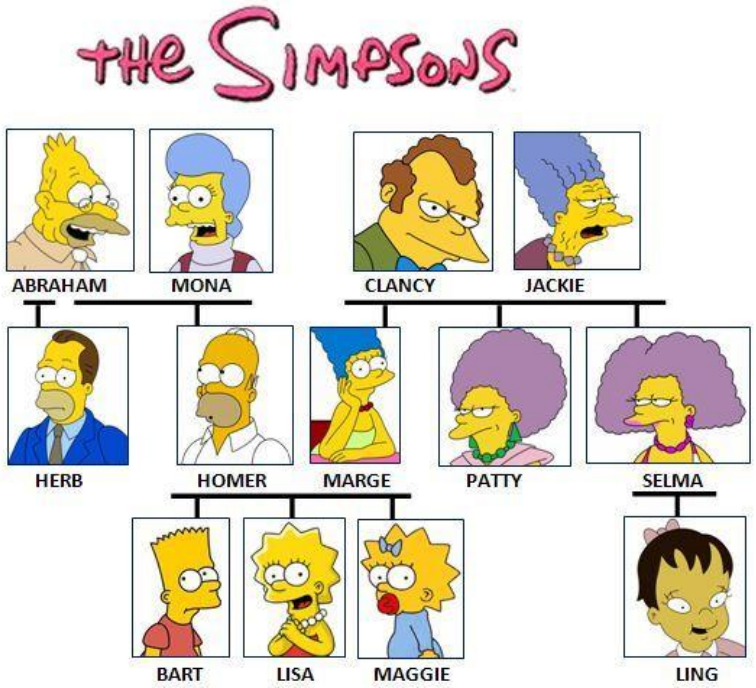
Considere um árvore genealógica com várias pessoas de uma mesma família





Herança

Na nossa árvore genealógica herdamos características de nossos pais, avós, bisavós, etc.

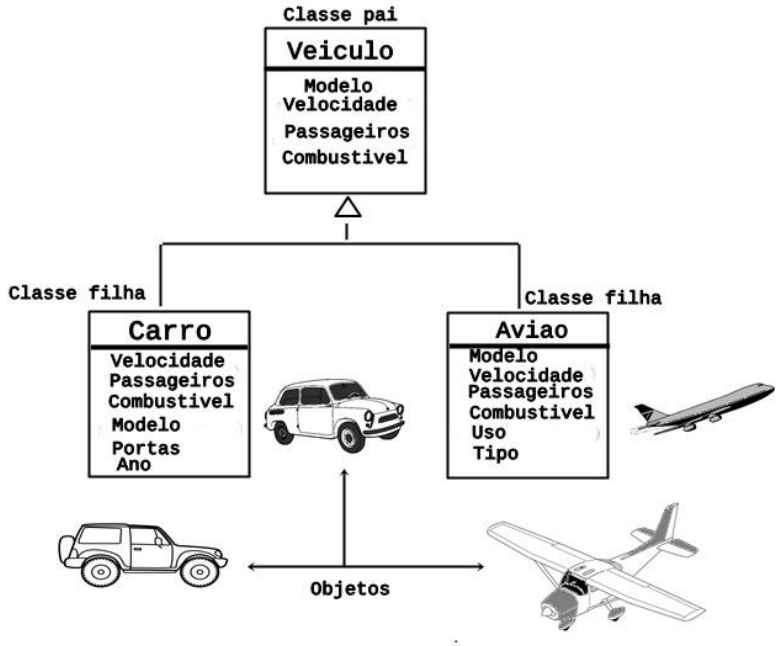


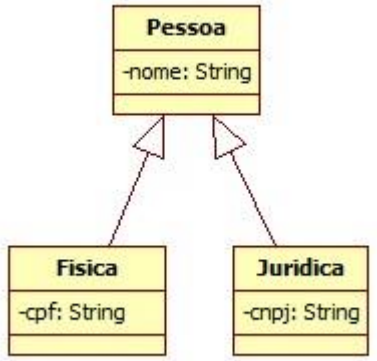
Bart herdou o cabelo da mãe e os olhos do pai
Lisa herdou o jeito da avó
Maggie herdou os olhos do pai

Se Bart e Lisa tem os mesmos olhos é porque herdaram de um mesmo ancestral. Irmão não herda de outro irmão.

Herança na OO

Na POO classes também podem herdar características (atributos e métodos) de outras classes.

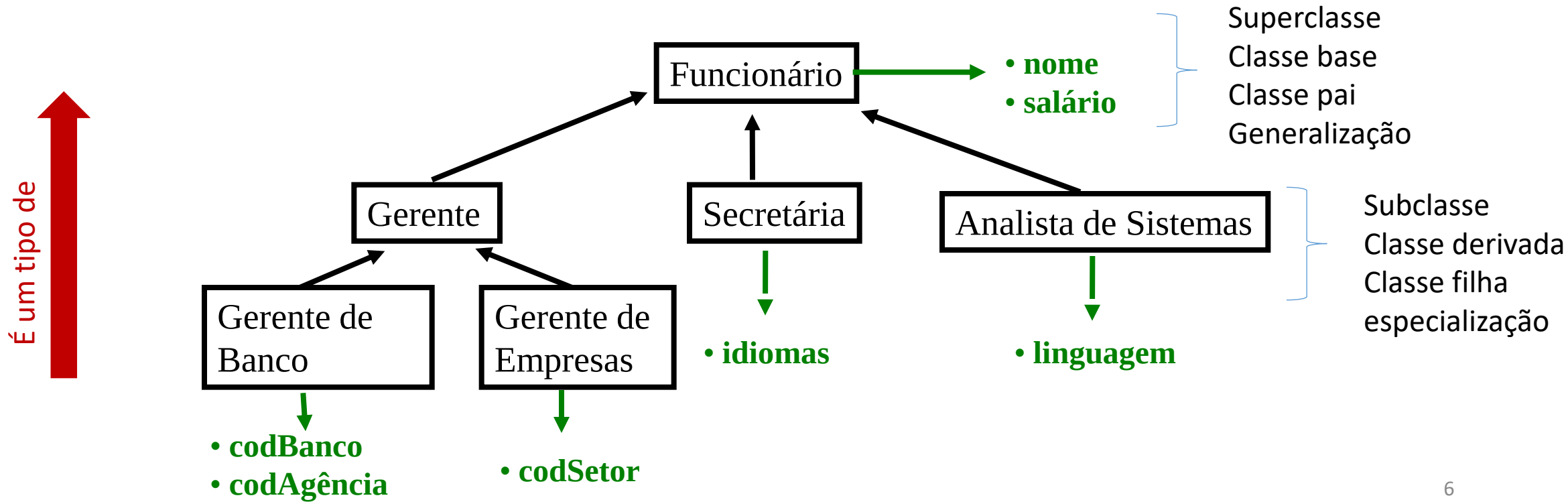




Herança: princípio da OO que permite que classes compartilhem atributos e métodos

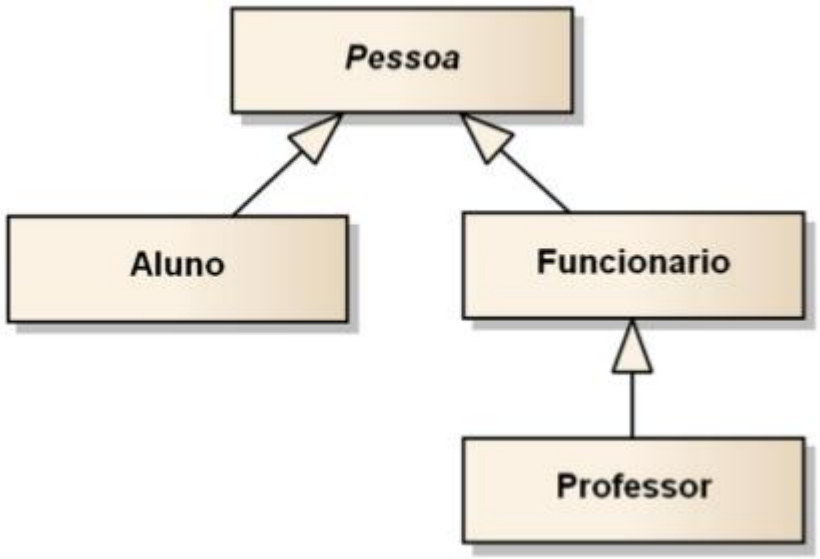
Herança em OO

Através do mecanismo de herança é possível definirmos classes genéricas que agreguem um conjunto de definições comuns a um grande número de objetos (**Generalização**). A partir destas especificações genéricas podemos construir novas classes, mais específicas, que acrescentem novas características e comportamentos aos já existentes (**Especialização**).



Herança em java

Usamos a palavra reservada *extends* para implementar herança em Java.




```
Class Pessoa{
  protected string cpf, nome;
  public Pessoa (String cpf, String nome){
     this.cpf=cpf;
     this.nome=nome;
    }
   // gets e sets
}

Class Funcionario extends Pessoa{
     protected float salario;
     public Funcionario (String cpf, String
nome, float Salario){
         super(cpf, nome);
        this.salario=salario;
       }
    //gets e sets
}
```

```
Class Professor extends Funcionario{
  protected string titulacao;

 public Professor (String cpf, String nome,
float Salario, String titulacao){
      super(cpf, nome, salario);
      this.titulação = titulação;
    }
    // gets e sets
}
```

Referencia Super

A palavra reservada **super**, nos possibilita:

- referência a classe base (pai) do objeto;

- acesso a variáveis escondidas (*shadow*), ou seja, variáveis de uma classe que tem o mesmo nome de uma variável definida em uma subclasse;

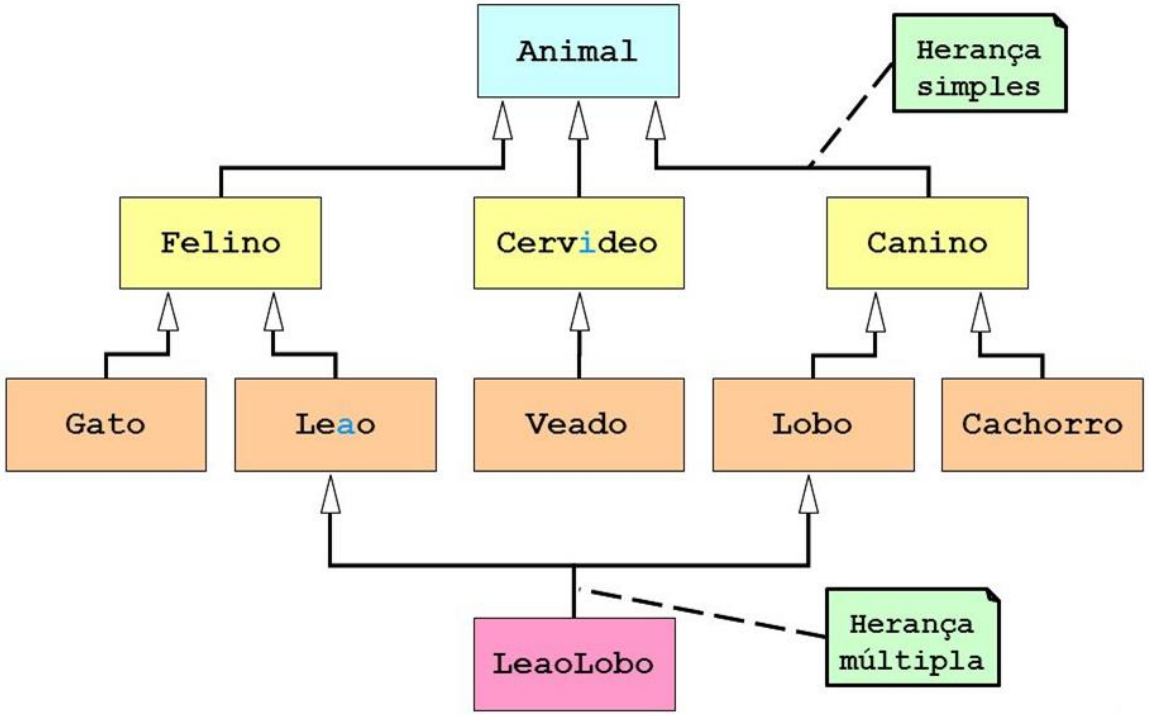
- acesso a métodos que foram redefinidos, ou seja, métodos que possuem a mesma assinatura na classe Pai e na classe derivada;

- utilização pela classe derivada do construtor da classe pai.

Herança Múltipla

Quando uma classe herda características de duas ou mais classes temos herança múltipla.

Em Java NÃO temos como implementar herança múltipla usando *extends*. Temos como simular a herança múltipla usando o conceito de Interface que estudaremos nas próximas aulas.



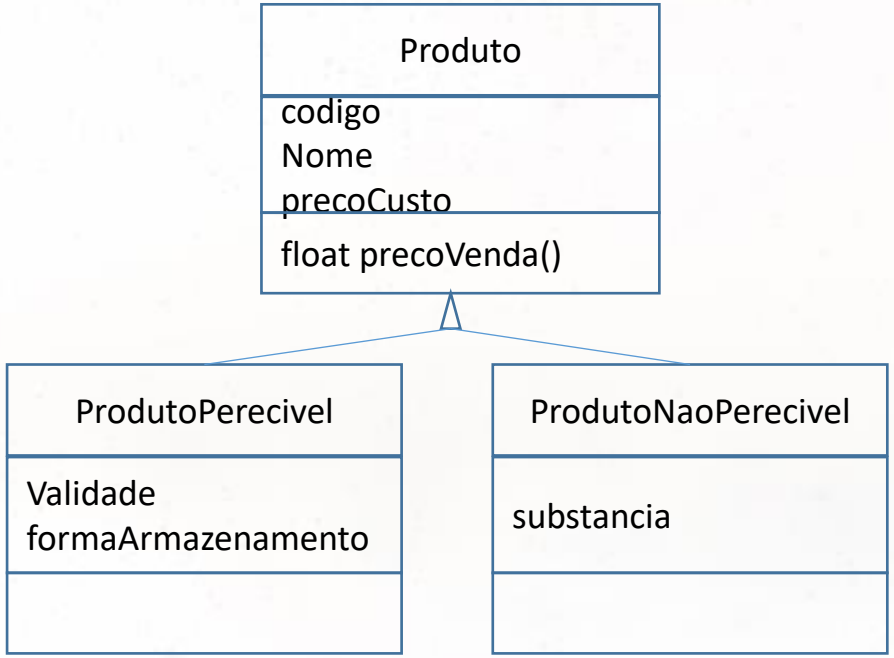
Vantagens do uso de Herança

Algumas vantagens do uso de herança são:

- Criação de uma hierarquia de classes
- Reuso de características entre classes
- Facilita a inserção de novas classes (evolução do sistema)
- Favorece o aumento da qualidade

Exercício 1

Vamos considerar a estrutura de herança abaixo para representar os produtos em um sistema de supermercado



Implemente o código referente à estrutura ao lado.

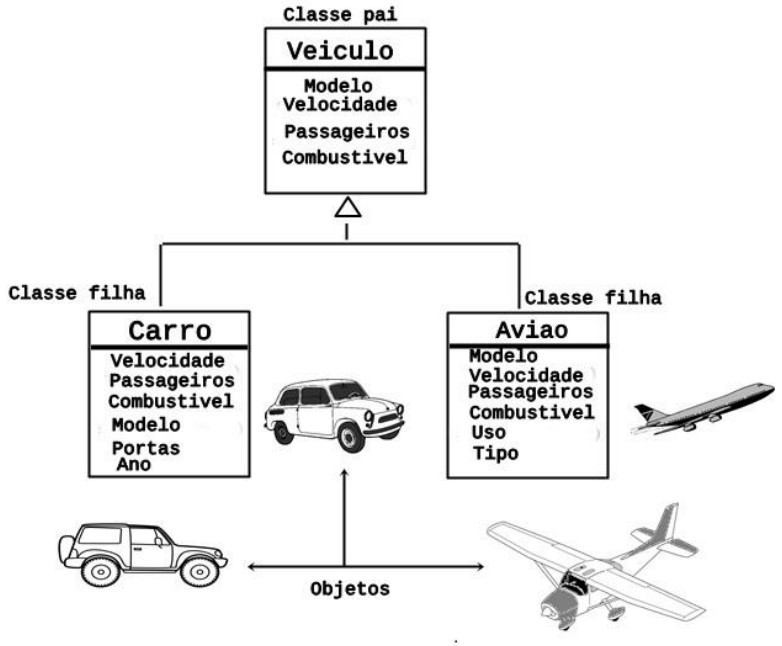
Exercício 2

Considere um sistema para um cinema. Neste sistema é preciso cadastrar as salas do cinema. Toda sala possui um número e uma capacidade de pessoas. Existem salas que são comuns, salas 3d e salas vip. As salas 3d precisam cadastrar também o nome do equipamento de projeção, que é especial. Para as salas vips é necessário informar no cadastro o horário de funcionamento da sala, pois ela não fica aberta todos os horários.

Implemente em Java uma estrutura de classes que possibilite cadastrar as salas do cinema.

Polimorfismo

Objetos pertencentes a classes distintas em uma estrutura de herança podem se comportar de maneira diferente na mesma situação...



Considere que todo veículo se movimenta.

Automóveis se movimentam no chão

Aviões voam

Barcos flutuam

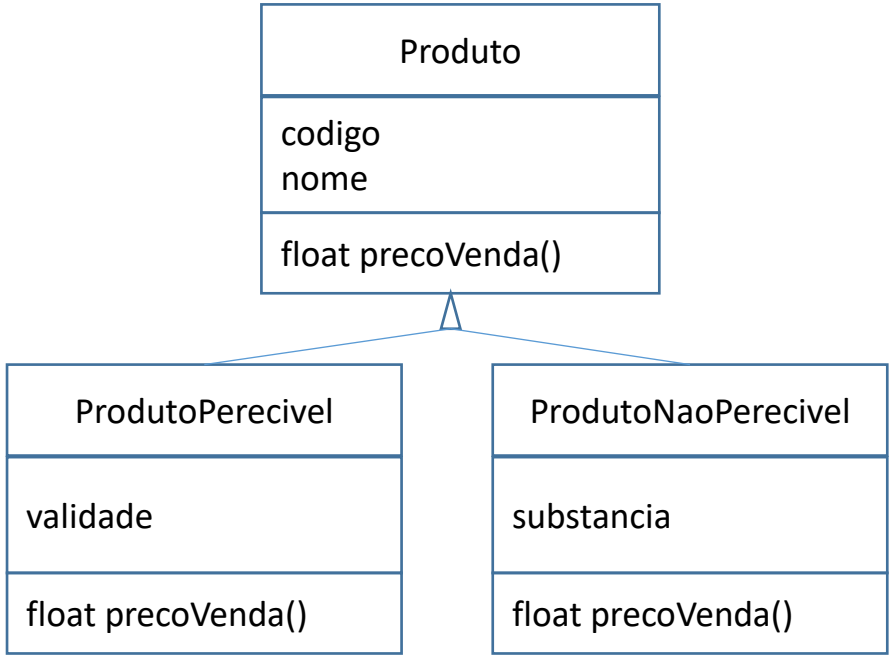
....

São diferentes formas de se movimentar

Polimorfismo

O mecanismo de herança permite que classes herdem um comportamento e possam modifica-lo de acordo com a sua necessidade.

Considere uma classe Produto e duas sub classes ProdutoPerecivel e ProdutoNaoPerecivel



O cálculo do preço de venda pode ser diferente, pois produtos perecíveis custam mais caros devido a possibilidade de perda.

Polimorfismo

O mecanismo de herança permite a criação de classes a partir de outras já existentes com relação ***é-um-tipo-de***, de forma que, a partir de uma classe genérica, classes mais especializadas possam ser criadas.

O polimorfismo decorre desta relação ***é-um-tipo-de.***

Podemos definir uma variável do tipo (pai) e colocar nela objetos do tipo (filho), pois o "filho" é-um-tipo-de "pai"

ProdutoPerecível ***é-um-tipo-de*** Produto
ProdutoNaoPerecível ***é-um-tipo-de*** Produto

Então é possível escrever **Produto p1 = new ProdutoPerecivel()**

Quando exercutarmos o método **p1.precoVenda()** é identificado que p1 contem um objeto ProdutoPerecivel e o método correto é executado. Isso é possível devido ao conceito de **lateBinding.**

Biding

O processo de ligação de uma invocação de um método ao corpo equivalente correto é conhecido por ***biding*** ou **ligação**.

Existem 2 métodos de *biding* possíveis:

- *Early Binding*
- *Late Biding*

Early Biding

- Realizada em tempo de compilação
- Baseia-se no tipo das variáveis
- Garante execuções mais rápidas
- Típicas das linguagens procedurais
- Em Java é usada para binding de método Final

Late Biding

- Realizada em tempo de execução
- Baseia-se no tipo do objeto
- Garante maior flexibilidade na execução
  - Upcast
  - Polimorfismo
- Típica das linguagens orientadas a objetos

Foi o que aconteceu no nosso exemplo de **p1.calcPrecoVenda()**
Em tempo de execução a ligação foi refeita e foi identificado o método correto a ser executado.

Polimorfismo

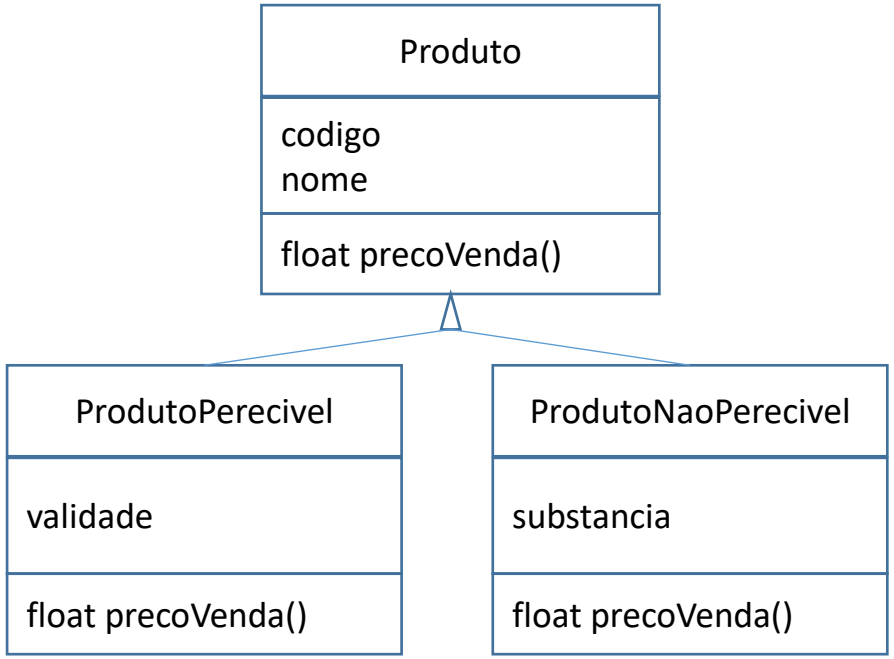
Vantagens do uso do Polimorfismo:

Permite o envio de mensagens a um objeto sem a determinação exata do seu tipo;

Permite a extensibilidade do sistema com pouco ou nenhum impacto nas classes existentes;

Permite a construção de classes genéricas para efetuar tarefas gerais comuns aos softwares, como as estruturas de dados.

Vamos implementar o exemplo abaixo



O cálculo do preço de venda pode ser diferente, pois produtos perecíveis custam mais caros devido a possibilidade de perda.

Analisando um exemplo mais completo

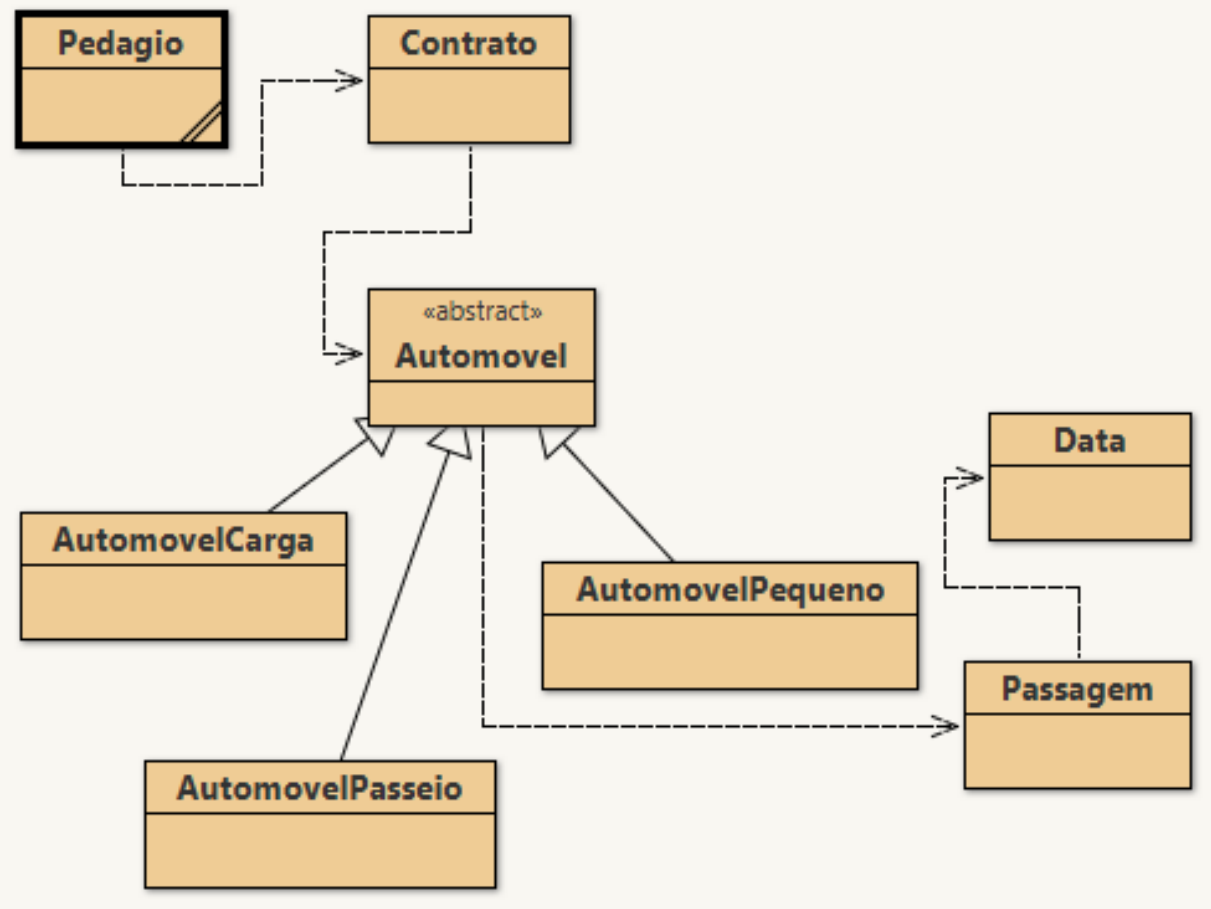
A empresa que administra o pedágio da BR101 oferece um serviço para clientes que trafegam diariamente pela rodovia onde é instalado um equipamento no automóvel para sempre que este passe pelo pedágio, não precise parar e fazer o pagamento. O equipamento emite um sinal para a empresa que registra a passagem e emite uma fatura para pagamento mensal no final do mês. A empresa solicita que você desenvolva esse sistema de acordo com a seguinte especificação

Todo automóvel que trafega na rodovia possui uma placa e um ano de fabricação. Existem vários tipos de automóveis. Os automóveis de carga devem registrar também o peso máximo que podem carregar; os automóveis de passeio devem registrar a quantidade de passageiros que pode comportar (em unidades); e os automóveis pequenos (exe. Moto) devem registrar o modelo do automóvel. Desta forma, os clientes fazem um contrato com a empresa informando seu CPF/CNPJ, nome, endereço, email e telefone.

Cada cliente pode cadastrar diversos automóveis em seu contrato. Cada vez que um automóvel passa pelo pedágio fica registrada uma passagem guardando a data e a hora. Por exemplo, o cliente de CPF 1234, chamado Maria, que mora na Rua Amazonas, tem email Maria@gmail.com e telefone 99999999, registra um automóvel de passeio de placa ABS-2233, ano 2009 com 5 passageiros. Durante o mês este automóvel passa 10 vezes pelo pedágio, então são registradas 10 passagens associados ao automóvel cada uma com data e hora.

O valor do pedágio varia a depender do tipo do automóvel. O automóvel de carga paga R$ 2.00 por quilo que o automóvel pode carregar. O automóvel de passeio paga R$ 5,00 por quantidade de pessoas que comporta. O automóvel pequeno para uma taxa única de R$ 6.00.

Modelo da solução, vamos implementar



Exercício

Lista de exercícios nr. 4