Herança e Polimorfismo

PROFESSOR: EDUARDO HABIB BECHELANE MAIA

HABIB@CEFETMG.BR

Herança

Capacidade de compartilhar estruturas comuns entre diversas classes derivadas

Reaproveitamento de código da classe pai

Herança

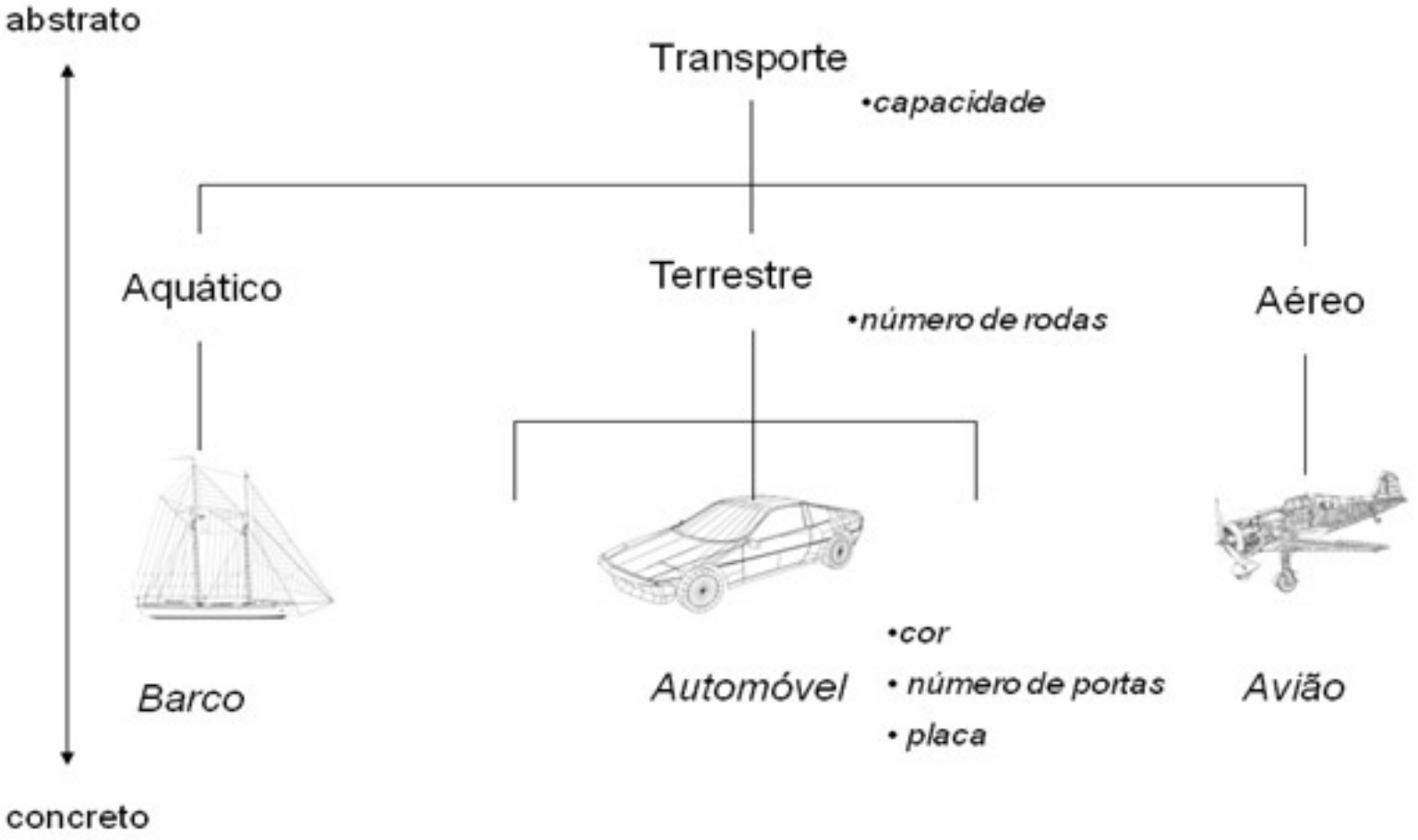
É o processo em que um objeto pode adquirir as características de outro objeto.

Herança Simples: um objeto herda as características de uma única classe.

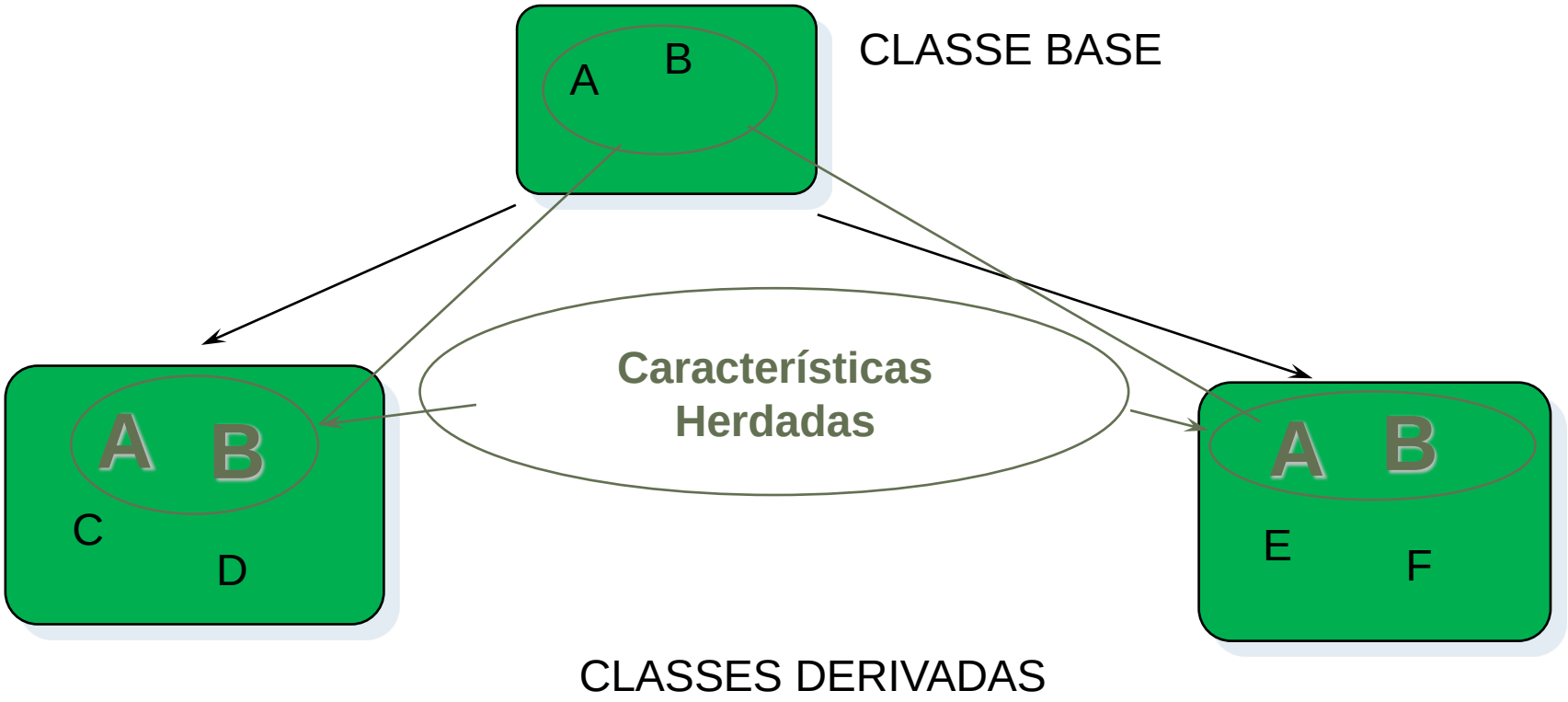
Herança Múltipla: uma objeto herda características de mais de uma classe.

- Algumas linguagens não permitem

Herança


abstrato
Transporte
•capacidade
Aquático
Terrestre
•número de rodas
Aéreo
Barco
Automóvel
•cor
• número de portas
• placa
Avião
concreto

Herança


A
B
CLASSE BASE
Características
Herdadas
A
B
C
D
A
B
E
F
CLASSES DERIVADAS

Herança

Relacionamento entre classes onde uma classe compartilha a estrutura e o comportamento de uma ou mais classes;

Define uma hierarquia de abstrações na qual a **subclasse** herda de uma ou mais **superclasses**:

Herança simples; **Herança múltipla**;

Uma herança é um relacionamento do tipo “é um tipo de”.

Herança

- A **subclasse** herda os atributos, operações e relacionamentos da **superclasse**;
- Cada subclasse pode definir novos atributos e/ou operações;
- Cada subclasse pode redefinir operações da superclasse;
- Cada subclasse pode participar de relacionamentos específicos.

Herança

Como identificar necessidade de heranças?

- Procure por similaridades entre as classes;
- Siga a regra: a subclasse é um **tipo** da superclasse;
- Evite herança de implementação, siga a regra;
- A herança **deve** ser total, pela subclasse;

Caso a regra não seja satisfeita, utilize composição.

Herança

A **herança** é uma forma de reuso de *software*

- O programador cria uma classe que absorve os dados e o comportamento de uma classe existente;
- Ainda é possível aprimorá-la com novas capacidades.

Além de reduzir o tempo de desenvolvimento, o reuso de *software* aumenta a probabilidade de eficiência de um *software*

- Componentes já testados e de qualidade provada contribuem para isto.

Herança

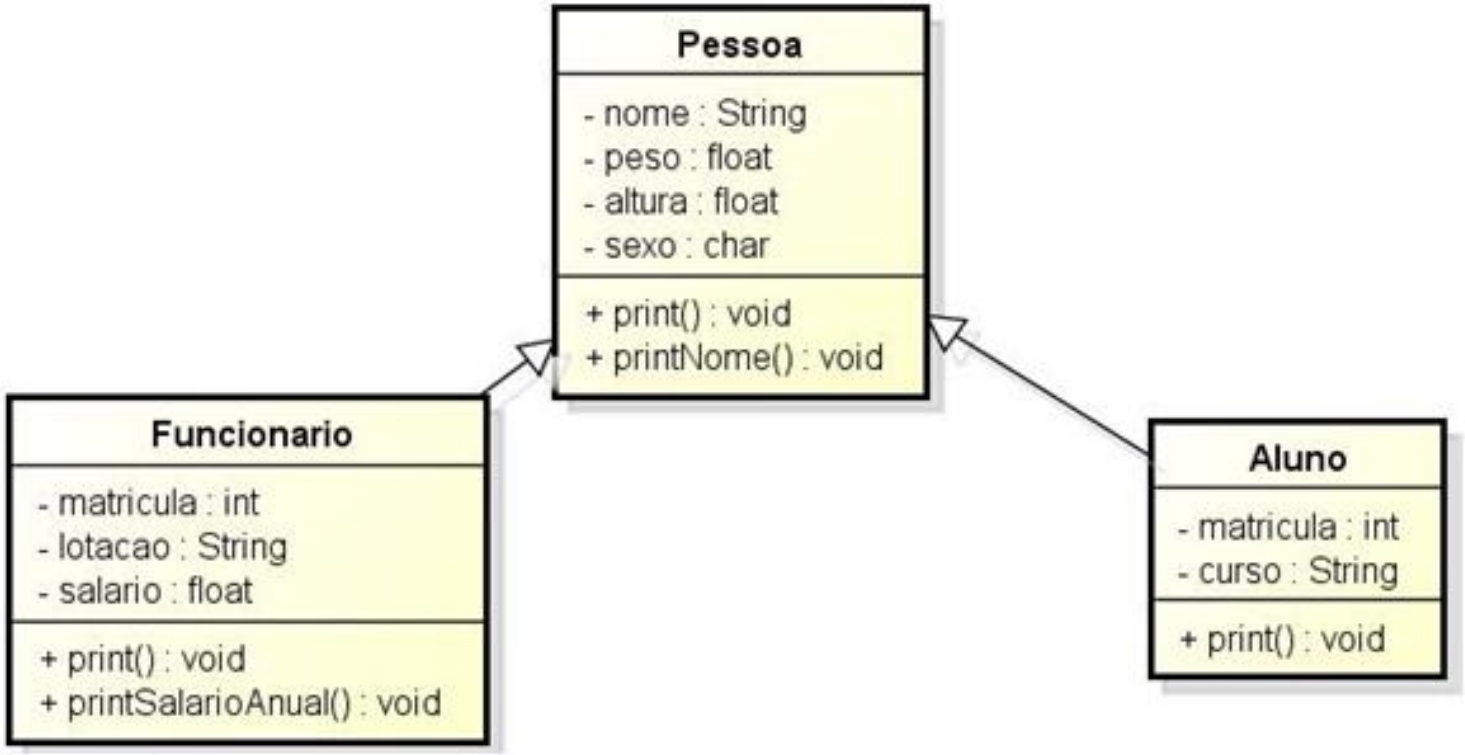
A herança define um relacionamento "**é um**"

- Um carro é um veículo
  - Todas as propriedades de um veículo são propriedades de um carro.
- Um objeto de uma classe derivada pode ser tratado como um objeto da classe base.

Os métodos de uma classe derivada podem necessitar acesso aos métodos e atributos da classe base

- Somente os membros não privados estão disponíveis;
- Ou seja, membros que não devem ser acessíveis através de herança devem ser **privados**;
  - Poderão ser acessíveis por *getters* e *setters* públicos, por exemplo.

Exemplo de herança



Polimorfismo

Um mesmo objeto pode ser de **vários tipos**;

Exemplo:

- Uma **Pessoa** pode ser um **Estudante** ou um **Professor**;

Não é viável exigir que todos os outros objetos saibam todos os possíveis tipos de um determinado objeto;

Todos os outros objetos devem reconhecer o objeto através de um **único** tipo;

Trechos de código para tratamento de diferentes tipos são eliminados;

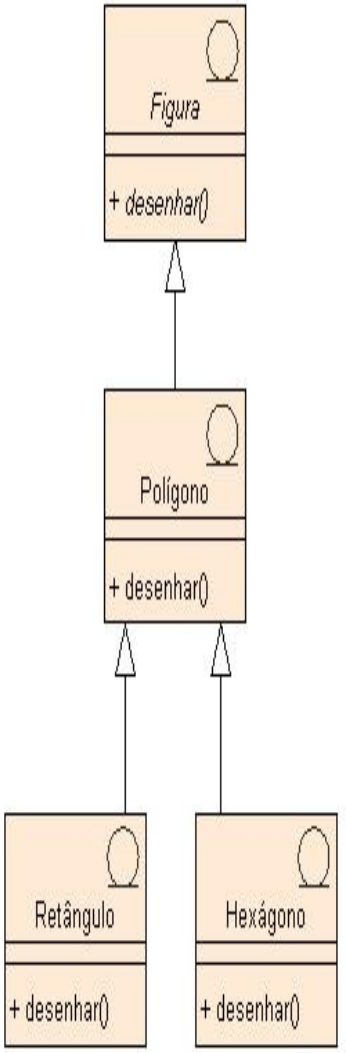
Através do **polimorfismo** instâncias de várias classes são tratadas de forma **única** em um sistema.

Polimorfismo

Cada tipo reimplementa alguma parte da interface em comum;

Outros objetos do sistema acessam a interface em comum de forma única;

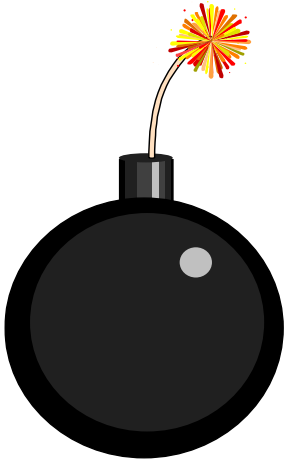
O comportamento do objeto será definido pela reimplementação contida no objeto.

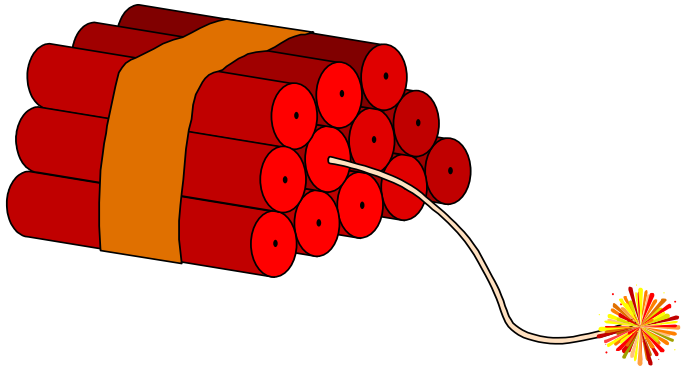


Polimorfismo

Através do **polimorfismo** instâncias de várias classes são tratadas de forma **única** em um sistema.

O Polimorfismo ocorre quando uma mesma mensagem chegando a objetos diferentes provoca respostas diferentes.





Polimorfismo

Polimorfismo de Inclusão:

- Modela **subtipos** e herança;
- O subtipo está incluído no tipo;
- Onde o objeto de um **tipo** for esperado, um objeto do **subtipo** deve ser aceito;

Exemplo:

- desenhar(Figura umaFigura).

Código em Java

https://onlinegdb.com/iwdtT7dHa

Código em C++

https://onlinegdb.com/L-WIt2yLs