REVISÃO ORIENTAÇÃO A OBJETOS

MARINA.GIROLIMETTO@UFFS.EDU.BR

ORIENTAÇÃO A OBJETOS

- **A UML está totalmente inserida no paradigma de orientação a objetos.**
- No início da infância, o ser humano aprende e pensa de maneira bastante semelhante à filosofia da orientação a objetos.
- As crianças aprendem conceitos simples, como pessoa, carro e casa, por exemplo, e, ao fazerem isso, **definem classes**, ou seja, grupos de objetos, **tendo características e comportamentos** de qualquer objeto do grupo em questão.

ORIENTAÇÃO A OBJETOS

- Uma criança deve se sentir um pouco confusa no começo ao descobrir que **o objeto amarelo e o objeto vermelho têm, ambos, a mesma classificação: carro**. “Carro” é um termo geral que se refere a muitos objetos.
- Cada **um dos objetos-carro tem características semelhantes entre si:**
  - todos têm quatro rodas;
  - tem no mínimo, duas portas;
  - tem luzes de farol e freio;
  - tem vidros frontais e laterais;
  - transportar pessoas de um lugar para outro.

ORIENTAÇÃO A OBJETOS

- O objeto é um exemplo do grupo carro, ou seja, uma instância da classe carro. Assim, **instanciação constitui-se simplesmente em criar um exemplo de um tipo, um grupo, uma classe.**

- Todos os objetos da classe carro possuem o atributo placa, mas cada um dos objetos possui um valor diferente para sua placa específica.

CLASSES DE OBJETOS

- Uma classe representa uma categoria e os objetos são os membros ou exemplos dessa categoria.
- Na UML, uma classe é representada por um retângulo que pode ter até três divisões.
- **A primeira divisão armazena o nome pelo qual a classe é identificada (e essa é a única divisão obrigatória), a segunda enuncia os possíveis atributos pertencentes à classe e a terceira lista as possíveis operações (métodos) que a classe contém.**

Figura 2.1 – Exemplo de uma classe.

ATRIBUTOS OU PROPRIEDADES

- **Os atributos representam as características**, são apresentados na segunda divisão da classe e contêm o nome do atributo e o tipo (integer, float, string...). Exemplos: o nome, o cpf ou a idade em um objeto da classe pessoa ou a placa, a cor em um objeto da classe carro.

- Os atributos podem assumir valores diversos em cada instância. Ex:
  pessoa1 – nome: “João”;
  pessoa2 – nome: “Paulo”.

Figura 2.2 – Exemplo de classe com atributos.

OPERAÇÕES, MÉTODOS OU COMPORTAMENTOS

- Classes costumam ter métodos. **Um método representa uma atividade que um objeto de uma classe pode executar.** Este, pode receber ou não parâmetros e tende a retornar valores.
- Os métodos são armazenados na terceira divisão de uma classe.

Figura 2.3 – Exemplo de classe com métodos.

VISIBILIDADE

- **A visibilidade é utilizada para indicar o nível de acessibilidade de um determinado atributo ou método.**
- Existem basicamente quatro modos de visibilidade:
  - Público (+): determina que o atributo ou método pode ser utilizado por qualquer objeto;
  - Protegido (#): além dos objetos da classe detentora do atributo ou método, também os objetos de suas subclasses poderão ter acesso a este;

VISIBILIDADE

- Privado (-): somente os objetos da classe detentora do atributo ou método poderão enxergá-lo; e
- Pacote (~): determina que o atributo ou método é visível por qualquer objeto dentro do pacote.

Figura 2.4 – Exemplo de Visibilidade.

VISIBILIDADE

- **Normalmente os atributos costumam ser privados ou protegidos, enquanto os métodos costumam ser públicos.**
- Um atributo privado, além de só ser visível por objetos de sua classe, só poderá ser acessado por meio de métodos. Assim, objetos de outras classes não terão conhecimento sobre quais atributos estão contidos na classe em questão e nem poderão acessá-los.

HERANÇA

- **A herança permite o reaproveitamento de atributos e métodos**, otimizando o tempo de desenvolvimento, além de permitir a diminuição de linhas de código, bem como facilitar futuras manutenções.
- A herança trabalha com os conceitos de superclasses e subclasses.
- Uma superclasse, também chamada de classe mãe, contém classes derivadas dela, chamadas subclasses, também conhecidas como classes-filha.

HERANÇA

- As subclasses, ao serem derivadas de uma superclasse, herdam suas características, ou seja, seus atributos e métodos.
- **A vantagem do uso da herança é óbvia: não precisamos redeclarar os atributos e métodos previamente definidos**: a subclasse herda-os automaticamente, permitindo reutilização do código já pronto.
- A herança permite trabalhar com especializações. Pode-se criar classes gerais, com características compartilhadas por muitas classes, mas que tenham pequenas diferenças entre si.

HERANÇA

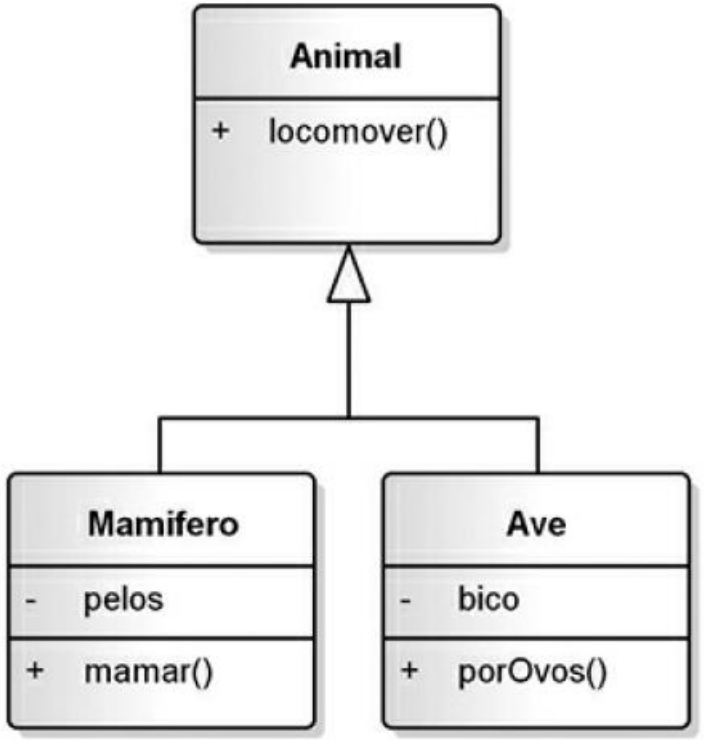

Figura 2.5 – Exemplo de Herança.

- Uma subclasse pode se tornar uma superclasse a qualquer momento, bastando para tanto que se derive uma subclasse dela.

HERANÇA MÚLTIPLA

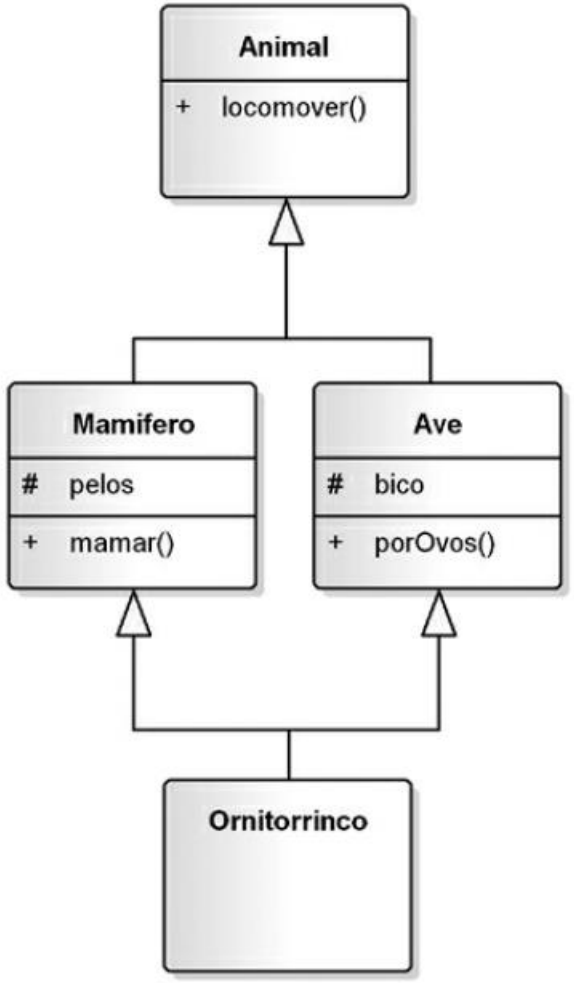

Figura 2.6 – Exemplo de Herança Múltipla.

- Basicamente, a herança múltipla ocorre quando uma subclasse herda características de duas ou mais superclasses. No caso, uma subclasse pode herdar atributos e métodos de diversas superclasses.

POLIMORFISMO

- **O polimorfismo trabalha com a redeclaração de métodos previamente herdados por uma classe**.
- Esses métodos, embora semelhantes, diferem de alguma forma da implementação utilizada na superclasse, sendo necessário, portanto, reimplementá-los na subclasse.

POLIMORFISMO

- Para evitar ter de modificar o código-fonte, redeclara-se o método com o mesmo nome declarado na superclasse.
- Assim, podem existir dois ou mais métodos com a mesma nomenclatura, diferenciando-se na maneira como foram implementados.

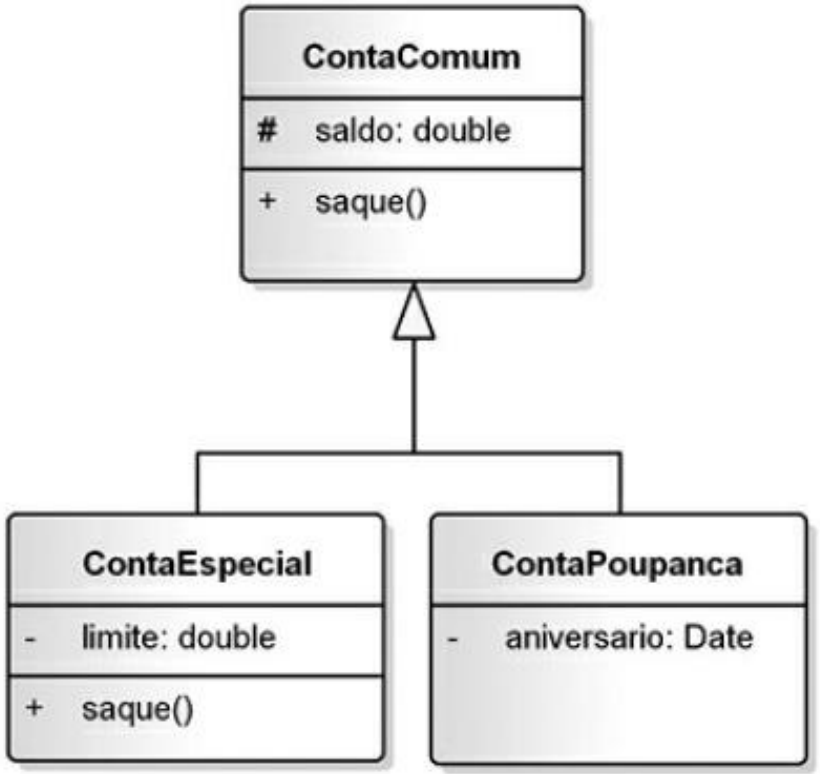

Figura 2.7 – Exemplo de Polimorfismo.

EXERCÍCIOS

1. O que é classe e como ela é dividida?
2. O que é uma instanciação de classe?
3. Explique sobre os quatro modos de visibilidade.
4. Explique a diferença entre Herança e Polimorfismo.

Aproveite o restante do tempo para pensar sobre o tema do seu trabalho.