Univiçosa – Centro Universitário de Viçosa
Cursos: Análise e Desenvolvimento de Sistemas e Engenharia de Computação
Disciplina: ADS101/ECO131 – Programação de Computadores 2

Aula 07 – Introdução ao conceito de Herança

1) Introdução

Um dos conceitos que está intimamente relacionada com a Orientação a Objetos é o uso de superclasses ou classes pai e subclasses ou classes filhas.

O uso deste recurso é chamado de herança e tem como característica principal a extensão das características da classe pai/superclasse para a classe filha/subclasse.

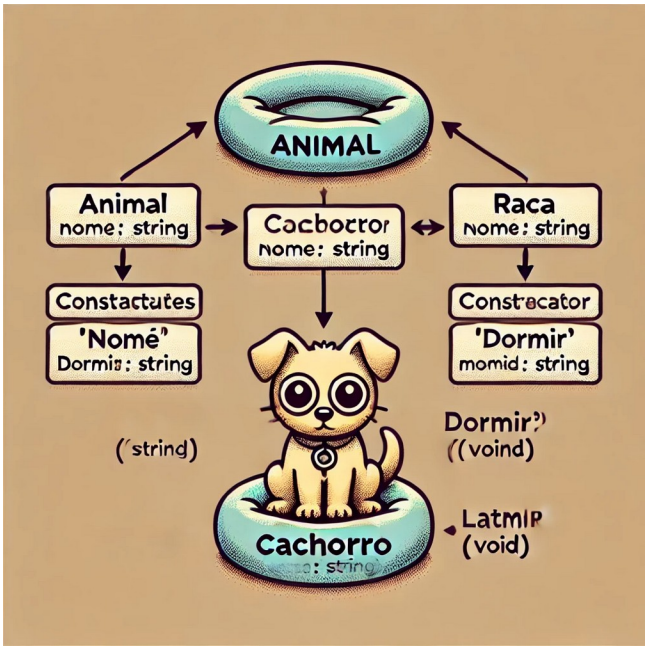
O uso deste recurso é importante pois diminui a digitação de código, como por exemplo os atributos da classe pai serem reescritas na classe filho.

Nesta aula veremos os conceitos iniciais da herança aplicada à programação C++ orientada à objetos.

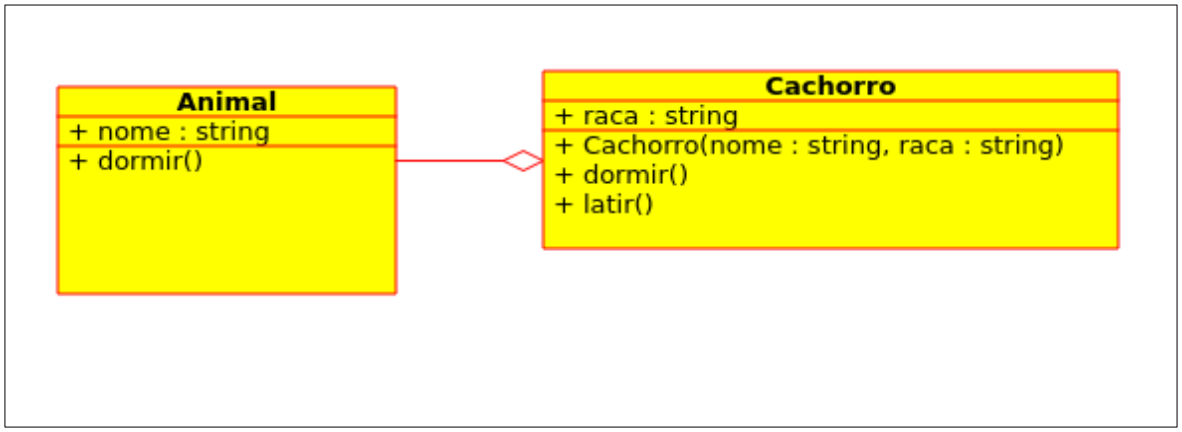
2) A especificação do programa usando duas classes

Vamos trabalhar com duas classes: Animal e Cachorro, onde Cachorro herda os atributos e métodos da classe Animal.

Observe a imagem abaixo.



O desenho acima pode ser transformado em um modelo UML segundo a imagem abaixo.



Implementando o código. Observe o quadro abaixo.

```
#include <iostream>
#include <string>

// Classe base (superclasse)
class Animal {
public:
    std::string nome;

    // Construtor da classe base
    Animal(std::string n) : nome(n) {}

    // Método da classe base
    void dormir() {
        std::cout << nome << " está dormindo." << std::endl;
    }
};

// Classe derivada (subclasse) que herda de Animal
class Cachorro : public Animal {
public:
    std::string raca;

    // Construtor da classe derivada
    Cachorro(std::string n, std::string r) : Animal(n), raca(r) {}

    // Método específico da classe derivada
    void latir() {
        std::cout << nome << " está latindo!" << std::endl;
    }

    // Sobrescrita de método (opcional, só se precisar)
    void dormir() {
        std::cout << nome << " um cachorro da raça " << raca << "está dormindo." << std::endl;
```

```
    }
};

int main() {

    // Criando um objeto da classe base
    Animal animal("Animal Genérico");
    animal.dormir();  // Chama o método da classe Animal

    // Criando um objeto da classe derivada
    Cachorro cachorro("Bolt", "Pastor Alemão");
    cachorro.dormir();  // Chama o método da classe derivada
    cachorro.latir();   // Chama o método específico de Cachorro

    return 0;
}
```

Explicando o código

Explicação:

1. Classe Base (Animal):
    - A classe Animal é a classe "pai" ou "base", que contém um atributo nome e um método dormir().
    - Esse método pode ser utilizado por qualquer objeto que seja da classe Animal ou de uma classe que herde de Animal.

2. Classe Derivada (Cachorro):
    - A classe Cachorro herda de Animal usando a sintaxe : public Animal. Isso significa que a classe Cachorro pode acessar os membros públicos da classe Animal.
    - Ela também tem um atributo extra chamado raca e um método latir(), específico para a classe Cachorro.
    - A classe derivada pode sobrescrever métodos da classe base, como o método dormir(), para personalizar o comportamento.

3. Construtores:
    - A classe Cachorro chama o construtor da classe base (Animal) usando : Animal(n) para inicializar o atributo nome herdado.

3) Considerações da aula

Nesta aula tivemos o primeiro contato com herança. Nas próximas aulas aprofundaremos o estudo para tornar este conceito bem sedimentado.

Bons estudos.