Polimorfismo

Aula 10 - Programação orientada a objetos

Professor Daniel Marques

INSTITUTO FEDERAL
Paraíba
Campus Campina Grande

Polimorfismo

Do Grego poly(muitas) + morpho(formas). Há dois aspectos de polimorfismo

Métodos de mesmo nome são definidos em várias classes, podendo assumir diferentes implementações em cada uma dessas

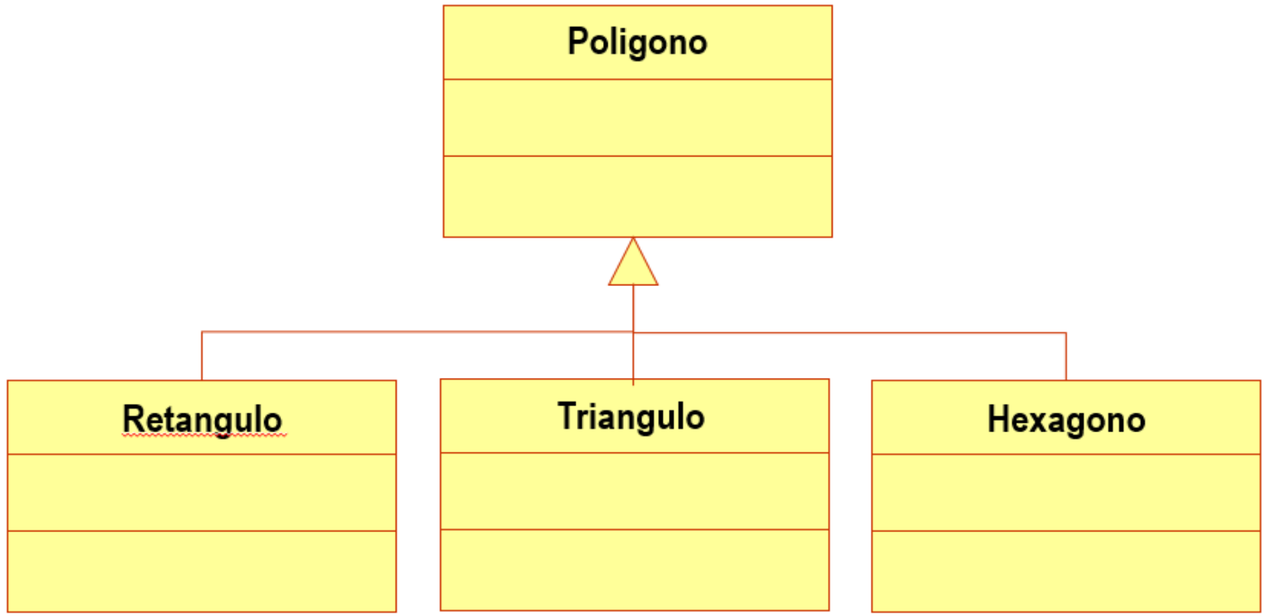
Propriedade pela qual uma variável pode apontar objetos de diferentes classes em momentos distintos

Estes aspectos são complementares, e trabalham juntos

Polimorfismo

Exemplo de polimorfismo em variáveis:

Uma variável do tipo Poligono pode assumir a forma de Poligono, Triangulo, Retangulo, etc.

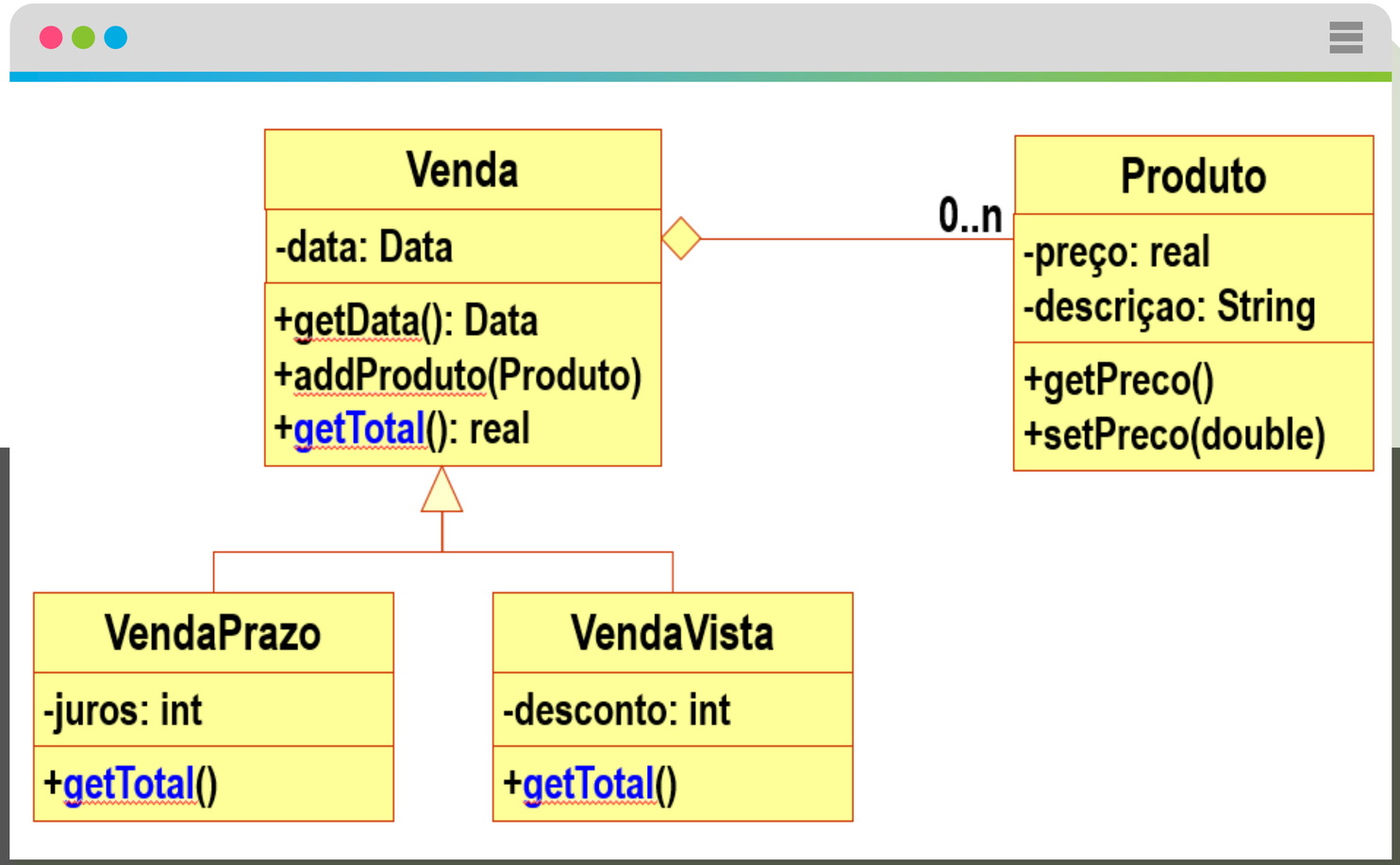


Polimorfismo

Exemplo de polimorfismo em variáveis:

Uma variável do tipo Poligono pode assumir a forma de Poligono, Triangulo, Retangulo, etc.

```
1  Poligono *p;
2
3  p = new Poligono();
4  ...
5  p = new Triangulo();
6  ...
7  p = new Retangulo();
8
```



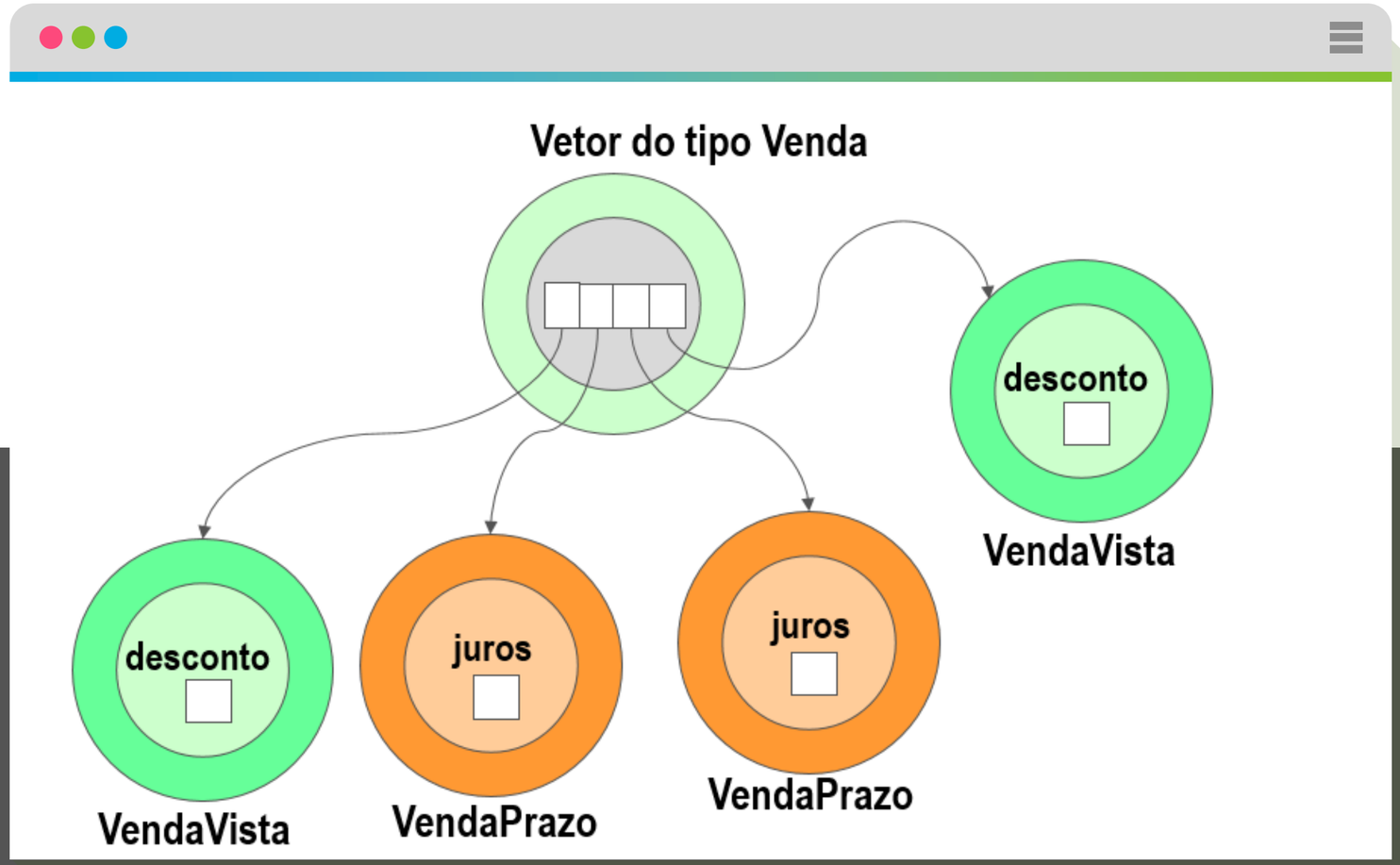
Polimorfismo em métodos

Considere a classe VendaPrazo

```
class VendaPrazo: public Venda{

    private:
        double juros;
    public:
        VendaPrazo(int q, string d, double j);
        double getTotal();
    };

    ...

    // trecho do arquivo vendaprazo.cpp
    double VendaPrazo::getTotal(){
        return Venda::getTotal()*(1+juros/100.0);
    }
```

Polimorfismo em métodos

Método getTotal() que está presente na classe mãe (Venda) e também na classe filha (VendaPrazo) reimplementado



Polimorfismo em métodos

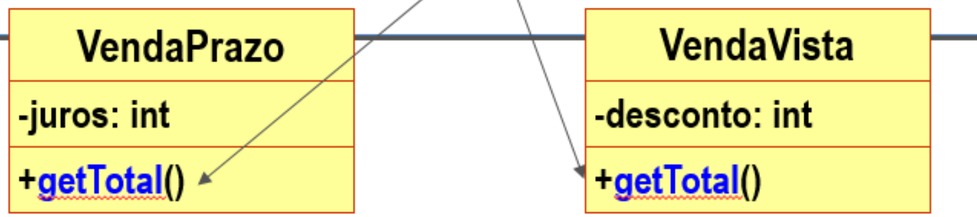
Ilustração de um vetor de Venda

Polimorfismo em métodos

Observe o trecho de código abaixo:

```
...
double total = 0.0;
for (int i=0; i<numVendas; i++) {
  total += vendas[i]->getTotal();
}
...
```

Qual getTotal() será chamado aqui?



Polimorfismo em métodos

Polimorfismo de sobreposição

- Métodos com mesma assinatura, em classes distintas
- Sobrescrita de métodos (override)

Polimorfismo de sobrecarga

- Métodos com assinaturas distintas, em uma mesma classe
- Sobrecarga de métodos (overload)

Ligação dinâmica

A ligação entre a mensagem vendas[i]->getTotal() e o método getTotal() de uma das classes só é estabelecida em tempo de execução, quando o tipo do objeto em vendas[i] for conhecido

- Venda é o tipo de cada referência do vetor de vendas
- VendaVista e VendaPrazo serão os tipos efetivos dos objetos referenciados no vetor

```
Venda *v = new VendaVista();
```

tipo estático *tipo dinâmico*

2

Classes abstratas

Classes e métodos abstratos

Classes abstratas

Uma classe abstrata é quase uma especificação

Classes abstratas estão em um nível intermediário entre especificação e código de programação

Estas classes permitem a definição das interfaces dos objetos sem entrar em detalhes de implementação

Classes abstratas

C++ permite que uma classe apresente métodos sem implementação

Métodos sem implementação são sempre virtuais

```
1  #include<iostream>
2  using namespace std;
3
4  class Shape {
5      public:
6          virtual int perimeter() = 0;
7          void width(int w) {
8            shape_width = w;
9          }
10         void height(int h) {
11           shape_height = h;
12         }
13
```

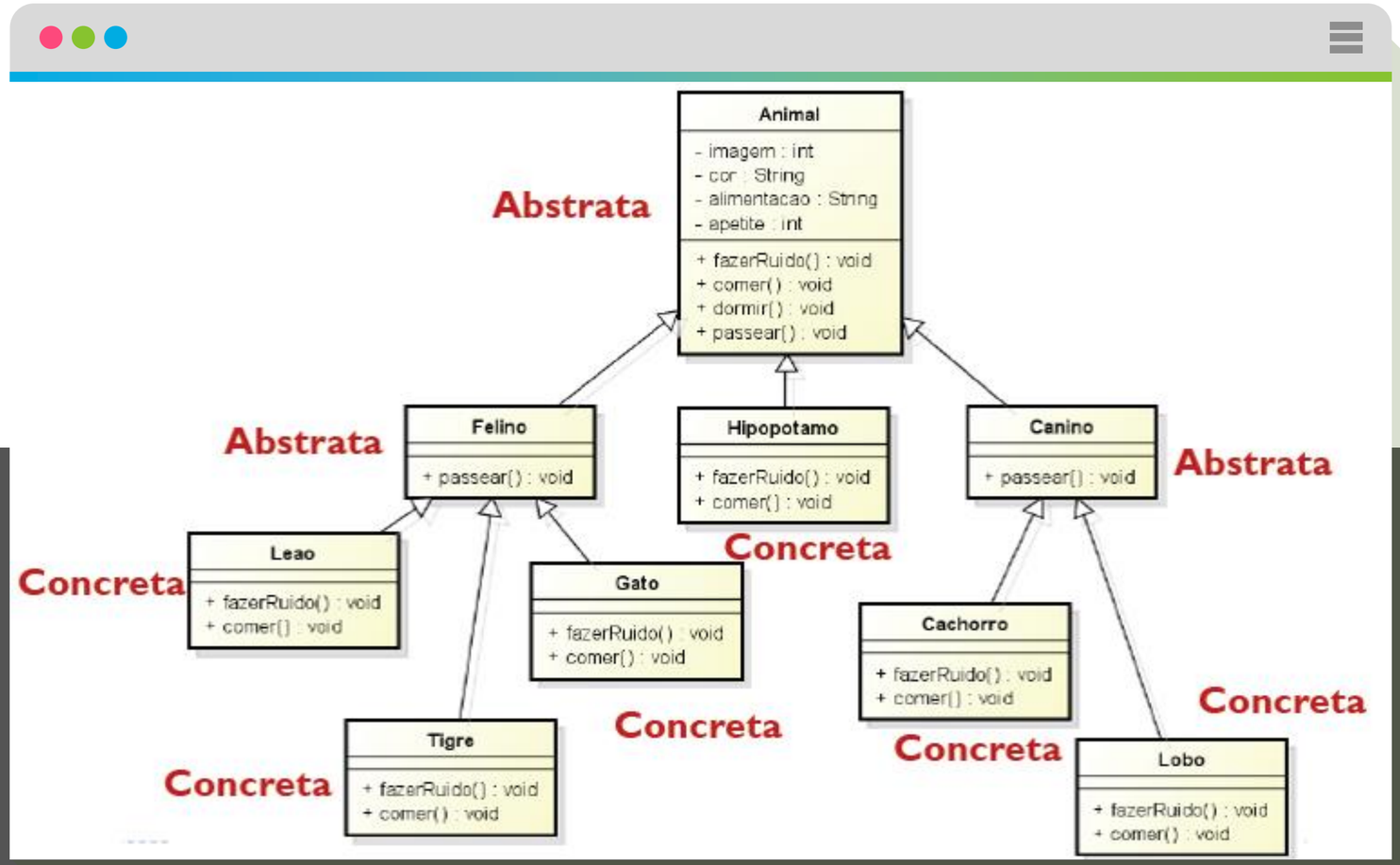
- Métodos definidos em uma classe sem implementação, apenas com a definição de sua assinatura, é denominado método virtual puro;
- Em C++, classes abstratas são aquelas que apresentam ao menos um método virtual puro

Classes abstratas

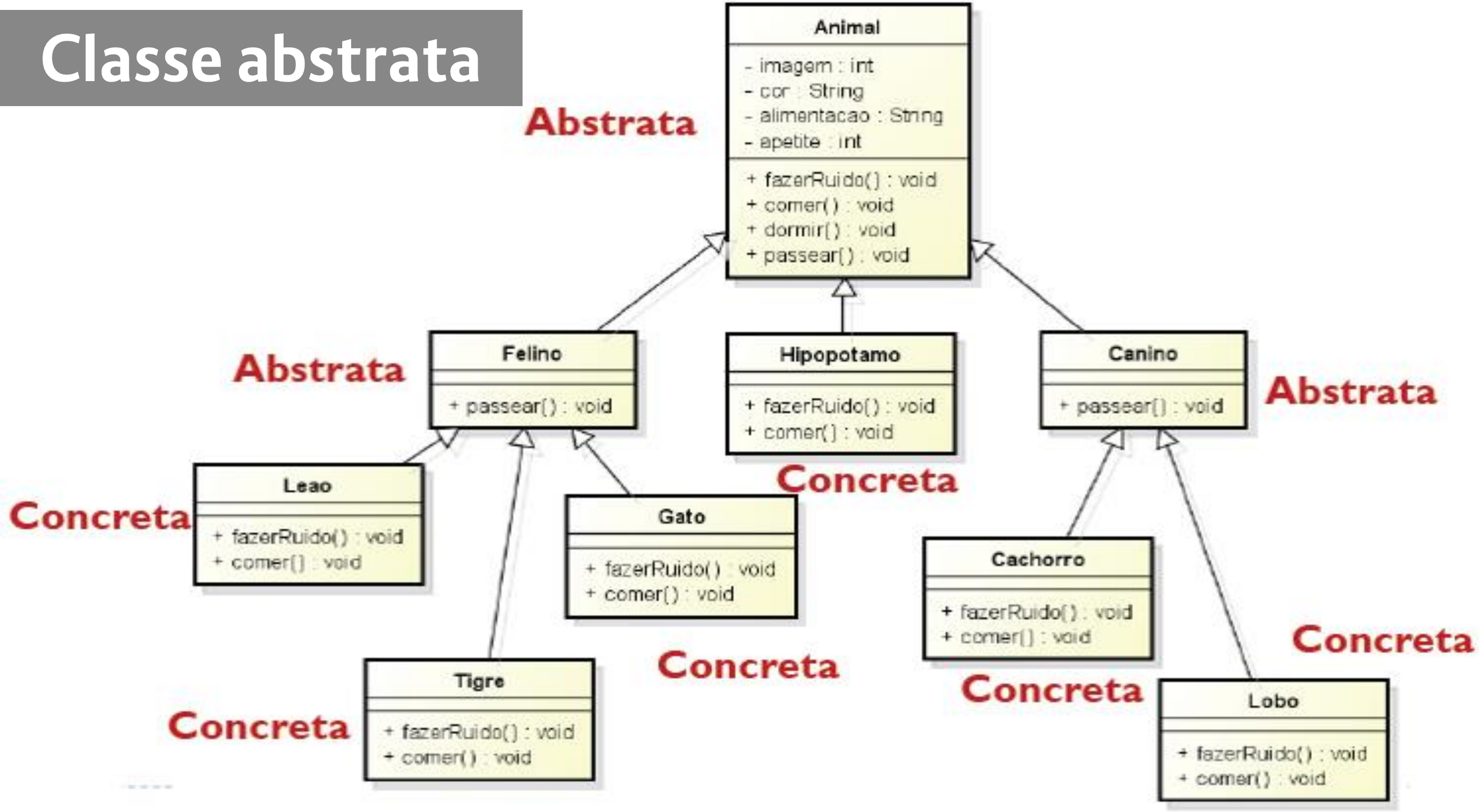
- Compilador impede que se crie uma instância desse tipo
- Pode ser usada para fins de polimorfismo
- Método deve ser obrigatoriamente implementado pela subclasse concreta

```
1  class Rectangle: public Shape {
2      public:
3          int perimeter() {
4            return (2 * (shape_width + shape_height));
5          }
6  };
7
8  class Square: public Shape {
9      public:
10         int perimeter() {
11           return (4 * shape_width);
12         }
13 };
```

- Método abstrato (em C++, é um método virtual puro)
- Todos os métodos abstratos DEVEM ser implementados nas subclasses concretas, mas não devem ser implementados na superclasse, embora possa.


Animal
- imagem : int
- cor : String
- alimentacao : String
- apetite : int
+ fazerRuido() : void
+ comer() : void
+ dormir() : void
+ passear() : void
Abstrata
Felino
+ passear() : void
Abstrata
Hipopotamo
+ fazerRuido() : void
+ comer() : void
Concreta
Canino
+ passear() : void
Abstrata
Leao
+ fazerRuido() : void
+ comer() : void
Concreta
Gato
+ fazerRuido() : void
+ comer() : void
Concreta
Cachorro
+ fazerRuido() : void
+ comer() : void
Concreta
Tigre
+ fazerRuido() : void
+ comer() : void
Concreta
Lobo
+ fazerRuido() : void
+ comer() : void
Concreta

Classe abstrata



Classes abstratas

Projete	Quando for projetar suas classes, você deve decidir quais classes serão concretas e quais serão abstratas.
Utilidade	Geralmente, classes abstratas só tem utilidade se forem estendidas.
Puros	Uma classe que todos os métodos são abstratos (virtuais puros). Ou seja, a subclasse deve implementar todos os métodos
Não implemente	Não deve haver implementação dos métodos.
Interface	Em outras linguagens (ex: Java), uma classe 100% abstrata é chamada também de Interface.

Alguma dúvida?

Não guardem dúvidas, perguntem

• • •

Referências

1. DA COSTA, Anderson Fabiano F. **Fundamentos de C++**. Instituto Federal da Paraíba. 2022.

2. Materiais de aula dos professores Guillermo Camara-Chavez, Tiago Maritan, Fred Guedes Pereira e Danielle Chaves.

3. DEITEL, **C++ Como Programar**, 5ª edição, Editora Prentice Hall, 2006

4. MANSSOUR, Isabel Harb. **Herança Múltipla**. Pontifícia Universidade Católica do Rio Grande do Sul. Disponível em: <https://www.inf.pucrs.br/~manssour/LinguagemC++/HerancaMultipla.pdf> . Acesso em: 17 Maio 2022.

5.