POLIMORFISMO

Programação Orientada a Objetos

educação a distância
UFSCar
virtual

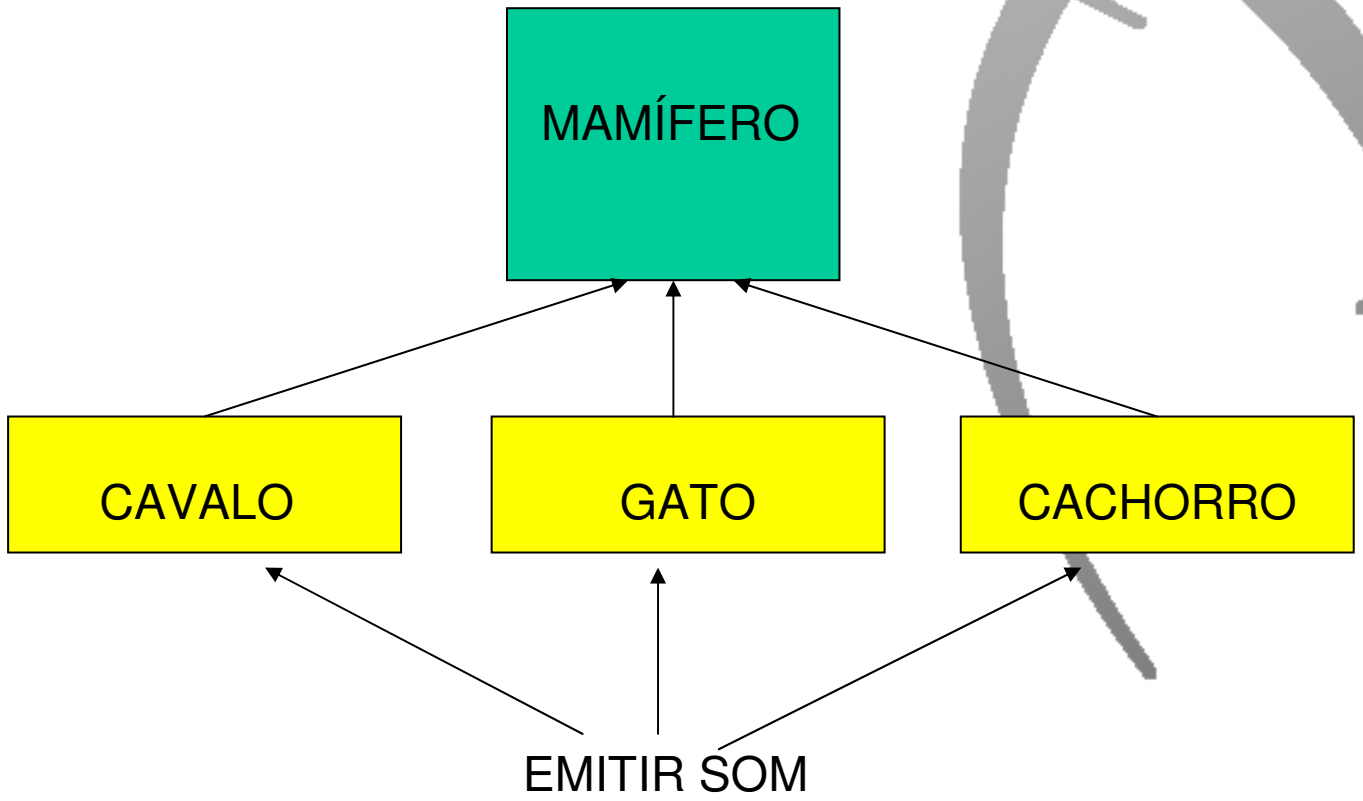
Programação Orientada a Objetos

Polimorfismo é a capacidade de assumir formas diferentes.

Em termos de programação, polimorfismo representa a capacidade de uma única variável chamar métodos diferentes, dependendo do que a variável contém.

Maneira de escrever programas de forma genérica que permite a manipulação de uma grande variedade de classes.

Programação Orientada a Objetos



Programação Orientada a Objetos

Aspectos importantes

→ Usa-se uma variável de um tipo único (normalmente do tipo da super-classe) para referenciar objetos variados do tipo das sub-classes.

→Envolve o uso automático do objeto armazenado na super-classe para selecionar um método de uma das sub-classes. O tipo do objeto armazenado não é conhecido até a execução do programa.

Programação Orientada a Objetos

```
class mamifero
{
public:
      mamifero( );
      virtual ~mamifero( );
      virtual void emitir_som( );
      void comer( );
      void andar( );
protected:
      int idade;
};
```

EXEMPLO EM C++

Programação Orientada a Objetos

```
class cachorro: public mamifero
{
public:
    cachorro( );
    ~cachorro( );
    void emitir_som( );
    void comer( );
protected:
    int raça;
};
```

EXEMPLO EM C++

Programação Orientada a Objetos

```
main( )
{
    mamifero *p_mamifero = new cachorro;
    p_mamifero→comer( );
    p_mamifero→emitir_som( );
    p_mamifero→andar( );
}
```

EXEMPLO EM C++

Programação Orientada a Objetos

- As construções apresentadas anterior-mente são válidas em C++, e fazem sentido na vida real, pois cavalo, gato e cachorro são mamíferos.
- Mais ainda, emitir_som é uma ação que todos eles fazem, mas cada um de forma diferente.

Programação Orientada a Objetos

Mamifero *p_mamifero = new cachorro;

Essa instrução cria um novo objeto cachorro e retorna um ponteiro para esse objeto, que é atribuído a um ponteiro para mamífero.

Como cachorro é um mamífero, OK !

Programação Orientada a Objetos

VOLTANDO AO EXEMPLO

P: A chamada **p_mamifero→comer( )** faz o que? Aciona o método de qual classe?

R: O método acionado será o da classe base, pois foi declarada como:

void comer( );

Programação Orientada a Objetos

P: E a chamada **p_mamifero→emitir_som( )** faz o que? Aciona o método de qual classe?

R: O método acionado será o da classe cachorro, pois foi declarada como:

virtual emitir_som( );

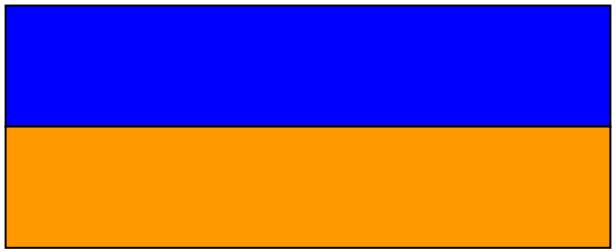
Programação Orientada a Objetos

P: E a chamada **p_mamifero→andar( )** faz o que? Aciona o método de qual classe?

R: O método acionado será o da classe base, pois não existe andar na classe derivada.

Programação Orientada a Objetos

Quando um objeto derivado, como um objeto de cachorro é criado, primeiro o construtor da classe base é chamado e, em seguida, o construtor da classe derivada.

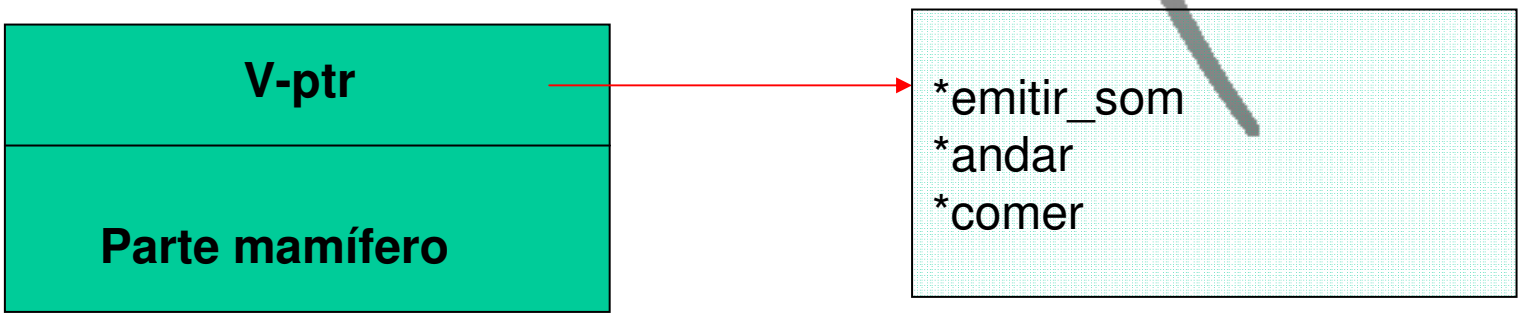


Parte mamífero

Parte cachorro

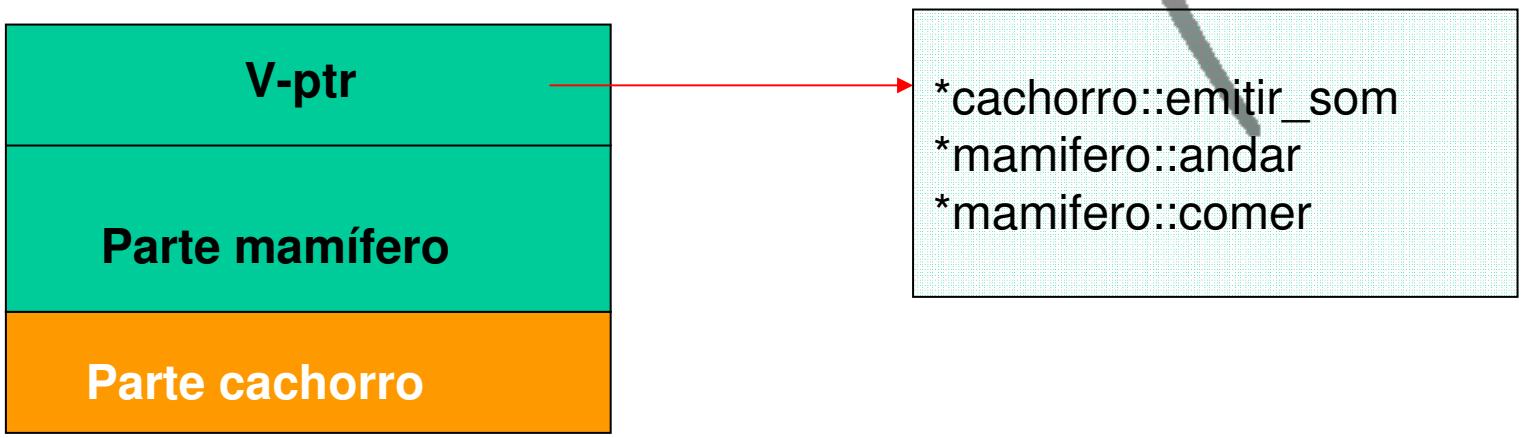
Programação Orientada a Objetos

Quando uma função virtual é criada em um objeto, esse deve monitorá-la. Muitos compiladores constroem uma tabela de funções virtuais chamada **tabela-v**.



Programação Orientada a Objetos

Quando o construtor de cachorro é chamado, o v-ptr é ajustado para apontar para as anulações de função virtual (se houver) no objeto cachorro.



Programação Orientada a Objetos

Em C++, a ***mágica*** da função virtual só opera em ponteiros e referências. A passagem de um objeto por valor não permitirá que as funções membros virtuais sejam invocadas.

Programação Orientada a Objetos

FIM