Template

Aula 12 - Programação orientada a objetos

Professor Daniel Marques

INSTITUTO FEDERAL
Paraíba
Campus Campina Grande

Introdução

Os templates (gabaritos) fornecem a base para existência da programação genérica

Possibilita desenvolver componentes de software reutilizável: funções, classes, etc.

Introdução

Exemplo:

Imagine uma mesma função somar dois números inteiros e somar duas strings

Mas qual a necessidade disso?

Os templates em C++ permitem criar uma família de funções e classes de templates para executar a mesma operação com diferentes tipos de dados.

Introdução

O mecanismo dos templates de C++ permite que um tipo ou valor seja um parâmetro na definição de uma classe ou função

Tipos de template

Templates de função

cada função “gerada” do template é chamada de especialização de template de função

Templates de classe

cada classe “gerada” do template é chamada de especialização de template de classe

Template em funções

Funcionam de forma similar a uma função (método);

Única diferença é que ela pode trabalhar com diferentes tipos de dados

```
1  template <class T>
2
3  T nome(T arg) {
4      //seu código da função
5  }
6
7
```

T é um argumento template que aceita diferentes tipos (int, chat, double, string)

- template <typename T>
- template <class T>
- template <class T1, class T2>

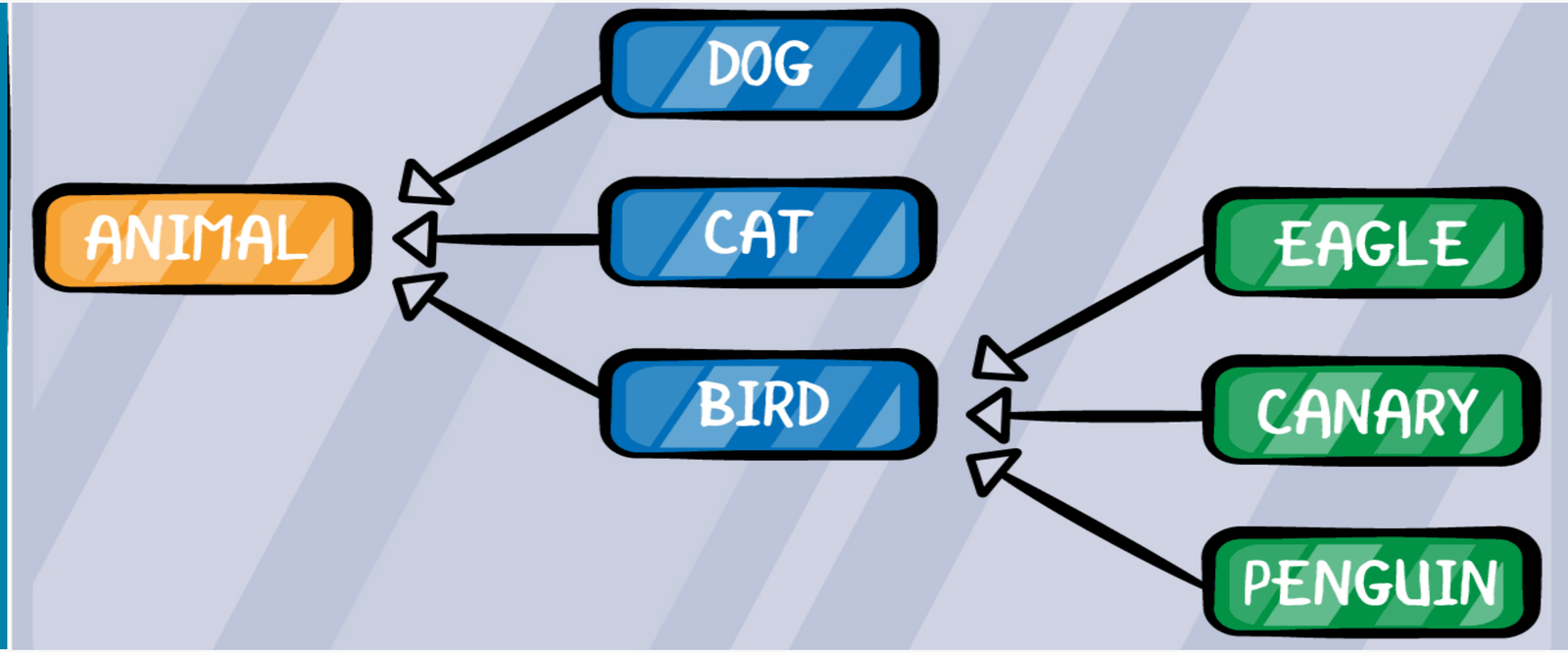
Precedendo o cabeçalho da função

```cpp
#include <iostream>
using namespace std;

template <class T>
T maior(T n1, T n2){
    if(n1 > n2){
        return n1;
    } else {
        return n2;
    }
}

int main(){
    int i1, i2;
    double n1, n2;
    char c1, c2;

    cin >> i1 >> i2;
    cout << maior(i1, i2) << endl;

    cin >> n1 >> n2;
    cout << maior(n1, n2) << endl;

    cin >> c1 >> c2;
    cout << maior(c1, c2) ;
    return 0;
}
```

Exemplo de template de função

Template de classe

Permitem criar classes genéricas, em outras palavras, operações genéricas para classes



- Útil quando o programador precisa de uma classe que compartilha código comum a várias classes, e muda-se apenas os tipos de dados
- Compartilham funções comuns de cadastro/inserção, remoção, consulta,...
- Ex: Listas: de Clientes, de Produtos, de Imóveis, etc

Template em classes

Sintaxe

```
1  template <class T>
2  class className {
3      ... ...
4
5      public:
6          T var;
7          T someOperation(T arg);
8          ... ...
9  };
10
```

```cpp
template <class T>
class Calculator {
    private:
        T num1, num2;

    public:
        Calculator(T n1, T n2){
            num1 = n1;
            num2 = n2;
        }

        T add() {
            return num1 + num2;
        }
        T subtract() {
            return num1 - num2;
        }
        T multiply() {
            return num1 * num2;
        }
        T divide() {
            return num1 / num2;
        }
};
```

Template de classe

Template em classe

Executando no main

```
int main() {
    Calculator<int> intCalc(2, 1);
    Calculator<float> floatCalc(2.4, 1.2);

    cout << "Resultado (int):" <<  intCalc.add() << endl;

    cout << "Resultado (float):" << floatCalc.add() << endl;

    return 0;
}
```

Template de classe

Qual a diferença entre polimorfismo e programação genérica?

- No polimorfismo nós temos o mesmo tipo se comportando de maneiras distintas.
- Na programação genérica nós temos tipos diferentes se comportando da mesma maneira.

Alguma dúvida?

Não guardem dúvidas, perguntem

• • •

Referências

1. DA COSTA, Anderson Fabiano F. **Template.** Instituto Federal da Paraíba. 2022.

2. DE OLIVEIRA, Victor André Pinho. **Template**. Instituto Federal da Paraíba. 2022.

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

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

5.