

Templates

PROFESSOR: EDUARDO HABIB BECHELANE MAIA
HABIB@CEFETMG.BR

Introdução

Introdução aos Templates em C++

- Recurso poderoso em C++ que permite aos programadores criar funções e classes genéricas.
- O propósito de um template é permitir que o mesmo código funcione com diferentes tipos de dados sem repetição de código.
- Aumentam a reusabilidade e a flexibilidade do código, permitindo que os programadores escrevam códigos mais genéricos que se adapta a diferentes situações.

Exemplo 1

```
template <typename T>
T myMax(T x, T y) {
    return (x > y) ? x : y;
}
```

T myMax(T x, T y)

- define uma função chamada myMax
- aceita dois parâmetros de qualquer tipo T
- retorna um valor do mesmo tipo T.

template <typename T> é a declaração do template,

- T é um parâmetro de tipo
- Será substituído pelo tipo real quando a função for usada.

Exemplo: <https://onlinegdb.com/2bc7T0l6R>

Vantagens

Reutilização de código

- Chamada da função myMax com diferentes tipos de argumentos
 - leva o compilador a instanciar uma nova versão da função para aquele tipo específico.
- feito em tempo de compilação.
- permitem que os programadores escrevam código genérico
- pode ser reutilizado com diferentes tipos de dados.
- reduz a necessidade de sobrecarregar funções para cada novo tipo de dado.
- Exemplo:
 - função de ordenação genérica que pode ser aplicada tanto a inteiros quanto a strings, ilustrando a reutilização do código.

Exemplo: <https://onlinegdb.com/7LKSqeVv8>

Vantagens

- Templates permitem :
 - Criação de bibliotecas como a Standard Template Library (STL)
 - oferece uma variedade de contêineres e algoritmos genéricos. (Vector, queue, etc)
 - Erros são capturados durante a compilação, não em tempo de execução
 - Pode economizar muito tempo de depuração e melhorar a qualidade do código.
 - aumentam a flexibilidade permitindo que uma única função ou classe trabalhe com qualquer tipo de dado.
 - Melhora a segurança de tipo
 - Código é verificado em tempo de compilação para cada tipo específico com o qual é usado, reduzindo os erros em tempo de execução.

Template de Classe

- Sintaxe e exemplo de um template de classe.
 - semelhante à de um template de função
 - Começa com a palavra-chave template, seguida por uma parâmetros de template entre < >.
 - Um template de classe é usado para definir uma classe genérica que pode operar com qualquer tipo de dado.
- Diferenças entre templates de classe e de função.

Explicação

- No exemplo abaixo, Box é um template de classe que encapsula um valor de um tipo genérico T.
- Pode ser utilizado para criar objetos que armazenam diferentes tipos de dados.
- Alguns desenvolvedores preferem usar uma extensão diferente, como .tpp ou .ipp, para o arquivo de implementação de template,
 - Deixa claro que este arquivo é diferente de um arquivo .cpp e .hpp tradicional.
 - Particularmente útil em ambientes onde o processo de compilação é automatizado e assume que todos os arquivos .cpp são compiláveis separadamente.

Exemplo com hpp: <https://onlinegdb.com/uLg16Nzll>

Separando hpp e tpp: <https://onlinegdb.com/mXOZ4Jodsp>

Exemplo de lista

Código de exemplo: <https://onlinegdb.com/D03Fi7L2E>

Limitações e Considerações

Templates em C++ são ferramentas poderosas, mas:

- vêm com suas próprias limitações e desafios
 - depuração
 - compilação
- Mensagens de erro podem ser longas e complexas
- Aumento do tempo de compilação
- Programadores menos experientes podem cair na armadilha de complicar desnecessariamente seu código
- Aumento do tamanho do binário

Conclusão e Recursos Adicionais

- Referência:

- DEITEL, H. M.; DEITEL, P. J. **C++: como programar**. 5. ed. São Paulo: Pearson, 2006. *E-book*. Disponível em: <https://plataforma.bvirtual.com.br>. Acesso em: 26 nov. 2023.
- SAVITCH, Walter. **C++ Absoluto**. 1. ed. São Paulo: Pearson, 2003. *E-book*. Disponível em: <https://plataforma.bvirtual.com.br>. Acesso em: 26 nov. 2023.