UNIVERSIDADE ESTADUAL DA PARAÍBA
CENTRO DE CIÊNCIA E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO

João Victor Gomes Barbosa

C++: Uma abordagem técnica

Projeto de Paradigmas de Programação

Campina Grande-PB
2022

João Victor Gomes Barbosa

C++: Uma abordagem técnica

Projeto de Paradigmas de Programação apresentado ao componente curricular, como parte dos requisitos necessários à obtenção da nota da primeira unidade.

Professor(a): Daniel Marques Vasconcelos Guimarães
Disciplina: Paradigmas de Programação
Turma: 2022.2

Campina Grande-PB
2022

Resumo

A linguagem de programação C++, em suma, é muito utilizada para o desenvolvimento de sistemas de alta performance, jogos, interfaces gráficas e no ensino sobre orientação a objetos na programação.

Considerada como uma extensão da linguagem C por utilizar comandos e estruturas dessa linguagem, essa linguagem proporciona maior desempenho, fácil aprendizado, é open source e continua em constante melhorias em suas novas versões.

Palavras-chave: C++, open source.

Abstract

The C++ programming language, in short, is widely used for the development of high-performance systems, games, graphical interfaces and in teaching object-oriented programming.

Considered as an extension of the C language because it uses commands and structures from that language, this language provides greater performance, is easy to learn,is open source and continues to be constantly improved in its new versions.

Keywords: C++, open source.

Sumário

1 Introdução

A linguagem C++ foi criada em 1979 por Bjarne Stroustrup, como uma extensão da linguagem de programação C, na Bell Labs. A intenção original era permitir a programação estruturada de sistemas, como os sistemas operacionais Unix. Com o tempo, a linguagem evoluiu para suportar programação orientada a objetos e programação genérica. Hoje, a linguagem é amplamente utilizada em muitos contextos diferentes, incluindo aplicativos, sistemas operacionais, jogos e dispositivos embarcados.

1.1 Breve história

Stroustrup começou a chamar a nova linguagem de “C com classes”. Em 1983, a linguagem foi renomeada para C++, e o nome “++” foi escolhido pelo fato de ser uma versão incrementada do C. Em 1985, o C++ foi adotado pelo comitê ANSI de padrões de linguagem de programação como parte do processo de padronização do C. O C++ se tornou uma das linguagens mais populares para programação orientada a objetos. Em uma pesquisa realizada em 2007, o C++ foi classificado como o terceiro idioma de programação mais popular, atrás apenas do C e Java.

Embora tenha sido criado há mais de 40 anos, o C++ ainda é amplamente utilizado pelos programadores. Empresas de tecnologia de ponta, como Google, Microsoft e Amazon, usam C++ para criar seus aplicativos. Além disso, o C++ ainda é usado para criar jogos. Jogos em 3D, como o Tomb Raider e o Halo, foram criados usando C++. O C++ também é usado em aplicativos de ciência de dados, como o Matlab, e em aplicativos de machine learning, como o TensorFlow.

1.2 Etimologia e filosofia da linguagem

- Stroustrup pretendia chamar o C ++, “C com classes”, mas descobriu que o nome já havia sido registrado por outra pessoa. Em seguida, ele optou pelo nome C prelusory
  ++, que significa “C pré-processado com classes”. No entanto, este nome não eraamigável para os usuários, e assim ele acabou ficando com o nome C ++.

- Sua filosofia remete que, ao longo da vida do C++, seu desenvolvimento e evolução foram guiados por um conjunto de princípios: o C++11 incluiu muitas adições à linguagem central e à biblioteca padrão. Os procedimentos de votação do Draft International Standard foram concluídos em meados de agosto de 2014. Após o C++14, uma grande revisão C++17, informalmente conhecida como C++1z, foi concluída pelo comitê ISO C++ em meados de julho de 2017 e foi aprovada e publicado em dezembro de 2017.

2 Paradigmas da linguagem de programação C++

C++ é uma linguagem de programação multiparadigma, originalmente desenvolvida por Bjarne Stroustrup, no Bell Labs, como uma extensão do linguajar de programação C.

A linguagem foi criada com o objetivo de proporcionar um modelo de programação abrangente que pudesse ser usado para o desenvolvimento de software de sistemas e aplicações. O C++ é uma linguagem de programação compilada, de tipagem estática e forte, que suporta a programação orientada a objetos, a programação genérica e a metaprogramação.

Desde a sua criação, a linguagem evoluiu de forma a incorporar novos paradigmas de programação e melhorar a sua expressividade. Atualmente, o C++ é uma das linguagens de programação mais populares, sendo utilizado em diversos domínios, tais como o desenvolvimento de software embarcado, a programação de jogos, a computação científica e a inteligência artificial.

Vamos analisar os principais paradigmas de programação suportados pelo C++ e exemplificar como eles podem ser aplicados na linguagem. Os principais paradigmas de programação suportados pelo C++ são:

1) **Programação Orientada a Objetos -** A programação orientada a objetos é um paradigma de programação que se baseia na manipulação de objetos. Esses objetos são entidades que encapsulam atributos e comportamentos. Os atributos representam as características do objeto e os comportamentos representam as ações que o objeto pode realizar.A programação orientada a objetos é uma forma de modularizar o código, o que torna mais fácil a manutenção e a evolução do software. Além disso, essa abordagem facilita a reutilização de código, uma vez que os objetos podem ser reaproveitados em outros contextos. Para implementar a programação orientada a objetos no C++, é necessário utilizar as classes, que são as estruturas de dados que representam os objetos. As classes definem os atributos e os comportamentos dos objetos, que podem ser acessados através dos métodos.

2) **Programação Genérica -** A programação genérica é um paradigma de programação que se baseia na criação de funções e classes genéricas, ou seja, que podem ser aplicadas a diversos tipos de dados. Esse paradigma permite a criação de funções e classes mais flexíveis e reutilizáveis.Por exemplo, ao criar uma função para imprimir os elementos de um vetor, é possível utilizar uma função genérica para imprimir qualquer tipo de vetor, basta passar o tipo de dado como parâmetro. No C++, a programação genérica é implementada através dos templates, que são funções e classes que podem ser aplicadas a diversos tipos de dados.

3) **Metaprogramação -** A metaprogramação é um paradigma de programação que se baseia na criação de código que gera ou manipula outro código. Esse paradigma é muito utilizado em bibliotecas e frameworks, uma vez que permite a criação de código mais flexível e reutilizável.No C++, a metaprogramação é implementada através dos macros, que são funções que podem gerar código C++. Além dos macros, o C++ também oferece outras ferramentas para a metaprogramação, taiscomo as templates metaprogramáveis.

Esses são os principais paradigmas de programação suportados pelo C++. Como podemos ver, a linguagem é muito flexível e permite a criação de código para diversos tipos de aplicações.

2.1 Funções básicas da linguagem de programação C++

A linguagem C++ oferece uma grande quantidade de recursos que permitem ao programador aumentar a produtividade e reduzir o tempo de desenvolvimento do software.Alguns dos recursos mais importantes da linguagem C++ são:

Classes: as classes permitem ao programador criar tipos de dados personalizados e estruturados. As classes também proporcionam um mecanismo de herança, que permite ao programador reutilizar o código existente e evitar a duplicação de trabalho.

Templates: os templates permitem ao programador criar funções e classes genéricas que podem ser usadas com qualquer tipo de dados. Os templates são extremamente úteis para a criação de bibliotecas de software genéricas.

Exceções: as exceções permitem ao programador tratar erros e outros problemas de forma mais eficiente. As exceções também proporcionam um mecanismo de controle de fluxo mais flexível.

Strings: as strings são uma das estruturas de dados mais importantes e úteis da linguagem C++. As strings são usadas para representar texto e fornecem uma variedade de funções e métodos para manipular strings.

Algoritmos: a linguagem C++ fornece uma grande variedade de algoritmos pré-definidos que podem ser usados para resolver problemas comuns de programação. Os algoritmos são implementados como funções e classes genéricas e podem ser facilmente estendidos pelo programador.

Bibliotecas: a linguagem C++ fornece uma grande quantidade de bibliotecas de software pré-compiladas que podem ser usadas pelos programadores. As bibliotecas são um conjunto de funções e classes que podem ser usadas para desenvolver software de forma mais eficiente.

2.1.1 Entrada e saída de dados

A entrada e saída de dados em C++ é feita através das funções de entrada e saída de streams. A biblioteca padrão do C++, <iostream>, fornece três streams de entrada e saída: <iostream.h> (input/output stream header), <fstream.h> (file input/output stream header) e <sstream.h> (string input/output stream header).

As funções de entrada/saída de stream usam o operador de inserção (<<) para a saída e o operador de extração (>>) para a entrada. O operador de inserção é sobrecarregado para todos os tipos de dados primitivos, como int, char e double, assim como para tipos de dados definidos pelo usuário, como classes e structs. O operador de extração é sobrecarregado apenas para tipos de dados primitivos.

Para ler ou escrever em um arquivo, primeiro é preciso criar um objeto de stream de arquivo. Isso é feito usando as palavras-chave ofstream, ifstream ou fstream, que são usadas para, respectivamente, criar um stream de saída, um stream de entrada ou um stream de entrada/saída. O objeto de stream de arquivo é criado usando o operador de inserção, passando o nome do arquivo como parâmetro.

Uma vez que o objeto de stream de arquivo é criado, a saída para o arquivo é feita usando o operador de inserção <<, da mesma forma que a saída para a tela. A entrada a partir do arquivo é feita usando o operador de extração >>.

Para ler ou escrever em uma string, primeiro é preciso criar um objeto de stream de string. Isso é feito usando a palavra-chave stringstream, que cria um stream de entrada/saída. O objeto de stream de string é criado usando o operador de inserção, passando a string como parâmetro.

Uma vez que o objeto de stream de string é criado, a saída para a string é feita usando o operador de inserção <<, da mesma forma que a saída para a tela. A entrada a partir da string é feita usando o operador de extração >>.

Um exemplo de entrada e saída:

```
#include
#include

using namespace std;

int main()
{
    string nome;
    int idade;

    cout << "Digite o seu nome: ";
    cin >> nome;

    cout << "Digite a sua idade: ";
    cin >> idade;
```

```
    cout << "Nome: " << nome << endl;
    cout << "Idade: " << idade << endl;
}

/*

Digite o seu nome: Maria
Digite a sua idade: 20
Nome: Maria
Idade: 20

*/
```

2.1.2. Estrutura de condição (IF e Else)

As estruturas de condição permitem que um programa tome decisões, executando um bloco de código apenas quando uma determinada condição é satisfeita. A linguagem C++ fornece as estruturas de condição if e else para isso. A estrutura if verifica se uma determinada condição é verdadeira e, se for, executa um bloco de código. Se a condição não for verdadeira, a estrutura if não executa o bloco de código. A estrutura else fornece um bloco de código que será executado se a condição não for verdadeira.

A sintaxe da estrutura de condição if é a seguinte:

```
if (condição)
{
  // Bloco de código a ser executado se a condição for verdadeira
}
```

A sintaxe da estrutura de condição else é a seguinte:

```
if (condição)
{
  // Bloco de código a ser executado se a condição for verdadeira
}
else
{
  // Bloco de código a ser executado se a condição não for verdadeira
```

```
}
```

Você também pode usar a estrutura else if para testar várias condições. A sintaxe da estrutura else if é a seguinte:

```
if (condição1)

{

  // Bloco de código a ser executado se a condição1 for verdadeira

}

else if (condição2)

{

  // Bloco de código a ser executado se a condição2 for verdadeira

}

else

{

  // Bloco de código a ser executado se nenhuma das condições for verdadeira

}
```

2.1.3. Funções

As funções podem ou não retornar valores. Se uma função retornar um valor, ela deve especificar o tipo de valor que está sendo retornado. Se uma função não retornar nenhum valor, ela deve especificar o tipo void.

Exemplo:

```
int max(int num1, int num2)
{
  // Local variable declaration
  int result;

  if (num1 > num2)
    result = num1;
  else
    result = num2;

  return result;
}
```