ESTRUTURAS DE DADOS

Introdução ao C++

Roteiro

- **Objetivo da Disciplina**
- **Linguagem da Disciplina**
- **Executando Programas**
- **Sintaxe Básica**

Objetivo da Disciplina

- **Familiarizar os alunos com a modelagem e a implementação de diferentes estruturas de dados, bem como os algoritmos para gerenciá-las.**
- **Para cada estrutura de dados vista na disciplina, discutiremos a lógica, modelaremos em C++ e depois implementaremos as operações principais:**
  - **Construção**
  - **Consulta**
  - **Inserção**
  - **Remoção**

Linguagem da Disciplina

- A linguagem utilizada na disciplina é o C++.
- A linguagem C++ é baseada na sintaxe da linguagem C. A primeira versão oficial surgiu em 1985.
- A linguagem é compilada, imperativa e de uso geral, com suporte a orientação a objetos.
- Note que a disciplina trata de modelar e gerenciar estruturas de dados em memória principal. Não se trata de um curso de C++.

Você tem estudado a linguagem Python no seu curso. Então, por que C++?

- **Porque a linguagem deixa a cargo do programador as operações para gerenciamento das estruturas de dados.**
- **A linguagem permite a manipulação de ponteiros de maneira explícita.**
- **A linguagem é orientada a objetos, permitindo a separação entre a visão lógica das estruturas e a implementação.**

A linguagem C++ difere do Python em diversos aspectos. Apenas para citar alguns:

1. **Linguagem Compilada.**
2. **Tipagem Estática.**
3. **Manipulação Explícita da Memória.**

Isso confere algumas diferenças:

1. **Python é mais fácil e rápido de programar.**
2. **C++ gera programas mais rápidos.**

Inicialmente, a comparação com Python nos ajudará a aprender a nova linguagem.

Executando Programas

Para executar os códigos, você precisará de um compilador C++:

1. **O compilador g++ é o mais conhecido. Você provavelmente já tem instalado se estiver usando Linux.**
2. **Em MacOS, instale o XCode para obter as ferramentas Clang.**
3. **No caso de Windows, recomenda-se o MinGW-x64. Não esqueça de adicionar o local de instalação como variável de ambiente.**

Você precisará escrever códigos:

- **Emacs**
- **Vim**
- **Visual Studio Code**
- **Eclipse**
- **Visual Studio**
- **CodeBlocks**
- **CLion**
- **Bloco de Notas**

Não assumiremos nenhuma dessas IDEs. O mínimo que você precisará é de um editor de arquivos de texto e de uma linha de comando.

Por exemplo, digite o código a seguir em um arquivo de extensão .cpp usando qualquer editor de texto.

```cpp
#include <iostream>

int main() {
  std::cout << "Hello World!" << "\n";
  return 0;
}
```

Para executar, fazemos:

```
codigos$ g++ hello_world.cpp -o hello_world
codigos$ ./hello_world
Hello World!
```

Sintaxe Básica

Veremos agora algumas construções que você irá se habituar no decorrer da disciplina. Estamos interessados nas seguintes classes de operações:

- **Operações Matemáticas**
- **Comandos Condicionais**
- **Comandos de Repetição**
- **Declarações de Funções**

Operações Matemáticas

```cpp
int number1;
int number2;

std::cout << "Digite o primeiro número: ";
std::cin >> number1;
std::cout << "Digite o segundo número: ";
std::cin >> number2;

int   sum  = number1 + number2;
int   sub  = number1 - number2;
int   mul  = number1 * number2;
int   div  = number1 / number2;
float fdiv = (float)number1 / (float)number2;
int   res  = number1 % number2;
```

Operações Condicionais

```
int number1;
int number2;

cout << "Digite o primeiro número: ";
std::cin >> number1;
cout << "Digite o segundo número: ";
std::cin >> number2;

if (number1 == number2)
  cout << number1 << " ==  " <<  number2 << std::endl;
if (number1 != number2)
  cout << number1 << " !=  " <<  number2 << std::endl;
if (number1 < number2)
  cout << number1 << " <  " <<  number2 << std::endl;
if (number1 > number2)
  cout << number1 << " >  " <<  number2 << std::endl;
```

Comandos de Repetição

```
int number1;
int counter = 0;
int amount  = 0;

while (counter < 10) {

  cout << "Digite um número (" << counter << ")" << endl;
  std::cin >> number1;

  if (number1 < 5) {
    amount++;
  }
  counter++;
}
```

Comandos de Repetição

```
#include <iostream>

using namespace std;

int main() {
  int total;

  for (int number = 2; number <= 20; number += 2)
    total += number;

  cout << "A soma da série é" << total << endl;

  return 0;
}
```

Comandos de Repetição

```
int count;
int number;
int sum;
do {
  cout << "Insira um novo número: " << endl;
  cin >> number;

  sum += number;
  count++;
}  while (number != 0);

float average = (float)sum/(float)(count-1);
cout << "A média da série é: " << average << endl;
```

Declarações de Funções

```
int sum(int number1, int number2){
  return number1 + number2;
}

int sub(int number1, int number2){
  return number1 - number2;
}

int mul(int number1, int number2){
  return number1 * number2;
}

int idiv(int number1, int number2){
  return number1 / number2;
}
```

Os parâmetros das funções podem ser passados por valor ou por referência:

Valor: a função recebe uma cópia da variável fornecida quando invocada. Alterações dentro da função não afetarão os valores originais.

Referência: a função recebe uma referência às variáveis, e não uma cópia. Alterações realizadas dentro da função irão alterar os valores contidos nas variáveis originais.

Parâmetro por Valor

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

void troca_por_valor(int a, int b){
        int temp;
        temp=a;
        a=b;
        b=temp;
}
int main(){
        int a=2, b=3;
        cout<<"Antes: a = "<<a<<" b = " << b<<endl;
        troca_por_valor(a,b);
        cout<<"Depois: a = "<<a<<" b = "<< b << endl;
        return 0;
}
```

Parâmetro por Valor

A execução do código anterior retornaria:

```
codigos$ g++ ex8_parametro_por_valor.cpp -o valor
codigos$ ./valor
Antes: a = 2 b = 3
Depois: a = 2 b = 3
```

Parâmetro por Referência

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

void troca_por_referencia(int &a, int &b){
        int temp;
        temp=a;
        a=b;
        b=temp;
}
int main(){
        int a=2, b=3;
        cout<<"Antes: a = "<<a<<" b = " << b<<endl;
        troca_por_referencia(a,b);
        cout<<"Depois: a = "<<a<<" b = "<< b << endl;
        return 0;
}
```

Parâmetro por Referência

A execução do código anterior retornaria:

```
codigos$ g++ ex9_parametro_por_referencia.cpp -o referencia
codigos$ ./referencia
Antes: a = 2 b = 3
Depois: a = 3 b = 2
```

ESTRUTURAS DE DADOS

Introdução ao C++