Introdução

ECT3201 – Linguagem de Programação

Prof. Éverton Santi

Cronograma e Avaliações

- SIGAA

Linguagens de Programação

Alto vs Baixo Nível

- **Linguagens de alto nível**: Próximas à linguagem humana, fáceis de entender (ex: C++, Python)
- **Linguagens de baixo nível**: Próximas ao hardware, mais difíceis (ex: Assembly)

Compiladas vs Interpretadas

- **Compiladas**: Código traduzido para executável antes da execução (ex: C++, Java)
- **Interpretadas**: Código executado linha por linha por um interpretador (ex: Python, JavaScript)

Processo de Compilação

Compilar significa transformar o arquivo `.cpp` em um programa que pode ser executado.

Etapas principais:

1. **Pré-processamento**: resolve diretivas como `#include` e `#define`
2. **Compilação**: verifica o código e gera arquivos intermediários
3. **Ligação (linking)**: junta tudo (incluindo bibliotecas) e produz o executável

Na prática, os compiladores modernos executam esse fluxo automaticamente.

Hello World em C++

```cpp
#include <iostream>

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

Este programa imprime "Hello, World!" na tela.

Programa de Computador

De modo geral, qual é o objetivo de criarmos um programa?

Dados → Processamento → Informação

Um programa:

1. recebe **dados**
2. executa **operações**
3. produz **informação**

Variáveis

Uma **variável** representa um espaço na memória.

Ela possui:

- um identificador
- um tipo
- um valor

Exemplo conceitual:

Endereço	Nome	Valor
0101	a	'd'
0110	resultado	12.45
1100	valor	864

Informações Importantes

Quando declaramos variáveis precisamos definir:

- **tipo de dado**
- **nome da variável**

O compilador decide:

- **endereço na memória**

Tipos Básicos em C++

Principais tipos primitivos:

```
char
int
float
double
bool
void
```

Tamanho dos Tipos

Tipo	Tamanho (bytes)	Mínimo	Máximo
`char`	1	-128	127
`bool`	1	false	true
`int`	4	-2.147.483.648	2.147.483.647
`float`	4	~-3,4 × 10^38	~3,4 × 10^38
`double`	8	~-1,7 × 10^308	~1,7 × 10^308

Tipos diferentes ocupam **quantidades diferentes de memória**.

Modificadores de Tipo

Modificadores alteram o alcance dos tipos.

Principais:

```
signed
unsigned
short
long
```

Exemplo:

```
unsigned int contador;
long int populacao;
```

Tipos Modificáveis e Limites

Valores **típicos** em compiladores modernos (arquitetura 64-bit):

Tipo	Tamanho (bytes)	Mínimo	Máximo
`signed char`	1	-128	127
`unsigned char`	1	0	255
`short int`	2	-32.768	32.767
`unsigned short int`	2	0	65.535
`int`	4	-2.147.483.648	2.147.483.647
`unsigned int`	4	0	4.294.967.295

Tipos Modificáveis e Limites (continuação)

Tipo	Tamanho (bytes)	Mínimo	Máximo
`long int`	4	-2.147.483.648	2.147.483.647
`unsigned long int`	4	0	4.294.967.295
`long long int`	8	-9.223.372.036.854.775.808	9.223.372.036.854.775.807
`unsigned long long int`	8	0	18.446.744.073.709.551.615
`long double`	8	~-1,7 × 10^308	~1,7 × 10^308

Identificadores

Regras para nomes de variáveis:

✔ podem conter letras, números e `_`
✔ devem começar com letra ou `_`
❌ não podem ser palavras reservadas

Exemplos válidos:

```
soma
peso_bruto
p2p
SOMA
```

Exemplos inválidos:

```
5altura
desvio-padrão
```

Declaração de Variáveis

Sintaxe geral:

```
tipo nome;
```

Exemplos:

```
double lucro;
int i, j;
char a, b, c;
```

Inicialização

Podemos declarar e atribuir ao mesmo tempo.

```
int idade = 35;
float peso = 82.4;
```

Isso é equivalente a:

```
int idade;
idade = 35;
```

Entrada/Saída de Dados

Escreva um programa para ler a idade de uma pessoa. Exiba o dado lidos na tela.

Exemplo

Distanciamento de Segurança

Escreva um programa para calcular a distância segura entre veículos:

$$d = \left(\frac{velocidade}{10} \right)^2$$

em que:

- velocidade em **km/h**
- distância **em metros**

Controlando Casas Decimais

Use a biblioteca `iomanip` para customizar a exibição de casas decimais do exemplo anterior:

```
#include <iostream>
#include <iomanip>

using namespace std;

int main() {
    cout << fixed << setprecision(2);
    return 0;
}
```

Exemplo Completo

Enunciado:

Crie um programa que:

1. Leia o **peso** (kg) e a **altura** (m) de uma pessoa
2. Calcule o IMC usando:

$$IMC = \frac{peso}{altura^2}$$

3. Mostre o resultado com **2 casas decimais**

Constantes

Constantes são valores que **não podem ser alterados**.

```
const int A = 10;
const float GRAVIDADE = 9.8;
const double T = 1e-10;
```

Constantes Simbólicas

Podemos usar o pré-processador:

```
#define GRAVIDADE 9.80665
```

Exemplo:

```
#include <iostream>

#define GRAVIDADE 9.80665

using namespace std;

int main() {
    cout << "Gravidade = " << GRAVIDADE << endl;
    return 0;
}
```

Casting (Conversão Forçada)

Quando queremos converter de forma explícita entre tipos, usamos **casting**.

```
double media = 8.75;
int nota_inteira = (int)(media);
```

Nesse caso, a parte decimal é descartada e `nota_inteira` recebe `8`.

Outro exemplo:

```
int a = 5, b = 2;
double resultado = (double)(a) / b; // 2.5
```

Conversão de Tipos

Cuidado ao misturar tipos.

```
#include <iostream>

using namespace std;

int main() {
    int x = 2.4;
    cout << x << endl;
    return 0;
}
```

Saída:

```
2
```

A parte decimal é perdida.