Tipos de Dados

Pesquisa de AEDs I - Roteiro 4

Aluno: Leonardo Aguilar Murça

1 Capacidade de armazenamento de tipos de dados

Logo abaixo temos uma tabela listando os principais tipos de dados da linguagem C++.

Tabela 1: Armazenamento de tipos de dados

Tipo	**Tamanho em bytes**	**Alcance típico**
int	4 bytes	-2.147.483.648 à 2.147.483.647
float	4bytes	$+/-3,4^{+/-38}$ (7 dígitos)
double	8 bytes	$+/-1,7^{+/-308}$ (15 dígitos)
long	8 bytes	$+/-1,7^{+/-308}$ (15 dígitos)

2 Modificadores de Tipos de Dados

A linguagem C++ permite a utilização de modificadores de tipos de dados precedendo os seguintes tipos **char**, **int** e **double**. Um modificador é usado para alterar a capacidade de um tipo de dado para se adequar mais precisamente à certos tipos de situação.

Os modificadores de tipo de dados estão listado abaixo:

- signed
- unsigned
- long
- short

Os modificadores **signed**, **unsigned**, **long** e **short** podem ser aplicados à tipos inteiros de dados. Além disso, **signed** e **unsigned** podem ser aplicados no **char**, e **long** pode ser aplicado em **double**.

Os modificadores **signed** e **unsigned** também podem ser usados como prefixos para os modificadores **long** ou **short**. Por exemplo: **unsigned long int**.

A linguagem C++ permite uma notação prática para a declaração dos modificadores inteiros **unsigned**, **short**, ou **long**. Você pode simplesmente usar a palavra **unsigned**, **short** ou **long** sem precisar digitar **int**. Por exemplo:

```
unsigned x;
unsigned int y;
```

3 Conversão implícita e explícita

3.1 Conversão implícita

Conversões implícitas são automaticamente executadas quando o valor é atribuído à um tipo compatível. Por exemplo:

```
short a = 2000;
int b;
b = a;
```

No exemplo acima, o valor de a foi convertido de **short** para **int** sem a necessidade de nenhum operador explícito. Isso é conhecido como *Conversão Padrão*. Conversões padrão afetam tipos de dados fundamentais e permitem conversões entre tipos numéricos (**short** para **int**, **int** para **float**, **double** para **int**...), e algumas conversões de ponteiros.

3.2 Conversão explícita

A linguagem C++ é fortemente tipada. Muitas conversões, especialmente aquelas que implicam uma interpretação diferente de valores, necessitam de uma *Conversão Explícita*, conhecida em C++ como *type-casting*. Existem 2 sintaxes principais para um conversão explícita genérica: **funcional** e **c-like**.

```
double x = 10.3;
int y;
y = int (x); // notação funcional
y = (int) x; // notação c-like
```

Vale salientar que as 2 notações acima estão corretas e aplicáveis à diversas situações.