Bài: Ép kiểu tường minh trong C++ (Explicit type conversion in C++)

Xem bài học trên website để ủng hộ Kteam: Ép kiểu tường minh trong C++ (Explicit type conversion in C++)

Mọi vấn đề về lỗi website làm ảnh hưởng đến bạn hoặc thắc mắc, mong muốn khóa học mới, nhằm hỗ trợ cải thiện Website. Các bạn vui lòng phản hồi đến Fanpage How Kteam nhé!

Dẫn nhập

Ở bài học trước, bạn đã nắm được tổng quan về ép kiểu dữ liệu và kỹ thuật ÉP KIỂU NGẦM ĐỊNH TRONG C++ (Implicit type conversion).

Hôm nay, mình sẽ chia sẽ thêm một cách ép kiểu dữ liệu thứ 2, đó là **Ép kiểu tường minh trong C++ (Explicit type conversion)**.

Nội dung

Để đọc hiểu bài này tốt nhất các bạn nên có kiến thức cơ bản về các phần:

- BIẾN TRONG C++ (Variables in C++)
- CÁC KIỂU DỮ LIỆU TRONG C++
- ÉP KIỂU NGẦM ĐỊNH TRONG C++ (Implicit type conversion)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Ép kiểu tường minh trong C++ (Explicit type conversion)

Ép kiểu tường minh trong C++ (Explicit type conversion)

Trong bài ÉP KIỂU NGẦM ĐỊNH TRONG C++ (Implicit type conversion), bạn đã biết trong một số trường hợp, trình biên dịch sẽ ngầm chuyển đổi một giá trị từ kiểu dữ liệu này sang kiểu dữ liệu khác. Khi muốn chuyển đổi một giá trị sang kiểu dữ liệu tương tự có miền giá trị lớn hơn, ép kiểu ngầm định nên được sử dụng.

Trong một số trường hợp, bạn sẽ gặp dòng lệnh như sau:

C++:

```
double d = 3 / 2;
```

Giá trị 3 và 2 là hai số nguyên, nên sẽ không có **ép kiểu ngầm định** (Implicit type conversion) trong biểu thức này, kết quả 3 / 2 là 1, sau đó 1 được chuyển đổi ngầm định thành 1.0 và gán cho biến d.

Để khắc phục trường hợp này, bạn có thể chuyển đổi giá trị một trong 2 toán hạng thành số chấm động (3.0 hoặc 2.0) để có kết quả đúng cho biểu thức:

C++:

```
double d1 = 3.0 / 2;
// Hoặc
double d2 = 3 / 2.0;
```

Xét ví dụ tương tự, nhưng 2 toán hạng của bạn là 2 biến:

C++:

```
int n1 = 3;
int n2 = 2;
double d = n1 / n2;
```

Trong trường hợp này, bạn cần sử dụng kỹ thuật Ép kiểu tường minh (Explicit type conversion) để trình biên dịch có thể hiểu và chuyển đổi kiểu dữ liệu theo ý của bạn.

> **Ép kiểu tường minh (Explicit type conversion)** là quá trình chuyển đổi kiểu dữ liệu một cách t**ường minh (rõ ràng) bởi lập trình viên**, sử dụng **toán tử ép kiểu (casting operator)** để thực hiện việc chuyển đổi.

Trong C++, có **5 cách ép kiểu tường minh**:

- C-style casts
- Static casts
- Const casts
- Dynamic casts
- Reinterpret casts

Phạm vi bài học này sẽ nói về **C-style casts** và **Static casts**, 2 cách phổ biến nhất trong C++.

Dynamic casts, Const casts và **Reinterpret casts** cần những kiến thức chuyên sâu hơn, nên sẽ được bỏ qua trong bài học này.

C-style casts

Trong **ngôn ngữ C chuẩn**, ép kiểu được thực hiện thông qua **toán tử ()**, và tên kiểu dữ liệu cần chuyển được đặt bên trong.

Ví dụ:

C++:

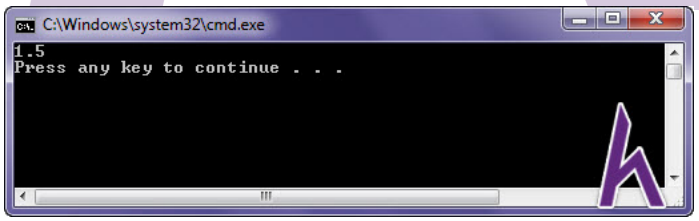
```
#include <iostream>
using namespace std;

int main()
{
	int n1 = 3;
	int n2 = 2;
	double d = (double)n1 / n2;

	cout << d << endl;

	return 0;
}
```

Outputs:



Trong chương trình trên, trình biên dịch (compiler) chuyển đổi biến n1 từ kiểu int sang kiểu double thông qua ép kiểu tường minh C-style. Sau đó biểu thức có 2 toán hạng có kiểu double và kiểu int, nên toán hạng kiểu int sẽ được chuyển đổi ngầm định sang kiểu double. Vì vậy, kết quả của biểu thức là phép chia giữa 2 số chấm động kiểu double.

Ngôn ngữ **C++ cho phép** thực hiện ép kiểu tường minh C-style với **cú pháp như một lời gọi hàm**:

C++:

```
int n1 = 3;
int n2 = 2;
double d = double(n1) / n2;
```

> Ép kiểu tường minh C-style không được trình biên dịch (compiler) kiểm tra tại thời điểm biên dịch (compile time), nên trình biên dịch sẽ không đưa ra những cảnh báo trong những trường hợp chuyển đổi không đúng

> **Chú ý**: **Tránh sử dụng** ép kiểu tường minh **C-style**.

static_cast

Ngôn ngữ C++ có 1 toán tử ép kiểu gọi là static_cast. Bạn đã biết đến nó trong bài KIỂU KÝ TỰ TRONG C++ (Character):

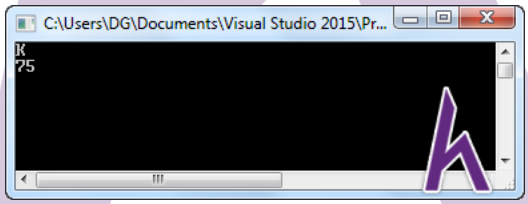
C++:

```
#include <iostream>
#include <iomanip>      // for std::setprecision()
using namespace std;

int main()
{
	int n{ 75 };
	cout << static_cast<char>(n) << endl; // in ký tự với mã ASCII 75

	char ch{ 'K' };
	cout << static_cast<int>(ch) << endl; // in mã ASCII của ký tự 'K'
	return 0;
}
```

Outputs:



Chương trình trên sử dụng toán tử ép kiểu **static cast** trong C++ để in một ký tự từ một số nguyên và ngược lại.

Sử dụng static cast trong biểu thức:

Ví dụ:

C++:

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

int main()
{
	int n1 = 3;
	int n2 = 2;
	double d = static_cast<double>(n1) / n2;

	cout << d << endl;

	return 0;
}
```

Outputs:

Ưu điểm của toán tử static_cast là nó yêu cầu compiler kiểm tra kiểu dữ liệu tại thời điểm biên dịch chương trình, hạn chế được những lỗi ngoài ý muốn.

Chú ý: Nên sử dụng toán tử static_cast thay vì ép kiểu C-style.

Kết luận

Qua bài học này, bạn đã nắm được phương pháp Ép kiểu tường minh trong C++ (Explicit type conversion).

Ở bài tiếp theo, mình sẽ chia sẽ cho các bạn về function trong bài CƠ BẢN VỀ HÀM & GIÁ TRỊ TRẢ VỀ (Basics of Functions and Return values).

Cảm ơn các bạn đã theo dõi bài viết. Hãy để lại bình luận hoặc góp ý của mình để phát triển bài viết tốt hơn. Đừng quên "**Luyện tập – Thử thách – Không ngại khó**".