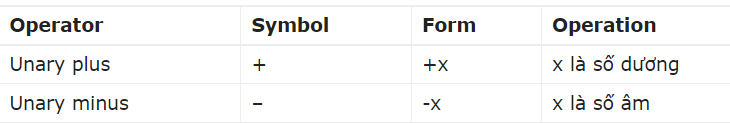
**GIỚI THIỆU CÁC TOÁN TỬ SỐ HỌC TRONG C++**

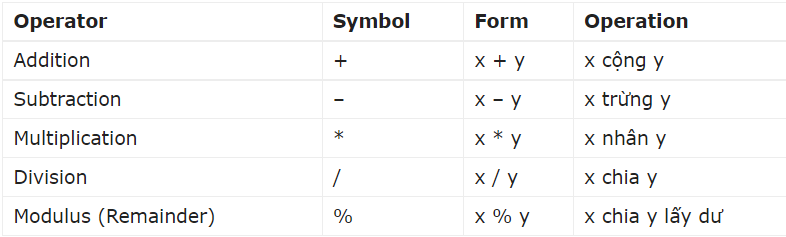
**1. Toán tử số học đơn phương**

Có hai toán tử số học đơn nguyên, cộng (+) và trừ (-). Xin nhắc lại, các toán tử đơn nguyên là các toán tư chỉ lấy một toán hạng.



**2. Toán tử số học nhị phân**

Có 5 toán tử số học nhị phân. Toán tử nhị phân là toán tử lấy toán hạng trái và phải.



3. Số nguyên và phép chia dấu phẩy động

Dễ dàng nhất khi nghĩ về toán tử phân chia là có hai chế độ khác nhau.

Nếu một trong hai (hoặc cả hai) toán hạng là giá trị dấu phẩy động, toán tử chia thực hiện phép chia dấu phẩy động. Phân chia điểm nổi trả về giá trị dấu phẩy động và phân số được giữ. Ví dụ: 7.0 / 4 = 1.75, 7 / 4.0 = 1.75 và 7.0 / 4.0 = 1.75. Như với tất cả các phép toán số học dấu phẩy động, lỗi làm tròn có thể xảy ra.

Nếu cả hai toán hạng là số nguyên, toán tử chia thực hiện phép chia số nguyên thay thế. Phân chia số nguyên giảm bất kỳ phân số và trả về một giá trị số nguyên. Ví dụ: 7/4 = 1 vì phần phân số của kết quả bị loại bỏ. Tương tự, -7 / 4 = -1 vì phân số bị giảm.

Cảnh báo

Trước C ++ 11, phép chia số nguyên với toán hạng âm có thể làm tròn lên hoặc xuống. Do đó -5 / 3 có thể dẫn đến -1 hoặc -2. Điều này đã được sửa trong C ++ 11, nó luôn giảm phân số (làm tròn về 0)

**4. Sử dụng static\_cast <> để thực hiện phép chia dấu phẩy động với số nguyên.**

Những điều trên đặt ra câu hỏi – nếu chúng ta có hai số nguyên và muốn chia chúng mà không mất phân số, chúng ta sẽ làm như thế nào?

**Trong bài Chars, chúng ta chỉ ra cách chúng ta có thể sử dụng toán tử static\_cast<> để chuyển đổi một char thành một số nguyên để nó in dưới dạng một số nguyên thay vì một ký tự.**

Chúng ta có thể sử dụng tương tự static\_cast<> để chuyển đổi một số nguyên thành số dấu phẩy động để chúng ta có thể thực hiện phép chia dấu phẩy động thay vì số nguyên. Hãy xem xét các đoạn code sau đây.

#include <iostream>

int main()

{

int x{ 7 };

int y{ 4 };

std::cout << "int / int = " << x / y << "\n";

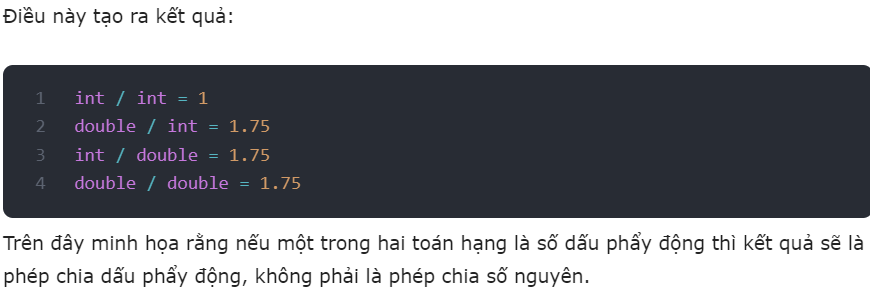
std::cout << "double / int = " << static\_cast<double>(x) / y << "\n";

std::cout << "int / double = " << x / static\_cast<double>(y) << "\n";

std::cout << "double / double = " << static\_cast<double>(x) / static\_cast<double>(y) << "\n";

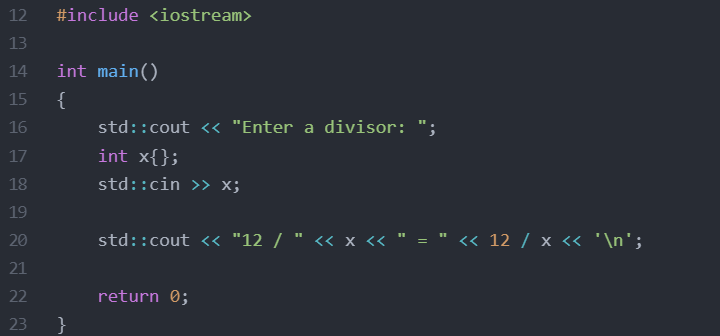
return 0;

}



5. Chia cho số không

Cố gắng chia cho 0 (hoặc 0, 0) thường sẽ khiến chương trình của bạn bị thoát, vì kết quả không được xác định về mặt toán học.



Nếu bạn chạy chương trình và nhập 0, chương trình của bạn sẽ bị sập hoặc chấm dứt bất thường.

