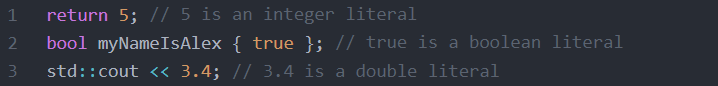
**Literals (Ký tự chữ)**

Trong lập trình, một hằng số là một giá trị cố định không thể thay đổi. C ++ có hai loại hằng số: **hằng số theo nghĩa đen** và **hằng số tượng trưng**. Cafedev sẽ đề cập đến các hằng chữ trong bài học này và hằng số tượng trưng trong bài học tiếp theo.

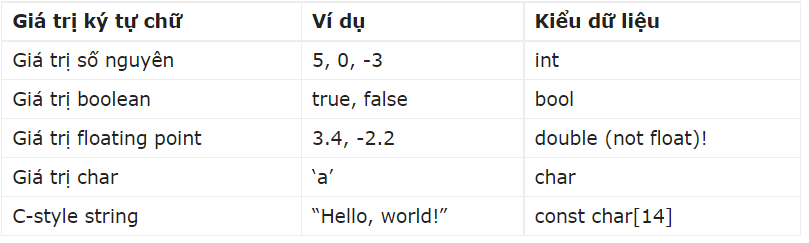
**Hằng số nghĩa đen (thường chỉ được gọi là các ký tự) là các giá trị được chèn trực tiếp vào code. Ví dụ:**



Chúng là hằng số vì các giá trị của chúng không thể thay đổi động (bạn phải thay đổi chúng, sau đó biên dịch lại để thay đổi có hiệu lực)

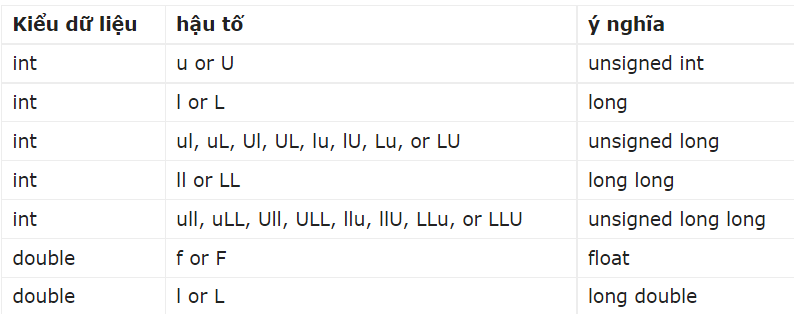
Gioongs như các đối tượng có một kiểu, tất cả các chữ đều có một kiểu dữ liệu. Kiểu của một chữ được giả định từ giá trị và định dạng của chính chữ đó.

Theo mặc định:



**1. Hậu tố của kỹ tự chữ (gọi tắt là ký tự)**

Nếu kiểu ký tự mặc định không như mong muốn, bạn có thể thay đổi kiểu của ký tự bằng cách thêm hậu tố.



Bạn thường không cần sử dụng hậu tố cho các kiểu số nguyên, nhưng đây là các ví dụ:



Theo mặc định, hằng số ký tự dấu chấm động có kiểu kép. Thay vào đó, để làm cho chúng nổi các ký tự, hậu tố f (hoặc F) nên được sử dụng:



Các lập trình viên mới thường bối rối về lý do tại sao những điều sau đây không hoạt động như mong đợi:

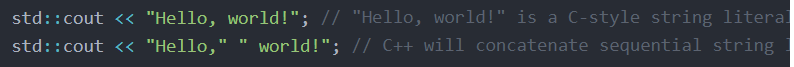


Vì 4.1 không có hậu tố nên nó được coi là một ký tự kép, không phải một ký tự float. Khi C ++ xác định kiểu của một ký tự, nó không quan tâm bạn đang làm gì với ký tự (ví dụ: trong trường hợp này, sử dụng nó để khởi tạo một biến float). Do đó, 4.1 phải được chuyển đổi từ double sang float trước khi nó có thể được gán cho biến f, và điều này có thể dẫn đến mất độ chính xác.

Chữ ký tự có thể sử dụng trong code C ++ miễn là nghĩa của chúng rõ ràng. Điều này thường xảy ra nhất khi được sử dụng để khởi tạo hoặc gán giá trị cho một biến, làm toán **hoặc in một số văn bản ra màn hình.**

**2. Chuỗi ký tự**

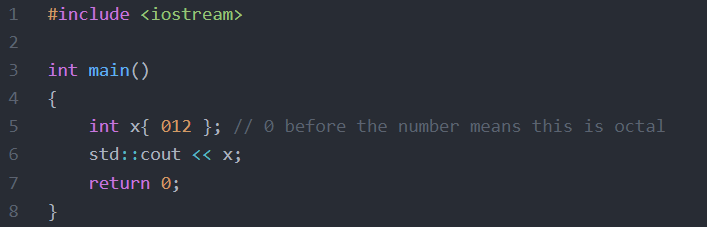
Trong bài -Char, chúng ta đã định nghĩa một chuỗi là một tập hợp các ký tự tuần tự. C++ hỗ trợ chuỗi ký tự:



Các ký tự chuỗi được xử lý rất kỳ lạ trong C ++ vì lý do lịch sử. Hiện tại, bạn có thể sử dụng chuỗi ký tự để in văn bản với std :: cout, nhưng đừng thử và gán chúng cho các biến hoặc chuyển chúng cho các hàm – nó sẽ không hoạt động hoặc sẽ không hoạt động như bạn  mong đợi. Chúng ta sẽ nói thêm về chuỗi kiểu C (và cách giải quyết tất cả những vấn đề kỳ quặc đó) trong các bài học trong tương lai.

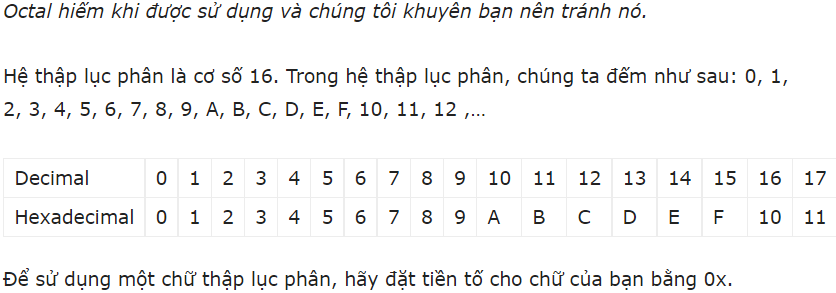
**3. Ký hiệu khoa học cho các ký tự dấu phẩy động**

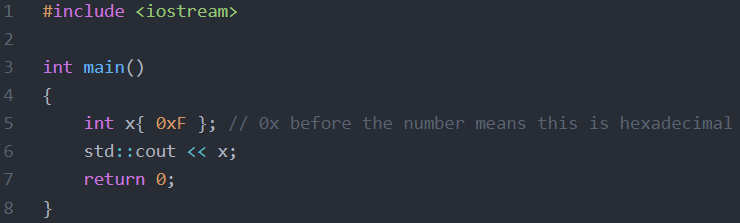
Để sử dụng một chữ bát phân, hãy đặt tiền tố cho chữ của bạn bằng một số 0.



Chương trình này in: 10

**Tại sao 10 thay vì 12? Bởi vì các số được in dưới dạng thập phân, và 12 bát phân = 10 thập phân.**





Chương trình này in: 15

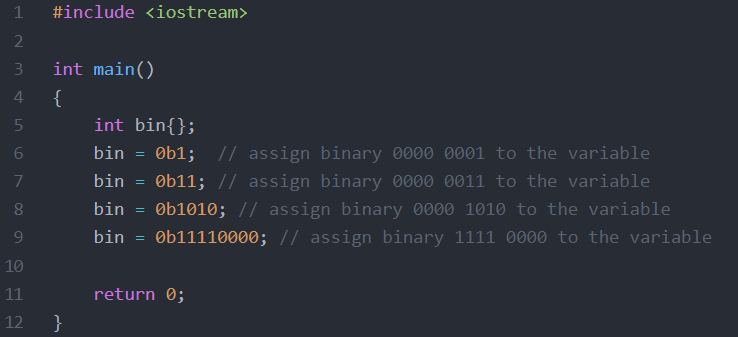
Hãy xem xét một số nguyên 32 bit có giá trị 0011 1010 0111 1111 1001 1000 0010 0110. Do độ dài và sự lặp lại của các chữ số nên điều đó không dễ đọc. Trong hệ thập lục phân, giá trị tương tự này sẽ là: 3A7F 9826. **Điều này làm cho các giá trị thập lục phân hữu ích như một cách ngắn gọn để biểu diễn một giá trị trong bộ nhớ. Vì lý do này, các giá trị thập lục phân thường được sử dụng để biểu diễn địa chỉ bộ nhớ hoặc giá trị thô trong bộ nhớ.**

Trước C++ 14, không có cách nào để gán một ký tự nhị phân Tuy nhiên, các cặp thập lục phân cung cấp cho chúng ta một giải pháp hữu ích.

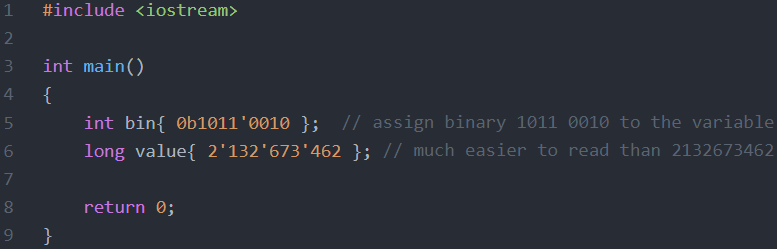


**4. C++ 14 ký tự nhị phân và dấu phân tách chữ số**

Trong C++ 14, chúng ta có thể gán các ký tự nhị phân bằng cahcs sử dụng tiền tố 0b:



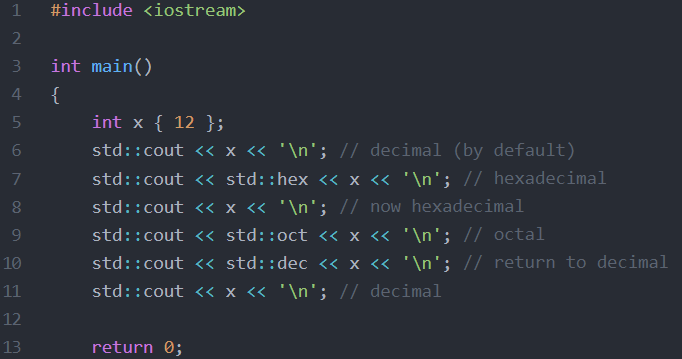
Vì các ký tự dài có thể khó đọc, C++ 14 cũng bổ sung khả năng sử dụng dấu ngoặc kép (‘) làm dấu phân tách chữ số.

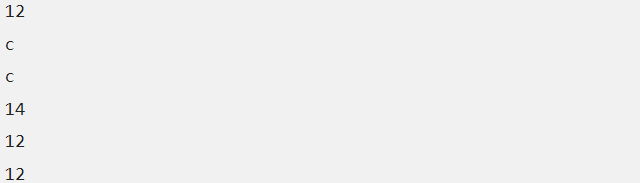


Nếu trình biên dịch của bạn không tương thích với C++ 14, trình biên dịch của bạn sẽ khiếu nại nếu bạn cố gắng sử dụng một trong hai.

**In số thập phân, bát phân, thập lục phân và số nhị phân**

Theo mặc định, C++ in giá trị ở dạng thập phân. Tuy nhiên, bạn có thể yêu cầu nó in ở các định dạng khác. Dễ dàng in ở dạng thập phân, bát phân hoặc hex bằng cách sử dụng std::dec, std::oct và std::hex





In ở dạng nhị phân khó hơn một chút vì std::cout không được tích hợp sẵn khả năng này. May mắn thay, thư viện chuẩn C++ bao gồm một kiểu là std::bitset sẽ thực hiện việc này cho chúng ta (trong phần tiêu đề <bitset>). Để sử dụng std :: bitset, chúng ta có thể xác định một biến std :: bitset và cho std :: bitset biết chúng ta muốn lưu trữ bao nhiêu bit. Số lượng bit phải là một hằng số thời gian biên dịch. std :: bitset có thể được khởi tạo bằng giá trị tích phân không dấu (ở bất kỳ định dạng nào, bao gồm thập phân, bát phân, hex hoặc nhị phân).



