Bài: Cơ bản về Hàm và Giá trị trả về (Basic of functions and return values)

Xem bài học trên website để ủng hộ Kteam: Cơ bản về Hàm và Giá trị trả về (Basic of functions and return values)

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 TƯỜNG MINH TRONG C++ (Explicit type conversion).

Hôm nay, mình sẽ giới thiệu cho các bạn về **Cơ bản về Hàm và Giá trị trả về (Basics of Functions and Return values)**, một phần không thể thiếu trong hầu hết các ngôn ngữ lập trình.

Nội dung

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

- CẤU TRÚC MỘT CHƯƠNG TRÌNH C++ (Structure of a program)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Tổng quan về hàm (functions overview)
- Giá trị trả về (return values)
- Giá trị trả về của kiểu void (return values of type void)

Tổng quan về hàm (functions overview)

Giả sử có một chương trình yêu cầu tính tuổi của người dùng với năm sinh được nhập từ bàn phím.

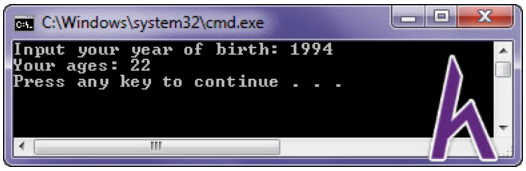
C++:

```
#include <iostream>
using namespace std;
int main()
{
	int year;
	cout << "Input your year of birth: ";
	cin >> year;

	int age = 2016 - year;
	cout << "Your ages: " << age << endl;

	return 0;
}
```

Outputs:



Chương trình bên trên dùng để tính tuổi của 1 người. Giả sử bây giờ bài toán cần được mở rộng thêm, yêu cầu tính tuổi của 3 người.

Vấn đề phát sinh từ đây, bạn phát hiện ra mình phải **lặp lại những dòng code tương tự** bên trên để tính tuổi cho 2 người tiếp theo. Dẫn đến tình trạng **trùng lắp code** và **mất nhiều thời gian** xây dựng chương trình. Để khắc phục vấn đề đó, khái niệm **Hàm (Function)** trong lập trình được ra đời.

> **Hàm (function) là một dãy các câu lệnh** có thể **tái sử dụng**, được thiết kế để thực hiện **một công việc cụ thể** trong chương trình.

Cú pháp của hàm trong C++:

```
<kiểu trả về> <tên hàm>([<danh sách tham số>])
{
   <các câu lệnh>
   [return <giá trị>;]
}
```

Trong đó:

- **<kiểu trả về>**: kiểu bất kỳ của C++ **(bool, char, int, double,...)**. Nếu không trả về thì là **void**.
- **<tên hàm>**: theo quy tắc đặt tên định danh.
- **<danh sách tham số>**: tham số hình thức đầu vào **giống khai báo biến**, cách nhau bằng **dấu phẩy ","**. **(Có thể không có)**
- **<giá trị>**: trả về cho hàm qua lệnh **return**. **(Có thể không có)**

Ở bài CẤU TRÚC MỘT CHƯƠNG TRÌNH C++ (Structure of a program), bạn đã biết mỗi chương trình C++ đều có một hàm tên là main(), hàm này là nơi bắt đầu của chương trình. Trong thực tế, một chương trình thường sẽ có rất nhiều hàm bên trong.

Ví dụ về chương trình đơn giản có 2 hàm: ***main***() và **sayHello**()

C++:

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

// Definition of function sayHello()
void sayHello() // sayHello() is the called function in this example
{
	cout << "Hello Howkteam.com!" << endl;
}

// Definition of function main()
int main()
{
	cout << "Starting main()" << endl;

	// Interrupt main() by making a function call to sayHello().  main() is the caller.
	sayHello();

	cout << "Ending main()" << endl;

	return 0;
}
```

Outputs:

Chương trình sẽ thực thi các câu lệnh một cách **tuần tự** bên trong một hàm. Khi gặp một lời gọi hàm, **CPU sẽ gián đoạn hàm hiện tại** để thực thi các câu lệnh bên trong hàm được gọi. Khi hàm được gọi kết thúc, CPU sẽ lại **tiếp tục thực thi hàm hiện tại**.

> **Chú ý:** Hàm có thể được gọi nhiều lần trong một chương trình **(tính tái sử dụng)**, và bất kỳ hàm nào cũng đều có thể gọi hàm khác.

Hiện tại, bạn nên đặt hàm **main()** ở **vị trí cuối cùng trong file code** của chương trình. Lý do tại sao sẽ được đề cập cụ thể trong bài TIỀN KHAI BÁO & ĐỊNH NGHĨA HÀM (Forward declarations and Definitions of Functions).

Giá trị trả về (return values)

Ở bài CẤU TRÚC MỘT CHƯƠNG TRÌNH C++ (Structure of a program), bạn đã biết hàm **main()** có kiểu **int** nên bắt buộc phải có một câu lệnh **return giá trị kiểu int**. Khi chương trình thực thi kết thúc, **hàm main() sẽ return một giá trị cho hệ điều hành**, để thông báo là nó chạy thành công hay không.

Khi tạo ra một hàm mới, tùy vào mục đích của hàm mà bạn có thể quyết định hàm đó có trả về một giá trị nào đó hay không.

Để tạo ra một hàm có giá trị trả về, bạn cần:

1. **Thiết lập kiểu trả về** trong định nghĩa của hàm
2. **Sử dụng câu lệnh return** để trả về một giá trị.

> **Chú ý:** Khi gặp câu lệnh return, hàm sẽ trả về giá trị ngay tại thời điểm đó. Tất cả câu lệnh trong hàm, sau dòng lệnh return sẽ được bỏ qua.

Ví dụ về chương trình có hàm trả về một số nguyên:

C++:

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

// int means the function returns an integer value to the caller
int return69()
{
	// this function returns an integer, so a return statement is needed
	return 69; // we're going to return integer value 69 back to the caller of this function
}

int main()
{
	cout << return69() << endl; // prints 69

	int sum = return69() + 1;
	cout << sum << endl; // prints 70

	return69(); // okay: the value 69 is returned, but is discarded

	return 0;
}
```

Outputs:

Hàm có giá trị trả về **có thể đặt riêng biệt**, hoặc **bên trong một biểu thức** như ở ví dụ trên.

Một câu hỏi thường được hỏi là: "Hàm có thể trả về nhiều giá trị thông qua câu lệnh return?". Câu trả lời là không. **Khi sử dụng câu lệnh return**, **hàm chỉ có thể trả về một giá trị duy nhất**.

Tuy nhiên, bạn có thể sử dụng phương pháp **truyền tham chiếu** hoặc **truyền địa chỉ** cho hàm để có thể **lấy được nhiều giá trị**:

- Phương pháp **truyền tham chiếu** sẽ được hướng dẫn ở bài: TRUYỀN THAM CHIẾU CHO HÀM (Passing Arguments by Reference).
- Phương pháp **truyền địa chỉ (con trỏ)** sẽ được hướng dẫn trong bài: TRUYỀN ĐỊA CHỈ CHO HÀM (Passing arguments by address).

Giá trị trả về của kiểu void (return values of type void)

Những hàm có mục đích tính toán thường sẽ return một giá trị khi kết thúc hàm. Đối với những hàm **không có mục đích tính toán** (**Vd:** hàm setter, hàm print, ...), C++ hỗ trợ **sử dụng kiểu dữ liệu void** cho những hàm **không có giá trị trả về**.

C++:

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

// void means the function does not return a value to the caller
void sayHello()
{
	cout << "Hello Howkteam.com!" << endl;
	cout << "Free Education" << endl;

	// This function does not return a value so no return statement is needed
}

int main()
{
	sayHello(); // okay: function sayHello() is called, no value is returned

	cout << sayHello(); // error: this line will not compile.  You'll need to comment it out to continue.

	return 0;
}
```

Outputs: "binary '<<': no operator found which takes a right-hand operand of type 'void' (or there is no acceptable conversion)"

Trong chương trình trên, hàm **sayHello()** có **kiểu void nên sẽ không trả về giá trị**. Nên compiler sẽ **thông báo lỗi** không thể in giá trị của hàm **sayHello()** ra màn hình trong lần gọi hàm thứ 2.

> **Chú ý:**
>
> - Hàm có kiểu **void** sẽ **không có giá trị trả về**.
> - Có thể **sử dụng câu lệnh return** trong hàm void để **kết thúc hàm ngay lập tức**.

Kết luận

Qua bài học này, bạn đã nắm được Cơ bản về Hàm và Giá trị trả về (Basics of Function and Return values) trong C++. Mình tóm tắt lại một số nội dung quan trọng các bạn cần nắm:

- **Hàm có thể được gọi nhiều lần** trong một chương trình (tính tái sử dụng).
- Khi gặp câu lệnh return, hàm sẽ **trả về giá trị ngay tại thời điểm đó**. Tất cả câu lệnh trong hàm, **sau dòng lệnh return sẽ được bỏ qua**.
- Hàm có **kiểu void sẽ không có giá trị trả về**.

Ở bài tiếp theo, mình sẽ chia sẻ cho các bạn về kỹ thuật TRUYỀN GIÁ TRỊ TRONG C++ (Passing Arguments by Value in C++).

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**".