Bài: Truyền Giá Trị cho Hàm (Passing Arguments by Value)

Xem bài học trên website để ủng hộ Kteam: Truyền Giá Trị cho Hàm (Passing Arguments by Value)

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 CƠ BẢN VỀ HÀM & GIÁ TRỊ TRẢ VỀ (Basics of Function and Return values) trong C++.

Hôm nay, mình sẽ giới thiệu cho các bạn về kỹ thuật **Truyền Giá Trị trong C++ (Passing Arguments by Value in C++)**.

Nội dung

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

- CƠ BẢN VỀ HÀM & GIÁ TRỊ (Basics of Function and Return values)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Tham số và đối số của hàm (Function parameters and arguments)
- Truyền giá trị cho hàm (Passing arguments by value)
- Tổng kết về phương pháp truyền giá trị cho hàm (Passing arguments by value)

Tham số và đối số của hàm (Function parameters and arguments)

Để chuyển thông tin vào một hàm để tính toán, bạn cần biết đến khái niệm **tham số** và **đối số của hàm (function parameters and arguments)**:

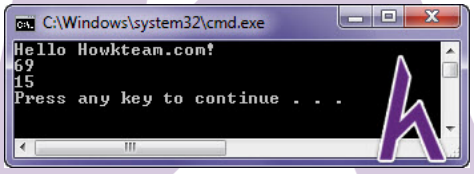
- **Tham số (parameters)**: là các biến được **sử dụng trong một hàm** mà giá trị của biến đó được **cung cấp bởi lời gọi hàm**. Các tham số được đặt bên trong dấu ngoặc đơn, cú pháp giống khai báo biến, **cách nhau bằng dấu phẩy ","**.
- **Đối số (arguments)**: là các **giá trị truyền vào** hàm qua lời gọi hàm, cách nhau bởi **dấu phẩy ","**. Số lượng đối số tương ứng với số lượng tham số của hàm.

Ví dụ: về 3 hàm có số tham số và đối số khác nhau:

C++:

```
#include <iostream>
using namespace std;
// This function takes no parameters
// It does not rely on the caller for anything
void sayHello()
{
	cout << "Hello Howkteam.com!" << endl;
}

// This function takes one integer parameter named x
// The caller will supply the value of x
void printValue(int x)
{
	cout << x << endl;
}

// This function has two integer parameters, one named x, and one named y
// The caller will supply the value of both x and y
int add(int x, int y)
{
	return x + y;
}

int main()
{
	sayHello();

	printValue(69); // 69 is the argument passed to function printValue()
	cout << add(6, 9) << endl; // 6 and 9 are the arguments passed to function add()

	return 0;
}
```

Outputs:



Trong C++, có 3 cách truyền đối số (arguments) cho một hàm:

- **Truyền giá trị (Call by value)**
- **Truyền tham chiếu (Call by reference)** (Chỉ có trong C++): Cách này sẽ được hướng dẫn trong bài sau: TRUYỀN THAM CHIẾU CHO HÀM (Passing Arguments by Reference)
- **Truyền địa chỉ (Call by address)**: Cách này sẽ được hướng dẫn trong bài TRUYỀN ĐỊA CHỈ CHO HÀM (Passing Arguments by Address), sau khi bạn đã được học về con trỏ.

Trong bài học này, mình sẽ chia sẻ về 2 cách đầu tiên.

Truyền giá trị cho hàm (Passing arguments by value)

Trong C++, mặc định đối số được truyền cho hàm ở dạng giá trị.

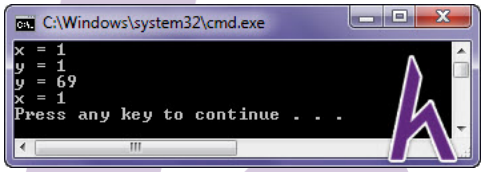
Khi truyền đối số cho hàm ở dạng giá trị, giá trị của đối số được sao chép vào tham số của hàm. Và đối số sẽ **không bị thay đổi** sau lời gọi hàm.

Ví dụ:

C++:

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

void callByValue(int y)
{
	cout << "y = " << y << endl;

	y = 69;

	cout << "y = " << y << endl;
}	// y is destroyed here

int main()
{
	int x(1);
	cout << "x = " << x << endl;

	callByValue(x);

	cout << "x = " << x << endl;

	return 0;
}
```

Outputs:



Trong chương trình trên, biến x truyền vào hàm **callByValue(int y)** ở dạng giá trị, nên nó không bị thay đổi sau lời gọi hàm. Kết quả cuối cùng của biến **x** vẫn là 1.

Tổng kết về phương pháp truyền giá trị cho hàm (Passing argument by value)

Ưu điểm:

- Đối số có thể là **biến** (Vd: x, y), **hằng** (Vd: 1, 2), **biểu thức** (Vd: x + 1), **structs**, **classes**, hoặc **enumerators**.
- Đối số **không bị thay đổi** sau lời gọi hàm, hạn chế tác động không mong muốn của hàm lên đối số.

Nhược điểm:

- **Gây tốn thêm vùng nhớ** do hàm phải tạo các tham số là bản sao của các đối số.
- Gây **giảm hiệu suất** trong trường hợp đối số là **kiểu cấu trúc (structs)** hoặc các **lớp (classes)**, đặc biệt là nếu hàm đó **được gọi nhiều lần**. Vì mỗi lần gọi hàm đều phải sao chép giá trị của đối số vào tham số của hàm.
- Hàm **chỉ có thể trả về một giá trị duy nhất** bằng câu lệnh **return**.

Khi nào nên sử dụng:

- Khi đối số là các **kiểu dữ liệu cơ bản**.
- Khi **không có nhu cầu thay đổi giá trị** của đối số sau khi thực hiện hàm.

Khi nào không nên sử dụng:

- Khi đối số là các **mảng (arrays)**, **kiểu cấu trúc (structs)**, hoặc các **lớp (classes)**.

> Trong đa số trường hợp, **truyền giá trị cho hàm (Passing arguments by value)** là phương pháp **thường được sử dụng nhất**, vì **tính linh hoạt (truyền đối số ở nhiều dạng)** và **an toàn (đối số không bị thay đổi bởi hàm)** của nó.

Kết luận

Qua bài học này, bạn đã nắm được phương pháp Truyền Giá Trị trong C++ (Passing Arguments by Value in C++). Và những ưu điểm, nhược điểm, khi nào nên và không nên sử dụng của phương pháp trên.

Trong bài tiếp theo, mình sẽ giới thiệu cho bạn phương pháp TRUYỀN THAM CHIẾU TRONG C++ (Passing Arguments by Reference in C++). Là một phương pháp khó hơn, và sẽ khắc phục được nhiều nhược điểm của phương pháp truyền giá trị trong bài học này.

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**".