Bài: Hàm trả về giá trị, tham chiếu và địa chỉ trong C++(value, reference, and address)

Xem bài học trên website để ủng hộ Kteam: Hàm trả về giá trị, tham chiếu và địa chỉ trong C++(value, reference, and address)

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

Cho đến bài viết này, mình đã giới thiệu cho các bạn ba phương pháp truyền đối số vào hàm:

- TRUYỀN GIÁ TRỊ CHO HÀM (Passing Arguments by Value)
- TRUYỀN THAM CHIẾU CHO HÀM (Passing Arguments by Reference)
- TRUYỀN ĐỊA CHỈ CHO HÀM (Passing arguments by address)

Trong bài học này, chúng ta sẽ tiếp tục tìm hiểu về vấn đề trả về của hàm thông qua ba phương pháp **Hàm trả về giá trị, tham chiếu và địa chỉ (value, reference, and address)**.

Nội dung

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

- CƠ BẢN VỀ HÀM & GIÁ TRỊ TRẢ VỀ (Basics of Functions and Return values)
- TRUYỀN GIÁ TRỊ CHO HÀM (Passing Arguments by Value)
- TRUYỀN THAM CHIẾU CHO HÀM (Passing Arguments by Reference)
- TRUYỀN ĐỊA CHỈ CHO HÀM (Passing arguments by address)

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Hàm trả về giá trị (return by value)
- Hàm trả về địa chỉ (return by address)
- Hàm trả về tham chiếu (return by reference)

Hàm trả về giá trị (return by value)

> **Chú ý:** Hàm trả về giá trị hoạt động gần như tương tự với phương pháp truyền giá trị cho hàm.

Hàm trả về giá trị là phương pháp **an toàn** và **dễ sử dụng** nhất:

- **Giá trị trả về** của hàm có thể là **biến** (vd: x, y), **hằng** (vd: 1, 2), **biểu thức** (vd: x + 1).
- Khi một giá trị được trả về, một **bản sao** của giá trị đó được tạo ra và trả về cho lời gọi hàm.

Ví dụ:

:

```
int add(int x, int y)
{
	int value = x + y;
	return value; // Bản sao của biến value được tạo ra và trả về
} // biến value bị hủy khi ra khỏi hàm

int return69()
{
	return (60 + 9); // Bản sao của hằng 69 được tạo ra và trả về
}
```

Hàm trả về giá trị là thích hợp nhất khi trả về các biến được khai báo bên trong hàm hoặc trả về các đối số hàm được truyền theo giá trị. Tuy nhiên, hàm trả về giá trị sẽ gây **giảm hiệu suất** trong trường hợp giá trị trả về là **kiểu cấu trúc (structs)** hoặc các **lớp (classes)** phức tạp.

Khi nào **nên** sử dụng:

- Khi trả về giá trị là các kiểu dữ liệu cơ bản.
- Khi trả về các biến được khai báo bên trong hàm, các đối số hàm được truyền bằng giá trị.

Khi nào **không nên** sử dụng:

- Khi trả về một mảng hoặc con trỏ (sử dụng **hàm trả về địa chỉ**).
- Khi trả về một **kiểu cấu trúc (structs)** hoặc **lớp (classes)** phức tạp (sử dụng **hàm trả về tham chiếu**).

Hàm trả về địa chỉ (return by address)

> **Chú ý:** Hàm trả về địa chỉ hoạt động gần như tương tự với phương pháp truyền địa chỉ cho hàm.

Hàm trả về địa chỉ là hàm trả về địa chỉ của một biến cho lời gọi hàm.

- **Giá trị trả về** của hàm **chỉ có thể là địa chỉ của biến**, không phải hằng hoặc biểu thức (không có địa chỉ).
- Hàm trả về địa chỉ **nhanh hơn** trả về giá trị, vì chỉ cần sao chép địa chỉ và trả về cho lời gọi hàm.
- **Không thể** trả về **địa chỉ của biến cục bộ** bên trong hàm.

Ví dụ:

:

```
int* add(int x, int y)
{
	int value = x + y;
	return &value; // trả về địa chỉ của biến value
} // biến value bị hủy khi ra khỏi hàm
```

Bạn có thể thấy, biến **value bị hủy** ngay sau khi ra khỏi hàm. Lúc này địa chỉ của biến **value** thuộc **vùng nhớ không được cấp phát (con trỏ lơ lửng)**, điều này có thể gây ra lỗi **undefined behavior** khi sử dụng nếu vùng nhớ này được cấp phát cho một chương trình khác.

Hàm trả về địa chỉ thường được sử dụng để trả về địa chỉ vùng nhớ được cấp phát động:

:

```
int* allocateArray(int size)
{
	return new int[size];
}

int main()
{
	int *array = allocateArray(10);

	// ...

	delete[] array;
	return 0;
}
```

Trong ví dụ trên, hàm **allocateArray(10)** trả về địa chỉ vùng nhớ được cấp phát động bên trong hàm. Sau khi ra khỏi hàm, vùng nhớ này vẫn được chương trình quản lý, và địa chỉ vùng nhớ được gán cho biến **array**.

Khi nào **nên** sử dụng:

- Khi trả về địa chỉ vùng nhớ được cấp phát động.
- Khi trả về các đối số hàm được truyền bằng địa chỉ.

Khi nào **không nên** sử dụng:

- Khi trả về các biến được khai báo bên trong hàm, hằng, biểu thức (sử dụng **hàm trả về giá trị**).
- Khi trả về một **kiểu cấu trúc (structs)** hoặc lớp (classes) phức tạp (sử dụng hàm trả về tham chiếu).

Hàm trả về tham chiếu (return by reference)

> **Chú ý:** Hàm trả về tham chiếu hoạt động gần như tương tự với phương pháp truyền tham chiếu cho hàm.

Hàm trả về tham chiếu là hàm trả về tham chiếu của một biến cho lời gọi hàm. **Giá trị trả về** của hàm **là biến**, không phải hằng hoặc biểu thức.

Tuy nhiên, giống hàm trả về địa chỉ, bạn **không nên trả** về các tham chiếu của **biến cục bộ**. Xét ví dụ sau:

:

```
int& add(int x, int y)
{
	int value = x + y;
	return value; // trả về tham chiếu của biến value
} // biến value bị hủy khi ra khỏi hàm
```

Trong chương trình trên, biến **value bị hủy** ngay sau khi ra khỏi hàm. Lúc này, hàm trả về một tham chiếu đến vùng nhớ rác. Điều này có thể gây ra lỗi **undefined behavior** khi sử dụng nếu vùng nhớ này được cấp phát cho một chương trình khác.

Hàm trả về tham chiếu thường được sử dụng để trả về các đối số được truyền theo tham chiếu:

:

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

#define MAX 10

int& getElement(array<int, MAX> &arr, int idx)
{
	return arr[idx]; // trả về tham chiếu tới phần tử idx
}

int main()
{
	array<int, MAX> arr;

	// gán phần tử thứ 2 = 20 bằng tham chiếu <=> arr[2] = 20
	getElement(arr, 2) = 20;

	cout << arr[2] << '\n'; // in 20

	system("pause");
	return 0;
}
```

Trong chương trình trên, hàm **getElement(arr, 2)** trả về một tham chiếu đến phần tử mảng **arr[2]**, sau đó tham chiếu này (**arr[2]**) được gán bằng 20.

Một ví dụ khác:

:

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

#define MAX 10

int& max(int &a, int &b)
{
	return a > b ? a : b; // trả về tham chiếu của biến lớn hơn
}

int main()
{
	int x = 6;
	int y = 9;

	max(x, y) = 69; // gán biến lớn hơn bằng 69

	cout << "x = " << x << '\n'; // 6
	cout << "y = " << y << '\n'; // 69: y lớn hơn nên bị thay đổi

	system("pause");
	return 0;
}
```

Khi tìm hiểu đến phần **Class**, bạn sẽ tìm thấy nhiều cách sử dụng hơn đối với việc trả về tham chiếu cho hàm.

Khi nào **nên** sử dụng:

- Khi trả về các đối số hàm được truyền bằng tham chiếu.
- Khi trả về một phần tử từ một mảng được truyền vào hàm.
- Khi trả về một **kiểu cấu trúc (structs)** hoặc **lớp (classes)** phức tạp mà không bị hủy khi ra khỏi hàm (được truyền vào hàm).

Khi **không nên** sử dụng:

- Khi trả về các biến được khai báo bên trong hàm (sử dụng **hàm trả về giá trị**)
- Khi trả về một giá trị mảng hoặc con trỏ (sử dụng **hàm trả về địa chỉ**)

Kết luận

Qua bài học này, bạn đã nắm được phương pháp Hàm trả về giá trị, tham chiếu và địa chỉ (value, reference, and address). Và những ưu điểm, nhược điểm, khi nào nên và không nên sử dụng của phương pháp trên.

Trong bài tiếp theo, chúng ta sẽ cùng tìm hiểu về HÀM NỘI TUYẾN TRONG C++ (Inline functions).

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**".