Bài 5: GHI CHÚ TRONG C++ (COMMENTS IN C++)

Xem bài học trên website để ủng hộ Kteam: Ghi chú trong C++ (Comments in C++)

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, CẤU TRÚC MỘT CHƯƠNG TRÌNH C++(Structure of a program), bạn đã hiểu được cấu trúc một chương trình cơ bản trong C++.

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

int main()
{
	cout << "Hello HowKteam.com!" << endl;
	return 0;
}
```

Trong bất cứ ngành nghề nào, chắc chắn bạn không chỉ làm việc một mình, đặc biệt trong lập trình, bạn muốn đồng nghiệp hoặc những thế hệ sau có thể dễ dàng hiểu được và kế thừa những dòng code của bạn viết ra, hoặc để vài năm sau đọc lại bạn vẫn đảm bảo hiểu được mình viết gì trong đó.

Để làm được chuyện đó, ngoài việc tuân thủ các **coding convention**, **naming convention**, ... thì một trong những cách truyền đạt ý nghĩa đoạn code của bạn cho mọi người sẽ được nói đến trong bài hôm nay: **Ghi chú trong C++ (Comments in C++)**.

Copyright © Howkteam.com

Nội dung:

Trong bài học này, chúng ta sẽ cùng tìm hiểu các vấn đề:

- Cú pháp comment trong C++
- Một số kinh nghiệm khi comment trong lập trình

Cú pháp comment trong C++

Trong lập trình, **Comment** là một dòng hoặc nhiều dòng văn bản, được chèn vào source code chương trình, nhằm làm cho source code trở nên dễ hiểu hơn với người đọc, được bỏ qua bởi **compiler** và **interpreter**. Trong C++, có 2 loại comment:

- **Ký hiệu //**: dùng cho comment 1 dòng. Với loại comment này, **compiler** sẽ bỏ qua mọi thứ từ ký hiệu **//** đến cuối dòng. Ví dụ:
    - Những bình comment này được dùng để giải thích cho 1 dòng code

```
cout << "Hello HowKTeam.com!" << endl; // Mọi thứ bên phải ký hiệu này đều bị bỏ qua
cout << "Free education!" << endl; // Dùng để giải thích cho dòng code này
cout << "Nice to meet you!" << endl;          // Hạn chế comment theo cách này
cout << "Hello HowKTeam.com! Free education!" << endl; // Đặc biệt với dòng code dài
```

    - Thông thường, comment // bên phải dòng code là không được khuyến khích, vì nó sẽ gây khó đọc cho cả code và comment của bạn, đặc biệt đối với những dòng code dài. Vì vậy, comment // thường được đặt phía trên của dòng code cần giải thích. Ví dụ:

```
// Mọi thứ bên phải ký hiệu này đều bị bỏ qua
cout << "Hello HowKTeam.com!" << endl;
```


Copyright © Howkteam.com

```
// Dùng để giải thích cho dòng code này
cout << "Free education!" << endl;

// Không nên comment theo cách này
cout << "Nice to meet you!" << endl;

// Đặc biệt với dòng code dài
cout << "Hello HowKTeam.com! Free education!" << endl;
```

- **Ký hiệu /* và */:** dùng cho comment nhiều dòng. Với loại comment này, **compiler** sẽ bỏ qua mọi thứ ở giữa ký hiệu **/* và */.** Ví dụ:
- Đây là comment nhiều dòng đơn giản:

```
/*
Đây là comment nhiều dòng
Mọi thứ bên trong ký hiệu này sẽ được bỏ qua
Bạn có thể viết cả bài văn vào đây...
*/
```

- Bạn có thể comment giữa dòng code của bạn. Ví dụ:

```
return /* Bỏ qua mọi thứ trong này */ 0;
```

- Hoặc bạn cũng có thể làm cho comment đẹp hơn bằng cách:

```
/*
 * Đây là comment nhiều dòng
 * Mọi thứ bên trong ký hiệu này sẽ được bỏ qua
 * Bạn có thể viết cả bài văn vào đây...
 */
```

- **Quy tắc: comment nhiều dòng** không được lồng nhau. Ví dụ:

```
/* Đây là comment cha /* comment con */ lỗi rồi */
// Comment 1 dòng /* comment nhiều dòng */ vẫn là comment
```


Copyright © Howkteam.com

Một số kinh nghiệm khi comment trong lập trình

Bạn đã nắm được các loại comment trong C++. Nhưng mới chỉ nắm được cú pháp không thôi vẫn chưa đủ, bạn cần phải biết sử dụng nó như thế nào cho hợp lý. Dưới đây là một số kinh nghiệm khi comment trong lập trình:

Thứ nhất, ở mức library, program hoặc function, một good comment sẽ mô tả được library, program hoặc function đó có nhiệm vụ gì:

```
//************************************
// Description: Thư viện này khai báo các standard input/output stream objects (iostream)
//************************************
#include <iostream>

//************************************
// Description: Chương trình tính thời gian xây được nhà dựa vào nơi bạn sống và ngành nghề của bạn
//************************************
double TinhThoiGianXayNha(string address, string job)


//************************************
// Description: Hàm sắp xếp mảng bằng thuật toán quick sort.
//************************************
void QuickSort(int * arr, int left, int right)
```

Những comment như trên sẽ giúp người khác nhanh chóng hiểu được một library, program hoặc function đó có mục đích gì, mà không cần phải nhìn vào những đoạn code của nó. Thông thường, những comment ở mức library có thể nằm trong file readme.txt, hoặc trên main function đối với một program.

Thứ hai, bên trong library, program hoặc function, một good comment sẽ mô tả được library, program hoặc function đó thực hiện như thế nào:

```
void QuickSort(int * arr, int left, int right)
```


Copyright © Howkteam.com

```
{
	////////////////////////////////////////////////////////////////////////
	//
	//	Để sắp sếp bằng thuật toán quick sort, hàm thực hiện theo các bước bên dưới:
	//	1. Tìm một giá trị trục
	//	2. Di chuyển tất cả phần tử lớn hơn hoặc bằng giá trị trục sang phải
	//	3. Di chuyển tất cả phần tử nhỏ hơn giá trị trục sang trái
	//	4. Sắp sếp đệ quy cho 2 mảng con bên trái và bên phải
	//
	////////////////////////////////////////////////////////////////////////
	// code here...

}


double TinhThoiGianXayNha(string address, string job)
{
	//************************************
	//	Để tính được thời gian bạn có thể xây nhà, chương trình sẽ :
	//	1. Thống kê mức lương trung bình ngành nghề của bạn
	//	2. Tính chi phí sinh hoạt trung bình nơi bạn đang sống
	//	3. Tính thêm tỉ lệ lạm phát nơi bạn sống,
	//	...
	//	Lấy kết quả nhân 69 sẽ ra.
	//************************************
	// code here...

	return 0;
}
```

Những comment như trên sẽ cho người khác biết **ý tưởng thực hiện cơ bản của library, program hoặc một function**, mà không cần phải xem đến từng dòng code. Những comment ở mức này không cần giải thích quá chi tiết.

Thứ ba, ở mức từng dòng code, một good comment sẽ giải thích tại sao. Một bad comment sẽ giải thích dòng code đó làm gì:

- **Bad comment:**

```
const int size = 10;
```


Copyright © Howkteam.com

```
// Gán smallestIndex bằng 0
int smallestIndex = 0;
for (int startIndex = smallestIndex + 1; startIndex < size; ++startIndex)
{
		//...
}
```

(Nhìn vào dòng code ai cũng hiểu smallestIndex được gán bằng startIndex, nhưng tại sao?)

Hoặc:

```
int array[] = { 6, 9, 69, 96 };
// Sử dụng insertion sort sắp xếp mảng
InsertionSort(array);
```

(Nhìn vào comment này ai cũng hiểu sắp xếp mảng bằng insertion sort, nhưng tại sao?)

- **Good comment:**

```
// smallestIndex1 là chỉ số của phần tử nhỏ nhất, giả sử phần tử đầu tiên
int smallestIndex1 = 0;
for (int startIndex = smallestIndex1 + 1; startIndex < size; ++startIndex)
{
		//...
}
```

Hoặc:

```
int array[] = { 6, 9, 69, 96 };
// Cần một thuật toán ổn định, hiệu suất không thực sự quan trọng
InsertionSort(array);
```

> **Có thể bạn đang viết một đoạn code rất phức tạp, và cần một comment để giải thích cho đoạn code đó**. Mình nghĩ bạn nên xem lại đoạn code của mình, xem đã tuân thủ coding convention, naming

Copyright © Howkteam.com

> convention chưa, và nên sửa lại đoạn code của bạn sao cho dễ hiểu, **không nên comment nó**.

- **Không nên viết những dòng code khó hiểu**, hoặc lạm dụng comment:

```
// Số lượng phần tử của mảng
const int n = 69;

// Gán x bằng max của a, b, c
int x = a > b ? (a > c ? a : c) : (b > c ? b : c);
```

- **Nên tuân thủ coding convention, naming convention** để dòng code trở nên dễ hiểu hơn, không lạm dụng comment:

```
const int nNumberOfElements = 69;

int nMax = a;
if (nMax < b)
{
	nMax = b;
}
if (nMax < c)
{
	nMax = c;
}
```

Commenting out code

Comment không chỉ dừng ở mục đích giải thích đoạn code của bạn. Đôi khi bạn sẽ gặp vài tình huống như:

- Bạn đang cần chạy một chương trình, và có vài dòng code trong chương trình của bạn đang gặp lỗi nên compiler không cho phép. Nhưng bạn muốn chạy chương trình ngay.

Copyright © Howkteam.com

- Bạn đang nâng cấp một đoạn code, bạn muốn giữ đoạn code cũ để tham khảo cho đến khi đoạn code mới của bạn hoàn thành. Hoặc để phục hồi lại đoạn code cũ nếu đoạn code mới của bạn chạy gặp vấn đề.

Khi gặp những tình huống này, bạn có thể dùng comment out code, mọi đoạn code mà bạn comment sẽ được bỏ qua bởi compiler. Ví dụ:

```
// const int n = 69;
// cout << "Hello HowKTeam.com!" << endl;
```

Tóm lại:

- Ở mức **library, program hoặc function**, một **good comment** sẽ mô tả được library, program hoặc function đó có **nhiệm vụ gì**
- Bên trong **library, program hoặc function**, một **good comment** sẽ mô tả được library, program hoặc function đó **thực hiện như thế nào**
- Ở mức **từng dòng code,** một **good comment** sẽ giải thích **tại sao**. Một **bad comment** sẽ giải thích dòng code đó **làm gì.**

Kết luận

Qua bài học này, bạn đã nắm được Ghi chú trong C++ (Comments in C++), và đã biết sử dụng nó như thế nào cho hợp lý.

Bài học tiếp theo, mình sẽ hướng dẫn các bạn về BIẾN TRONG C++ (Variables)

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**".

Copyright © Howkteam.com