Bài 28: VÒNG LẶP WHILE TRONG C++ (WHILE STATEMENTS)

Xem bài học trên website để ủng hộ Kteam: Vòng lặp While trong C++ (While statements)

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, mình đã chia cho các bạn về CÂU LỆNH GOTO TRONG C++ (Goto statements). Nhìn chung, việc sử dụng câu lệnh Goto bị xa lánh trong C++ (và hầu hết các ngôn ngữ bậc cao khác). Bạn nên hạn chế sử dụng chúng.

Trong bài hôm nay, mình sẽ giới thiệu cho các bạn về **Vòng lặp While trong C++ (While statements)**.

Nội dung

Trong bài ta sẽ cùng tìm hiểu các vấn đề:

- Tổng quan về cấu trúc vòng lặp
- Vòng lặp while

Tổng quan về cấu trúc vòng lặp

Trong cuộc sống, có rất nhiều tình huống giống nhau được lặp đi lặp lại nhiều lần. Lập trình cũng vậy, các chương trình máy tính luôn có những đoạn code được lặp đi lặp lại.

Ví dụ:

- Chương trình yêu cầu xuất các số từ **1** đến **10**. => sử dụng **10** câu lệnh cout.
- Chương trình yêu cầu xuất các số từ **1** đến **1000**. => sử dụng **1000** câu lệnh cout **!!!**

Trong ví dụ trên, lập trình viên không thể tự tay viết **1000** câu lệnh cout, vì nó mất khá nhiều thời gian và công sức.

Vì vậy, C++ đã cung cấp 4 loại vòng lặp: **while**, **do while**, **for** và **for each** (C++ 11) cho phép thực hiện lặp đi lặp lại một công việc nào đó. Trong bài học này, chúng ta cùng tìm hiểu về vòng lặp **while**.

Vòng lặp while (while statements)

Vòng lặp while là cấu trúc lặp **đơn giản nhất** trong bốn cấu trúc lặp mà C++ cung cấp, và nó có một nét rất giống với câu lệnh if:

```
while (expression)
    statement;
```

Hoặc:

```
while (expression)
{
    statements;
}
```

Nếu **expression** là true (khác 0), các câu lệnh bên trong khối lệnh sẽ được thực thi. Nếu vòng lặp while thực thi **nhiều câu lệnh**, bạn cần đặt các câu lệnh vào **khối ngoặc nhọn {}**.

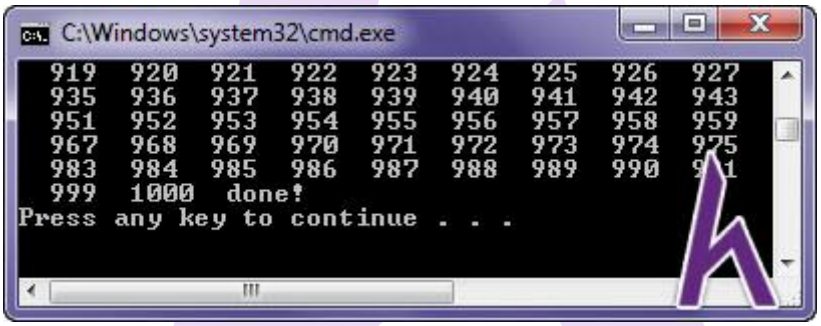
Ví dụ: Chương trình yêu cầu xuất các số từ **1** đến **1000**.

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

int main()
{
	int count(1);
	while (count <= 1000)
	{
		cout << count << " ";
		++count;
	}
	cout << "done!" << endl;

	return 0;
}
```

Outputs:



Trong ví dụ trên, khi sử dụng vòng lặp, bạn **không cần** phải viết đến **1000** lần dòng lệnh cout. Vòng lặp sẽ **chấm dứt** khi điều kiện lặp không còn đúng, nghĩa là biến **count > 1000**.

Một vòng lặp có thể không được thực hiện lần nào, nếu biểu thức điều kiện sai ngay từ đầu:

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

int main()
{
```

```
	int count(1000);
	while (count <= 10)
	{
		cout << count << " ";
		++count;
	}
	cout << "done!" << endl;

	return 0;
}
```

Vòng lặp vô hạn (Infinite loops)

> Nếu **biểu thức điều kiện luôn đúng**, vòng lặp while sẽ **thực hiện mãi mãi**. Đây gọi là một **vòng lặp vô hạn**.

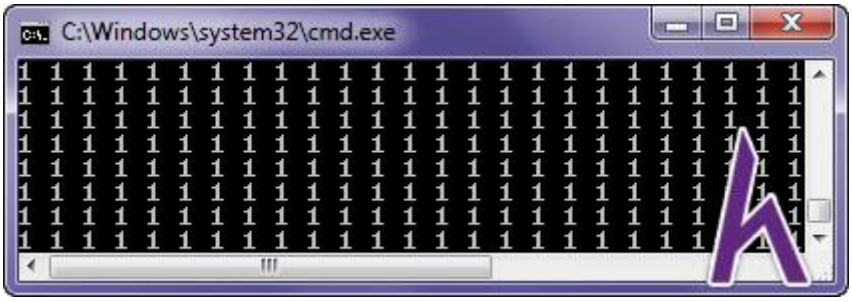
Ví dụ:

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

int main()
{
	int count(1);
	while (count < 10) // điều kiện luôn đúng vì count luôn = 1
		cout << count << " ";

	return 0; // dòng lệnh này không bao giờ được thực thi
}
```

Outputs:



Trong ví dụ trên, biến **count** **không bao giờ thay đổi giá trị**, nên biểu thức **count** **< 10** luôn đúng. Vì vậy, vòng lặp sẽ lặp mãi mãi.

Bạn cũng có thể cố ý tạo ra một vòng lặp vô hạn:

```
while (1) // or while (true)
{
	// vòng lặp này sẽ lặp mãi mãi

	// có thể thoát khỏi vòng lặp bằng cách:
	// return, break, exit(), goto, throw hoặc bạn tự tắt chương trình.
}
```

Cách **duy nhất** để thoát khỏi một vòng lặp vô hạn là sử dụng một trong những từ khóa: **return**, **break**, **exit()**, **goto**, **throw** hoặc bạn tự tắt chương trình.

Biến vòng lặp (Loop variables)

Thông thường, người ta thường sử dụng một **biến vòng lặp** để **giới hạn số lần lặp** của vòng lặp. **Biến vòng lặp là một biến số nguyên** với mục đích duy nhất **là đếm số lần lặp** đã được thực hiện.

Trong những ví dụ trên, các biến **count** là một biến vòng lặp.

> **Nguyên tắc:** Không sử dụng kiểu số nguyên không dấu (**unsigned**) cho các biến vòng lặp.

Ví dụ:

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

int main()
{
	unsigned int count = 10;

	// count from 10 down to 0
	while (count >= 0)
	{
		cout << count << " ";
		--count;
	}

	return 0;
}
```

Outputs:

Ví dụ trên là một chương trình **lặp vô hạn**, nó in ra màn hình dãy số: "10 9 8 7 6 5 4 3 2 1 0 4294967295 4294967294 ...". Tại sao như vậy, biến **count** có kiểu dữ liệu **unsigned int**, nên sẽ không có giá trị âm, nên vòng lặp sẽ không bao giờ chấm dứt. Nếu giá trị của **count = 0**, khi giảm đi 1 sẽ tràn số và quay lại 4294967295, suy ra điều kiện lặp **count >= 0** sẽ luôn luôn đúng.

Các biến vòng lặp thường được đặt những tên đơn giản (**ví dụ:** i, j, k, iii, jjj, kkk, ...). Nhưng để dễ phân biệt hơn, bạn nên đặt cho nó những tên có ý nghĩa cho từng mục đích, ví dụ như count.

Vòng lặp lồng nhau (Nested loops)

Một vòng lặp while có thể lồng vào trong một vòng lặp khác. **Ví dụ:**

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

int main()
{
	// Loop between 1 and 5
	int outer(1);
	while (outer <= 5)
	{
		// loop between 1 and outer
		int inner(1);
		while (inner <= outer)
		{
			cout << inner << " ";
			++inner;
		}

		// print a newline at the end of each row
		cout << "\n";
		++outer;
	}

	return 0;
}
```

Outputs:

Kết luận

Qua bài học này, bạn đã nắm rõ về Vòng lặp While trong C++ (While statements). Vòng lặp while là một cấu trúc đơn giản, dễ sử dụng, lập trình viên thường sử dụng vòng lập while khi **số lần lặp lại của một công việc là chưa biết trước**.

Trong bài tiếp theo, mình sẽ giới thiệu cho các bạn về cấu trúc vòng lặp thứ 2 trong C++, đó là VÒNG LẶP DO WHILE TRONG C++ (Do while statements).

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**".