Bài 5: Vòng Lặp For, While, Do While, Format number Trong C++

1. Mục đích

## Tại Sao Cần Phải Học Vòng Lặp

Vòng lặp là một công cụ mạnh mẽ trong lập trình giúp tự động hóa và xử lý các nhiệm vụ lặp đi lặp lại. Việc học vòng lặp sẽ giúp bạn:

- \*\*Giảm Thiểu Mã Lệnh\*\*: Vòng lặp giúp bạn viết mã lệnh ngắn gọn hơn, tránh việc phải lặp lại nhiều lần.

- \*\*Tăng Tính Linh Hoạt\*\*: Cho phép xử lý dữ liệu và thực hiện các nhiệm vụ lặp lại một cách hiệu quả.

- \*\*Giải Quyết Các Bài Toán Thực Tế\*\*: Vòng lặp là giải pháp quan trọng cho nhiều bài toán trong lập trình như tính tổng, tìm giá trị lớn nhất, và nhiều hơn nữa.

Với những lợi ích này, việc nắm vững cách sử dụng vòng lặp là một kỹ năng cơ bản nhưng rất quan trọng trong lập trình.

1. Các loại vòng lặp
   1. Vòng Lặp for

Vòng lặp for thường được sử dụng khi số lần lặp biết trước. Cú pháp cơ bản:

for (khởi\_tạo; điều\_kiện; bước\_lặp) {  
    // Mã lệnh thực thi  
}

**Bài Tập Thực Hành 1:**

Viết một chương trình sử dụng vòng lặp for để in ra các số chẵn từ 2 đến 10.

**Bài Tập Thực Hành 2:**

Viết một chương trình sử dụng vòng lặp for để tính tích của các số từ 1 đến 5.

**Bài Tập Thực Hành 3:**

Viết một chương trình sử dụng vòng lặp for để in ra các số từ 1 đến 10.

**Bài Tập Thực Hành 4:**

Viết một chương trình sử dụng vòng lặp for để tính tổng của các số từ 1 đến 10.

* 1. Vòng Lặp while\

Vòng lặp while thường được sử dụng khi số lần lặp không xác định trước và phụ thuộc vào điều kiện. Cú pháp cơ bản:

while (điều\_kiện) {  
    // Mã lệnh thực thi  
}

**Bài Tập Thực Hành 1:**

Viết một chương trình sử dụng vòng lặp while để in ra các số lẻ từ 1 đến 9.

**Bài Tập Thực Hành 2:**

Viết một chương trình sử dụng vòng lặp while để đếm số lần người dùng nhập vào số 5 cho đến khi người dùng nhập số khác.

* 1. Vòng Lặp do-while

Vòng lặp do-while đảm bảo mã lệnh bên trong vòng lặp ít nhất được thực hiện một lần. Cú pháp cơ bản:

 do {  
    // Mã lệnh thực thi  
} while (điều\_kiện);

**Bài Tập Thực Hành 1:**

Viết một chương trình sử dụng vòng lặp do-while để yêu cầu người dùng nhập vào một số từ 1 đến 100 cho đến khi số đó lớn hơn 50.

**Bài Tập Thực Hành 2:**

Viết một chương trình sử dụng vòng lặp do-while để tính tổng của các số nhập vào cho đến khi người dùng nhập số 0.

1. Break và Continue
   1. Break

* **Chức năng**: Từ khóa break được sử dụng để thoát khỏi vòng lặp hiện tại ngay lập tức, bất kể điều kiện vòng lặp còn đúng hay không.
* **Khi sử dụng**: break thường được sử dụng khi bạn cần dừng vòng lặp sớm hơn khi một điều kiện cụ thể được thỏa mãn.
  1. Continue
* **Chức năng**: Từ khóa continue được sử dụng để bỏ qua phần còn lại của vòng lặp hiện tại và chuyển đến vòng lặp tiếp theo.
* **Khi sử dụng**: continue thường được sử dụng khi bạn muốn bỏ qua một số phần của vòng lặp và tiếp tục với lần lặp tiếp theo.

1. Vòng lặp lồng nhau

* **Khái niệm**: Vòng lặp lồng nhau là khi một vòng lặp nằm bên trong một vòng lặp khác. Kỹ thuật này rất hữu ích trong các bài toán liên quan đến ma trận hoặc các cấu trúc dữ liệu nhiều chiều khác.
* **Cú pháp cơ bản**:

for (int i = 0; i < rows; ++i) {  
    for (int j = 0; j < columns; ++j) {  
        // Mã lệnh thực thi  
    }  
}

Ví Dụ 1: In Hình Chữ Nhật

**Mô tả**: In một hình chữ nhật với chiều cao và chiều rộng xác định.

Ví Dụ 2: In Bảng Cửu Chương

**Mô tả**: In bảng cửu chương từ 1 đến 9.

1. Sử Dụng std::fixed Để Định Dạng Số Thực Khi Xuất Ra Màn Hình Trong C++

### **1. Giới Thiệu**

Trong C++, khi làm việc với số thực (floating-point numbers), dữ liệu thường được hiển thị theo dạng khoa học (scientific notation) hoặc dạng số thập phân. std::fixed là một công cụ giúp bạn kiểm soát định dạng số thực khi xuất ra màn hình, cho phép hiển thị số theo dạng thập phân cố định.

### **2. Thư Viện Cần Thiết**

Để sử dụng std::fixed, bạn cần bao gồm thư viện iomanip:

#include <iomanip>

### **3. Tại Sao Cần** std::fixed

Mặc định, số thực có thể được hiển thị theo dạng khoa học nếu số quá lớn hoặc quá nhỏ. Điều này có thể gây khó khăn trong việc đọc và hiểu dữ liệu, đặc biệt khi bạn muốn hiển thị số với một số lượng chữ số thập phân cụ thể.

### **4. Cú Pháp Cơ Bản**

Để sử dụng std::fixed, bạn cần bao gồm thư viện iomanip và kết hợp nó với std::setprecision để xác định số lượng chữ số thập phân hiển thị.

### **Lưu Ý**

* **Đặt Lại Định Dạng**: Nếu bạn cần quay lại định dạng mặc định, sử dụng std::defaultfloat:

**std::cout << std::defaultfloat;**

* **Kết Hợp Với std::setprecision**: std::setprecision xác định số lượng chữ số sau dấu thập phân. Nếu không sử dụng std::setprecision, số lượng chữ số thập phân mặc định sẽ được áp dụng.