TIN ĐẠI CƯƠNG

Bài 5: CẤU TRÚC LẶP

Trần Thị Ngân

Bộ môn Công nghệ phần mềm, Khoa CNTT

Trường đại học Thủy Lợi

Nội dung chính

1. Cấu trúc chương trình
2. Vòng lặp FOR
3. Vòng lặp WHILE
4. Vòng lặp DO-WHILE
5. Từ khoá break và continue
6. Bài tập

1. Cấu trúc chương trình

Có 3 loại cấu trúc

- Tuần tự
- Lặp
- Lựa chọn (sẽ học ở bài sau)

Cấu trúc tuần tự

- Các lệnh được lần lượt thực hiện
- Tất cả các chương trình học đến thời điểm này đều theo cấu trúc tuần tự

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

int main()
{
  int a = 10;
  std::cout << "a = " << a << std::endl;
  const int b = a * 4;
  std::cout << "b = " << b << std::endl;
  return 0;
}
```

Cấu trúc lặp

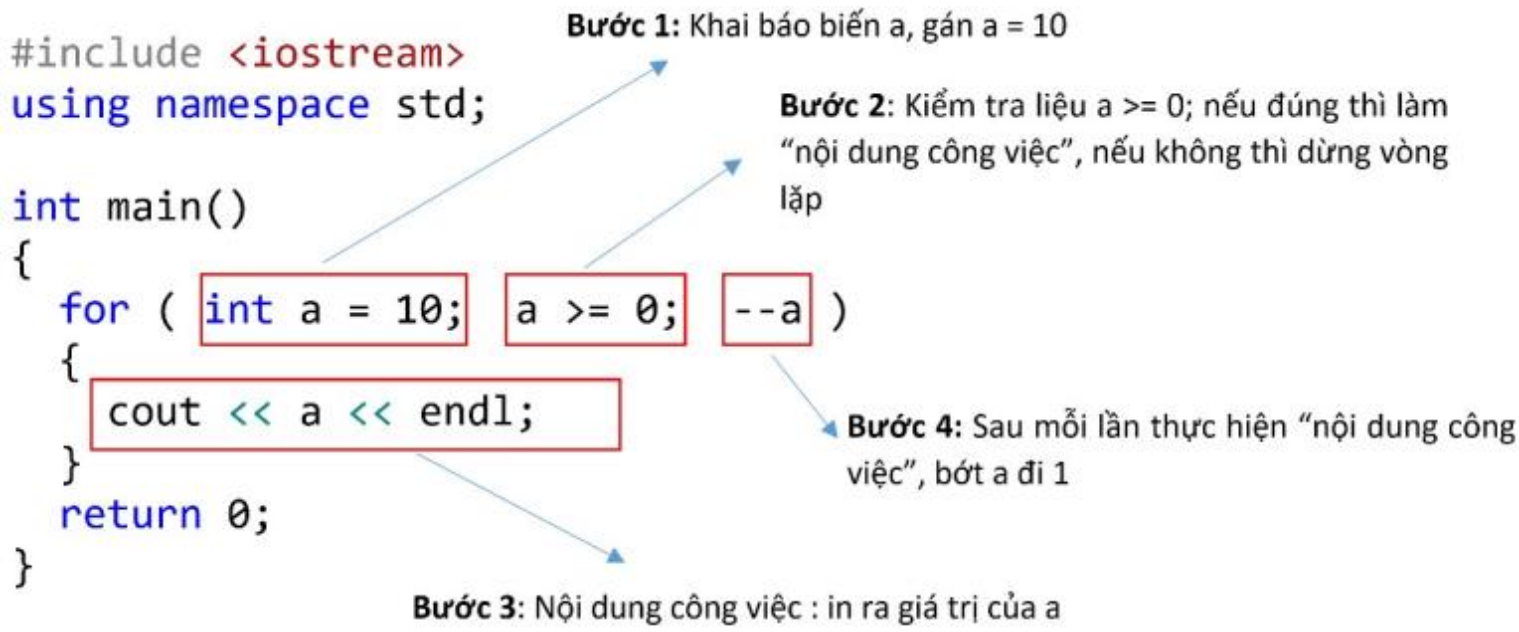
- Ví dụ : Nhập dữ liệu điểm môn Tin học đại cương cho 120 sinh viên lớp N03 và tính điểm trung bình của lớp. Nếu dùng cấu trúc tuần tự :
  - Khai báo 120 biến để lưu điểm của 120 sinh viên
  - Viết 120 lần lệnh nhập dữ liệu
  - Viết 120 lần lệnh cộng các biến
- Vấn đề : chương trình quá dài, nhàm chán, rất dễ phát sinh lỗi, không tổng quát hóa

→ Giải pháp : sử dụng cấu trúc lặp, làm lặp đi lặp lại công việc nào đó cho đến khi thỏa mãn một điều kiện

Cấu trúc lặp

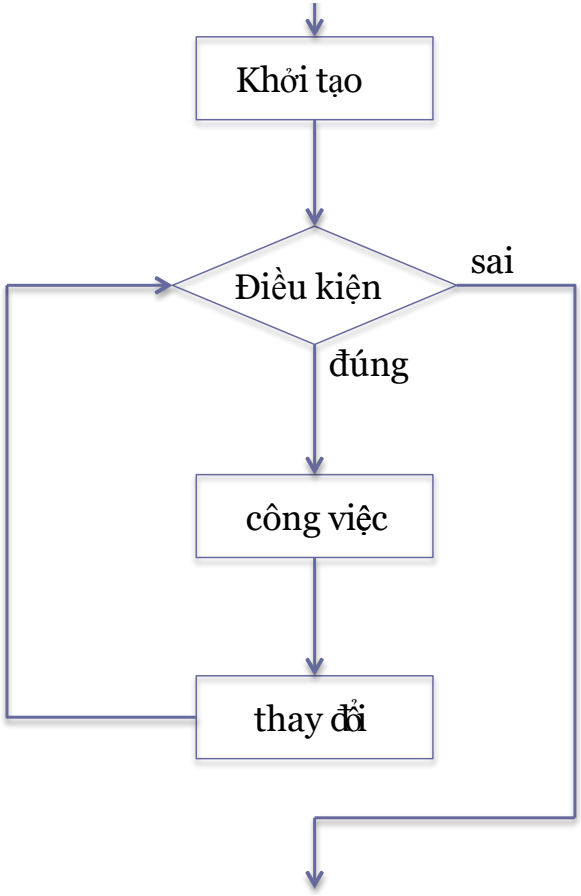
- Có hai kiểu lặp thông dụng
  - lặp sử dụng biến đếm (biết trước số lần lặp), ví dụ : nhập điểm của 120 sinh viên
  - lặp sử dụng điều kiện dừng, ví dụ : nhập mật khẩu cho đến khi nhập đúng
- Tương ứng với những kiểu lặp này, C++ cung cấp các lệnh lặp for, while và do-while

2. Vòng lặp FOR



- Khởi gán a = 10; 10 có >= 0 không? → có → in ra 10
- Bớt a đi 1 → a = 9; 9 có >= 0 không? → có → in ra 9
- ...
- Bớt a đi 1 → a = 0; 0 có >= 0 không? → có → in ra 0
- Bớt a đi 1 → a = -1 ; -1 có >= 0 không ? → không → dừng vòng lặp, không in -1 ra màn hình

Vòng lặp FOR



Cú pháp :

```
for (<khởi tạo> ; <điều kiện> ; <thay đổi>)
{
        <công việc>

}
```

Quá trình thực hiện

1. <khởi tạo> : gán giá trị ban đầu cho biến điều khiển
2. Kiểm tra <điều kiện> (biểu thức logic). Nếu sai : dừng lặp
3. Thực hiện <công việc> (một lệnh hoặc khối lệnh)
4. Thực hiện <thay đổi> (tăng hoặc giảm giá trị biến điều khiển)
5. Quay về bước 2

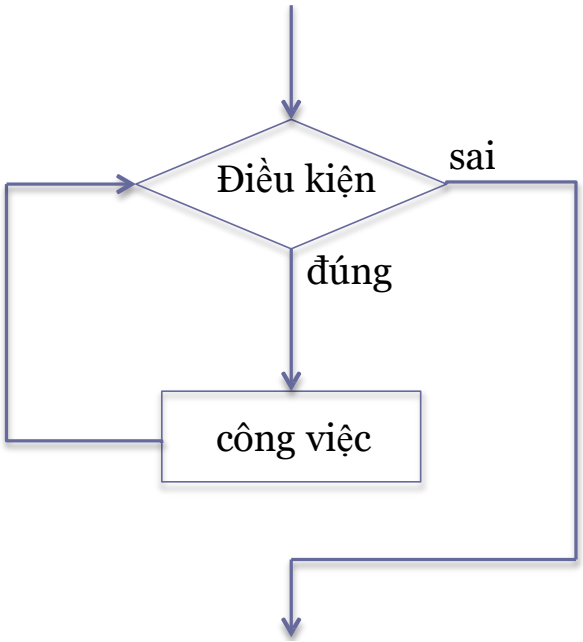
Vòng lặp FOR : ví dụ

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

int main()
{
  for ( int a = 10;  a >= 0;  a-=3)
  {
    cout << a << endl;
  }
  return 0;
}
```

Có thể tăng/giảm a với bước nhảy tùy ý

```
//tính tổng các số nguyên từ 10 đến 20
for (int i = 10, tong = 0; i <= 20; i++)
{
    tong = tong + i;
}
//vòng lặp này làm gì?
for (int i = 10, tich = 1; i <= 20; i = i+2)
{
        tich = tich * i;

}
```

3. Vòng lặp WHILE



Cú pháp :

```
while (<điều kiện>)
{
        <công việc>
}
```

Quá trình thực hiện

1. Kiểm tra <điều kiện>. Nếu sai : dừng lặp
2. Thực hiện <công việc>
3. Quay về bước 1

Vòng lặp WHILE

- Vòng lặp While được dùng khi không biết chính xác số lần lặp, chỉ biết điều kiện dừng
- Chú ý khởi tạo các biến cần thiết trước khi vào vòng lặp
- Các lệnh trong khối <công việc> có thể không được thực hiện lần nào nếu biểu thức <điều kiện> sai ngay từ đầu
- Trong <công việc> thường có ít nhất một lệnh ảnh hưởng đến giá trị của biểu thức <điều kiện>, làm cho biểu thức <điều kiện> đang đúng trở thành sai
- Lỗi hay gặp: vòng lặp vô hạn

Vòng lặp WHILE - ví dụ

```
int a = 10; while
(a >= 0)
{
        cout << a << endl;
        a--;
}
```

- Khởi tạo giá trị của a
- Khi a vẫn còn lớn hơn hoặc bằng 0 thì làm công việc sau

1. In ra màn hình giá trị của a
2. Bớt a đi 1, lệnh này ảnh hưởng đến giá trị của biểu thức <điều kiện>

Vòng lặp WHILE - ví dụ

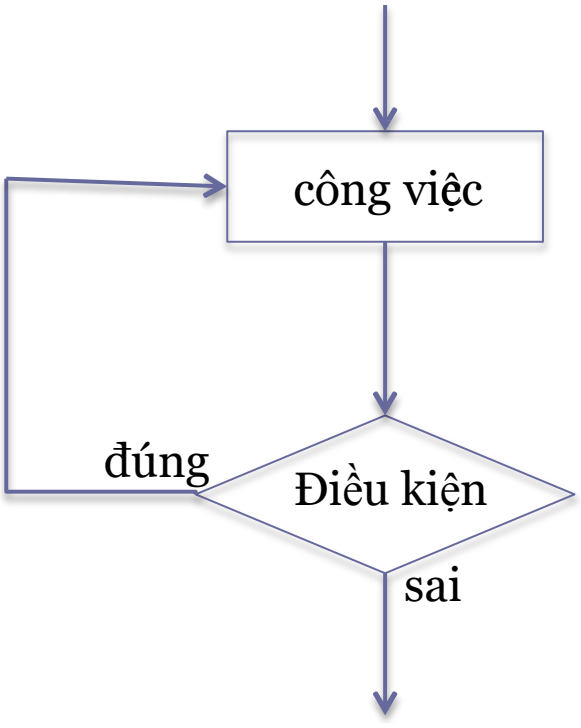
- Vòng lặp vô hạn

```
int a = 10; while (a >= 0)
   {
        cout << a << endl;
   }
```

- Chương trình sau đây làm gì?

```
int a = n-1;
   while ( (n % a) != 0 )
   {
        a = a - 1;
   }
```

4. Vòng lặp DO-WHILE



Cú pháp :

```
do
{
        <công việc>
}
while (<điều kiện>);
```

Quá trình thực hiện

1. Thực hiện <công việc>
2. Kiểm tra <điều kiện>. Nếu sai: dừng lặp
3. Quay về bước 1

Vòng lặp DO-WHILE - ví dụ

```
int a = 10;
do
{
    cout << a << endl;
    a--;
}
while (a >= 0);
```

Khác nhau giữa DO-WHILE và WHILE

- DO-WHILE: làm trước, kiểm tra điều kiện dừng sau,

  <công việc> được thực hiện ít nhất 1 lần

- WHILE: kiểm tra điều kiện dừng trước, làm sau

5. Từ khoá break và continue

- break : được dùng khi cần thoát khỏi vòng lặp
- continue : được dùng khi cần dừng bước lặp hiện tại để tiếp tục bước lặp mới

6. Bài tập

Bài 1

Nhập vào số nguyên dương n. Tính tổng các số từ 0 đến n. Viết ba hàm cùng làm công việc trên nhưng sử dụng các cấu trúc lặp khác nhau : do-while, while, for.

Bài 2

Nhập vào số nguyên dương n. Tính giá trị của biểu thức

$$X = 1 + 1/2 + 1/3 + \ldots + 1/n.$$

Bài 3

Nhập các số thực từ bàn phím cho đến khi tổng của chúng lớn hơn hoặc bằng 100 thì dừng.