



# IT001 - NHẬP MÔN LẬP TRÌNH

## CHƯƠNG 4: CẤU TRÚC ĐIỀU KHIỂN (FLOW CONTROL STRUCTURES)

Cấu trúc điều khiển (control structure) là một phần quan trọng trong ngôn ngữ lập trình C++. Nó giúp lập trình viên kiểm soát luồng thực thi của chương trình, cho phép chương trình thực hiện các hành động khác nhau tùy thuộc vào điều kiện nhất định. Cấu trúc điều khiển là một phần thiết yếu trong lập trình C++ để tạo ra các chương trình logic và linh hoạt.

Khoa Khoa học Máy tính



## NỘI DUNG

- 4.1 Khái niệm câu lệnh và khối lệnh
- 4.2 Phạm vi hoạt động của biến
- 4.3 Giới thiệu về cấu trúc điều khiển
- 4.4 Cấu trúc rẽ nhánh if, if-else
- 4.5 Cấu trúc rẽ nhánh switch-case
- 4.6 Cấu trúc vòng lặp for
- 4.7 Cấu trúc vòng lặp while
- 4.8 Cấu trúc vòng lặp do-while
- 4.9 Lệnh break, continue, goto, return
- 4.10 Một số ví dụ minh họa

### Bài tập



## 4.1 Khái niệm câu lệnh và khối lệnh



## 4.1 Khái niệm câu lệnh và khối lệnh

- **Câu lệnh:**

- Một câu lệnh (statement) xác định một công việc mà chương trình phải thực hiện để xử lý dữ liệu đã được mô tả và khai báo.
- Các câu lệnh được ngăn cách với nhau bởi dấu chấm phẩy (;).

- **Ví dụ:**

```
int n;  
  
std::cout << "Nhap vao so nguyen n = ";  
  
std::cin >> n;  
  
std::cout << "So n= " << n;
```



## 4.1 Khái niệm câu lệnh và khối lệnh

- **Khối lệnh:** Một dãy các câu lệnh được bao bởi các dấu { } gọi là một khối lệnh.

```
{  
    int n;  
    cout << "Nhập số nguyên n = ";  
    cin >> n;  
    cout << "So n= " << n;  
}
```



## **4.2 Phạm vi hoạt động của biển**



## 4.2 Phạm vi hoạt động của biến

- Biến phải được khai báo trước khi sử dụng.

```
#include <iostream>
int main() {
    std::cout << x; // Error identifier 'x' is undefined
    // Build: error C2065: 'x' : undeclared identifier
    int y=5;
    std::cout << y; // 5
}
```

- Phạm vi hoạt động của một biến chính là khối lệnh mà nó được khai báo.
- Nếu nó được khai báo trong một hàm, tầm hoạt động sẽ là hàm đó.
- Còn nếu được khai báo trong vòng lặp thì tầm hoạt động sẽ chỉ là vòng lặp đó.



## 4.2 Phạm vi hoạt động của biến

- Ví dụ:

```
#include <iostream>
using namespace std;
int x=3;
int main() {
    cout << x << endl;
    int x=5;
    {    cout << x << endl;
        int x=7;
        cout << x << endl;
        cout << ::x << endl;
    }
    cout << x << endl;
    cout << ::x << endl;
}
```



## 4.3 Giới thiệu về cấu trúc điều khiển



## 4.3 Giới thiệu về cấu trúc điều khiển

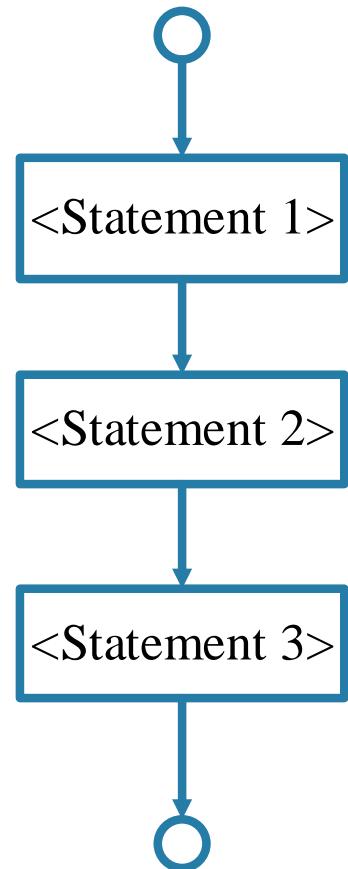
Có 3 loại cấu trúc điều khiển các lệnh cơ bản:

- **Cấu trúc tuần tự (Sequential Structure)**
- **Cấu trúc rẽ nhánh (Selection Statements): if, if-else, switch-case**
- **Cấu trúc lặp (Iteration Statements - Loops): for, while, do while**



# Cấu trúc tuần tự - Sequential Structure

- Cấu trúc tuần tự là một trong những cấu trúc lập trình cơ bản nhất và là nền tảng của nhiều chương trình lớn, phức tạp.
- Cấu trúc tuần tự cho phép thực hiện các câu lệnh theo một thứ tự cụ thể.



Các lệnh tuần tự

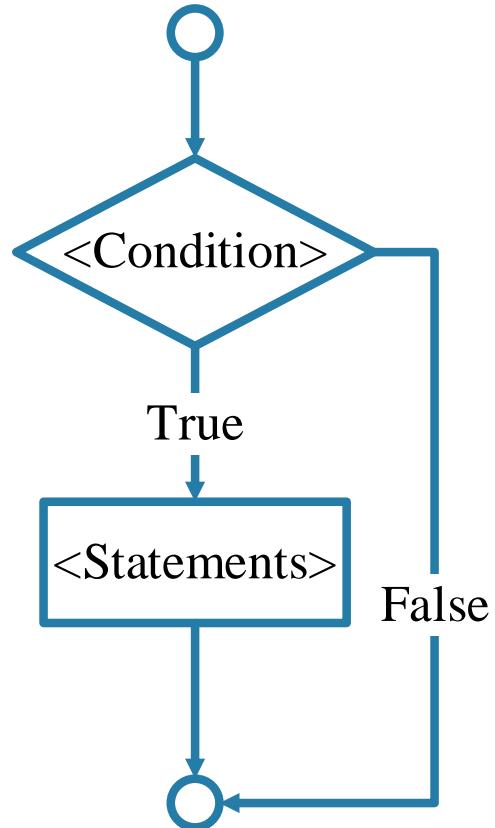


# Cấu trúc rẽ nhánh - Selection Statements

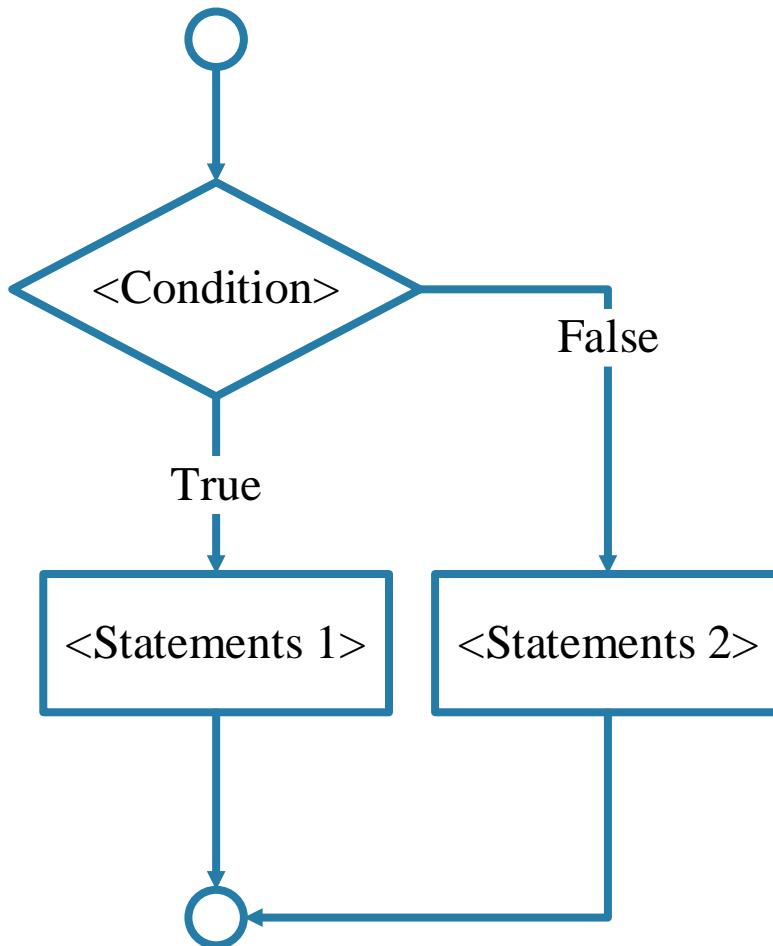
- Cấu trúc rẽ nhánh dùng để diễn đạt một việc sẽ được thực hiện khi một điều kiện cụ thể được thỏa mãn.
- Cấu trúc rẽ nhánh chia làm hai loại:
  - Cấu trúc rẽ một trong hai nhánh: như cấu trúc *if*, *if...else* và toán tử điều kiện (*? :*)
  - Cấu trúc rẽ một, hai hoặc nhiều nhánh: cấu trúc *switch-case*



# Cấu trúc rẽ nhánh



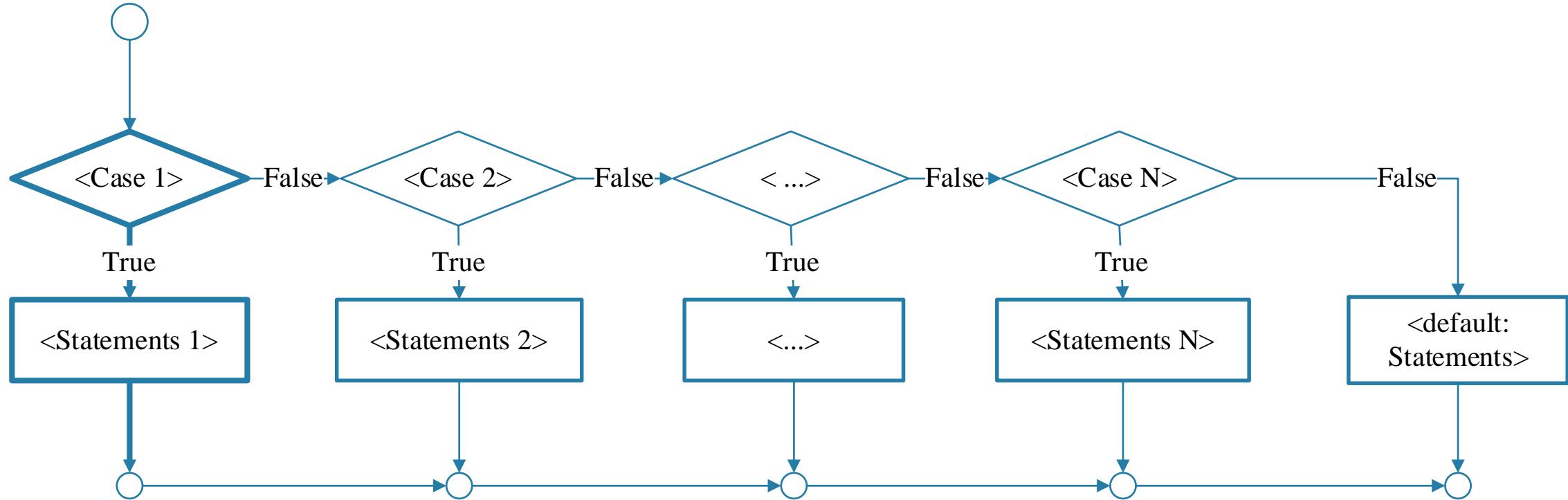
Cấu trúc điều kiện  
rẽ nhánh **if**



Cấu trúc điều kiện  
rẽ nhánh **if-else**



# Cấu trúc rẽ nhánh

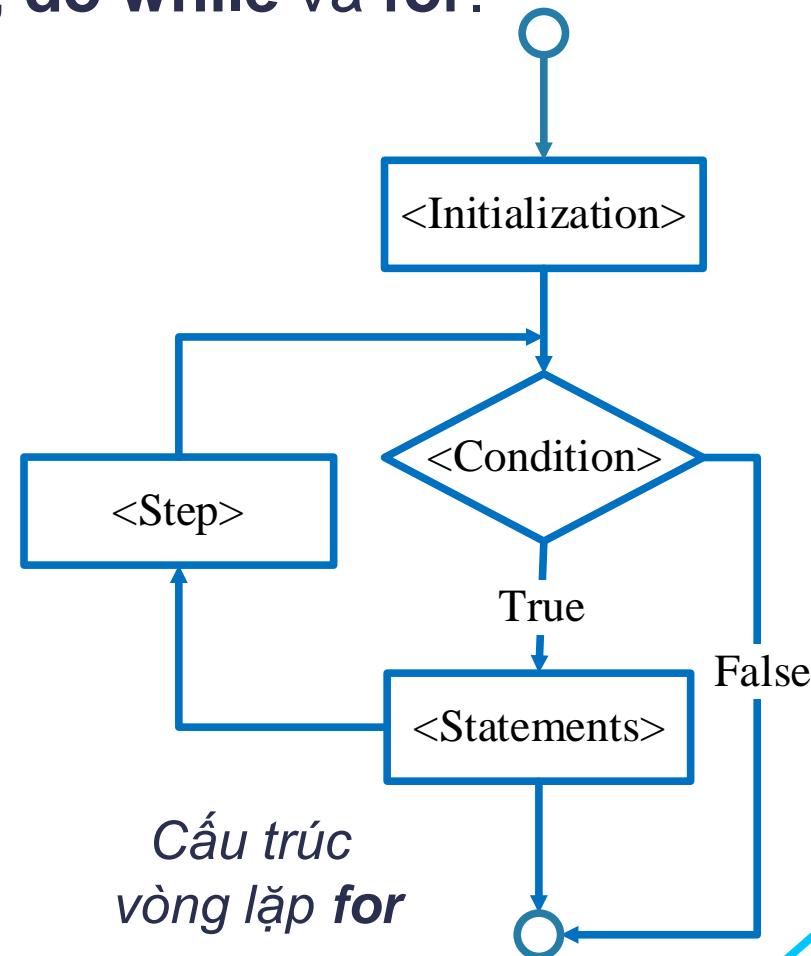
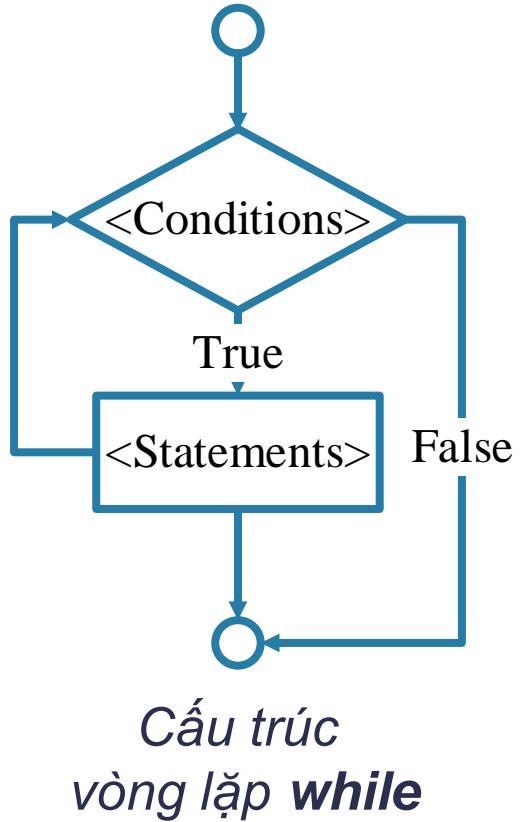
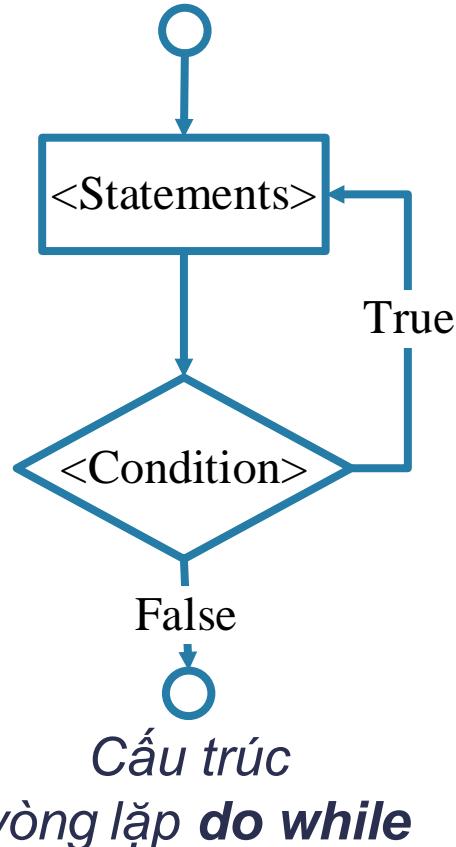


*Cấu trúc điều kiện rẽ nhánh **switch-case***



# Cấu trúc lặp - Iteration statements (Loops)

- Cấu trúc lặp: là việc lặp lại một câu lệnh một số lần nhất định hoặc thực hiện đến khi thỏa một điều kiện dừng, gồm: **while**, **do while** và **for**.





## 4.4 Cấu trúc rẽ nhánh if, if-else

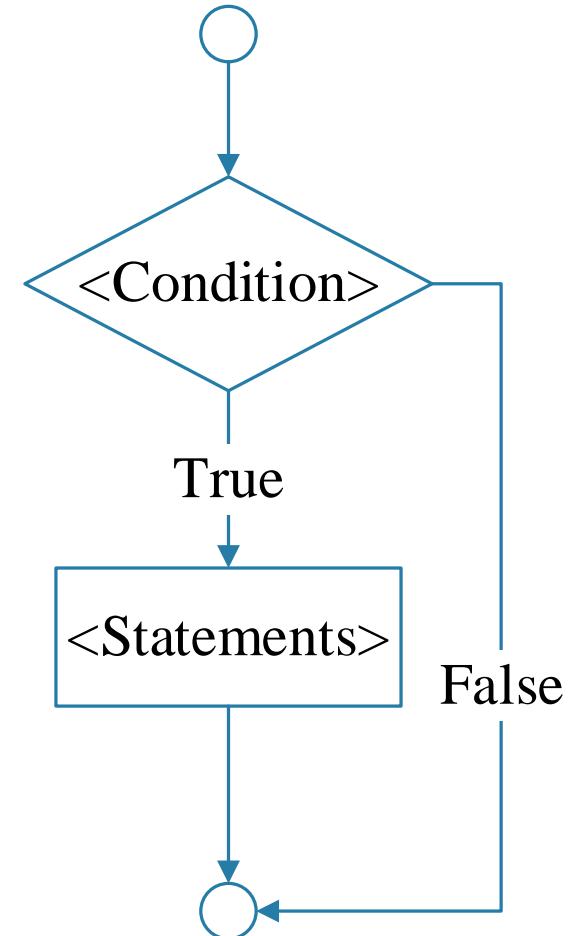


# Cấu trúc rẽ nhánh if

- Cú pháp:

```
if (Condition == True)  
    Statements;
```

Trong đó: Lệnh *Statements* có thể là lệnh  
đơn hay khối lệnh



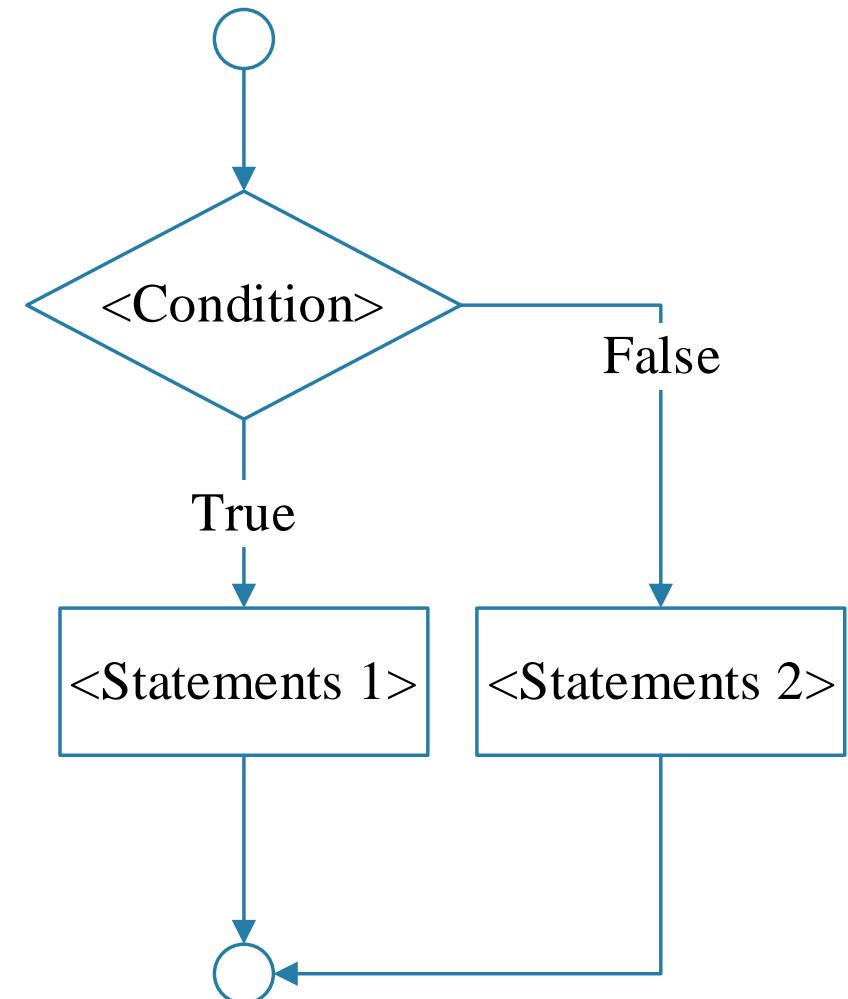
Lưu đồ: Cấu trúc điều kiện  
rẽ nhánh if



# Cấu trúc rẽ nhánh if-else

- Cú pháp:

```
if (Condition==true)
    Statements 1;
else
    Statements 2;
```



Lưu đồ: Cấu trúc điều kiện if else



# Ví dụ

- Viết chương trình tìm giá trị bé nhất của ba số a, b, c cho trước.

```
#include <iostream>
using namespace std;
int main(){
    int a = 10, b = 15, c= 8;
    int min;

    //Cách 1
    min = a;
    if (b < min)
        min=b;
    if (c < min)
        min=c;

    cout << "Gia tri be nhat la " << min;
}
```

```
//Cách 2
if (a<b)
    if (a<c) min=a;
    else min=c;
else
    if (b<c) min=b;
    else min=c;
```

```
//Cách 3
min = (a<b) ? ((a<c) ? a:c) : ((b<c) ? b:c);
```



# Lưu ý

- Không được thêm ; sau điều kiện của if.

```
if(a==0);    → SAI
```

```
cout << "a bang 0";
```

- Câu lệnh if và câu lệnh if... else là một câu lệnh đơn.

```
{  
    if(a==0) cout << "a bang 0";  
}  
{  
    if(a==0) cout << "a bang 0";  
    else cout << "a khac 0";  
}
```



# Lưu ý

- Câu lệnh **if** có thể lồng vào nhau
- **else** sẽ tương ứng với **if** gần nó nhất.

```
if (a != 0)
    if (b > 0) cout << "a != 0 va b > 0";
else cout << "a != 0 va b <= 0";
```



```
if (a != 0) {
    if (b > 0) cout << "a != 0 va b > 0";
    else cout << "a != 0 va b <= 0";
}
```



# Lưu ý

- Nên dùng **else** để loại trừ trường hợp.

```
if (delta < 0)
    cout << "VN";
if (delta == 0)
    cout << "Nghiem kep";
if (delta > 0)
    cout << "Co 2 nghiem phan biet";
```

```
if (delta < 0)
    cout << "VN";
else // delta >=0
    if (delta == 0)
        cout << "Nghiem kep";
    else // delta > 0
        cout << "Co 2 nghiem phan biet";
```

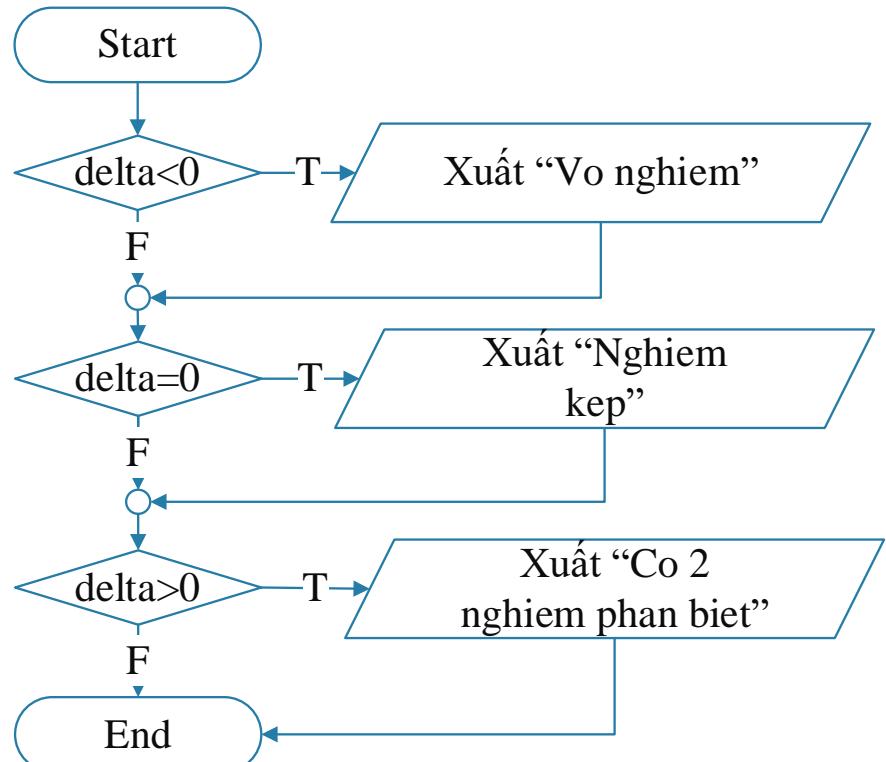
Hãy nêu điểm khác nhau giữa 3 đoạn code này?

```
if (delta < 0)
    cout << "VN";
else if (delta == 0)
    cout << "Nghiem kep";
else cout << "Co 2 nghiem phan biet";
```

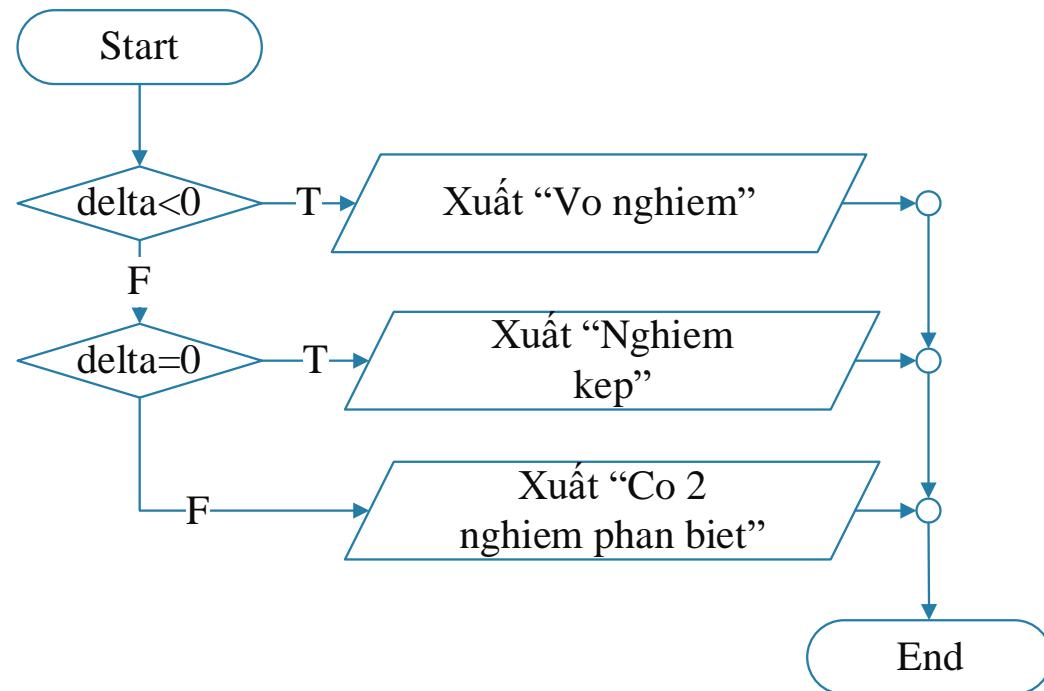


# Lưu ý: Vẽ lưu đồ các lệnh if đơn và lệnh if-else

```
if (delta < 0)
    cout << "VN";
if (delta == 0)
    cout << "Nghiem kep";
if (delta > 0)
    cout << "Co 2 nghiem phan biet";
```



```
if (delta < 0)
    cout << "VN";
else if (delta == 0)
    cout << "Nghiem kep";
else
    cout << "Co 2 nghiem phan biet";
```





# Lưu ý

- Thực hiện 1 trong n quyết định:

```
double toan, ly, hoa;  
cin >> toan >> ly >> hoa;  
double DTB = (toan+ly+hoa)/3;  
DTB = round(DTB*100)/100;  
if (9 <= DTB && DTB <= 10) cout<<"XUAT SAC";  
else if (8 <= DTB && DTB < 9) cout<<"GIOI";  
else if (7 <= DTB && DTB < 8) cout<<"KHA";  
else if (6 <= DTB && DTB < 7) cout<<"TB KHA";  
else if (5 <= DTB && DTB < 6) cout<<"TB";  
else if (4 <= DTB && DTB < 5) cout<<"YEU";  
else cout<<"KEM";
```



# Ví dụ:

- Cho biết kết quả đoạn code sau:

```
#include <iostream>  
using namespace std;  
int main() {  
    bool a=true;  
    if (!a)  
        cout << "in";  
    else  
        cout << "out";  
    return 0  
}
```

Câu điều kiện ( $!a$ ) tương đương ( $a==false$ )



## Ví dụ:

- Cho biết kết quả đoạn lệnh sau:

```
#include <iostream>
using namespace std;
int x=3;
int main() {
    int i=5;
    int n=20;
    if(i<10 && ++i<n)
        cout << n << endl;
    cout << i << endl;
    i=5;
    if(i<10 || ++i<n)
        cout << n << endl;
    cout << i << endl;
}
```

Kết quả:

```
20
6
20
5
```

- Cho biết kết quả đoạn code sau:

```
#include <iostream>
using namespace std;
int main() {
    int a = 0, b = 2, c;
    if (a = b % 2)
        c = 2;
    else
        c = 3;
    cout << a << b << c;
    return 0;
}
```

Kết quả:

```
023
```



# Ví dụ:

- Cho biết kết quả đoạn code sau:

```
#include <iostream>
using namespace std;
int main() {
    int a = 5, b=10, c=15;
    if (++a==6 || ++b!=11 || ++c!=15)
        cout << "in" << endl;
    else
        cout << "out" << endl;
    cout << a << " " << b << " " << c;
    return 0;
}
```

Kết quả thực thi:

```
in
6 10 15
```



## 4.5 Cấu trúc rẽ nhánh switch - case



## 4.5 Cấu trúc rẽ nhánh switch - case

- Cú pháp:

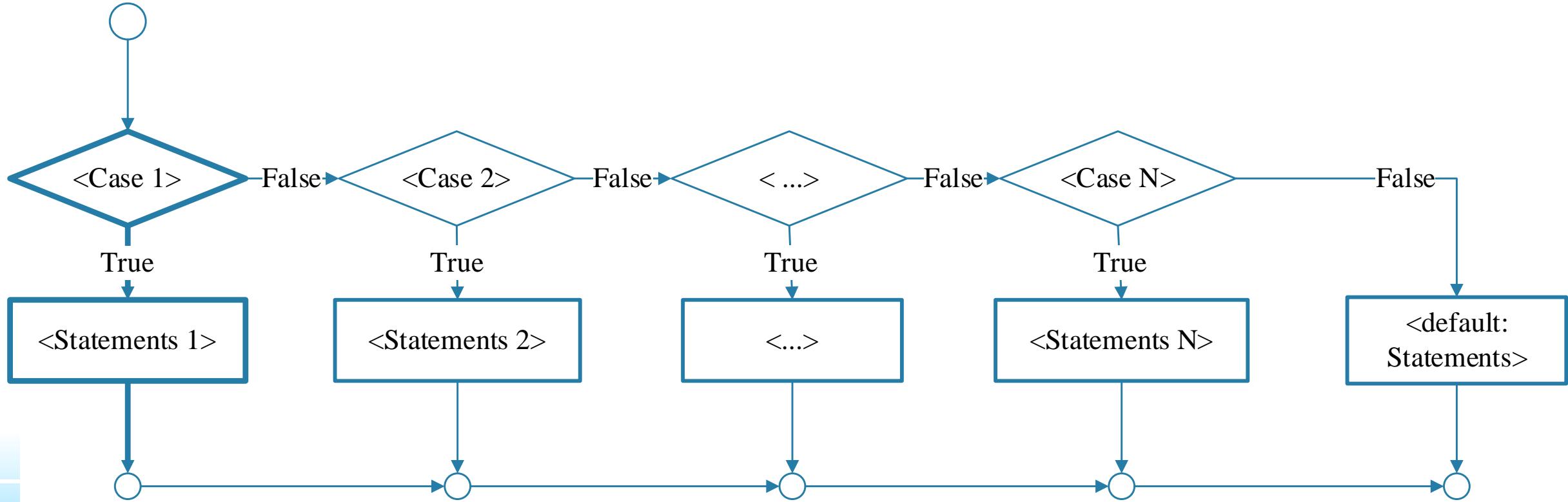
```
switch (expression) {  
    case constant1:  
        Statements 1;  
        break;  
    case constant2:  
        Statements 2;  
        break;  
    ...  
    default:  
        default Statements;  
}
```

- Trong đó:

- **expression**: là một biểu thức số học nhận giá trị nguyên
- **constant1, constant2, ...** : số nguyên, hằng ký tự, biểu thức hằng (các giá trị hằng số khác nhau tương ứng với từng case khác nhau)
- Nhánh **default** là nhánh lựa chọn mặc định khi không có nhánh nào khác được chọn. Nhánh này là không bắt buộc phải có.
- **Statements 1, Statements 2, ... , default Statements**: Tập các câu lệnh
- **break**: thoát khỏi switch sau khi thực hiện xong câu lệnh trước nó



## 4.5 Cấu trúc rẽ nhánh switch - case



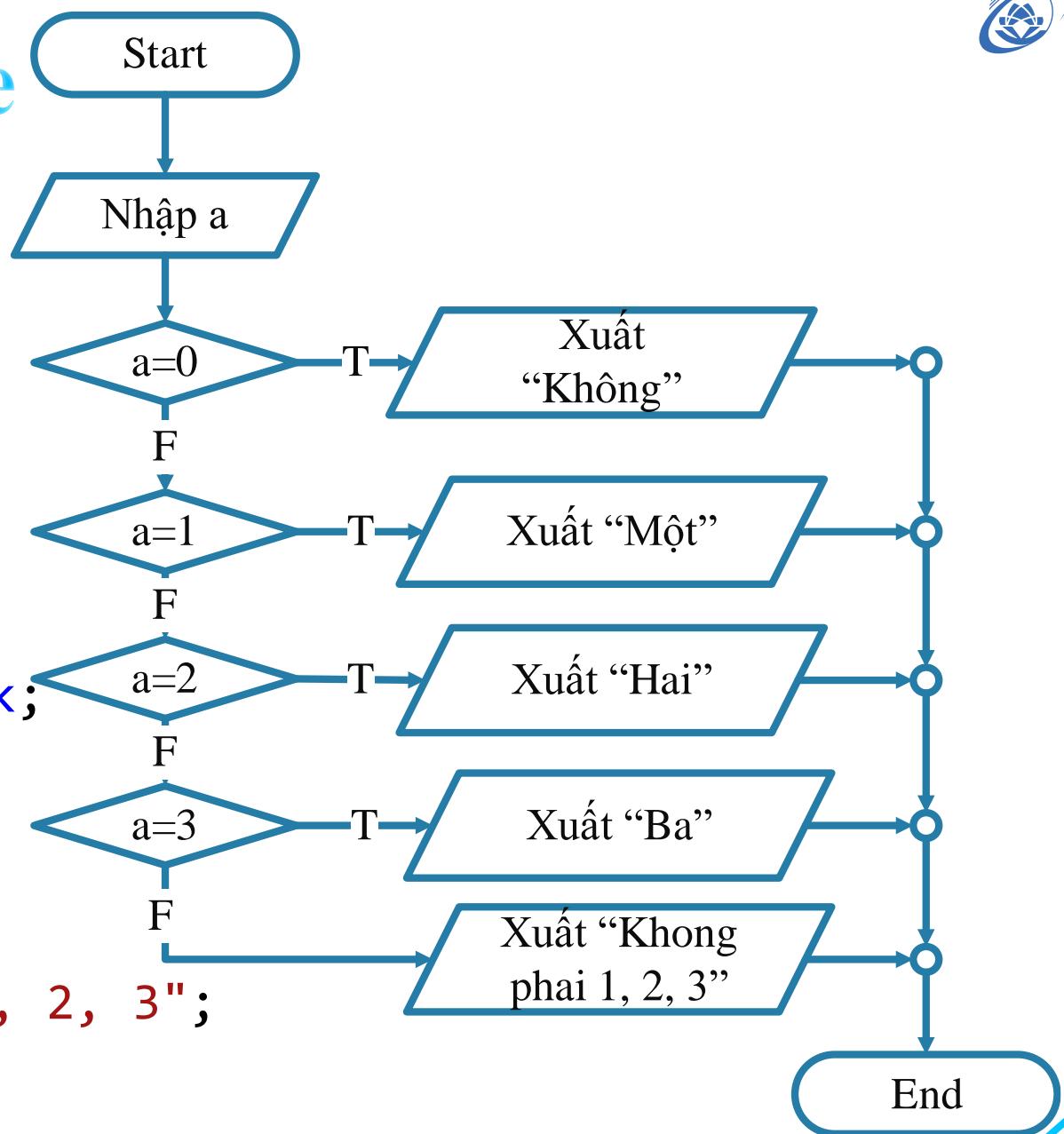
Lưu đồ: Cấu trúc rẽ nhánh switch - case



# Ví dụ 1: dùng switch-case

```
#include <iostream>
int main() {
    int a;
    std::cout << "Nhập a: ";
    std::cin >> a;

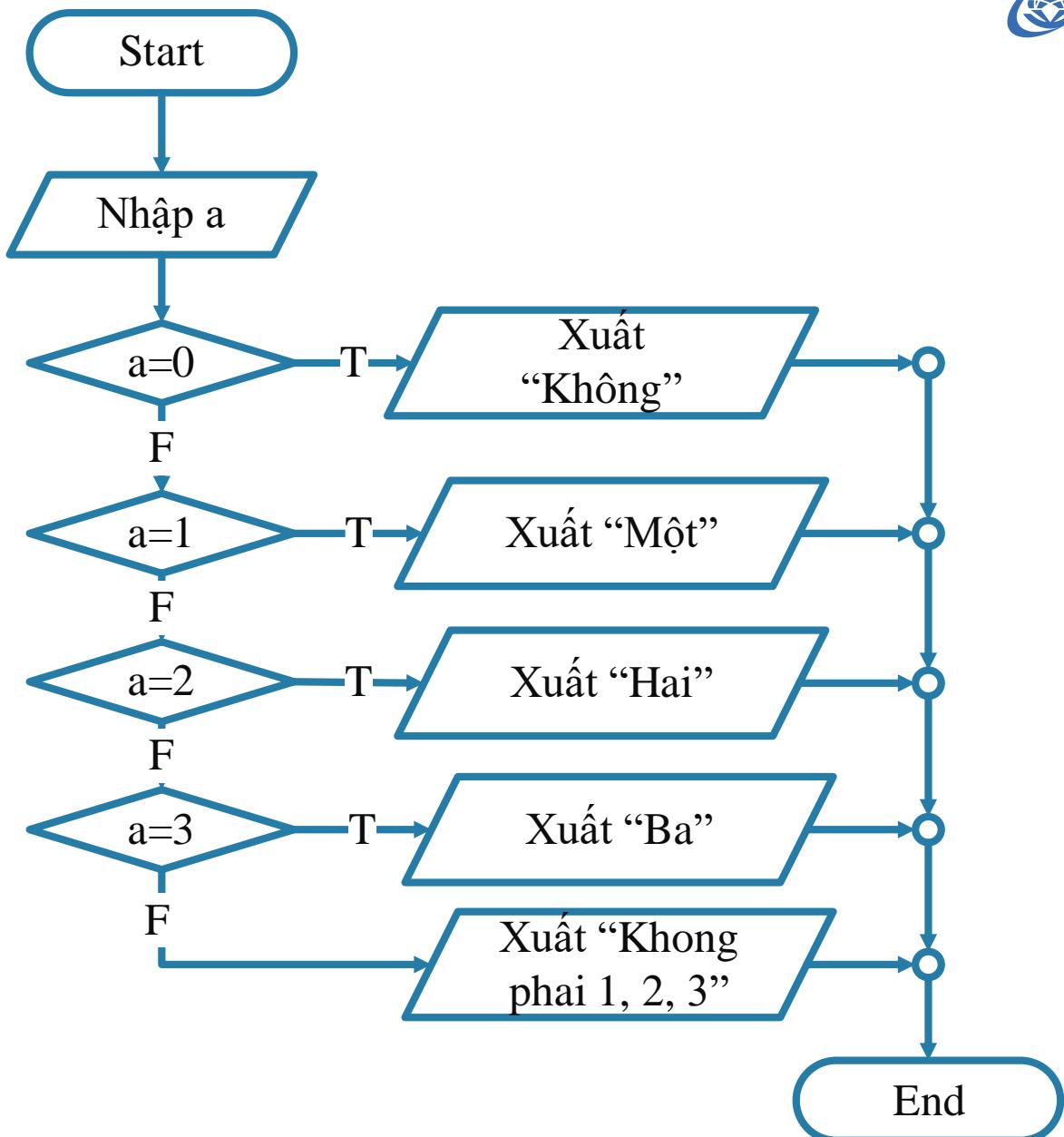
    switch (a) {
        case 0: std::cout << "Khong"; break;
        case 1: std::cout << "Mot"; break;
        case 2: std::cout << "Hai"; break;
        case 3: std::cout << "Ba"; break;
        default: std::cout << "Khong phai 1, 2, 3";
    }
}
```





# Ví dụ 1: dùng if- else

```
#include <iostream>
int main() {
    int a;
    std::cout << "Nhập a: ";
    std::cin >> a;
    if(a==0)
        std::cout << "Khong";
    else if(a==1)
        std::cout << "Mot";
    else if(a==2)
        std::cout << "Hai";
    else if(a==3)
        std::cout << "Ba";
    else
        std::cout << "Khong phai 1, 2, 3";
    return 0;
}
```





## Ví dụ 2

```
#include <iostream>
int main() {
    enum PLAYER_COMMANDS { Off, On, Stop, Play, Pause, Eject };
    switch (Eject) {
        case Off:    std::cout << "Off";      break;
        case On:     std::cout << "On";       break;
        case Stop:   std::cout << "Stop";     break;
        case Play:   std::cout << "Play";     break;
        case Pause:  std::cout << "Pause";    break;
        default:    std::cout << "Eject";
    }
    return 0;
}
```



## Ví dụ 3

```
#include <iostream>
using namespace std;
int main() {
    char ch;
    cout << "Nhập giá trị ch=";
    cin >> ch;
    switch (ch) {
        case 'a': cout << "Ký tự a đã được nhập"; break;
        case 'b': cout << "Ký tự b đã được nhập"; break;
        default : cout << "Ký tự khác a và b đã được nhập";
    }
}
```



# Lưu ý 1: Case label

- Các giá trị trong mỗi trường hợp phải **khác nhau**, nếu không sẽ phát sinh lỗi.

```
switch (a){  
    case 1 : cout << "Mot"; break;  
    case 1 : cout << "MOT"; break;  
    case 2 : cout << "Hai"; break;  
    case 3 : cout << "Ba"; break;  
    case 1 : cout << "1"; break;  
    case 1 : cout << "mot"; break;  
    default : cout << "Khong biet doc";  
}
```

➔ PHÁT SINH LỖI



## Lưu ý 2: Lệnh break

- switch sẽ nhảy đến case tương ứng và thực hiện đến khi nào gặp **break** hoặc cuối switch sẽ kết thúc.

```
int a=1;
switch (a){
    case 1 : cout << "Mot";
    case 2 : cout << "Hai";
    case 3 : cout << "Ba";
}
```

Kết quả: MotHaiBa

```
int a=1;
switch (a) {
    case 1 : cout << "Mot"; break;
    case 2 : cout << "Hai"; break;
    case 3 : cout << "Ba"; break;
}
```

Kết quả: Mot



# Lưu ý 3: Tận dụng tính chất khi bỏ break

- Ví dụ 1:

```
switch (a) {  
    case 1 : cout << "So le";  
              break;  
    case 2 : cout << "So chan";  
              break;  
    case 3 : cout << "So le";  
              break;  
    case 4 : cout << "So chan";  
              break;  
}
```

```
switch (a) {  
    case 1 :  
    case 3 : cout << "So le";  
              break;  
    case 2 :  
    case 4 : cout << "So chan";  
              break;  
}
```



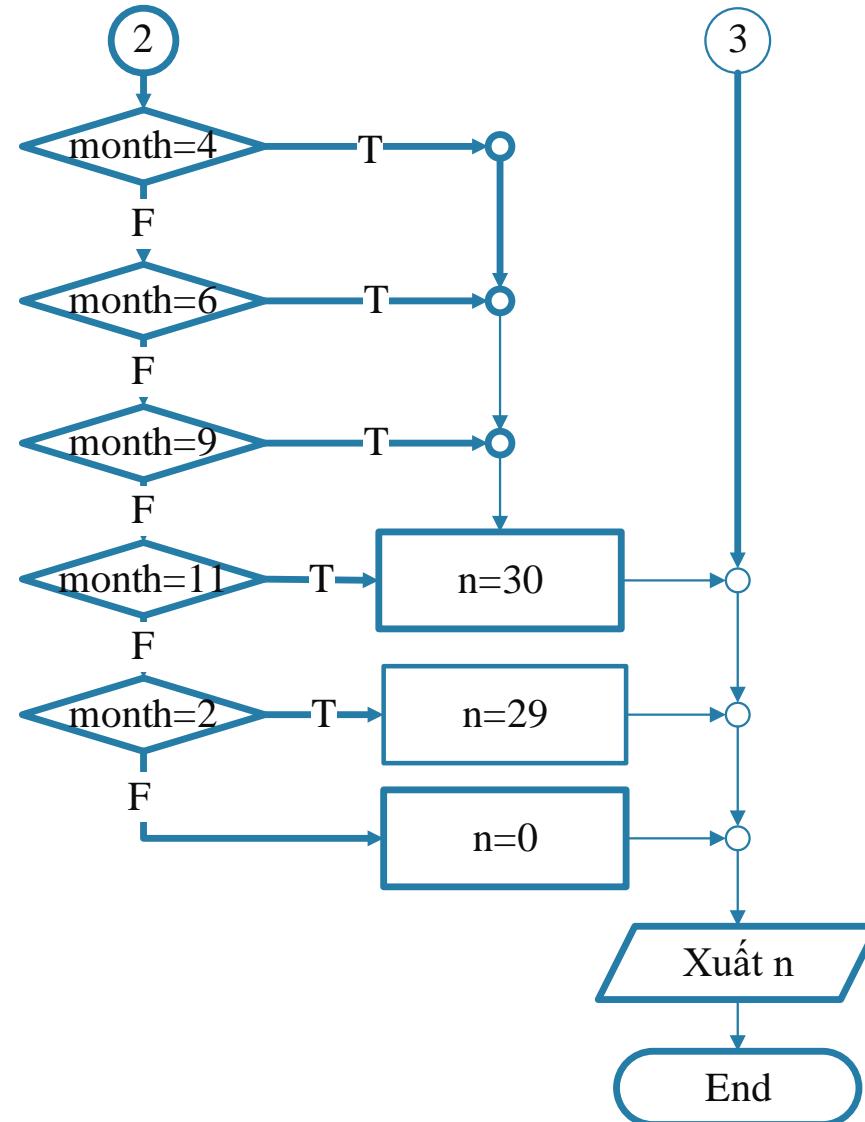
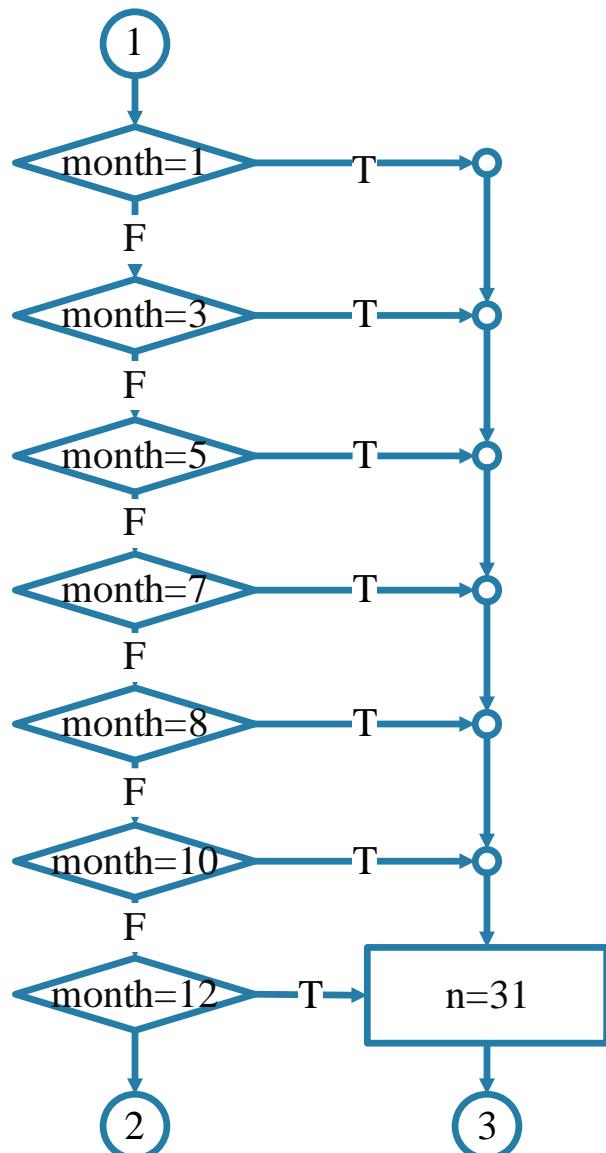
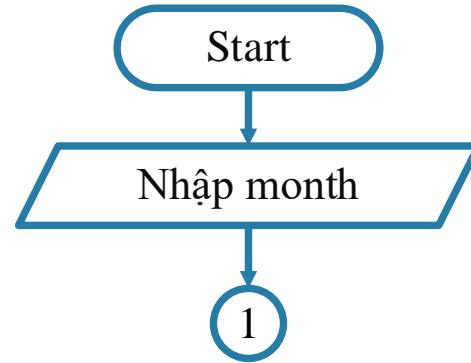
# Lưu ý 3: Tận dụng tính chất khi bỏ break

- Ví dụ 2: Xuất số ngày của tháng trong năm.

```
#include <iostream>
int main() {
    int n, month; cin >> month;
    switch (month) {
        case 1: case 3: case 5: case 7: case 8: case 10: case 12:
            n = 31; break;
        case 4: case 6: case 9: case 11:
            n = 30; break;
        case 2: n = 29; break;
        default: n=0;
    }
    std::cout << "Thang " << month << " co " << n << " ngay.";
}
```



# Lưu ý 3: Tận dụng tính chất khi bỏ break



Lưu đồ: Xuất số ngày  
của tháng trong năm



# Lưu ý 4: default có thể đặt ở bất kỳ vị trí nào trong switch

- Cú pháp:
- Trong đó: lệnh **break** phải được đặt sau khi thực hiện xong các lệnh trong **default**.

```
switch (expression) {  
    default:  
        default Statements;  
        break;  
    case constant1:  
        Statements 1;  
        break;  
    case constant2:  
        Statements 2;  
        break;  
    ...  
}
```

# Lưu ý 4: default có thể đặt ở bất kỳ vị trí nào trong switch



- Ví dụ:

```
#include <iostream>
using namespace std;
int main() {
    char ch;
    cin >> ch;
    switch (ch) {
        default : cout << "Ky tu khac a va b."; break;
        case 'a': cout << "Ky tu a."; break;
        case 'b': cout << "Ky tu b."; break;
    }
}
```



# Lưu ý 5: Câu lệnh switch là một câu lệnh đơn và có thể lồng nhau

- Ví dụ:

```
switch (a){  
    case 1 : cout << "Mot"; break;  
    case 2 : switch (b) {  
        case 1 : cout << "A"; break;  
        case 2 : cout << "B"; break;  
    }  
    break;  
    case 3 : cout << "Ba"; break;  
    default: cout << "Khong biet doc";  
}
```



# Lưu ý 6: Switch case hoạt động theo case label, không quan tâm scope

- Ví dụ 1:

```
#include <iostream>

int main() {
    switch (3) {
        case 1: std::cout << "case 1." << std::endl; break;
        case 2: {
            std::cout << "case 2." << std::endl; break;
        case 3: std::cout << "case 3." << std::endl; break;
    }
    default: std::cout << "default." << std::endl;
}
return 0;
}
```

Kết quả:

case 3.



# Lưu ý 7: Khai báo biến trong switch

- Ví dụ 2:

```
#include <iostream>
int main() {
    switch (4) {
        int x; // ✓, khai báo được phép trước gọi case label
        case 1:
            int y; // Ok, các case (cả default) bên dưới
                    // có thể sử dụng biến này mà không cần khai báo lại
            std::cout << "case 1."; break;
        case 2: std::cout << "case 2. "; break;
        case 4:
            y = 7; // ok, y đã được khai báo ở trên
            std::cout << "case 4: " << y; break;
        default: std::cout << "default. ";
    }
}
```

→ Khai báo biến trong một case có thể được dùng cho các case khác bên dưới kể cả khi case chưa biến khai báo không được thực thi.



## Lưu ý 7: Khai báo biến trong switch

- Bởi vì các câu lệnh trong mỗi case không nằm trong một khối lệnh, nên tất cả các câu lệnh bên trong switch là một phần của cùng một scope. Do đó, một biến được định nghĩa trong một case có thể được sử dụng trong case khác, ngay cả khi case mà biến được định nghĩa không bao giờ được thực thi!



# Lưu ý 7: Khai báo biến trong switch

- Ví dụ 3:

```
#include <iostream>
int main() {
    switch (2) {
        case 1:
            std::cout << "case 1."; break;
        case 2:
            int i2 = 2; // Báo lỗi
            std::cout << "case 2. " << i2; break;
        default:
            int i = 4; // Ok, khởi tạo biến được
                        // chấp nhận ở case cuối cùng
            std::cout << "default. " << i;
    }
}
```

→ Khởi tạo biến trực tiếp trong mỗi case không được phép và sẽ phát sinh lỗi. Lý do: Vì việc khởi tạo giá trị yêu cầu phải thực thi, nhưng case chưa lệnh khởi tạo có thể sẽ không bao giờ được thực thi!



# Lưu ý 7: Khai báo biến trong switch

- Ví dụ 4:

```
#include <iostream>
int main() {
    switch (2) {
        case 1:
            std::cout << "case 1."; break;
        case 2: {
            int i2 = 2; // okay, variables can be initialized
                        // inside a block inside a case
            std::cout << "case 2. " << i2; break;
        }
        default:
            int i = 4; // Ok, khởi tạo biến được
                        // chấp nhận ở case cuối cùng
            std::cout << "default. " << i;
    }
}
```

➔ Nếu khởi tạo biến được dùng trong 1 case thì cần phải đặt nó vào 1 khối lệnh trong case (hoặc trước câu lệnh switch)



# Bài tập



# Bài tập

1. Viết chương trình Nhập vào 3 số thực a, b, c. In ra theo thứ tự tăng dần.
2. Viết chương trình Giải phương trình bậc hai  $ax^2+bx+c=0$
3. Viết chương trình Xác định học lực của SV dựa trên điểm trung bình dùng switch-case
4. Viết chương trình Nhập ba số a, b, c. Kiểm tra xem chúng có thể là độ dài của các cạnh của một tam giác hay không. Nếu có thì cho biết đó là tam giác gì: NHON, VUONG, TU?
5. Viết chương trình Tính tiền đi taxi từ số km nhập vào. Biết: 1 km đầu giá 15000đ, từ km thứ 2 đến km thứ 5 giá 13500đ, từ km thứ 6 trở đi giá 11000đ, nếu trên 120km được giảm 10% trên tổng số tiền.
6. Viết chương trình Nhập vào tháng và năm (năm > 1975), kiểm tra tính hợp lệ của tháng, năm và cho biết tháng đó có bao nhiêu ngày.



# Chúc các em học tốt !

