NHẬP MÔN LẬP TRÌNH

CHƯƠNG 4: CẤU TRÚC ĐIỀU KHIỂN-FLOW CONTROL STRUCTURES

4.1 CẤU TRÚC RẼ NHÁNH-SELECTION STRUCTERS

ThS. Nguyễn Thị Ngọc Diễm
diemntn@uit.edu.vn

CĐR buổi học

- Sau khi học xong buổi học, sinh viên có khả năng:
  - Hiểu và vận dụng được các cấu trúc điều khiển để viết được chương trình trên máy tính

Nội dung

1. Khái niệm câu lệnh và khối lệnh trong lập trình
2. Phạm vi hoạt động của biến trong các khối lệnh
3. Giới thiệu về cấu trúc điều khiển
4. Cấu trúc rẽ nhánh - Selection statements
    1. Cấu trúc rẽ nhánh if
    2. Cấu trúc rẽ nhánh if-else
    3. Cấu trúc rẽ nhánh switch-case
5. Một số ví dụ minh họa

1. Khái niệm câu lệnh và khối lệnh trong lập trình

- 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 (;).
  - VD:

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

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

```
{
  int n;
  cout << "Nhap so nguyen n = ";
  cin >> n;
  cout << "So n= “ << n;
}
```

2. Phạm vi hoạt động của biến trong các khối lệnh

- Tất cả các biến mà chúng ta sẽ sử dụng đều phải được khai báo trước.

```
#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.

2. Phạm vi hoạt động của biến trong các khối lệnh

- Ví dụ:

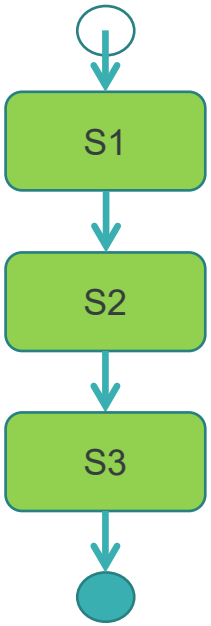
```
#include <iostream>
using namespace std;
int x=3;
int main() {
  cout << x << endl; // 3
  int x=5;
  {
     cout << x << endl; // 5
     int x=7;
     cout << x << endl; // 7
     cout << ::x << endl; // 3
  }
  cout << x << endl; // 5
  cout << ::x << endl; //3
}
```

3. Giới thiệu về cấu trúc điều khiển

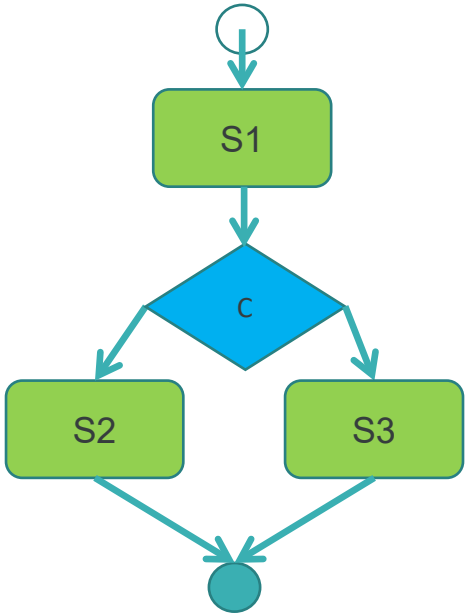
Có 3 loại cấu trúc điều khiển các lệnh cơ bản:

- **Cấu trúc tuần tự - Sequence structure**: Là cách tổ chức các lệnh thành từng khối. Phần cấu trúc khối lệnh đã được trình bầy trong chương 1.
- **Cấu trúc rẽ nhánh - Selection structure**: có các cấu trúc *if và switch.*
- **Cấu trúc lặp – Iteration/Loop/repetition structure**: có các cấu trúc *for, while, do while.*

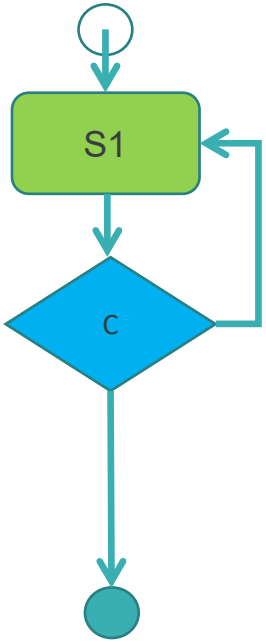
3. Giới thiệu về cấu trúc điều khiển



Sequence structure



Selection structure



Iteration structure

4. Các cấu trúc rẽ nhánh Selection structure

- Cấu trúc rẽ nhánh có thể 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.*
- Trong hai cấu trúc này thì cấu trúc hai nhánh tổng quát hơn vì nó có thể áp dụng cho mọi loại biểu thức điều kiện rẽ nhánh và cấu trúc này cho phép lồng nhau để tạo thành các cấu trúc rẽ nhiều nhánh. Còn cấu trúc rẽ nhiều nhánh switch chỉ có thể áp dụng với biểu thức điều kiện rẽ nhánh với các giá trị rời rạc.

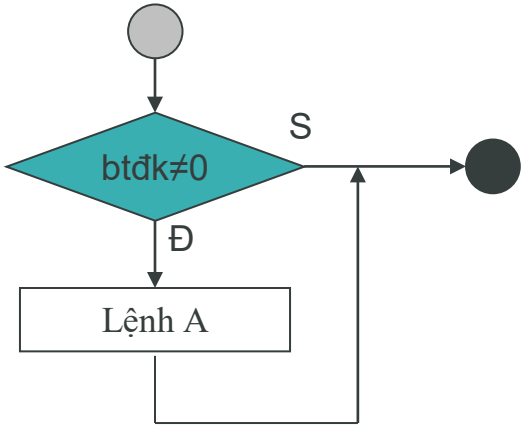
4.1 Cấu trúc rẽ nhánh if

- Cú pháp:

```
if (biểu thức điều kiện- btđk)
    Lệnh A;
```

Trong đó: Lệnh A có thể là lệnh đơn hay khối lệnh

- Lưu đồ:

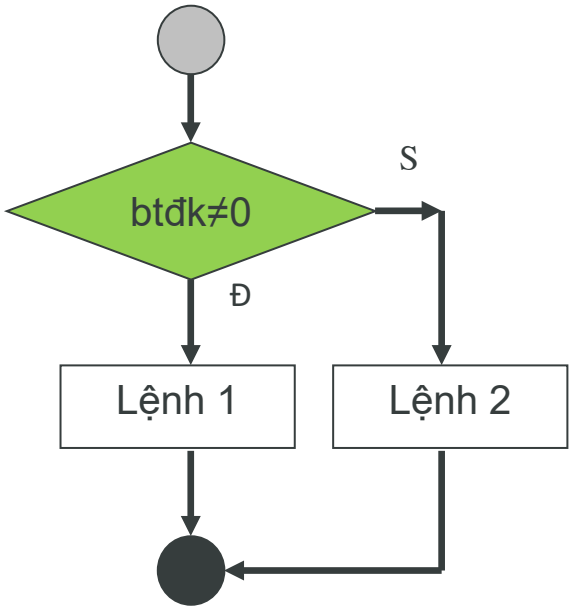


4.2 Cấu trúc rẽ nhánh if..else..

- Cú pháp:

```
if (btđk)
    Lệnh 1;
else
    Lệnh 2;
```

- Lưu đồ:



Ví dụ minh hoạ

- 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;
```

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

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

```
    cout << "Gia tri be nhat là " << min;
} // end main
```

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";
```

? SV hãy nêu điểm khác nhau

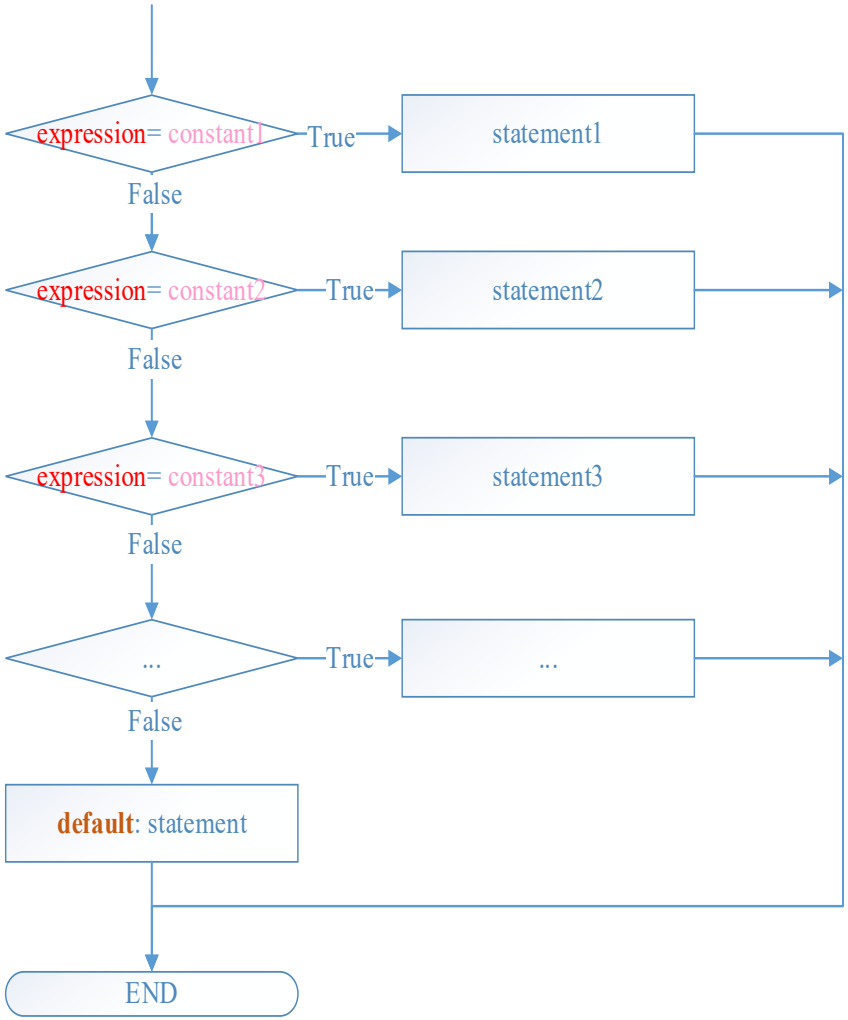
```
if (delta < 0)
        cout << "VN";
else // delta >=0
        if (delta == 0)
                cout << "Nghiem kep";
        else // delta > 0
                cout << "Co 2 nghiem phan
biet";
```

4.3 Cấu trúc rẽ nhánh switch..case..

Cú pháp:

```
switch (biểu thức điều kiện) {
    case hằng số 1 : Lệnh 1; break;
    case hằng số 2 : Lệnh 2; break;
    …
    [default : Lệnh default]
}
```

- **Biểu thức điều kiện:** là một biểu thức số học nhận giá trị nguyên.
- Hằng số 1, hằng số 2,…: số nguyên, hằng ký tự, biểu thứ 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.
- Câu lệnh **break**; thoát khỏi switch sau khi thực hiện xong câu lệnh trước nó.



Lưu ý

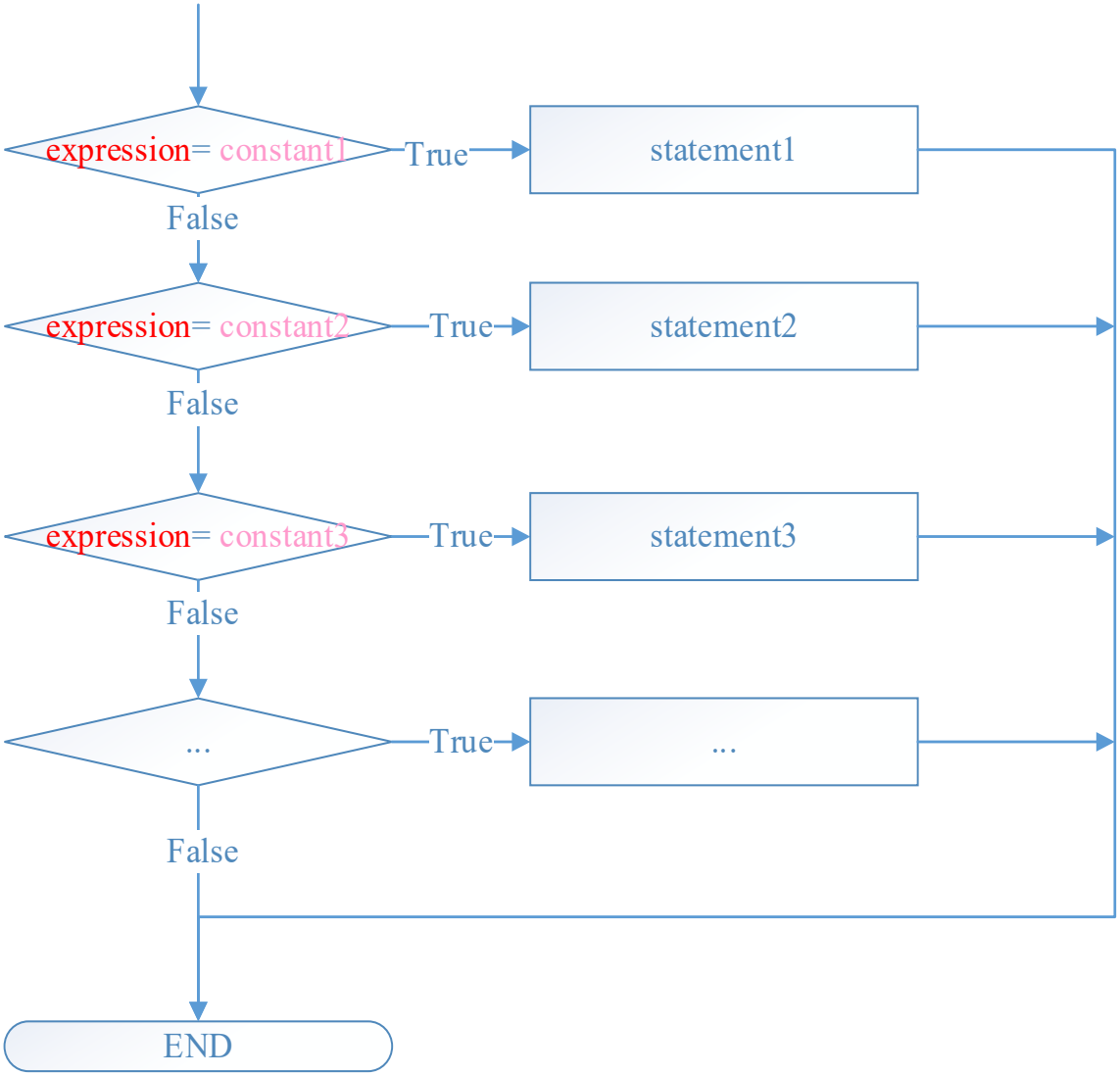
- Thực hiện 1 trong n quyết định

```
if (biểu thức 1)
        Khối lệnh 1;
else if (biểu thức 2)
        Khối lệnh 2;
...
else if (biểu thức n - 1 )
        Khối lệnh n - 1;
else
        Khối lệnh n;
```

```
if (balance > 50000.00)
    interestRate = 0.07;
else
    if (balance >= 25000.00)
        interestRate = 0.05;
    else
        if (balance >= 1000.00)
            interestRate = 0.03;
        else
            interestRate = 0.00;
```

```
if (balance > 50000.00)
    interestRate = 0.07;
else if (balance >= 25000.00)
    interestRate = 0.05;
else if (balance >= 1000.00)
    interestRate = 0.03;
else
    interestRate = 0.00;
```

Lưu đồ cho câu lệnh switch không có nhánh default

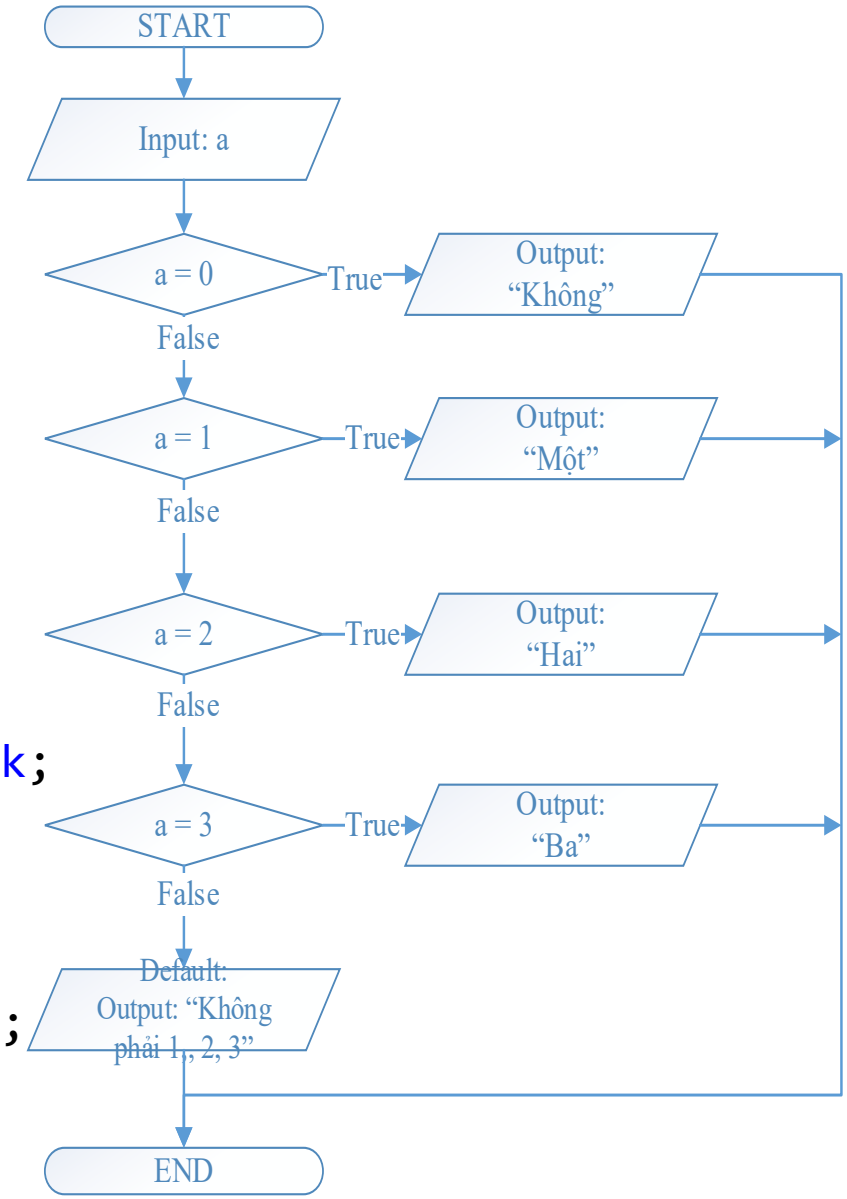


Ví dụ 1

```
#include <iostream>

int main() {
   int a;
   std::cout << "Nhap 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 << "o biet doc";
   }
}
```



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 << "Nhap gia tri ch=";
    cin >> ch;
    switch (ch) {
        case 'a': cout << "Ky tu a da duoc nhap";
            break;
        case 'b': cout << "Ky tu b da duoc nhap";
            break;
        default: cout << "Ky tu khac a va b da duoc nhap";
    }
}
```

Lưu ý 1

- Câu lệnh switch là một câu lệnh đơn và có thể lồng nhau.

```
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 2: 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 3: default có thể đặt ở bất kỳ vị trí nào trong switch

```
switch (controlling_expression)
{
default: // not typical location, but perfectly legal
... some code ...
break;
case label_1:
... some code ...
break;
};
```

OK

Lưu ý 4

- 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 : printf("Mot");
  case 2 : printf("Hai");
  case 3 : printf("Ba");
}

Kết quả: MotHaiBa
```

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

Lưu ý 5

- Tận dụng tính chất khi bỏ break

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

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

Lưu ý 5

```
#include <iostream>
int main() {
   int days, month=9;

   enum m { January, February, March, April, May, June, July,
   August, September, October, November, December};

   switch (month-1) {
   case January: case March: case May: case July: case August: case
   October: case December:
         days = 31; break;
   case April: case June: case September: case November:
         days = 30; break;
   case February:
         days = 28;
   }

   std::cout << "Thang" << month << " co " << days << "ngay.";
}
```

Lưu ý 6

Switch case hoạt động theo case label, không quan tâm scope.

```
#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: cout << "case 3." << std::endl; break;
       }
     default:
            std::cout << "default." << std::endl;
  }
  return 0;
}
```

Lưu ý 7: Khai báo biến trong switch

```
#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 khái 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 8: Khởi tạo biến trong switch

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 chứa lệnh khởi tạo có thể sẽ không bao giờ được thực thi!

```
#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;
   }
}
```

Lưu ý 8: Khởi tạo biến trong switch

➔ 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)

```
#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;
  }
}
```

5. Một số ví dụ minh họa

- Ví dụ 1: Viết chương trình Kiểm tra tính chẵn, lẻ của một số nguyên
- Ví dụ 2: Viết chương trình Tính max của 2 số thực
- Ví dụ 3: Viết chương trình Giải phương trình bậc nhất
- Ví dụ 4: Viết chương trình Xác định học lực của SV dựa trên điểm trung bình dùng switch-case

Bài tập bắt buộc

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 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?
4. 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.
5. 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!