

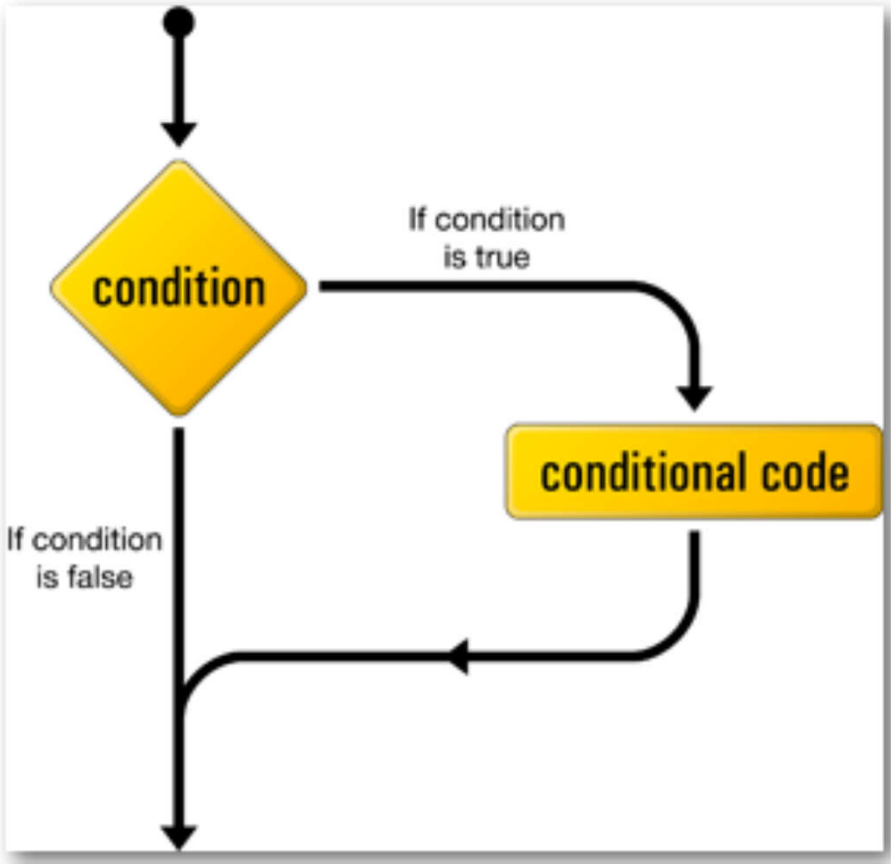

Nội dung chính

1. Cấu trúc rẽ nhánh
2. Cấu trúc lặp
3. Một số cấu trúc điều khiển khác
3. Bài tập

1. Cấu trúc if (Lựa chọn có điều kiện)

- Dạng 1:

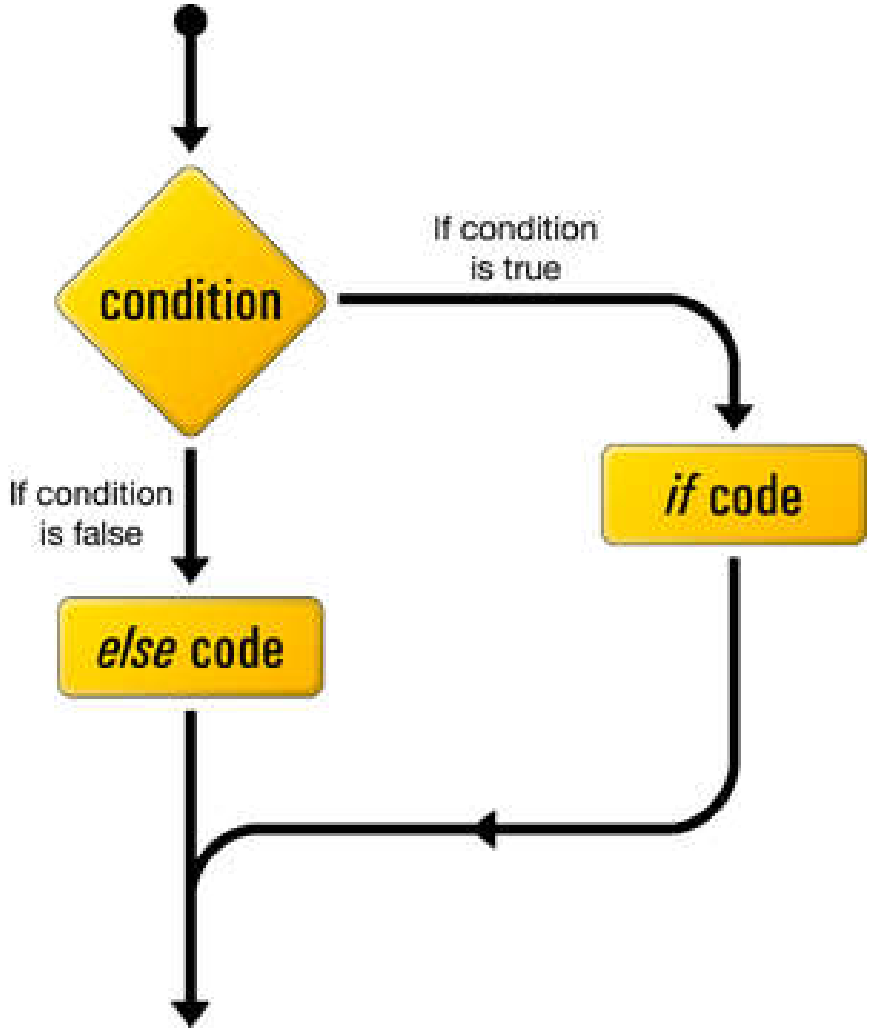
```
if (condition)
{
    conditional code;
}
```



Ngôn ngữ C cho phép sử dụng cấu trúc if lồng nhau

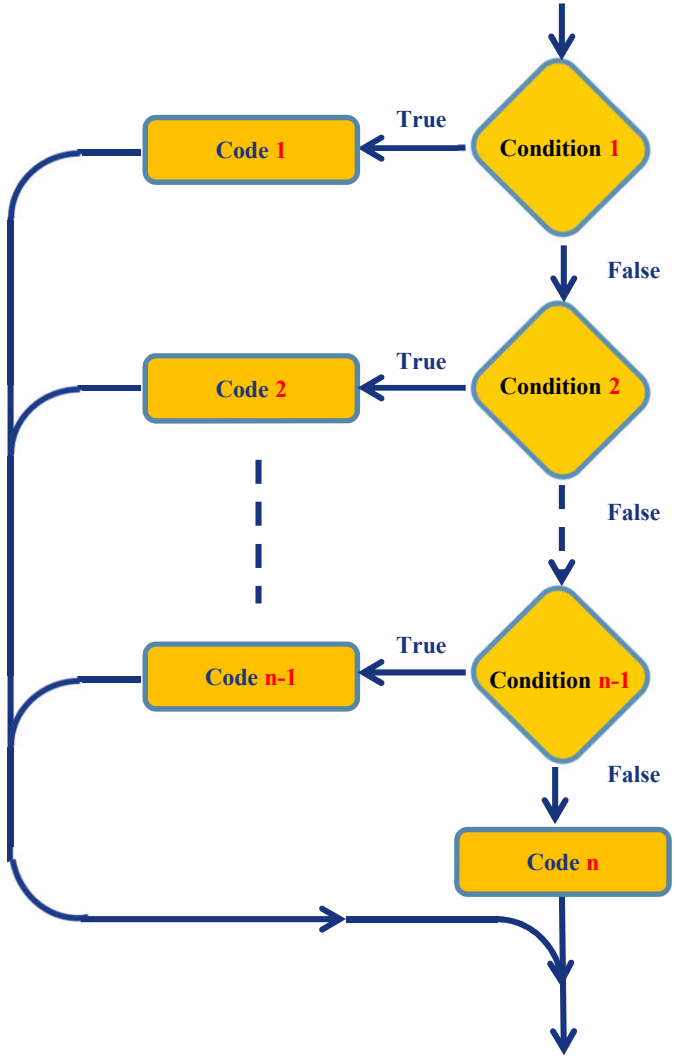
- Dạng 2:

```
if (condition)
{
    if_code;
}
else
{
    else_code;
}
```



Dạng 3:

```
if (condition_1)
 {
   code_1;
 }
 else  if (condition_2)
 {
   code_2;
 }
 else  if (condition_3)
 .
 .
 .
 else
   code_n;
```



Ví dụ: So sánh hai số a và b

```
# include <stdio.h>
main()
{
    int a, b;
    printf("Nhap vao so a: ");
    scanf("%d", &a);
    printf("Nhap vao so b: ");
    scanf("%d", &b);
    if (a>b)
     {
         printf("a lon hon b.\n");
     }
     else if (a<b)
     {
        printf("a nho hon b.\n");
     }
    else
     {
        printf("a bang b.\n");
     }
}
```

Nhập số nguyên a

Nhập số nguyên b

Nếu a lớn hơn b

Nếu a nhỏ hơn b

Nếu a bằng b

Cấu trúc switch (Lựa chọn)

- switch (expression)
  {
  case n_1: Các câu lệnh_1;
  break;
  case n_2: Các câu lệnh_2;
  break;
  case n_3: Các câu lệnh_3;
  break;
  case n_i: Các câu lệnh_i;
  break;
  default: Các câulệnh_mặc_định;
  }

- Nếu expression nhận giá trị n_i nào thì sẽ thực hiện các câu lệnh tương ứng.

- Hoạt động của switch
  - Xét giá trị của expression.
  - Nếu exp. bằng n_i thì thực hiện các câu lệnh có nhãn n_i
    Nếu exp. khác n_i thì nhảy tới thực hiện các câu lệnh ở default, hoặc thoát ra khỏi switch nếu không có default
- Thoát khỏi switch khi:
  - Gặp lệnh break
  - Gặp dấu } cuối của switch
  - Gặp lệnh goto nhảy ra bên ngoài

- *Chú ý*: giá trị n là các hằng

Ví dụ: Nhập vào số và in ra thông báo bằng chữ

```
# include <stdio.h>
int main()
{
    int x;
    printf("Nhap vao so 1, 2 hoặc 3: ");
    scanf("%d", &x);
    switch(x)
    {
        case 1: printf(“Mot”); break;
        case 3: printf(“Ba”); break;
        case 2: printf(“Hai”); break;
    };
    printf(“Ket thuc chuong trinh \n");
    return 0;
}
```

Nhập giá trị của x

Xem xét các trường hợp của biến x

Nếu x = = 1

Nếu x = = 3

Nếu x = = 2


Nội dung chính

2 **Cấu trúc lặp**

2. Cấu trúc lặp for

- Cú pháp for tổng quát:

```
for (bt1; bt2; bt3)
{
    Các lệnh;
}
```

  - bt1: biểu thức chứa *giá trị khởi tạo,* chỉ thực hiện đúng một lần duy nhất.
  - bt2: biểu thức chứa *điều kiện dừng* để thoát khỏi vòng for.
  - bt3: biểu thức chứa toán tử *thay đổi giá trị biến khởi tạo.*

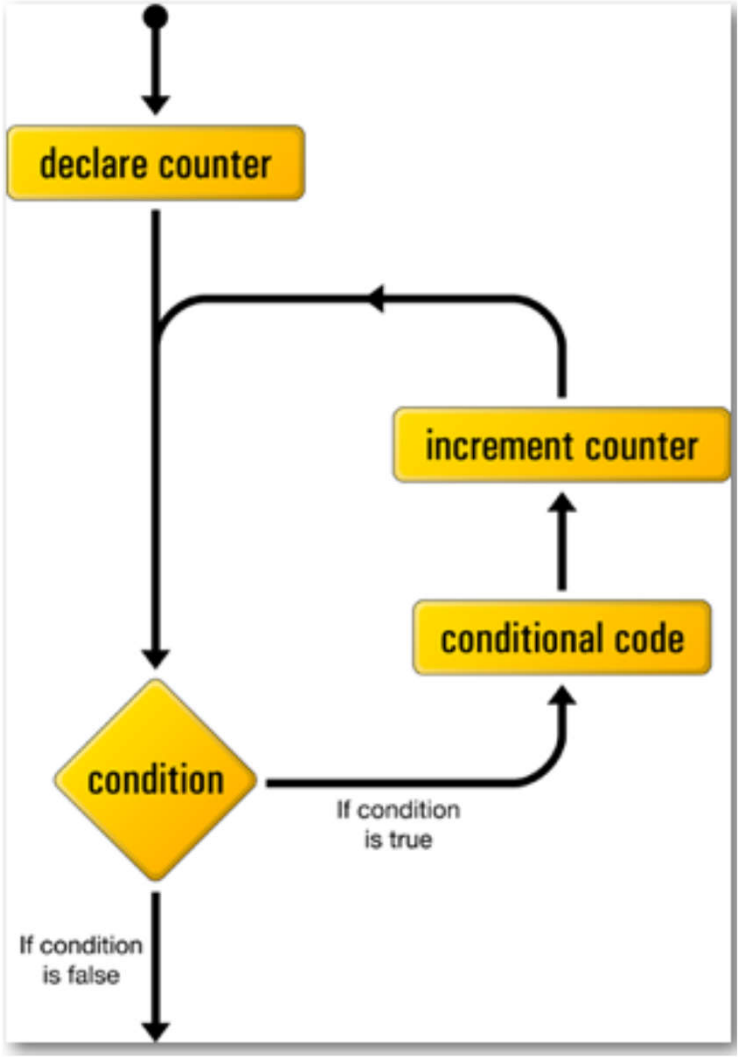
- Hoạt động của for
  1. Tính bt1
  2. Xác định bt2
     - Nếu bt2 *sai* thì thoát khỏi for
     - Nếu bt2 *đúng* thực hiện các lệnh trong thân for
  3. Tính bt3, sau đó quay lại bước 2

- Muốn thoát khỏi vòng lặp không qua *điều kiện dừng*(bt2), phải dùng lệnh break hoặc goto hoặc return

- Có thể dùng vòng lặp for lồng nhau

- **Một cách đơn giản hơn**
  - Tiếp tục công việc trong khi điều kiện còn đúng
  - Cú pháp:

```
for (de. counter; condition; increment)
{
        // conditional code;
}
```



Ví dụ:

- Viết ra 10 dòng chữ *Hello world* trên màn hình

```
# include <stdio.h>
main()
{
    int i;
    for(i = 1; i<=10; i++)
    {
       printf("Hello world \n");
    }
}
```

- Tính tổng của các số tự nhiên từ 1→100. In kết quả ra màn hình

```
# include <stdio.h>
main()
{
      int i, s;
      s=0;
      for(i = 1; i<=100; i++)
        {
        s = s+i;
        }
      printf("Tong la:  %d", s);
}
```

Ví dụ:

- In ra màn hình ma trận 3 x 3

```
# include <stdio.h>
int a[3][3]={{2, 4, 6},{8, 1, 3},{5,  7, 9}};
main()
{
    int i, j;
    for(i = 0; i<=2; i++)
    {
        for(j = 0; j<=2; j++)
        {
            printf("%d   ",a[ i][j]);
        }
        printf("\n");
    }
}
```

Vòng for thứ nhất theo i

Vòng for thứ hai theo j

In phần tử ma trận

Xuống dòng

2. Cấu trúc lặp while

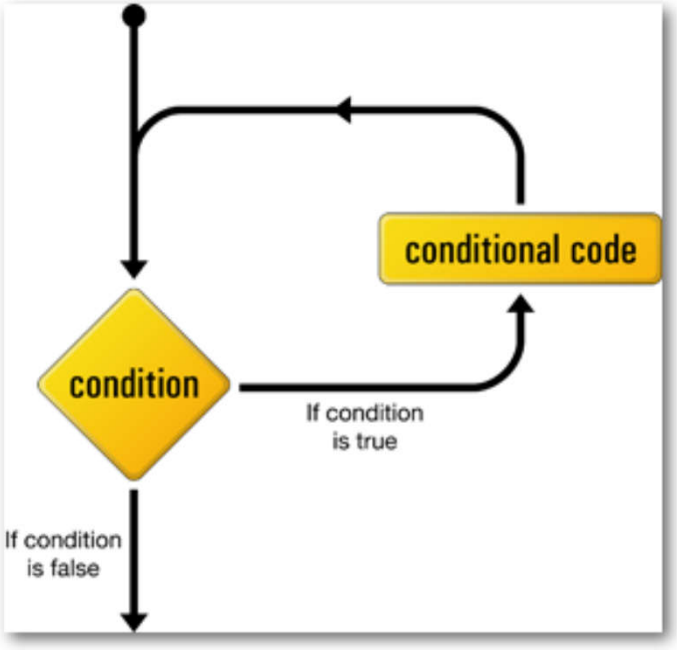
- Cú pháp while tổng quát:

```
while (btdk)
{
    Các lệnh;
}
```

- Dùng cú pháp này khi không biết trước số lần lặp.
- Ta có thể dùng câu lệnh break để thoát khỏi vòng lặp.
- btdk phải thay đổi qua mỗi vòng lặp.

- Hoạt động của **while**
  1. Đánh giá btdk.
  2. Nếu btdk sai thì thoát khỏi while

     Nếu btdk đúng thì thực hiện các lệnh. Khi thực hiện xong các lệnh thì quay lại bước 1.



Ví dụ: Tính tổng số nguyên từ 1 đến n

```
# include <stdio.h>
main()
{
    int i, n, tong;
    printf("Nhap vao so n>0: ");
    scanf("%d", &n);

    i = 0; tong = 0;
    while (i <=n)
    {
       i++; //i+=1 hoac i=i+1;
       tong += i;
    }
   printf("Tong: %d", tong);
}
```

Khởi tạo i, tong

Kiểm tra nếu i < n

Tính tổng

In kết quả

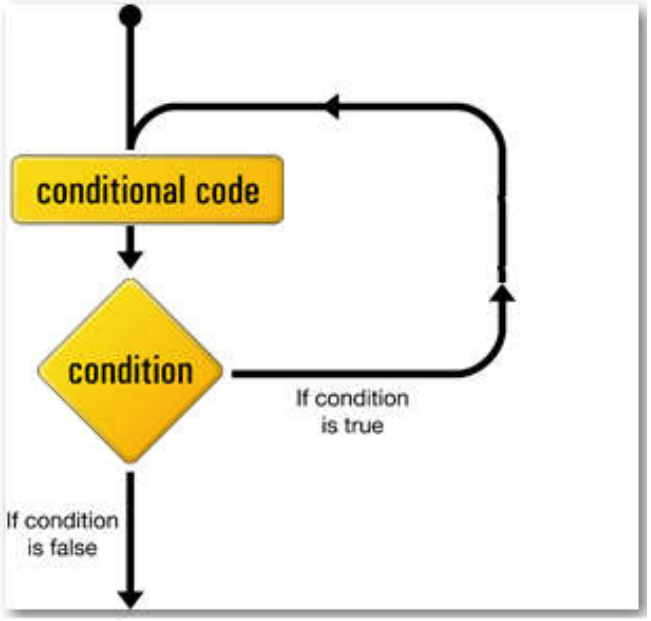
2. Cấu trúc lặp do …while

- Cú pháp do while tổng quát:

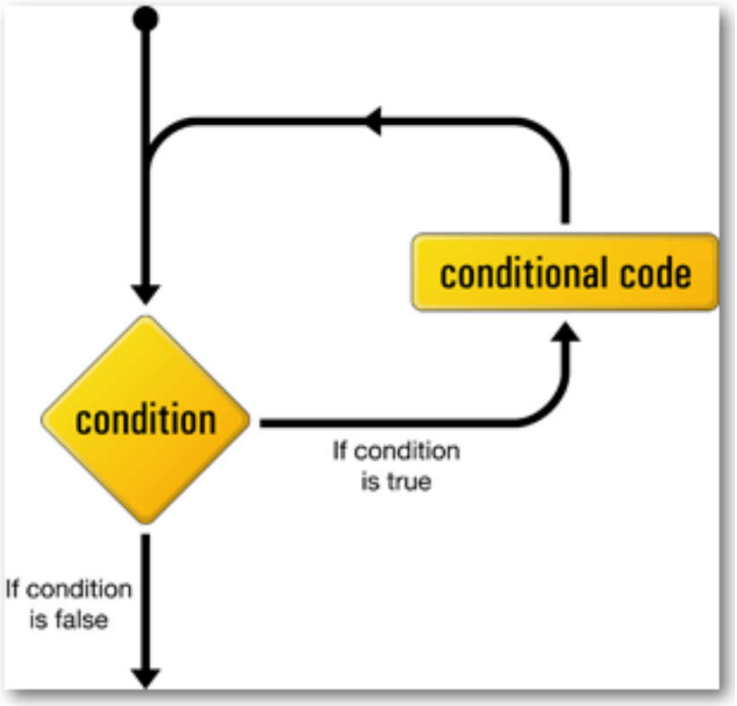
```
do
{
        Các lệnh;
} while (btdk);
```

- Dùng cú pháp này khi không biết trước số vòng lặp
- Ta có thể dùng câu lệnh break để thoát khỏi vòng lặp theo ý muốn
- btdk phải thay đổi qua mỗi vòng lặp

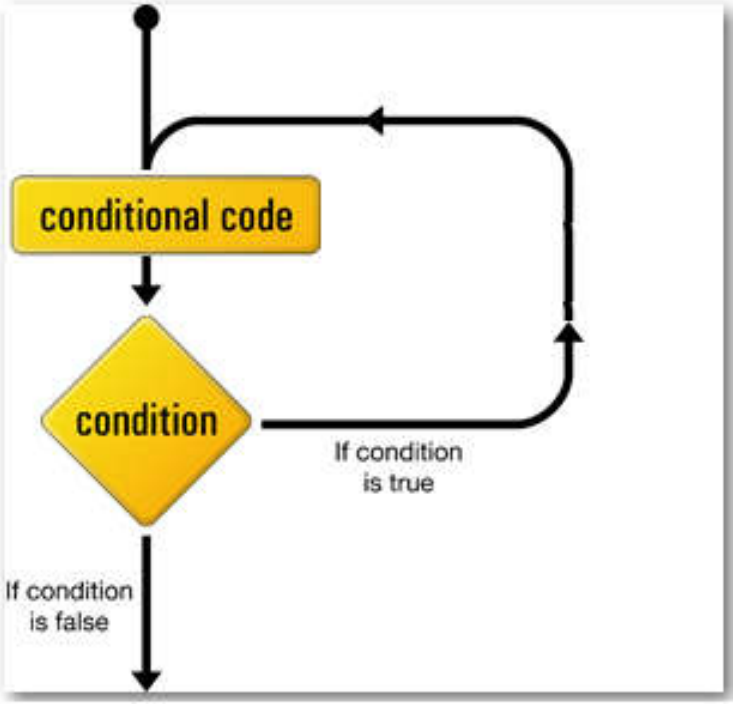
- Hoạt động của do while
  1. Thực hiện các lệnh trong do while.
  2. Khi thực hiện xong lệnh cuối cùng trong các lệnh, chương trình sẽ đánh giá btdk.
  - ✓ Nếu btdk đúng thì quay lại bước 1.
  - ✓ Nếu btdk sai thì thoát khỏi do while



while



do-while



Ví dụ: Kiểm tra mật khẩu

```
# include <stdio.h>
# define PASSWORD  123456

main()
{
   int  a;
   do
   {
      printf("Nhap vao password: ");
      scanf("%d", &a);
   } while (a ! = PASSWORD);

    printf("\nBan da nhap dung mat khau");
    printf("\nLay tien thoi \n");
}
```

Mật khẩu cài đặt sẵn

Thực hiện nhập mật khẩu

Kiểm tra mật khẩu vừa nhập


Câu hỏi

- Thiết kế vòng lặp nhằm mục đích gì?
- Ý nghĩa của vòng lặp ?
- Lúc nào thì nên dùng vòng lặp ?


Nội dung

3 **Một số cấu trúc điều khiển khác**

Thoát khỏi chu trình break

break

- Khi break được đặt trong thân các vòng lặp ***for***, ***while***, ***do while*** nó cho phép thoát khỏi các vòng lặp đó mà không cần dùng đến điều kiện kết thúc.

- Khi có nhiều chu trình lồng nhau, break sẽ đưa máy ra khỏi chu trình *bên trong nhất* chứa nó.

- break cũng cho phép thoát khỏi cấu trúc lựa chọn ***switch*** tại vị trí gặp câu lệnh này

Nhảy tới vị trí mong muốn goto

- Cú pháp: goto lb ;
- Trong đó:
  - lb là tên(nhãn) của vị trí mà ta muốn nhảy tới. Nó là tên do người lập trình tự đặt.
  - Quy tắc đặt tên cho lb giống như đặt tên biến.
  - Tại vị trí viết tên của lb thì ngay sau nó là dấu :
- Hoạt động của goto

  Khi gặp lệnh goto máy sẽ nhảy tới vị trí câu lệnh nằm ngay sau khu vực được đặt nhãn lb:
- Lưu ý:

  - Không cho phép nhảy hàm này sang hàm khác

  - Không cho phép nhảy từ ngoài vào bên trong một khối lệnh

Ví dụ: Nhập vào một số, nếu < 0 thì nhập lại

```
# include <stdio.h>
main()
{
    float   a;
    printf("Nhap vao a: ");
abcd:     scanf("%f", &a);
    if (a<0)
   {
        printf("Nhap lai a ");
        goto  abcd;
   }
    printf("So vua nhap la:  %f ",a);
}
```

Tới lần lặp kế tiếp continue

- Khi gặp lệnh continue trong thân for, máy sẽ:
  1. Bỏ qua các câu lệnh đứng sau continue
  2. Chuyển tới thực hiện bt3
  3. Kiểm tra bt2

     - Nếu bt2 sai thì thoát khỏi for

     - Nếu bt2 đúng thì thực hiện các câu lệnh chứa trong for

- Khi gặp continue trong while hoặc do while
  - Bỏ qua các câu lệnh đứng sau continue và sẽ chuyển tới xét giá trị của btdk

Tóm tắt bài học

- Cấu trúc rẽ nhánh: if, switch
- Cấu trúc lặp
  - for
  - while
  - do-while
- Một số cấu trúc điều khiển khác
  - break
  - goto
  - continue


4 Bài tập

Bài tập

1. Nhập 2 số thực a, b từ bàn phím. Tìm và in ra màn hình số lớn nhất và số bé nhất
2. Viết chương trình tính n!. Với n nhập từ bàn phím
3. Viết chương trình nhập vào N số nguyên, đếm xem có bao nhiêu số âm, bao nhiêu số dương và bao nhiêu số không
4. Viết chương trình tính tổng của n số đầu tiên của dãy số sau:

$$S = 1+1/2+1/3+1/4+...+1/n$$