

## Bài 6: Condition (Điều kiện)

Trong bài 1, ta đã thấy rằng có rất nhiều ngôn ngữ lập trình, trong đó một số ngôn ngữ hoạt động tương tự nhau, ví dụ như ngôn ngữ PHP khá giống với C, tuy nhiên người ta sử dụng PHP để thiết kế web nhiều hơn. 😊

Các ngôn ngữ thường có một số điểm giống nhau do chúng cùng sử dụng lại các nguyên tắc cơ bản của ngôn ngữ đời trước. Ngôn ngữ C được tạo ra cách đây khá lâu, và mô hình của phần lớn các ngôn ngữ mới hiện giờ đều được tạo ra dựa trên C.

Nguyên tắc cơ bản thì có rất nhiều như cách khai báo biến số, các cách thực hiện phép tính (hầu như các ngôn ngữ lập trình đều giống nhau ở mặt này!) và cách sử dụng condition.

Phần này gồm :

- Condition "if... else"
- Boolean, trung tâm của những conditions
- Condition "switch"
- Ternaries: những condition rút gọn
- TRẮC NGHIỆM KIẾN THỨC.

---

### Condition « if...else »

Chúng ta thường muốn kiểm tra giá trị của một biến số. Ví dụ « Nếu biến số maymoc có giá trị là 50, hãy thực hiện công việc ... » . Hoặc nếu biến số nhỏ hơn 50, nhỏ hơn hoặc bằng 50, lớn hơn, lớn hơn hoặc bằng...

Điều đó có thể được thực hiện trong C thông qua việc sử dụng condition if...else. Condition dùng để kiểm tra giá trị của biến số. Và để biết cách sử dụng nó, chúng ta sẽ đi theo sơ đồ sau:

1. Một số kí tự cần biết trước khi bắt đầu.
2. Test if
3. Test else
4. Test “else if”
5. Cách thiết lập nhiều conditions cùng lúc
6. Những sai phạm thường gặp mà người mới học cần tránh

## Một vài kí hiệu cơ bản cần biết trước khi học cách sử dụng condition “if...else” trên C

Chúng ta cần phải **thuộc lòng** bảng kí hiệu này: 😊

Kí tự	Ý nghĩa
=	Bằng
>	Lớn hơn
<	Bé hơn
>=	Lớn hơn hoặc bằng
<=	Bé hơn hoặc bằng
!=	Khác



Hãy chú ý, để kiểm tra bằng nhau, ta cần nhập 2 kí tự « == ». Những người bắt đầu học lập trình thường mắc lỗi chỉ nhập một kí tự =, chúng không có cùng ý nghĩa trong ngôn ngữ C, tôi sẽ nói về vấn đề này ở dưới.

### if

Chúng ta sẽ thực hiện một chương trình đơn giản, nó sẽ nói với máy tính:

**NẾU** biến số thỏa điều kiện ...  
**THÌ** thực hiện ...

Trong tiếng anh, từ « nếu » sẽ dịch thành « if », từ này cũng được sử dụng trong C để khai báo một condition.

Để viết một condition if, đầu tiên hãy viết từ **if**, kế đó mở ngoặc đơn. Trong ngoặc đơn, hãy viết điều kiện.

Sau đó, mở một dấu gộp { và hãy đóng lại ở phía sau }. Trong đó sẽ chứa tất cả những instruction sẽ được thực hiện nếu điều kiện thỏa mãn.

Chúng ta sẽ viết như sau:

**C code:**

```
if (/* Điều kiện của bạn */)
{
    // Các Instructions sẽ được thực hiện nếu như điều kiện thỏa mãn
}
```


Tại vị trí chú thích « điều kiện của bạn », chúng ta sẽ viết một điều kiện để kiểm tra biến số.

Ví dụ chúng ta có thể kiểm tra một biến số « tuổi » sẽ chứa giá trị là tuổi của bạn. Chúng ta sẽ xét xem bạn có phải là người trưởng thành hay không, có nghĩa là tuổi của bạn có lớn hơn hoặc bằng 18 hay không:

**C code:**

```
if (tuoi >= 18)
{
    printf ("Ban la nguoi truong thanh !");
}
```

Kí tự >= có nghĩa là « lớn hơn hoặc bằng », chúng ta đã thấy trong bảng liệt kê các kí tự đặc biệt ở trên.

 Các dấu gộp {...} không bắt buộc nếu bên trong nó chỉ chứa duy nhất một instruction. Bạn có thể viết như bên dưới nhưng tôi khuyên bạn hãy luôn đặt những dấu gộp để chương trình bạn được rõ ràng.

**C code:**

```
if (tuoi >= 18)
    printf ("Ban la nguoi truong thanh !");
```

## Test thử đoạn code trên

Nếu như bạn muốn test những đoạn mã trước để xem if hoạt động như thế nào, chúng ta phải đặt condition if bên trong một function main và đừng quên khai báo biến số tuổi, và cho nó một giá trị nào đó theo ý thích của bạn. 😊

Tôi nghĩ rằng bạn có thể tự mình viết ra đoạn mã này và sau đó chạy thử nó.

Đây là đoạn mã hoàn chỉnh:

**C code:**

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    long tuoi = 20;
    if (tuoi >= 18)
    {
        printf ("Ban la nguoi truong thanh !\n");
    }
    return 0;
}
```

Ở đây, biến số tuổi bằng 20, vậy « Ban la nguoi truong thanh ! » sẽ được hiển thị  
Hãy thử thay đổi giá trị của biến số. Cho giá trị mới là 15: điều kiện không thỏa mãn và « Ban la nguoi truong thanh ! » không hiển thị lần này. 😊  
Hãy giữ lại đoạn mã này để sử dụng cho ví dụ tiếp theo. 😊

### ***Vấn đề cần giải thích***

Cách bạn đặt những dấu gộp { } không quan trọng, chương trình của bạn sẽ cũng chạy tốt nếu như bạn viết tất cả trên cùng một hàng. Ví dụ:

**C code:**

```
if (tuoi >= 18) { printf ("Ban la nguoi truong thanh !"); }
```

Mặc dù bạn có thể viết như vậy nhưng cách viết này **không hề được khuyến khích** (điều này thực sự rất quan trọng nhé). 😊

Thực sự là viết tất cả trên cùng một hàng sẽ khiến cho **việc đọc đoạn mã của bạn vô cùng khó khăn**. Bạn cần tập cách trình bày mã nguồn của mình ngay từ bây giờ, nếu không sau này khi bạn viết các chương trình lớn hơn, bạn sẽ không tìm thấy được cái bạn cần tìm trong đó!

Hãy thử trình bày lại mã nguồn của bạn theo cách thức của tôi: một dấu gộp mở { duy nhất trên 1 hàng, các dòng sau đó là các instruction (nhấn tab để có thể « cách về bên phải »), sau đó một dấu gộp đóng } duy nhất trên một hàng.

⚠️ Có rất nhiều cách thức hay để trình bày mã nguồn và nó không làm thay đổi hoạt động của chương trình.

Trong suốt quá trình từ giờ về sau bạn sẽ bắt gặp các đoạn mã được trình bày bởi các style hơi khác. Nhưng về cơ bản hầu hết các đoạn mã đó đều có cách trình bày thoáng và dễ nhìn.

## « else » để nói « nếu không »

Chúng ta đã biết cách viết condition đơn giản, hãy đi xa hơn một chút: điều kiện không thỏa mãn (sai), chúng ta sẽ yêu cầu máy tính thi hành một instruction khác.

Trong ngôn ngữ của chúng ta, điều đó sẽ được ghi với dạng tương tự như sau :

**NẾU** biến số thỏa điều kiện...  
**THÌ** thực hiện ...  
**NẾU KHÔNG** hãy thực hiện ...

Chỉ cần thêm vào từ **else** sau dấu gộp kết thúc của **if**.

Một ví dụ nhỏ :

**C code:**

```
if (tuoi >= 18) // Neu tuoi lon hon hoac bang 18
{
    printf ("Ban la nguoi truong thanh !");
}
else // Neu khong...
{
    printf ("Hehe, ban con la con nit !");
}
```

Ta có thể hiểu đơn giản: nếu như biến số tuổi có giá trị lớn hơn hoặc bằng 18, sẽ hiển thị « Ban la nguoi truong thanh ! »

Nếu không hiển thị « Hehe, ban con la con nit ! »

## « else if » để nói « nếu không nếu »

Chúng ta đã thấy bằng cách nào để tạo một « nếu » và một « nếu không ». Chúng ta cũng có thể tạo một “nếu không nếu”. Nếu như điều kiện đầu không thỏa mãn, chúng ta sẽ kiểm tra biến số với một điều kiện khác. “Nếu không nếu” đặt bên trong if và else.

Chúng ta yêu cầu máy tính:

**NẾU** biến số thỏa điều kiện 1  
**THÌ** thực hiện việc 1  
**NẾU KHÔNG NẾU** biến số thỏa điều kiện 2  
**THÌ** thực hiện việc 2  
**NẾU KHÔNG** thực hiện việc 3

Dịch sang ngôn ngữ C:

### C code:

```
if ( tuoi >= 18) // Neu tuoi lon hon hoac bang 18
{
    printf ("Ban la nguoi truong thanh !");
}
else if ( tuoi > 4 ) // Neu khong, Neu tuoi nho hon 18 va lon hon 4
{
    printf ("Hehe, ban con la con nit !");
}
else // Neukhong...
{
    printf ("Oe oe"); // Ngon ngu tre so sinh, ban khong the hieu duoc dau
}
```

Máy tính sẽ thực hiện những kiểm tra theo thứ tự:

- Đầu tiên nó sẽ kiểm tra **if** đầu tiên: nếu như điều kiện được thỏa mãn, nó sẽ thi hành các instructions có trong các dấu gộp {...} đầu tiên.
- Nếu không, nó sẽ đi đến « nếu không nếu » để kiểm tra các điều kiện mới: nếu đúng, thì nó sẽ thi hành các instruction trong các dấu gộp {...} thứ 2 của **else if**.
- Cuối cùng, nếu không có điều kiện nào được thỏa mãn, nó sẽ chạy những instruction của **else** « nếu không»



« else » và « else if » không bắt buộc phải có. Cần phải có ít nhất một “if” để tạo một condition

**Chúng ta có thể đặt bao nhiêu “else if” nếu như ta muốn.** Chúng ta có thể viết như sau:

```
NẾU biến số thỏa điều kiện 1
THÌ thực hiện việc 1
NẾU KHÔNG NẾU biến số thỏa điều kiện 2
THÌ thực hiện việc 2
NẾU KHÔNG NẾU biến số thỏa điều kiện 3
THÌ thực hiện việc 3
NẾU KHÔNG NẾU biến số thỏa điều kiện 4
THÌ thực hiện việc 4
NẾU KHÔNG thực hiện việc 5
```

## Thiết lập nhiều điều kiện cùng lúc

Ta có thể kiểm tra nhiều điều kiện cùng lúc trong if. Ví dụ, các bạn muốn xem thử nếu như tuổi lớn hơn 18 VÀ nhỏ hơn 25.

Chúng ta cần phải sử dụng các kí hiệu sau đây:

Kí hiệu	Ý nghĩa
&&	VÀ
	HOẶC
!	KHÔNG

### Test VÀ

Đoạn mã sử dụng kí hiệu vừa nêu

**C code**

```
if (age > 18 && age < 25);
```

Những kí hiệu « && » có nghĩa là VÀ. Diễn đạt bằng ngôn ngữ của chúng ta : « *Nếu tuổi lớn hơn 18 VÀ bé hơn 25* »

### Test HOẶC

Để tạo một HOẶC, chúng ta sử dụng 2 kí tự ||.

Chúng ta hãy tưởng tượng một chương trình kiểm tra và quyết định quyền mở tài khoản ngân hàng. Để mở một tài khoản ngân hàng, tuổi khách hàng không được nhỏ quá (chúng ta sẽ tùy tiện đặt điều kiện tuổi phải lớn hơn 30) hoặc khách hàng có nhiều tiền (tất nhiên ngân hàng sẽ dang tay đón chào kể cả khi tuổi bạn bé hơn 10) 😊

Đây là đoạn mã nhận biết quyền mở một tài khoản ngân hàng :

**C code**

```
if (tuoi > 30 || tien > 100000)
{
    printf("Chao mung ban den voi chung toi !");
}
else
{
    printf("Cut !");
}
```

Điều kiện sẽ được thỏa mãn nếu người này có tuổi trên 30 hoặc có số tiền nhiều hơn 100.000 euros.

## Test KHÔNG

Kí tự cuối cùng còn lại để test là dấu chấm than. Trong tin học, dấu chấm than có nghĩa là «Không»  
Các bạn phải đặt kí tự này trước điều kiện để nói rằng «Nếu điều này không đúng »:

C code:

```
if (!(tuoi < 18))
```

Đoạn mã trên có thể dịch là « *Nếu người này không phải là trẻ con* »

Nếu dấu « ! » ở phía trước bị lấy đi, đoạn mã sẽ có nghĩa trái ngược : « *Nếu người này là trẻ con* »

## Một số lỗi thường gặp của người mới học

**Đừng quên có đến 2 dấu ==**

Nếu chúng ta muốn kiểm tra xem người này có phải 18 tuổi hay không, chúng ta phải ghi:

C code:

```
if (tuoi == 18)
{
    printf ("Ban vua moi truong thanh !");
}
```

Đừng quên việc đặt **2 kí tự « bằng »** trong một if, như thế này : ==

Nếu bạn chỉ đặt mỗi một kí tự =, thì biến số của bạn sẽ nhận giá trị 18 (giống như ta đã học trong phần biến số). Tất cả những điều ta muốn ở đây, là kiểm tra giá trị của biến số chứ không phải thay đổi nó! Hãy chú ý điểm này, có rất nhiều người trong các bạn chỉ đặt một dấu = và khi chương trình bạn được bắt đầu thì tất nhiên là nó không chạy giống như ý họ muốn. 😊

### Dấu chấm phẩy dư thừa



Một lỗi khác rất thường xuyên của những người mới bắt đầu học lập trình: có khi các bạn đặt một dấu chấm phẩy sau dòng của if.

if là một condition, chúng ta sẽ không đặt dấu chấm phẩy ở cuối condition mà phải là cuối một instruction.



Đoạn code sau đây sẽ không chạy vì có một dấu chấm phẩy ở cuối condition if :

**C code:**

```
if (tuoi == 18); // dấu chấm phẩy nay KHÔNG được phép ở đây
{
    printf ("Ban chi vua moi truong thanh");
}
```

Có một vấn đề khó khăn nữa là "condition if...else..." không biết phải dịch như thế nào là đúng, vì nếu dịch là "điều kiện if...else..." thì có thể bạn sẽ nhầm lẫn bởi điều kiện bên trong dấu ngoặc (...), nếu dịch là "hàm điều kiện..." thì có thể hiểu nhầm là cần đặt một dấu chấm phẩy ở cuối cùng vì nó cũng là một hàm.

Nên mình quyết định, gọi là "condition" nếu đề cập đến "if...else...", gọi "điều kiện" nếu đề cập đến điều kiện bên trong (...) của if.

## Booleans, trung tâm của những conditions

Chúng ta sẽ tìm hiểu chi tiết hơn về cách thức hoạt động của condition if...else.

Thực tế, condition được tác động bởi một thứ mà trong tin học người ta gọi là những Boolean.

Đây là một khái niệm khá quan trọng, cần phải nghe rõ (đúng hơn là phải nhìn rõ 😊)

### Một số ví dụ để có thể hiểu rõ hơn

Trong những bài học về Vật Lý–Hóa học, thầy giáo của tôi thường có thói quen bắt đầu từ một số thí nghiệm nhỏ trước khi giảng về một khái niệm mới.

Bây giờ tôi sẽ bắt chước ông ấy 😊

Bạn hãy chạy thử đoạn mã đơn giản sau :

**C code:**

```
if (1)
{
    printf ("Dung");
}
else
{
    printf ("Sai");
}
```

Kết quả :

**Console:**

Dung

❓ Nhưng ??? Chúng ta không hề đưa điều kiện vào trong if, chúng ta chỉ cho nó một số. Vậy đó nghĩa là gì ?

Chúng ta thử một ví dụ khác, bây giờ bạn hãy thay thế 1 bởi 0 :

**C code:**

```
if (0)
{
    printf ("Dung");
}
else
{
    printf ("Sai");
}
```

Kết quả:

**Console:**

Sai

Thử một số những ví dụ khác, thay thế 0 bởi bất kì một số nguyên nào, như là 4, 15, 226, -10, -36 .v.v...

Mỗi lần như vậy, nó đưa ra kết quả là gì ? Nó trả lời là : « **Điều đó đúng** ».

**Tóm tắt lại các thử nghiệm trên:** nếu đặt 0, điều kiện không thỏa mãn, và nếu chúng ta để vào giá trị 1 hay bất kì một số hạng nào, điều kiện thỏa mãn.

## Giải thích

Thực tế, mỗi khi ta kiểm tra một điều kiện trong một if, nó sẽ trả về giá trị là 1 nếu điều đó đúng và 0 nếu điều đó sai.

Ví dụ :

**C code:**

```
if (tuoi >= 18)
```

Ở đây, chúng ta cần kiểm tra điều kiện “tuoi >= 18”.

Giả định rằng tuổi có giá trị là 23, sẽ cho kết quả đúng, và máy tính « thay thế » « tuoi >= 18 » bởi 1. Sau đó, máy tính sẽ ghi nhớ một « if (1) ». Khi mà số này là 1, như chúng ta đã thấy, máy tính sẽ hiểu là điều kiện này là đúng, và nó sẽ thi hành các instruction khi điều kiện đúng.

Tương tự nếu như điều kiện này là sai, máy tính sẽ thay thế “tuoi >= 18” bởi số 0, và ngay tức khắc máy tính hiểu điều kiện là sai. Máy tính sẽ thi hành instruction của « else ».

## Kiểm tra với một biến số

Bây giờ chúng ta làm cái khác: gửi kết quả của điều kiện vào trong một biến số, như chúng ta làm một phép tính (bởi vì đối với máy tính điều kiện cũng là một phép tính !).

### C code:

```
int tuoi = 20;
int truongthanh = 0;
truongthanh = tuoi >= 18;
printf ("truongthanh co gia tri: %ld\n", truongthanh);
```

### Console:

```
Truongthanh co gia tri: 1
```

Như các bạn vừa thấy, điều kiện `tuoi >= 18` đã trả về giá trị 1 vì `20 >= 18` (đúng).

Biến số `truongthanh` nhận giá trị 1, chúng ta kiểm chứng bằng cách làm một *printf* để thấy rõ `truongthanh` đã thay đổi giá trị từ 0 thành 1.

Thực hiện lại ví dụ trên, cho `tuoi = 10`. Trong trường hợp này, `truongthanh` có giá trị 0.

## Biến số `truongthanh` là một Boolean

Hãy nắm vững :


Biến số nhận giá trị 0 và 1 được gọi là một **Boolean**.

Và :

0 = Sai

1 = Đúng

Chính xác hơn, 0 = sai và tất cả các số còn lại là đúng (đã được kiểm chứng). Để đơn giản, chúng ta không sử dụng những số khác, sử dụng 0 và 1 để nói « điều đó là đúng hay sai »

 Trong ngôn ngữ C, không có dạng biến số « Boolean ». Một dạng biến số mới là “bool” được thêm vào trong C++ , sử dụng để tạo ra những biến số dạng Boolean.

Trong thời điểm này, chúng ta chỉ làm việc trên C, chúng ta không có các dạng biến số đặc biệt khác.

## Booleans trong các conditions

Chúng ta sẽ làm một kiểm tra condition "if" bằng cách sử dụng một boolean :

**C code:**

```
int truongthanh = 1;
if (truongthanh)
{
    printf ("Ban la nguoi truong thanh !");
}
else
{
    printf ("Ban la con nit");
}
```

Vì biến số truongthanh mang giá trị 1, điều kiện đúng, "Ban la nguoi truong thanh !" sẽ hiển thị. Điều này rất tiện lợi, người khác đọc điều kiện của bạn sẽ được dễ dàng hơn. Khi họ thấy « if (truongthanh) » thì họ có thể hiểu là « Nếu bạn là người trưởng thành »

Những điều kiện dùng boolean thường rõ ràng và dễ hiểu nhưng ít nhất bạn cũng phải đặt tên rõ ràng cho biến số, như tôi đã nói kể từ khi bắt đầu. 😊

Đây là một ví dụ khác giúp bạn hiểu rõ hơn :

**C code:**

```
if (truongthanh && nam)
```

Có nghĩa là "Nếu bạn là người trưởng thành và bạn là nam".

nam ở đây là một biến số khác dạng boolean có giá trị là 1 nếu bạn là nam, và 0 nếu bạn là nữ.

❓ Câu hỏi : nếu chúng ta viết « if (truongthanh == 1) » thì nó vẫn hoạt động đúng không?

Vẫn hoạt động như bình thường nhưng nếu ghi là « if (truongthanh) » thì dễ hiểu hơn 😊

**Nắm vững vấn đề :** nếu biến số của bạn mang giá trị là một số, hãy viết điều kiện dưới dạng « if (variable == number) ».

Ngược lại, nếu biến số của bạn mang giá trị là một boolean (có nghĩa là sẽ mang giá trị 1 hoặc 0 để nói rằng điều đó là đúng hay sai), thì hãy viết điều kiện dưới dạng «if (boolean)».

## Condition "switch"

Chúng ta thấy rằng condition "if... else" là dạng condition được dùng thường xuyên nhất. Đôi khi "if... else" bị lặp lại khá thường xuyên. Xem ví dụ này:

### C code:

```
if (tuoi == 2)
{
    printf ("Chao baby !");
}
else if (tuoi == 6)
{
    printf ("Chao nhoc !");
}
else if (tuoi == 12)
{
    printf ("Chao cau be !");
}
else if (tuoi == 16)
{
    printf ("Chao chang trai !");
}
else if (tuoi == 18)
{
    printf ("Chao anh !");
}
else if (tuoi == 68)
{
    printf ("Chao ong !");
}
else
{
    printf ("Toi khong co cau chao danh cho ban ");
}
```

## Thiết lập một switch

Những người lập trình rất ghét việc làm một việc gì đó nhiều lần, chúng ta đã có cơ hội chứng minh điều này trước đó. Và để tránh lặp đi lặp lại nhiều lần việc kiểm tra giá trị của mỗi một biến số, người ta đã tạo ra một cấu trúc khác ngoài "if... else"

Và người ta gọi nó là "switch". Đây là một switch được thực hiện trên ví dụ mà ta vừa thấy :

**C code:**

```
switch (tuoi)
{
    case 2:
        printf("Chao baby !");
        break;
    case 6:
        printf("Chao nhoc !");
        break;
    case 12:
        printf("Chao cau be !");
        break;
    case 16:
        printf("Chao chang trai !");
        break;
    case 18:
        printf("Chao anh!");
        break;
    case 68:
        printf("Chao ong !");
        break;
    default:
        printf("Toi khong co cau chao danh cho ban ");
        break;
}
```

Hãy lấy ví dụ của tôi làm cơ sở cho việc tạo một switch khác của riêng bạn. Tuy rằng chúng ta ít sử dụng nó, nhưng việc này khá tiện lợi vì nó giúp ta viết ít hơn. 😊

Ý nghĩa viết "**switch (bienso)**" là "**Tôi sẽ kiểm tra giá trị của biến số bienso**".

Tiếp đó bạn hãy mở một dấu gộp và đóng nó lại ở phía sau { ... }

Bên trong, bạn sẽ tạo nên tất cả các trường hợp: **case 1, case 2, case 4, case 5, case 45 ...**



Các bạn bắt buộc phải ghi instruction break ở cuối mỗi trường hợp. Nếu bạn không làm, nó sẽ đọc những instruction dành cho các trường hợp khác ở phía dưới.  
Instruction break ; là lệnh yêu cầu máy tính ra khỏi những dấu gộp.

Cuối cùng, trường hợp « default » tương ứng với « else » mà chúng ta đã biết trước đó. Nếu biến số không mang giá trị nào được nêu ra ở trước đó, thì máy tính sẽ đọc instruction nằm trong default.

## Tạo một menu với switch

switch thường xuyên được sử dụng để tạo những menu trên console.

Trên console, để tạo một menu, chúng ta sử dụng *printf* để hiển thị những lựa chọn khác nhau. Mỗi lựa chọn sẽ được đánh số, người sử dụng phải nhập vào số của lựa chọn mà họ muốn

Đây là một ví dụ mà console sẽ phải hiển thị:

### Console:

```
==== Menu ====  
1. Pho  
2. Bun bo Hue  
3. Mi Quang  
4. Thit cay  
Lua chon cua ban ?
```

*(Bạn cần phải biết là khi tôi đánh ra các dòng này tôi đang cảm thấy rất đói bụng)*

Và đây là nhiệm vụ của các bạn: Hãy tạo lại một menu như thế nhờ vào hàm *printf* (quá dễ), sử dụng *scanf* để lưu lại lựa chọn của người sử dụng vào biến số *luachonMenu* (quá dễ 😊), và cuối cùng hãy sử dụng một *switch* để nói với người sử dụng biết rằng "Bạn đã lựa chọn ".

Nào, tiến hành thôi.



### ***Đáp án***

Đây là kết quả mà tôi mong muốn rằng tự bạn có thể tìm ra :

#### **C code:**

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    int luachonMenu;

    printf("=== Menu ===\n\n");
    printf("1. Pho\n");
    printf("2. Bun bo Hue\n");
    printf("3. Mi Quang\n");
    printf("4. Thit cay\n");
    printf("\nLua chon cua ban ? ");
    scanf("%d", &luachonMenu);

    printf("\n");

    switch (luachonMenu)
    {
        case 1:
            printf("Ban da chon Pho. Lua chon tuyet voi !");
            break;
        case 2:
            printf("Ban da chon Bun bo Hue. Lua chon chinh xac !");
            break;
        case 3:
            printf("Ban da chon Mi Quang. Qua tuyet !");
            break;
        case 4:
            printf("Ban da chon Thit cay. Hay den quan nhau !");
            break;
        default:
            printf("Ban da khong nhap dung so can thiet, ban khong duoc an gi het !");
            break;
    }
    printf("\n\n");
    return 0;
}
```



Tôi hi vọng rằng bạn đã không quên việc đặt "default" ở cuối của *switch* !

Trên thực tế, nếu bạn lập trình thì bạn phải nghĩ đến tất cả các trường hợp. Bạn đã nói rõ ràng rằng hãy nhập vào một số từ 1 đến 4, nhưng sẽ có một thằng đàn nào đó sẽ nhập vào "10" hay có thể là "gkfhgs". Tất nhiên đó không phải là điều chúng ta mong muốn. 😊

Tóm lại bạn hãy luôn cẩn trọng: đừng bao giờ tin tưởng vào người dùng, đôi khi họ có thể nhập vào bất cứ cái gì.

Hãy tiên đoán trước một trường hợp « default » hoặc một « else » nếu bạn thực hiện ví dụ trên bởi *if*.

⚠️ Tôi khuyên bạn hãy làm quen với cách hoạt động của những menu trên console. Vì chúng ta sẽ thường xuyên lập trình những chương trình chạy trên console và tôi chắc là bạn sẽ cần đến.

## Ternary: những conditions rút gọn

Có một cách khác để viết những condition, nhưng rất hiếm.  
Chúng ta gọi đó là **ternary expression**.

Cụ thể, nó cũng tương tự như "if... else", chỉ trừ việc tất cả chỉ nằm trên 1 dòng !  
Thay vì phải giải thích dài dòng, tôi sẽ cho bạn 2 condition giống nhau: cái thứ nhất sử dụng "if... else", và cái thứ 2 cũng như thế nhưng sẽ sử dụng dạng ternary.

### Một condition if... else được biết đến khá nhiều

Hãy giả định rằng chúng ta có một biến số dạng boolean "truongthanh" mang giá trị đúng (1), và sai (0) nếu người đó là con nít.

Chúng ta muốn thay đổi giá trị của biến số "tuoi" dựa vào hoạt động của boolean, sẽ cho giá trị « 18 » nếu người đó trưởng thành và « 17 » nếu người đó là con nít. Tôi đồng ý rằng đây là một ví dụ khá ngu ngốc, nhưng để giới thiệu cho bạn thấy làm cách nào chúng ta có thể sử dụng những **ternary**.

Chúng ta sẽ thực hiện điều đó với *if... else* :

**C code:**

```
if (truongthanh)
    tuoi = 18;
else
    tuoi = 17;
```

⚠️ Chú thích rằng tôi đã xóa đi những dấu gộp { .. } vì ở đây chỉ có một instruction duy nhất, tôi đã giải thích cho bạn điều này trước đó.

## Condition ternary

Đây là một đoạn mã hoàn toàn tương tự đoạn mã vừa rồi, nhưng lần này chúng ta sẽ viết dưới dạng **ternary**:

C code:

```
tuoi = (truongthanh) ? 18 : 17;
```

Các **ternary** cho phép, chỉ trên 1 dòng, thay đổi giá trị của biến số dựa vào hoạt động của một điều kiện. Ở đây, điều kiện của chúng ta chỉ đơn giản là « **truongthanh** », nhưng nó còn có thể hoạt động trên bất kì điều kiện khác không kể là dài hay ngắn. 😊

Dấu chấm hỏi "?" ở đây có nghĩa là « **có phải bạn là người trưởng thành ?** ». Nếu đúng, nó sẽ đưa giá trị 18 vào biến số **tuoi**, nếu không ( dấu ":" có nghĩa là **else** ở đây), nó sẽ đưa giá trị 17.

Những **ternary** thật sự không cần thiết, về cá nhân tôi nghĩ là không nên sử dụng nó nhiều quá vì nó có thể khiến cho việc đọc một đoạn mã khó khăn hơn.

Tuy nhiên, bạn cũng phải hiểu rõ nó vì sẽ có một ngày, bạn rơi vào một đoạn mã với đầy những **ternary** với mọi cách 😊. Bạn sẽ hiểu được nó hoạt động như thế nào.

Và kể từ giây phút này, bạn sẽ thực hiện các **condition** khắp mọi nơi trong chương trình của bạn, vì vậy tốt hơn bạn hãy luyện tập với nó 😊

Đây là một ý tưởng để luyện tập (lần này sẽ không có đáp án 😊): hãy tạo một công cụ tính toán trên console. Hiển thị đầu tiên menu yêu cầu người sử dụng chọn lựa những phép tính: (cộng, trừ, nhân, chia... có thể thêm vào căn bậc 2, bằng cách sử dụng thư viện toán học)

Khi mà người sử dụng lựa chọn xong, hãy yêu cầu họ nhập vào các giá trị cần thiết và hiển thị đáp án! 😊

Bạn sẽ sử dụng những gì bạn học được từ trong phần này, tôi muốn nhấn mạnh ở một điểm khác: những boolean.

Thật sự cực kì quan trọng việc nắm vững rằng boolean là những biến số có nghĩa là đúng hay sai tùy theo giá trị của nó (0 là sai, 1 là đúng).

Chương tiếp theo sẽ sử dụng lại những **boolean** và các **condition**, vì vậy bạn hãy chuẩn bị tốt trước khi sẵn sàng 😊

Cố gắng lên nào !

---