

[Log in](#)☐ Nhớ tôi vào?[Đăng ký](#)[Help](#)[What's New?](#)[Tất cả diễn đàn](#)[Lập trình cho người mới](#)[Tutorials & Thủ thuật](#)[Các bài viết mới](#) [Lịch](#) [Forum Actions](#) [Quick Links](#)[Tìm kiếm cao cấp »](#)

🏠 [Diễn đàn](#) ➡ [LẬP TRÌNH C++](#) | [LẬP TRÌNH C](#) | [LẬP TRÌNH C++0X](#) ➡ [Thủ thuật, Tutorials và Mã nguồn C/C++/C++0x](#) ➡ [C++0x Tutorial](#) [Tìm hiểu bản chất của con trỏ - từ cơ bản đến nâng cao](#)

Nếu đây là lần đầu tiên bạn ghé thăm diễn đàn cộng đồng C Việt, vui lòng tìm hiểu luật lệ tham gia, đọc các hướng dẫn trước khi bạn tiến hành đăng ký một tài khoản. Bạn phải **đăng ký thành viên** trước, hoặc đăng nhập bằng **tài khoản facebook của bạn** bạn mới có thể gửi bài viết, tải các đính kèm.

Từ 1 tới 10 trên tổng số 178 kết quả

[Trang 1 trên tổng số 18](#) [1](#) [2](#) [3](#) [11](#) ... [Cuối cùng](#)

## Đề tài: Tìm hiểu bản chất của con trỏ - từ cơ bản đến nâng cao

[Các công cụ đề tài](#) [Display](#)

09-12-2010, 02:35 PM

#1



**langman**  
Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

### Tìm hiểu bản chất của con trỏ - từ cơ bản đến nâng cao

Hôm nay đang học Oracle nhưng đêm qua lúc 3h hứa với 1 người em là sẽ viết 1 bài viết hướng dẫn về con trỏ, ở mức cực kì dễ hiểu, cực kì đầy đủ, thấu toán toàn bộ **bối cảnh** về con trỏ trong C và C++. Vì vậy mình quyết định dành toàn bộ buổi chiều nay cho bài viết này. Hy vọng

sẽ là 1 bài viết truyền thần, cực kì dễ hiểu , và đầy đủ hết mức có thể (hết mức langman có thể)

# Chap I : Bộ nhớ

## Bộ nhớ vật lý

## Bộ nhớ ảo

**Hình 1** chúng ta thấy những thứ được gọi là bộ nhớ, bộ nhớ vật lý, sở nắm nghịch thoải mái ý hơ hơ, cái này là thiết bị bạn à

**Hình 2** là mô hình bộ tổ chức bộ nhớ ảo mức khái niệm

**Hình 3** là mình chụp lại các vùng của bộ nhớ ảo của 1 tiến trình quen thuộc Unikey

## I. Bộ nhớ ảo là gì?

Quản lý bộ nhớ vật lý (cấp phát, thu hồi) là 1 vấn đề cực kì phức tạp trong hệ thống máy tính , để bảo đảm sự hiệu quả, đúng đắn, an toàn cho việc quản lý đó, hệ điều hành xây dựng lên các vùng nhớ ảo

Trong hệ thống máy tính, bộ nhớ ảo (tiếng Anh: virtual memory) là một kĩ thuật cho phép một chương trình ứng dụng tưởng rằng mình đang có một dải bộ nhớ liên tục (một không gian địa chỉ), trong khi thực ra phần bộ nhớ này có thể bị phân mảnh trong bộ nhớ vật lý và thậm chí có thể được lưu trữ cả trong đĩa cứng. So với các hệ thống không dùng kĩ thuật bộ nhớ ảo, các hệ thống dùng kĩ thuật này cho phép việc lập trình các ứng dụng lớn được dễ dàng hơn và sử dụng bộ nhớ vật lý thực (ví dụ RAM) hiệu quả hơn.

Lưu ý rằng khái niệm "bộ nhớ ảo" không chỉ có nghĩa "sử dụng không gian đĩa để mở rộng kích thước bộ nhớ vật lý" - nghĩa là chỉ mở rộng hệ thống bộ nhớ để bao gồm cả đĩa cứng. Việc mở rộng bộ nhớ tới các ổ đĩa chỉ là một hệ quả thông thường của việc sử dụng các kĩ thuật bộ nhớ ảo. Trong khi đó, việc mở rộng này có thể được thực hiện bằng các phương pháp khác như các kĩ thuật overlay hoặc chuyển toàn bộ các chương trình cùng dữ liệu của chúng ra khỏi bộ nhớ khi các chương trình này không ở trạng thái hoạt động. Định nghĩa của "bộ nhớ ảo" có nền tảng là việc định nghĩa lại không gian địa chỉ bằng một dải liên tục các địa chỉ bộ nhớ ảo để "đánh lừa" các chương trình rằng chúng đang dùng các khối lớn các địa chỉ liên tục.  
(theo wiki)

## II. Địa chỉ ảo là gì?

Trong cái vùng bộ nhớ ảo kia, để cho tiến trình dễ sử dụng, hệ điều hành để hiểu, 2 thằng này cùng nhau quy định rằng, chỉ nhỏ ra theo từng byte, và đánh số từ 1 đến hết

**cái ô nhớ nào đó, đã được đánh số là i thì ta nói địa chỉ của cái ô nhớ đó là i**

ok?!!!

giả sử tôi có biến a khai báo như sau

```
int a;
```

và a nằm trong cái ô thứ 452321 tại cái vùng nhớ trên, vậy a có địa chỉ là 452321

tiến trình hiểu là thế, còn hệ điều hành thì hiểu hơn 1 tí : "à, cái địa chỉ này tương ứng với cái ô nhớ nào trong thanh ram mà ta đang quản lý, he he he he he he"

thêm 1 tí nữa là : người ta ko dùng hệ thập phân (decimal, hệ đếm cơ số 10) để viết địa chỉ đâu, nên thui, chuyển qua hệ thập lục phân (hexadecimal , **hệ đếm cơ số 16** nha )

452321 hệ cơ số 10, chuyển lại thành 6E6E1 ở hệ cơ số 16

ở trong C tôi viết là 0x6E6E1

ở ngôn ngữ ASM tôi viết là 6E6E1h << thêm chữ h vài cuối để hiểu hệ cơ số ấy mà

thôi viết là 0006E6E1h đi

tại sao vậy ? tại vì như này nè

trong windows 32bit (xp, vista, 7) thì địa chỉ ảo có độ dài là 32 bit, tương ứng với số hexa có 8 chữ số, thế à, nên tôi viết thêm 0 vào cho dễ hiểu ấy mà

Để ko bị loãng bài viết mình xin trình bày các điều cần nhớ sau đây :

- + Mỗi tiến trình có 1 vùng nhớ ảo riêng
- + Vùng nhớ ảo là 1 ko gian địa chỉ ảo trải dài từ thấp đến cao ( từ 0x0000 -> cao hơn)
- + Ở trong windows 32bit thì ko gian địa chỉ ảo có địa chỉ từ 00000000h trải dài đến 7fffffffh
- + Bạn cần hiểu nó chỉ là ảo, ko phải vùng nào cũng có bộ nhớ vật lý thật đâu nhá,
- + Khái niệm về bộ nhớ phân đoạn : segment offset bạn hãy bỏ qua đi, vì nó quá cũ rồi

## III. Ví dụ vui về địa chỉ ảo

để làm ví dụ vui này bạn cần 2 cái đó là

+ pokemon : <http://forums.congdongcviet.com/atta...3&d=1282105506>  
+ armoney active code là **dot68** : <http://forums.congdongcviet.com/atta...1&d=1282119896>

Khi chơi game, ta thấy điểm hiện lên trên màn hình, vậy thì chắc chắn nó sẽ được lưu trữ ở đâu đó trong bộ nhớ và sẽ có địa chỉ VA cụ thể. Dân lập trình chúng ta gọi chúng là **biến**, và có địa chỉ cụ thể, hj hj

Để thay đổi điểm từ phía app của mình, đầu tiên chúng ta phải tìm được địa chỉ VA của biến điểm này đã nhỉ.

Để tìm được địa chỉ của biến này ko quá khó với 1 tool cơ bản như artmoney (**Chưa có download ở đây**, active code là **dot68**) :

#### Bước 1

Đầu tiên bật pikachu lên chơi lấy 20 điểm và bật artmoney lên, đầu tiên là chọn tiến trình, pikachu ở đây có cái tên là D4S rồi click vào **Search** lên 1 hộp thoại

#### Bước 2

click vào **...** để chọn kiểu dữ liệu, mình hack nhiều lần rồi nên biết nó là kiểu float 4byte, nếu chưa hack bao giờ, các bạn có thể để ALL để tìm với mọi loại dữ liệu

#### Bước 3

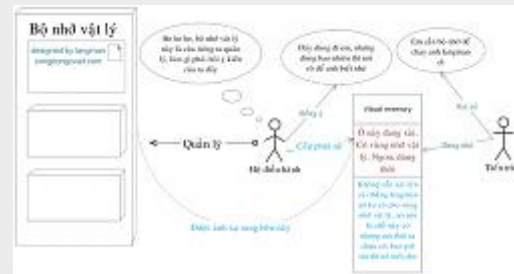
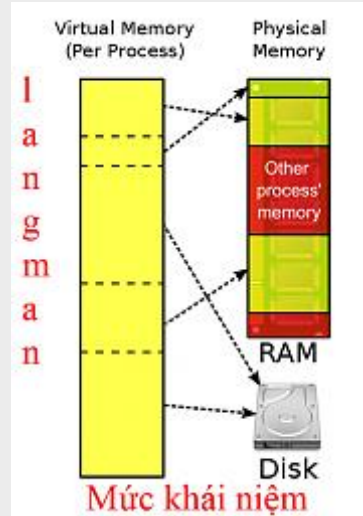
chúng ta sẽ thu được 1 loạt địa chỉ đang chứa giá trị 20, bây giờ chúng ta vào trong game để chơi cho điểm trở thành 40 rồi vào artmoney, click vào nút **Filter** gõ giá trị mới là 40 rồi ok

#### Bước 4

Vậy là ta đã biết địa chỉ của biến điểm là **004B6088**

ai quan tâm đến hack game thì tham khảo bài viết này  
<http://forums.congdongcviet.com/showthread.php?t=35324>

Attached Thumbnails



Đã được chỉnh sửa lần cuối bởi langman : 21-03-2011 lúc 07:29 PM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang





langman  
Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

# Chap II : Tổng quan

## I. Cái nhìn vấn đề

A: Con trở là gì, chả hiểu cái khi khô gì cả, nghe nói khó lắm....

B: Hoặc có bạn học 2,3 buổi xong nói , úi trời dễ ợt ấy mà,.....

thứ 1 đối với A : Con trở dễ ợt ấy mà, chỉ cần bạn theo dõi đầy đủ tut này, làm theo hướng dẫn

thứ 2 đối với B : thôi đi nhé pa, đọc xong cái bài viết này đã, rồi hăng kết luận nha



tôi xì pam linh tinh thế thôi

đi thẳng vào vấn đề đi

### 1. con trở chỉ là 1 biến nguyên bình thường

con trở chỉ là 1 biến nguyên bình thường như cân đường hộp sữa ý bạn à

nó là 1 biến, biến nguyên giá trị của nó là **nguyên**

nó chứa cái được gọi là **địa chỉ ảo** mà ta đã nói ở bên trên đó bạn

ví dụ như là : **0x6E6E1** hoặc **0x4B6088** hoặc **454321**

đó bạn à

sau này nha

dù bạn khai báo

void \*p;

```
char *p;  
hay là  
double *p;  
long long *p;
```

thì p vẫn là 1 biến, nó là 1 biến, biến nguyên,

**2. trong hệ điều hành 32bit thì nó có độ dài là 32 bit,**  
trong windows 32bit (xp, vista, 7) thì địa chỉ ảo có độ dài là 32 bit, tương ứng với số hexa có 8 chữ số,

vì sao lại chỉ có 32bit ?

vì nó cần 32bit là vừa đủ để chỉ trỏ hết vùng nhớ ảo đó

## II. Con trỏ dùng để làm gì?

Vâng, tôi chưa từng bao giờ nghĩ đến 1 câu hỏi đơn giản mà tuyệt vời như này vì tôi luôn .... Nói thế nào nhỉ, tôi luôn..... tôi cứ tiện tay là dùng, hợp lý tôi dùng, cần thiết tôi dùng mà cho đến nay tôi chưa hề nghĩ đến câu trả lời câu hỏi

Con trỏ dùng để làm gì nhỉ

+ à à, đơn giản, đúng như cái bản chất của nó thì nó để chỉ trỏ lung tung trong vùng nhớ ảo của tiến trình hiện tại

+ có người nói với tôi để dùng làm tham biến cho hàm, tôi hok nói gì cả, vì cái câu ni đúng thì đúng với các bạn mới thui, chứ đi sâu vào vấn đề thì lại sai lè ra ý (tại sao xem tiếp ở các chap sau nha)

Tôi ko thể nói rõ 1 cách đơn giản ngay từ đây là con trỏ để làm gì cho bạn, thậm chí cả sau này cũng thế

Nhưng tôi tin chắc rằng mình sẽ mang lại cho các bạn những sự tuyệt vời mà tôi biến đến từ cách sử dụng con trỏ .....

À quên , có 1 điều này cực kì quan trọng : con trỏ chỉ là 1 công cụ, là 1 kiểu dữ liệu, để ta cài đặt các giải thuật, chứ ko phải là 1 giải thuật hay thuật toán, nên câu nói như là "dùng con trỏ để giải bài A", "giải bài tập B bằng con trỏ" là hoàn toàn sai.  
Nói đúng phải là "giải bài tập C sử dụng con trỏ"

*Đã được chỉnh sửa lần cuối bởi langman : 12-12-2010 lúc 03:34 AM.*

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#3



langman

Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

## Chap III : Khai báo

Chà chà, dẫn nhập thật là dài dài, nhưng bạn ơi, hãy chắc chắn với tôi rằng bạn đã cảm thấy ok ở 2 chap đầu (xin đừng đọc lướt qua nó với về thờ ơ) vì đó là tiền đề cực kì quan trọng để bạn có thể vượt qua khỏi mức cơ bản sau này

### I. Cấu trúc khai báo

`kiểudulieu *tenConTro;`

kiểu dữ liệu ở đây có thể là

- + kiểu dữ liệu có sẵn (built-in data type) : int , char , void , double , long , .....
- + kiểu dữ liệu cấu trúc do người dùng định nghĩa (user-defined data type) : struct , union
- + kiểu dữ liệu là lớp do người dùng định nghĩa (C++)
- + kiểu dữ liệu dẫn xuất + kiểu con trỏ hàm (các chap adv nhé)

nhắc lại lần nữa, kiểu dữ liệu này là kiểu dữ liệu của cái vùng nhớ mà nó trỏ đến nha



tenConTro : là tên của con trở nha  
ra khỏi câu khai báo rồi thì **tenConTro** sẽ là tên của con trở,

int \*a;  
ra khỏi câu khai báo này ta sẽ nói : **a là con trở**

## II. Ví dụ

PHP Code:

```
int *a,*p;
```

ta sẽ được 2 con trở a, và p  
xin chú ý để cách tôi viết nhé  
+ a, p là con trở  
+ \*a,\*p không phải là con trở  
+ kí tự \* đứng gần a, đừng gần p, tại sao vậy?

## III. Chú ý

Chú ý 1 :

PHP Code:

```
int *a,b; // thì a là con trở, b là biến nguyên
```

Chú ý 2:

PHP Code:

```
int* a,b; //thì a là con trở, b là biến nguyên  
//và cách viết như này cực kì đáng ghét vì gây ra toàn hiểu lầm đáng ghét
```

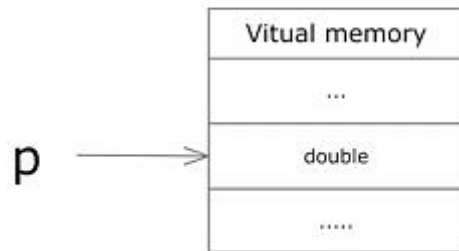
Chú ý 3:

PHP Code:

```
void *a;//đúng , hoàn toàn có con trỏ void nha
```

Attached Thumbnails

```
double *p;
```



Đã được chỉnh sửa lần cuối bởi langman : 09-12-2010 lúc 06:03 PM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#4



langman  
Thành viên mới  
★★★★★

Ngày gia nhập: 06/2007  
Nơi ở: C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

# Chap IV : Khởi tạo

## I. Khởi tạo là gì

Có 1 số bạn sẽ lạ lẫm vì cái tiêu đề khai báo với khởi tạo nghe có vẻ giống nhau..... Nhưng bạn ơi, khai báo (declared, register) và khởi tạo (initialize) hoàn toàn khác nhau nha

```
int a; // khai báo biến a
```

```
int b=2; //khai báo biến b và kết hợp với khởi tạo giá trị cho biến b bằng 2
```

Khi ta khai báo 1 biến thì câu lệnh đầu tiên thiết lập giá trị cho biến đó thì đó là khởi tạo. Trong C03, C++03 trở lên khi ta khai báo 1 biến local, chưa khởi tạo giá trị mà đã đem sử dụng thì sẽ phát sinh lỗi runtime .

Ví dụ đoạn code sau vẫn dịch được, vẫn run nhưng khi chạy sẽ tung ra lỗi "Run-Time Check Failure #3 - The variable 'a' is being used without being initialized."

PHP Code:

```
#include <iostream>
void main()
{
    int a;
    if (a==2) printf("ok"); // có lỗi run-time sinh ra ở dòng này
}
```

## II. Khởi tạo giá trị cho biến con trỏ

cấu trúc khởi tạo:

**TênConTrỏ = ĐịaChỉ;**

+ trong đó tên con trỏ là tên của biến con trỏ

+ địa chỉ là vùng địa chỉ mà ta muốn trở đến

## Ví dụ

**Chú ý 1:** Bản thân p cũng là 1 biến (nguyên), p cũng nằm trong bộ nhớ, cũng có địa chỉ riêng đó bạn à

**Chú ý 2:** Toán tử & ở đây chính xác phải gọi là unary operator &, toán tử & 1 ngôi, nó hoàn toàn khác với toán tử & 2 ngôi (bitwise). Toán tử & 1 ngôi này dùng để lấy địa chỉ của 1 biến. Trước khi động đến lý thuyết về con trỏ, chúng ta đã từng sử dụng toán tử này rồi đó : `scanf("%d",&a);`.

PHP Code:

```
a=3&2 //toán tử & 2 ngôi, là toán tử dạng bitwise
p=&a; // toán tử & 1 ngôi, là toán tử lấy địa chỉ của 1 biến
scanf("%d",&a) // toán tử & 1 ngôi, là toán tử lấy địa chỉ của 1 biến
```

**Chú ý 3:** Có thể viết ví dụ trên ngắn gọn lại thành

PHP Code:

```
int a=1987,p=&a;
```

## III. Có được điều gì sau khi khởi tạo như ví dụ trên

+ Khi giá trị nằm trong p là địa chỉ của a thì ta nói p trỏ vào a

+ Lúc này thì \*p hoàn toàn tương đương với a, người ta coi \*p là bí danh của a, thao tác với \*p cũng như thao tác với a, thao tác với a cũng như thao tác với \*p

ví dụ :

a. câu lệnh `a=2;` hoàn toàn tương đương với câu lệnh `*p=2;`

b. câu lệnh `a++;` hoàn toàn tương đương với `(*p)++`

// chú ý khác với `*p++` nhé, phải cho \*p vào trong đóng mở ngoặc vì toán tử \* có độ ưu tiên thấp hơn ++

c. câu lệnh `b=a+c-9;` hoàn toàn tương đương với câu lệnh `b=(*p)+c-9;`

d. câu lệnh `(*p)=(*p)-1227;` hoàn toàn tương đương với `a=a-1227;`

+Lúc này câu lệnh `scanf("%d",&a);` ta có thể thay bằng `scanf("%d",p);`

## Chú ý : Toán tử \*

Toán tử \* ở đây là toán tử 1 ngôi , tác dụng là truy xuất đến ô nhớ mà con trỏ đang trỏ đến

Để tránh những hiểu lầm ko đáng có, khi có sự nhập nhằng mà bạn ko thể đoán được, bạn hãy thêm cặp `()` nha

`(*p)++`

`a+(*p)*c` // thêm vào cho nó sáng sủa code ra

## IV. Một số trường hợp

1. Hiểu lầm về cách cho p trỏ vào a

2. Cùng trỏ vào 1 biến

3. Con trỏ đa cấp

4. Con trỏ trỏ đến ô nhớ đã biết

## 5. Con trỏ void

Con trỏ void là 1 con trỏ đặc biệt, thích trỏ đi đâu thì trỏ

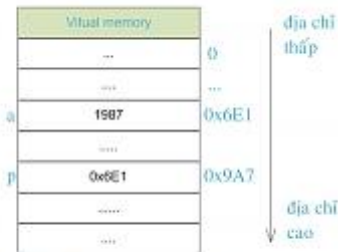
PHP Code:

```
int ham()  
{  
    return 1;  
}  
  
void main()  
{  
    int a;  
    void *p,*q;  
    p=ham;  
    q=&a;  
}
```

Con trỏ voi khác với con trỏ hươu ở chỗ nào ?  
vui lòng đón xem ở các chap tiếp

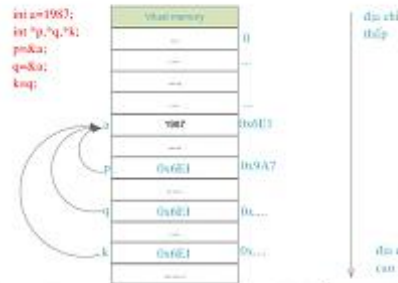
được viết bởi langman đến từ congdongviet.com

```
int a=1987;
int *p;
p=&a;
```



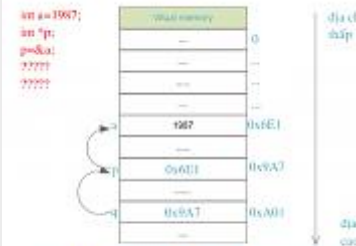
Khi mà giá trị nằm trong p bằng địa chỉ ảo của a ta nói p trỏ đến a

### Hai con trỏ cùng trỏ vào 1 ô nhớ



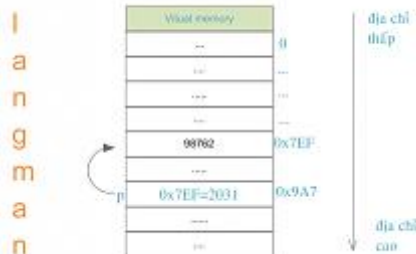
Tuy cùng trỏ vào 1 ô nhớ nhưng 3 biến p,q,k vẫn là 3 biến con trỏ khác nhau  
Được viết bởi langman đến từ congdongviet.com

### Con trỏ trỏ đến con trỏ



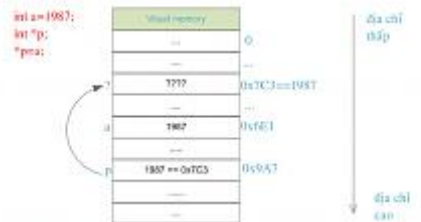
Vui lòng xem ở các chap tiếp theo nhé  
Được viết bởi langman đến từ congdongviet.com

### Trỏ đến 1 ô nhớ đã biết



```
int *p=(int*)0x7EF;
Hoặc char *p=(char*)2031;
Hoặc void *p=(double*)0x7EF;
//riêng con trỏ void ép thế nào
// cũng được hết, he he he
```

Có người nói rằng có thể cho p trỏ vào a bằng cách viết \*p=a. Đây là sai lầm đấy bà con. Sau đây là bản chất của vấn đề



Bạn thấy đó, p trỏ vào 1 ô nhớ mà ko hề biết tới ko hề biết rằng ô nhớ này đã có bộ nhớ vật lý hay chưa ko hề biết rằng nó chứa cái gì trong đấy, nằm ở vùng nhớ nào Do đó, khả năng gây ra lỗi run-time là vô cùng lớn!!!!!!  
Được viết bởi langman đến từ congdongviet.com

Đã được chỉnh sửa lần cuối bởi langman : 18-02-2013 lúc 02:55 PM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#5



langman  
Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

# Chap V : Kiểu dữ liệu con trỏ và các phép toán trên con trỏ

## I. Kiểu dữ liệu con trỏ

Khi ta viết `int *p, b`; chúng ta luôn viết `*` gần `a`, vì sao? vì `*` này là của `p`, `p` là con trỏ, `b` ko phải con trỏ  
kiểu dữ liệu của `b` là `int`  
kiểu dữ liệu của `p` là gì ?????????????? (1)

bạn xem lại hình ảnh của mục 4. Con trỏ trỏ đến ô nhớ đã biết thấy  
`p=(int *)....; (2)`

-----từ (1) và (2) chúng ta có thể nhận thấy điều này, kiểu dữ liệu của `p` là `(int *)`



Thật ra chúng ta đã từng gặp kiểu dữ liệu con trỏ dạng này rồi. Ví dụ khi tra MSDN tôi có được cái này :

PHP Code:

```
char* gets(char* str);
```

Tôi rất tin vào cách viết chuẩn mực của Microsoft, vì thế tôi cũng khuyên các bạn code theo chuẩn mực này :

- + trong câu lệnh khai báo con trỏ tôi viết \* gần tên con trỏ
- + khi viết kiểu dữ liệu tôi viết \* đứng gần kiểu dữ liệu cơ bản : cụ thể ở kiểu dữ liệu trả về của hàm, ở tiêu đề và nguyên mẫu hàm
- + Ở câu lệnh ép kiểu thì manual theo bạn muốn, có thể viết cách ra cho thoáng code

các bạn có thể xem lại nguyên mẫu hàm gets ở bên trên để hiểu thêm về cách viết code này

## II. Các phép toán trên con trỏ

### a. Phép gán

Phép gán đối với con trỏ thì tham khảo phần khởi tạo nhưng có 1 vài yếu tố xâu đây :

- + Tất cả các loại con trỏ đều có phép gán
- + Phép gán với con trỏ yêu cầu vế trái là 1 con trỏ và vế phải là 1 địa chỉ
- + Phép gán yêu cầu sự tương xứng về kiểu dữ liệu, nếu ko tương xứng chúng ta phải ép kiểu

ví dụ `p=(int*)8232;`

`p` có kiểu dữ liệu là `int*`

còn 8232 là 1 hằng số nguyên, nên phải ép kiểu về `int*` rồi thực hiện phép gán

- + Phép gán với 1 con trỏ kiểu `void` ko cần thiết phải tương xứng hoàn toàn về kiểu dữ liệu, `void*` có thể tương ứng với tất cả (như ở ví dụ chấp trước), thậm chí là vượt cấp (vượt hẳn 2 cấp) như ví dụ sau

PHP Code:

```
void *p,**q;  
p=&q;
```

### b. Phép so sánh

Phép so sánh ngang bằng dùng để kiểm tra 2 con trỏ có trỏ vào cùng 1 vùng nhớ hay không, hoặc kiểm tra 1 con trỏ có phải là đang trỏ vào NULL hay không (trong trường hợp cấp phát động, mở file, mở resource,.....)

Phép so sánh lớn hơn nhỏ hơn : `>` , `<` , `>=` , `<=` sử dụng để kiểm tra về độ thấp cao giữa 2 địa chỉ . Con trỏ nào nhỏ hơn thì trỏ vào địa chỉ

thấp hơn.

+ Được quyền so sánh mọi con trỏ với 0, vì 0 chính là NULL

PHP Code:

```
void main()
{
    int a=197,*p=&a;
    double *x;
    p=&a;
    main==0; // học các chap sau để hiểu sâu hơn dòng lệnh này, he he he he he
    p==0;
    x==0;
}
```

+ Ngoài ra thì khi so sánh 2 con trỏ hoặc con trỏ với 1 địa chỉ xác định (số nguyên) cần có sự tương xứng về kiểu dữ liệu

PHP Code:

```
int main()
{
    int a=197,*p=&a;
    double b=0,*x=&b;

    // so sánh 2 con trỏ
    (int)p==(int)x;
    p==(int *)x;
    (double*)p==x;
    (void*)p==(void*)x;
    p==(void*)x;
    (float*)p==(float*)x;

    //so sánh con trỏ với số nguyên
    p==(int*)9999;
    int(p)==9999;

    // phân nâng cao và thâm thúy về con trỏ
    (int)p==int(main);
    p==(int*)main;
    (int(*)())p==main;
    p==(void*)main;
    // bình tĩnh tự tin theo hết tut này bạn sẽ hiểu được cái gì đang xảy ra ở 4 dòng code này
```

+ Con trỏ void có thể đem ra so sánh với tất cả các con trỏ khác

xì pam tí , thử cái này

PHP Code:

```
void main()
{
    int a=197,*p=&a;
    (int(*)())p==main;
}
```

### c. Phép cộng trừ và phép tăng giảm : + += - -= ++ --

Bản chất của việc tăng/ giảm con trỏ p đi 1 đơn vị là cho p trỏ đến ô nhớ bên cạnh phía dưới/trên.

Chú ý:

- + Khi tăng giảm con trỏ p đo 1 đơn vị không có nghĩa là trỏ sang byte bên cạnh
- + Việc tăng giảm con trỏ đi 1 đơn vị phụ thuộc vào kiểu dữ liệu và nó trỏ đến, quy tắc là  $p+1 \gg \text{giá trị chứa trong } p + \text{sizeof(kiểu dữ liệu của biến mà } p \text{ trỏ đến)}$
- + Không có phép tăng giảm trên con trỏ void
- + Không có phép tăng giảm trên con trỏ hàm
- + Không có phép cộng 2 con trỏ với nhau
- + Phép trừ 2 con trỏ trả về độ lệch pha giữa 2 con trỏ

Vậy ta có kết luận như sau : kiểu dữ liệu trỏ đến có tác dụng xác thực sự rõ ràng tất cả các phép toán trên con trỏ (bao gồm cả phép  $= * \&$ )

## III. Ứng dụng

Mình demo trước một ứng dụng của việc thao tác các phép toán trên con trỏ

### ứng dụng duyệt xâu

PHP Code:

```
#include <stdio.h>
#include <conio.h>
#include <ctype.h>

void main()
{
    char xau[200];

    printf("Nhap xau : ");
    scanf("%[a-zA-Z ]",xau); //nếu bạn chưa hiểu dòng lệnh này hãy xem bài viết này để hiểu sâu thêm về scanf
    //http://forums.congdongcviet.com/showthread.php?t=34612
```

```

//Viết hoa xâu (duyệt xuôi)
printf("Viet hoa : ");
for (char *p=xau;*p;p++) //p trở đến xâu; kí tự trở đến khác NULL;p=p+1
    printf("%c",toupper(*p));

//viết đầy đủ sẽ là (char *p=xau;*p!=NULL;p++)
//viết ngắn gọn lại cho độc đáo

//Viết đảo ngược xâu (duyệt ngược)
printf("\nDao nguoc xau : ");
for(char *p=xau+strlen(xau)-1;p>=xau;p--) // cho p trở vào từ cuối cùng; p còn lớn hơn xau;p=p-1
    printf("%c",*p);

```

### ứng dụng đổi số thực thành số nhị phân

Cách 1 : C style

PHP Code:

```

#include <stdio.h>
#include <conio.h>

void nhiphan(float n)
{
    for(int i=0,*temp=(int *) (void*)&n;i<sizeof(n)*8;i++,(*temp)<=&1)
        printf("%d",*temp>=0);
}

void main()
{
    nhiphan(3.9f);
    getch();
}

```

Cách 2: C++ style

PHP Code:

```

#include <iostream>
using namespace std;

void nhiphan(unsigned n)
{
    n>>1?nhiphan(n>>1):0;
    printf("%d",n&1);
}

```

```
void nhiphan(float n)
{
    nhiphan(*(unsigned*)(void*)&n);
}

void main()
{
    nhiphan(3.9f);
    getch();
}
```

Ứng dụng tìm (số float lớn hơn ko) nhỏ nhất  
đây chính là số 00000000 00000000 00000000 00000001

PHP Code:

```
#include <iostream>

int main()
{
    float x = 0;
    char *p = (char*)&x;
    (*p) |= 1;
    std::cout<<x<<std::endl;

    return 0;
}
```

Attached Thumbnails



Đã được chỉnh sửa lần cuối bởi langman : 23-03-2011 lúc 04:19 AM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#6



**langman** ●  
Thành viên mới  
★★★★★

Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

 Tìm hiểu bản chất của con trỏ - từ cơ bản đến nâng cao

# Chap VI : Con trỏ với mảng, xâu, cấp phát bộ nhớ động

## I. Hằng con trỏ - const pointer ????? Con trỏ hằng ,



```

1. void main()
2. {
3.     char *p="Bui Tan Quang";
4.
5.     p++;
6.
7.
8.     (*p)++; <<<<<<<<<<<<<<<<<<< báo lỗi tại đây (không báo lỗi khi biên dịch nhưng có lỗi trong run-time)
9.     p[2]='b';<<<<<<<<<<<<<<<<<<< báo lỗi tại đây (không báo lỗi khi biên dịch nhưng có lỗi trong run-time)
10.
11. }

```

ví dụ tiếp về loại (2) by clamvn

### C++ Code:

Select All | Show/Hide

```
1. char buf[] = "bonjour";
2. char const * p = buf; /* hay const char * p = buf; */
3.
4. p++; /* được */
5. p[4]++; /* ko được, sai */
```

ví dụ về kết hợp by clamvn

### C++ Code:

Select All | Show/Hide

```
1. char buf[] = "bonjour";
2. char const * const p = buf;
3.
4. p++; /* Sai */
5. p[4]++; /* Sai */
```

### Ví dụ tiếp với hàm

### C Code:

Select All | Show/Hide

```
1. void ConvertUnicodeTextToSomething(const unsigned short int *wstr)
2. {
3.     unsigned short int const * p=wstr; //okies
4.
5.     unsigned short int * q=wstr; //báo lô~i
6.
7. }
```



8. }

Ứng dụng lớn nhất của `char const *` đó là chú ý khi khai báo và sử dụng các hàm trả về `const`, nếu ko biết điều này có thể bạn sẽ ko biết cách xài code của người khác khi đang ở trong 1 project lớn và mọi người dùng các hàm của nhau. Chú ý nhé, mấy newbie hay gặp vấn đề với cái này lắm nhé

C Code:

Select All | Show/Hide

```
1. const char * HamGiDoCuaNguoiKhacViet(void)
2. {
3.     return "abc";
4. }
5.
6.
7. void HamCuaToi(void)
8. {
9.     //gọi và sử dụng đến kết quả hàm bên trên thế nào?
10.
11.     char const * pstr=HamGiDoCuaNguoiKhacViet(); // thế này nè
12. }
```

## II. Mảng liên quan gì đến con trỏ và cho vào bài viết này chi ?

Khi ta khai báo mảng thì tương đương với : xin cấp phát 1 vùng nhớ có kích thước như bạn khai báo và khai báo ra 1 hằng con trỏ trỏ vào đầu vùng nhớ đó

**int a[100];**

+ có thể coi a là 1 hằng con trỏ trỏ vào phần tử thứ 0 của mảng nhé, a mang đầy đủ tính chất của 1 hằng con trỏ nhưng có thêm 1 số khác biệt nhỏ (ví dụ khi dùng `sizeof`)

+ các phép toán nhằm làm a trỏ tới vùng khác (thay đổi giá trị của a) là ko thể (`++ -- =` )

- + a tương đương với &a[0]
  - + a+i tương đương với &a[i]
  - + \*a tương đương với a[0]
  - + \*(a+i) tương đương với a[i]
- Chú ý : trình biên dịch luôn hiểu a[i] là \*(a+i)

## Biết điều này để làm gì ?

Mình demo 2 điều

1. nhập mảng

PHP Code:

```
#include <stdio.h>
#include <conio.h>

void main()
{
    float a[100];
    int n;
    //nhập n
    printf("Nhap n :");
    scanf("%d",&n);
    // nhập mảng
    for(int i=0;i<n;i++)
    {
        printf("Nhap vao phan tu thu %d",i+1);
        scanf("%f",a+i);
    }

    // xuất mảng
    printf("mang vua nhap : \n");
    for(int i=0;i<n;i++)
        printf("%f ",*(a+i));
```

## 2. bài toán vui

### PHP Code:

```
#include <stdio.h>
#include <conio.h>

void main()
{
    int a[100]={0,1,2,3,4,5,6};
    printf("%d",2[a]); //in ra 2, tại sao vậy ?

    getch();
}
```

chắc chắn lúc nhìn thấy 2[a] ko ít người sẽ thấy là lạ, nghĩ nó là lỗi,.....

có người thì nghĩ là nó in ra 2, nhưng tại sao vậy, thì nhiều người

thật ra :  $2[a]$  trình biên dịch sẽ hiểu là  $*(2+a)$

$*(2+a)$  hoàn toàn tương đương với  $*(a+2)$

mà  $*(a+2)$  chính là  $a[2]$

vậy  $2[a]$  cũng đơn giản là  $a[2]$

>>> cool phải hok nào

Ngoài 2 điều này ra còn nhiều cái thú vị lắm, bạn hãy thử khám phá xem sao

### III. À, thế còn con trỏ hằng là cái gì thế ?

(đây là phần nâng cao)

con trỏ hằng là 1 optional ability trong lập trình, tác dụng của nó tựa như là (gần như thôi, ko thể bằng được) phương thức hằng trong C++;

ý nghĩa là 1 con trỏ, trỏ đến 1 ô nhớ, nhưng ko được quyền thay đổi giá trị của ô nhớ đó!!!!!!!!

### PHP Code:

```
int a=3;
const int *p;
p=&a;          <<<<<<<<<<<< bản thân p thì có thể thay đổi, cho p gán vào chỗ khác được nhưng
(*p)++;<<<<<<<<<<<< báo lỗi tại đây!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

đổi với các bạn mới thì chắc ko hiểu hoặc có hiểu cũng nghĩ là úi giờ ới, biết làm quái gì!!! hì hì.....

Ví dụ điển hình nhất ở đây là hàm strlen của chúng ta

PHP Code:

```
int strlen(const char *Str)
```

Khi bạn code trong 1 project C lớn 1 tí hoặc lớn nhiều tí, hơ hơ, giả sử bạn có 1 hàm, thao tác với 1 mảng , **hàm này chỉ đọc mảng thôi, ko làm thay đổi các giá trị trong mảng** . Và quan trọng là, khi share code cho các bạn khác trong cùng project, làm sao để họ biết điều này ??????????

Vậy ta sẽ cài đặt hàm của mình như sau

demo ví dụ mẫu

PHP Code:

```
// đối với trường hợp hằng con trở là tham số hình thức thì
// void ham(const int *) và void ham(int const *)
// là như nhau, từ const khi đóng góp vào trong tham số hình thức là như nhau
void ham(const int *a,int n)
{
    //xử lý gì đó
}
void main()
{
    int a[100]={1,2},n=2;

    ham(a,n); // khi sử dụng hàm này tôi hiểu là, à, nó ko thay đổi mảng a của tôi đâu
    //yên tâm xài, nếu có lỗi gì đó thì ko phải sinh ra từ đây
}
```

## IV. Thế còn chuỗi ký tự thì sao ?

+ Chuỗi ký tự là trường hợp đặc biệt của mảng 1 chiều khi mà cách thành phần của mảng là 1byte

+ Chuỗi ký tự kết thúc bằng NULL. NULL là 1 ký tự đặc biệt có mã là 0,

Có 3 cách viết NULL trong C như sau : **NULL** , **'\0'** , **0**

### A. Sai lầm thường gặp khi làm việc với chuỗi ký tự

đối với chuỗi ký tự thì các bạn phải nhớ được những trường hợp sau

1. Chưa có bộ nhớ đã sử dụng như đúng rồi >>>> sai lè ra

PHP Code:

```
char *xau;
gets(xau); // vẫn biên dịch được
//nhưng khi chạy sẽ sinh ra lỗi run-time
```

```
// ở 1 sô'trình biên dịch cùi bắp ngày xưa thì có thể'ko bị lỗi đâu
// nhưng sai thì vẫn là sai, code này sai thuộc loại chưa cấp phát
```

## 2. Thay đổi giá trị của một hằng >>>> sai lè ra tiếp

PHP Code:

```
char *xau="langman-congdongcviet";
xau[6]='A';// vẫn biên dịch được
//nhưng khi chạy sẽ sinh ra lỗi run-time
// lỗi này là lỗi cố'tình thay đổi giá trị của 1 hằng
```

Nguyên nhân sâu xa của vấn đề như sau :

khi khai báo char \*xau="langman-congdongcviet"; thì bản chất là

+ trong vùng nhớ data của chương trình sẽ có 1 hằng chuỗi "langman-congdongcviet" . <<<< là hằng chuỗi, đã là hằng thì ko thể bị thay đổi

+ cho con trỏ xau trỏ đến đầu của vùng nhớ đó.

Câu lệnh tiếp theo xau[6]='A'; cố tình thay đổi giá trị của hằng , rõ ràng là sinh ra lỗi rồi

## 3. Cố tình thay đổi giá trị của hằng con trỏ <<<<<<< sai lè tiếp nữa

PHP Code:

```
char xau[100];
xau="bùi tấn quang"; // không biên dịch được
// vì phép toán trên có nghĩa là khai báo 1 chuỗi "bùi tấn quang" trong vùng nhớ code
// rồi sau đó cho hằng con trỏ xau trỏ vào đó
// rất tiếc xau là hằng con trỏ nên ko thể'trỏ đi đâu khác được
// ngoài vị trí đã được khởi tạo trong câu lệnh khai báo
```

chú ý char xau[100]="bùi tấn quang"; hoặc char xau[100]={0}; thì hoàn toàn hợp lệ

trích :

## 4. Dùng phép toán so sánh để so sánh nội dung 2 xâu <<<<<<< sai lè tiếp nữa

C++ Code:

Select All | Show/Hide

```
1. void main()
2. {
3.     char xau[100]="quangxeng";
4.     if (xau=="quangxeng") ... // code này ko sai về` ngữ pháp, ko sinh ra lỗi runtime
5.     //nhưng mang lại kết quả' ko như người dùng mong muốn
6.     // vì theo mục b. ở trên ta có
7.     //Phép so sánh ngang bằng dùng để' kiểm tra 2 con trỏ' có trỏ' vào cùng 1 vùng nhớ hay không,
8.     //hoặc kiểm tra 1 con trỏ' có phải là đang trỏ' vào NULL hay không
9.     //(trong trường hợp cấp phát động, mở file, mở resource,.....)
```

```
10. // chứ ko pha'i là phép so sánh nội dung cu'a xâu
11. //đê' so sánh nội dung cu'a xâu ta pha'i dùng những hàm strcmp (string compare) hoặc stricmp
12. // hoặc những hàm bạn tự định nghĩa
13.
14. }
```

Phụ lục

## B. Biết thêm 1 style duyệt xâu mới

Xem chap V, phần 3

mở rộng ứng dụng duyệt xâu để làm bài xâu sau : Nhập vào dạng "họ đệm tên", viết ra màn hình "Tên Đệm Họ"

PHP Code:

```
#include <stdio.h>
#include <conio.h>

void main()
{
    char xau[100],*p=xau,*q,*i;
    printf("Nhap : "),scanf("%[a-z ]",xau); // nhap vao "ho dem ten"

    while(*p!=' ') p++;
    q=xau+strlen(xau)-1;
    while(*q!=' ') q--;

    //viet hoa
    *xau=toupper(*xau);
    p[1]=toupper(l[p]);
    q[1]=toupper(l[q]);

    //viet
    printf("Xuat :%s",q); //ten
    for(i=p;i<=q;i++) printf("%c",*i); // dem
    for(i=xau;i<p;i++) printf("%c",*i); // ho

    getch();
}
```

## V. Thế còn cái từ cấp phát động thì sao nhỉ? Nghe quen quá đi...

### 1. Bản chất của việc cấp phát động.

Đầu tiên để hiểu được cấp phát động, bạn hãy nghe lời tôi, tạm thời bỏ qua tất cả các lý thuyết, các câu lệnh, các code mà bạn biết, tạm thời

chưa quan tâm đến nó vội, hãy đọc cho tôi bài viết này đã : <http://forums.congdongcviet.com/showthread.php?t=36221> (rất cần thiết đấy)

(và làm ơn ko hỏi đáp, thắc mắc gì trên tất cả các topic mình hướng dẫn, cần đặt câu hỏi hãy qua box hỏi đáp lập, mình sẽ tận tình trả lời bạn bằng tất cả những gì mình biết)

## 2. cấp phát động như thế nào (cú pháp làm ơn xem sách giáo khoa nhé)

### a. C

contro = (ép kiểu) malloc(...)

Trong C chúng ta cấp phát động chủ yếu sử dụng các hàm trong alloc.h

các bạn có thể tham khảo các hàm ở đây

<http://forums.congdongcviet.com/showpost.php?p=30657>

chú ý là :

+ malloc trả về 1 địa chỉ đến 1 vùng nhớ và coi vùng nhớ này là void \*, nên trong câu lệnh malloc luôn đi kèm với việc ép kiểu

+ cấp phát là luôn phải đi kèm với giải phóng, ở đâu cũng thế, malloc là phải free, ok ? Code mà để thoát chương trình rồi chưa giải phóng cho dù là có hệ thống có tự giải phóng đi nữa vẫn bị coi là bad!!!!

+ Trong java chỉ cần cho reference = null là nó giải phóng nhưng trong C thì bắt buộc phải có thao tác giải phóng free()

### b. C++

trong C++ chúng ta dùng new và delete để cấp phát động

new và delete về cú pháp tham khảo trong sách

Câu hỏi của quyết 1991 : sự khác nhau giữa malloc và new?

Trả lời :

new và malloc khác nhau cực kỳ nhiều đó các bạn à

sơ bộ như sau, chưa phân tích kĩ

malloc là **hàm**, cấp phát trả về kiểu **void \***, malloc thì **ko gọi hàm tạo**

free ko gọi hàm hủy

malloc trả về NULL nếu thất bại

new là toán tử, new gọi hàm tạo, new có thể được đa năng hóa (nạp chồng),

new ném ra exception nếu thất bại

toán tử new và toán tử new[] ko có khả năng realloc

## VI. Mảng 2 chiều, bản chất như thế nào, khác gì mảng một chiều ?

cũng chả cần nói nhiều làm gì, chỉ cần bạn xem cái này là hiểu rồi

khi khai báo như trên ta có a trở vào `a[0][0]`

Con trở đa cấp và mảng 2 chiều!!! Bản chất nó thế nào ta ơi???? Có 1 vài hiểu lầm giữa con trở đa cấp, con trở mảng, nó ra sao? Vui lòng xem chap tiếp

Update 13/4/2013: Chứng minh có thể coi tên mảng là hằng con trở (Chứng minh này chỉ dành cho các adv pointer, các newbie đừng đọc vội nhé. để khi các bạn thành thục rồi quay lại đọc cũng okies):

Cách 1 chứng minh thuận

a. Chứng minh a là 1 con trở (tất nhiên có 1 số sự khác biệt so với con trở thuần, nếu ko nó đã ko define lên mảng làm gì)

Đầu tiên : a có các tính chất cơ bản của con trở với unary operator \*, unary operator [] dùng để truy xuất

+ a tương đương với `&a[0]`



+ a+i tương đương với &a[i]  
+ \*a tương đương với a[0]  
+ \*(a+i) tương đương với a[i]

Và đối với static

++> a ở đây là 1 hằng,  
++> giá trị của hằng đó chính là địa trị của ô đầu tiên trong 100 ô nhớ kia,  
vậy đây có phải đã đủ điều kiện là 1 hằng con trỏ ko?

Tiếp theo đối với local :

gần giống như static, nhưng ở đây ko a lại ko phải là 1 hằng số fix cứng như trên vì nhớ nằm trong stack, tất nhiên rồi, code :

C Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3. void main(void)
4. {
5.     int a[100]={3,1,2,3,4,5,6,7,8};
6.     int *pr;
7.
8.     //chứng minh a có 1 ít adn di truyên cu'a int *
9.     pr=123; // lờ~i
10.    pr=main; // lờ~i
11.    pr='a'; // lờ~i
12.    pr=void;// lờ~i
13.    pr =(int *)123;//ok built được
14.    pr =(int *)main;//ok built được
15.    pr =(int *)'a';//ok built được
16.    pr= a;////ok built được
17.
18.    //chứng minh giá trị cu'a a tro' đến đầu vùng nhớ
19.    int x;
20.    printf("%X\nHay nhap so hexa van vua thay : ",a);
21.    scanf("%X",&x);// ta lấy luôn cái số đó lưu vào x
22.    int *p=(int*)x;
23.    (*p)++;// ta thấy số đó tro' vào vào phần từ đầu tiên cu'a mảng
24.    printf("%d ",a[0]);
25.    getch();
26. }
```

b. Chứng minh a là 1 hằng (cái này chắc chắn cần chứng minh nhĩ, vì nó base quá mà)

C Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3. void main(void)
```

```

4. {
5.     int a[100]={0,1,2,3,4,5,6,7,8};
6.     a++; //nếu bật dòng này lên là lỗi, vậy a ko ++ -- được, rõ ràng a được trình biên dịch coi là hằng
7.     a--; //nếu bật dòng này lên là lỗi, vậy a ko ++ -- được, rõ ràng a được trình biên dịch coi là hằng
8.     int *pxx=a; //đem a gán cho int * được, vậy rõ ràng base của a là 1 int * (giống giống đa hình)
9.     getch();
10. }

```

## Cách 2 chứng minh bằng phản biện

Giả sử a ko phải là con trỏ, a sẽ ko thể thỏa mãn 1 số vấn đề sau

C Code:

Select All | Show/Hide

```

1. #include <stdio.h>
2. #include <conio.h>
3. void main(void)
4. {
5.     int a[100]={0,1,2,3,4,5,6,7,8};
6.     printf("%d",2[a]); ////////////////////
7.     getch();
8. }

```

C Code:

Select All | Show/Hide

```

1. #include <stdio.h>
2. #include <conio.h>
3. void ham(int *arr)
4. {
5. }
6. void main(void)
7. {
8.     int a[100]={0,1,2,3,4,5,6,7,8};
9.     ham(a); // tại sao lại gọi ham(con trỏ' nguyên) đối với a được
10.    getch();
11. }

```

C Code:

Select All | Show/Hide

```

1. #include <stdio.h>
2. #include <conio.h>
3. void ham(int *arr)
4. {
5. }
6. void main(void)
7. {
8.     int a[100]={0,1,2,3,4,5,6,7,8};
9.     int *x;

```

```
10.     x=a;//biến = hằng giá trị tương đương hoặc biến có kiểu gâ`n tương đương
11.     //ko có bất kì ca`nh báo warning gì ở` đây với vs2012 nhé
12.     _getch();
13. }
```

....More trích dẫn từ bạn kimcy92

💬 Nguyên bản được gửi bởi kimcy1992 ▶▶

Đã đọc hết toàn bộ các cuộc trao đổi và đọc lại hết tài liệu được coi là tin tưởng, hay kiểm chứng code em thấy anh đưa ra là không sai trích xuất thêm một ít tư liệu trong quyển how to Programming C++ 8th có đoạn nói về mối quan hệ giữa con trỏ và mảng như sau

3rd Party Hosting has been temporarily disabled.



to unlock your account visit:  
[photobucket.com/p500](https://photobucket.com/p500)

...mà đọc lại sách thầy ất cũng nói tên mảng là một hằng địa chỉ, chứa phần tử đầu tiên của mảng. Thêm cái ảnh nữa vẫn trong quyển how to java kia, nhưng có cái lưu ý em thấy rất đúng viết sao cho dễ sửa đổi nhận biết tường minh nhất như việc anh langman luôn muốn hướng tới

3rd Party Hosting has been temporarily disabled.



to unlock your account visit:  
[photobucket.com/p500](http://photobucket.com/p500)

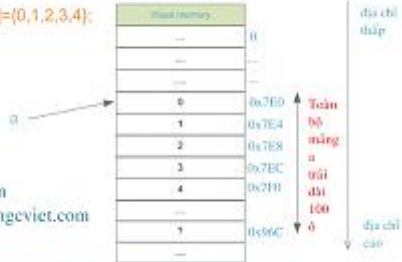
Chọn trang cpp có tên tuổi

<http://www.cplusplus.com/doc/tutorial/pointers/>

*Because numbers is an array, so it operates as a constant pointer, and we cannot assign values to constants.*

Mảng một chiều

```
int a[100]={0,1,2,3,4};
```



- + Các phần từ được sắp xếp liên tiếp trên bộ nhớ.
- + Là đặc điểm cơ bản của mã ngữ nghĩa.
- + Có thể sử dụng toán tử sizeof đối với mảng trên để trả về kích thước của mảng.
- + Do đó với 1 con trỏ vào đầu mảng là ta đã có thể kiểm soát được cả mảng rồi.
- + bằng cách lấy tên mảng + 1 là độ lệch.
- + Việc sử dụng [] để kiểm soát cũng không khác chỉnh là +1 mà thôi mà.

m-by-n matrix

Diagram illustrating a matrix  $a_{i,j}$  with  $m$  rows and  $n$  columns. The row index  $i$  is labeled vertically on the left, and the column index  $j$  is labeled horizontally at the top. A blue arrow labeled  $i$  changes points downwards, and a red arrow labeled  $j$  changes points to the right. The matrix elements are shown as  $a_{1,1}, a_{1,2}, a_{1,3}, \dots$  in the first row,  $a_{2,1}, a_{2,2}, a_{2,3}, \dots$  in the second row,  $a_{3,1}, a_{3,2}, a_{3,3}, \dots$  in the third row, and so on, with vertical dots indicating further rows and columns.

langman-congdongcviet

```
int a[3][2] = {1,2,99,54,5}, //a[dòng][cột]
```

1	2
99	54
5	0

langman-congdongcviet

Địa chỉ VA	Giá trị trong ô nhớ	Tên biến tương ứng
0xC0	1	a[0][0]
0xC4	2	a[0][1]
0xC8	99	a[1][0]
0xCC	54	a[1][1]
0xD0	5	a[2][0]
0xD4	0	a[2][1]

```
{
    static int ar[100];
    int *par;
00831ABC mov     dword ptr [ebp-3Ch],B392C8h
    ar[2]=3;
00831AC3 mov     eax,4
00831AC8 shl     eax,1
00831ACA mov     dword ptr [eax+8392C8h],3
    ham(ar);
00831AD4 push    B392C8h
00831AD9 call    ham (0831145h)
00831ADE add     esp,4
}
```

Name	Value
EAX	00000001
ESP	006CF810
EBP	00000000
ESI	006CF908

Đã được chỉnh sửa lần cuối bởi langman : 19-04-2013 lúc 08:57 PM.

$$\begin{array}{cc} \wedge & \wedge \\ & \text{—} \end{array},$$

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#7



langman  
Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

# Chap VII : Con trỏ với hàm, con trỏ hàm

## I. Hàm cũng có địa chỉ

Khi 1 chương trình(1 pe file) chạy (tiến trình) thì các hàm nằm bên chương trình đó được được load lên không gian nhớ ảo, VA space, chúng nằm trong vùng nhớ code.

Các bạn có thể tham khảo hình dưới đây , hình ảnh khi debug 1 ứng dụng với ollydbg và debug 1 ứng dụng bằng IDE VS2010 :

## II. Con trỏ hàm

Con trỏ hàm là 1 điều thú vị trong C/C++. bản chất của con trỏ hàm cũng là 1 con trỏ có định kiểu. ta có thể sử dụng con trỏ hàm để gọi hàm (invoke ) khi đã biết địa chỉ của hàm

## a. gọi nội ứng dụng

demo 1 ví dụ

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3. int min(int a,int b)
4. {
5.     if (a>b) return a;
6.     return b;
7. }
8. void main()
9. {
10.    int (*p)(int,int);
11.    p=min;
12.    printf("min của 4 va 5 la %d",p(4,5));
13.    getch();
14. }
```

Chú ý : khi khai báo ta phải dùng toán tử () với ý nghĩa là \* này thuộc về p, là 1 con trỏ hàm. int (\*p)(int,int);

## b. gọi từ ứng dụng khác (bản chất thì vẫn là nội nhưng ở 1 hình thái khác, remote + nội)

Bạn có thể ý thấy auto game võ lâm ko ? Làm sao khi ta ấn Ctrl+Z nó sẽ mở hòm đồ ra ?

nguyên tắc của nó như sau : nó sử dụng kĩ thuật cài hook để cài 1 thread vào trong game võ lâm.

thread này khi người dùng ấn nút Ctrl+Z nó sẽ gọi hàm mở hòm đồ có sẵn trong game võ lâm.

giả sử a có hàm dạng void hamMoHomDo(int a); tại địa chỉ 0x873AB chẳng hạn thì a sẽ làm thế này

```
void (*p)(int);
p=(void (*)(int)) 0x873AB;
p(3); //gọi hàm với tham số là 3
```

## III. Hằng con trỏ hàm

Khái niệm hằng con trỏ hàm cũng gần gần giống như khái niệm hằng con trỏ với mảng 1 chiều,

khi bạn khai báo 1 hàm, thì tên của hàm chính là 1 **hằng con trỏ hàm**, con trỏ này trỏ cố định vào vùng nhớ của hàm. Vâng, đó là lý do vì sao ở code bên trên, tôi có thể có những dòng lệnh này

PHP Code:

```
(int)p==int(main);
p==(int*)main;
(int(*)())p==main;
p==(void*)main;
```


```
p==(void*)main;
```

chúng ta thấy đó, chúng ta khai báo ra 1 hàm main. vậy rõ ràng ta có 1 hằng con trỏ main, là 1 hằng thì ta hoàn toàn có thể sử dụng để so sánh rồi

## IV. Ứng dụng của con trỏ hàm

Con trỏ hàm được ứng dụng trong nhiều trường hợp khác nhau khá rộng rãi. Sau đây mình xin tiến cử vài ví dụ điển hình

+ Trường hợp đơn giản tất cả chúng ta đều sử dụng ko ít lần rồi, nhưng vẫn ko hiểu ko biết là mình dùng, đó là `cout<<endl;`

 Nguyên bản được gửi bởi langman,[url

<http://forums.congdongcviet.com/showthread.php?t=24853>[/url]

*endl, hex, oct được định nghĩa như nào ?*

hôm nay có người hỏi tôi câu hỏi rất hay như này  
endl nó được định nghĩa như này ?

PHP Code:

```
ostream& endl ( ostream& os )  
{  
    os.push( '\n' );  
    return os;  
}
```

vậy thì câu lệnh

`cout<<endl;`

ko hiểu thằng endl được truyền tham số vào như nào



câu hỏi khá hay và khá ảo, bạn ơi vấn đề là thằng toán tử << có 1 hàm overload như này  
tôi demo lại cho dễ hiểu nha

PHP Code:



```
friend ostream& operator<<(ostream &os, ostream& (*p)(ostream& ) )
{
    return p(os);
}
```

## + Sử dụng trong các hàm mẫu, lớp mẫu , có tính tùy chọn cao

\* bạn đã bao giờ nghe nói về hàm qsort trong namespace std chưa, tại sao khi sử dụng nó ta lại phải truyền vào 1 tên hàm , hay nói chính xác là 1 hằng con trỏ hàm?

mình xin đề mô 1 cái đơn giản (khà khà)

C++ Code:

Select All | Show/Hide

```
1. #include <iostream>
2. using namespace std;
3.
4. #include <stdio.h>
5. #include <conio.h>
6. void xapxep(void *a,int sizeofElement,int n,int (*hamsosanh)(void*,void*))
7. {
8.     int i,j;
9.     void *temp=new char[sizeofElement];
10.    for(i=0;i<n-1;i++)
11.        for(j=i+1;j<n;j++)
12.            if (hamsosanh((char*)a+i*sizeofElement,(char*)a+j*sizeofElement)>0)
13.            {
14.                memcpy(temp,(char*)a+i*sizeofElement,sizeofElement);
15.                memcpy((char*)a+i*sizeofElement,(char*)a+j*sizeofElement,sizeofElement);
16.                memcpy((char*)a+j*sizeofElement,temp,sizeofElement);
17.            }
18.    delete[] temp;
19. }
20.
21.
22.
23. int hamsosanhungdung1(void *a,void *b)
24. {
25.     return (*(double*)a)>(*(double*)b);
26. }
27.
28. int hamsosanhungdung2(int *a,int *b)
29. {
30.     return (*a)>(*b);
31. }
32. void main()
33. {
```

```

34.     double a[100]={1.,2.,3.,4.,6.,5.};
35.     int n=6;
36.
37.     xapxep(a,sizeof(double),n,hamsosanhungdung1);
38.
39.     for(int i=0;i<n;i++)
40.         cout<<a[i]<<" ";
41.
42.     cout<<endl;
43.     int b[100]={1,2,6,3,5,4};
44.     int m=6;
45.     xapxep(b,sizeof(int),m,(int*)(void*,void*))hamsosanhungdung2);
46.
47.     for(int i=0;i<n;i++)
48.         cout<<b[i]<<" ";
49.
50.     getch();
51. }

```

+ Sử dụng để gọi hàm trong các dll mà ko có thư viện nhập

tham khảo bài viết <http://forums.congdongcviet.com/showthread.php?t=47180>

+ Sử dụng để gọi hàm trong 1 ứng dụng khác khi đã biết địa chỉ của hàm đó

(xem mục II)

## V. Con trỏ với hàm (quan trọng)

### 1. Overview lại về hàm trong C

### 2. Sai lầm trong suy nghĩ

Có nhiều thật nhiều người nói rằng trong C, ta có thể sử dụng con trỏ trong tham số của hàm như là 1 tham biến, qua hàm ta có thể thay đổi được giá trị của tham số.

tôi xin khẳng định lại, điều này thật là 1 hiểu lầm, sai lầm trong suy nghĩ, 1 sự hiểu biết nông cạn, 1 câu phát biểu kiểu ù ù cạc cạc!!! 😂😂😂

Nguyên nhân

+ Hàm trong C ko hề có tham biến, hàm trong C đều hoạt động theo nguyên tắc sau :

Khi gọi hàm, 1 bản sao của tham số được tạo ra (cấp phát vùng nhớ mới, copy giá trị sang. quá trình này theo giáo trình của đại học FPT gọi là shadow copy, là 1 yếu tố cần quan tâm, 1 C/C++ Developer đừng bao giờ quên điều này), và hàm sẽ làm việc với bản sao này

(trong C++ nó sẽ dùng hàm tạo sao chép để tiến hành quá trình shadow copy này)

+ Vậy khi làm việc với con trỏ thì hàm làm thế nào

vâng, hàm vẫn cứ làm theo nguyên tắc 1 và 1 bản sao của con trỏ được tạo ra, và hàm làm việc với bản sao hàm, và **trước khi gọi hàm con trỏ trỏ vào đâu thì nó vẫn được trỏ vào đấy** chứng minh :

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4. int ham(int *a)
5. {
6.     *a=2;
7.     a++;
8. }
9. void main()
10. {
11.     int *a;
12.     printf("Truoc : %x",a); //trước và sau khi gọi hàm
13.     ham(a);                //con trỏ a trỏ vào đâu
14.     printf("Sau %x",a);    // thì nó vẫn trỏ vào đó
15.     getch();
16. }
```

+ Vậy tại sao lại có sự thay đổi và tại sao lại sử dụng con trỏ trong hàm? Con trỏ ko thay đổi thì cái gì thay đổi được ?

vâng, các bạn chú ý nhé, **giá trị nằm trong vùng nhớ trỏ đến thay đổi**. Vâng đúng thế đấy bạn à, do biến của ta nằm trong vùng nhớ được trỏ đến nên nó được thay đổi

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4. int ham(int *a)
5. {
6.     *a=2; // làm việc với địa chỉ nhận được
7. }
8. void main()
9. {
10.     int n;
11.     ham(&n); // truyền địa chỉ của n vào đây
12.     // do đó sau hàm này n =2
13.     getch();
14. }
```

### 3. Sai lầm trong hành động

Một trong những sai lầm cơ bản nhưng lại hay gặp đó là ví dụ sau.

sai lầm vì trong hàm chúng ta cấp phát bộ nhớ rồi cho bản sao đang làm việc trở đến. ra khỏi hàm rồi thì x của ta vẫn chưa có trở vào bộ nhớ nào cả

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4. void nhap(int *a,int n)
5. {
6.     //a=new int[n]; //sai lầm
7.     a=(int*)malloc(n * sizeof(int)); //sai lầm
8.     for(int i=0;i<n;i++)
9.         cin>>a[i];
10. }
11. void main()
12. {
13.     int *x;
14.     int n=6;
15.     nhap(x,n);
16.     //xuất
17.     delete[] x; // sa'n sinh ra lỗi run-time , tung là 1 exception, do x chưa tro' vào đâu mà đòi gia'i phóng
18. }
```

## VI. Vậy tôi phải làm thế nào để mà thay đổi giá trị của 1 con trỏ qua 1 hàm

Vâng, hôm nay có người bạn hỏi mình như vậy, hì hì, lại nhớ ra bài này mình chưa trả lời, vậy nên tôi đề xuất ra đây 2 cách để có thể thay đổi giá trị của 1 con trỏ qua 1 hàm

Cách 1 : dùng tham chiếu trong C++

C++ Code:

Select All | Show/Hide

```
1. void ham(int *&a)
2. {
```

```
3.     a=new int[100];
4. }
5. void ham(int **&a)
6. {
7.     a=new int*[100];
8. }
```

xin chú ý là \* đứng trước &

Cách 2 : up level của \* dùng con trỏ cấp cao hơn con trỏ hiện tại

Cách 2 này mình chỉ demo thôi, bạn cần phải đọc chi tiết ở chap con trỏ đa cấp

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4. void ham(int **a)
5. {
6.     *a=(int*)malloc(100*sizeof(int));
7.     //a[0]=(int*)malloc(100*sizeof(int));
8.     // 2 cách nay như này
9. }
10.
11. void main()
12. {
13.     int *a;
14.     ham(&a);
15.
16.     free(a);
17. }
```

## VII. Nâng cao về con trỏ hàm, mảng con trỏ hàm và kĩ năng phân tích vấn đề

mới các bạn đọc 3 bài viết sau

Giải thích ý nghĩa của dòng lệnh khai báo int(\*) : <http://forums.congdongcviet.com/showthread.php?t=49779>

Kĩ năng phân tích vấn đề : <http://forums.congdongcviet.com/showthread.php?p=117404>

So sánh (\*ptr)[10] và \*ptr[10] trong C! : <http://forums.congdongcviet.com/showthread.php?t=34085>

Hỏi đáp :

Nguyên bản được gửi bởi ddatduong

a ơi, có thể giải thích dùm e cái lệnh này là như nào ko?

C Code:

Select All | Show/Hide

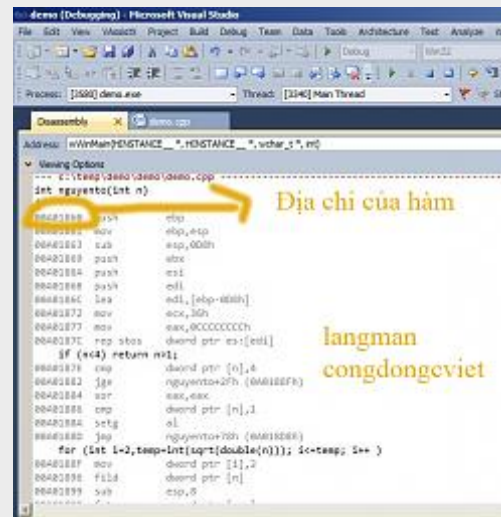
1. `void (*p)(int);`
2. `p=(void (*)(int)) 0x873AB; //<< lệnh này ạ`

C Code:

Select All | Show/Hide

1. `void (*p)(int);` // khai báo ra 1 con trỏ hàm
2. // hàm này có dạng [COLOR="Red"]void ham(int a)[/COLOR]
3. // vậy thì con trỏ để trỏ tới hàm này phải có dạng [COLOR="Red"]void (\*)(int)[/COLOR]
4. // ở đây ta có thể hiểu ca' cùm đây là kiểu dữ liệu
5. `p=(void (*)(int)) 0x873AB;`
6. // 0x873AB là địa chỉ của 1 hàm nào đó mà ta qua quá trình debug phát hiện ra nó nằm tại địa chỉ trên
7. // nó là 1 số nguyên nên ta ép kiểu để nó về đúng kiểu dữ liệu với p
8. //câu lệnh này có ý nghĩa là cho p trỏ vào đầu hàm đó,
9. //sau khi trỏ rồi, nếu ta `p(2)` chính là gọi hàm với tham số thực là 2

Attached Thumbnails



Đã được chỉnh sửa lần cuối bởi langman : 30-03-2011 lúc 12:29 PM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:55 PM

#8



langman  
Thành viên mới



Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

# Chap VIII : Con trỏ đa cấp

## I. Con trỏ đa cấp là gì ?

Mình cũng ko rõ định nghĩa của nó , nhưng ta có thể tạm hiểu đó là những con trỏ có dạng 2 hoặc nhiều \*

ví dụ :

C++ Code:

Select All | Show/Hide

```
1. int **a; // cấp 2
2. char ***b; // cấp 3
```

3. `int *****a; //cấp ??`

+ Phép toán trên con trỏ cấp n ( $n > 1$  và con trỏ cấp 2 thuần túy như trong ví dụ vừa khai báo trên) tương tự như với con trỏ cấp 1 tương ứng

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4.
5. void main()
6. {
7.     int **a=NULL;
8.     printf("%x\n",a); //0x0
9.     a++;
10.    printf("%x",a); //0x4
11.    getch();
12. }
```

## II. Con trỏ đa cấp dùng để làm gì ?

Con trỏ đa cấp dùng để làm gì và tại sao tôi lại phải quan tâm?

Vâng,

+ con trỏ đa cấp thường được dùng trong trường hợp cần thay đổi giá trị của 1 con trỏ cấp thấp hơn khi ra khỏi hàm.

(thật ra thường thì rất ít khi mình và bạn cần đá xoáy vào vấn đề này. Tại vì theo như các framework, và như 1 OS core linux mình đã từng làm việc, người ta thường hay dùng con trỏ void với sự linh hoạt rất cao)

+ Con trỏ cấp 2 còn được dùng như là "con trỏ" trỏ tới một "con trỏ", có thể dùng để xử lý 1 matrix 2 chiều

+ Con trỏ cấp 3 còn được dùng như là con trỏ trỏ tới một "con trỏ", mà con trỏ này đang trỏ tiếp tới 1 con trỏ khác  
có thể dùng như matrix 3 chiều

....

## III. Vài ví dụ hay gặp

trường hợp 1 hay gặp : Xử lý con trỏ mảng các chuỗi

C Code:

Select All | Show/Hide

```
1.
```



```

2. #include <stdio.h>
3. #include <conio.h>
4. #include <malloc.h>
5. void main(void)
6. {
7.     char** lines;
8.     int    numberline=10;
9.     int    linelen=200;
10.
11.     //câ'p phát
12.     //malloc **
13.     lines=(char**)calloc(1,sizeof(char*)*numberline);
14.
15.     for (int i=0;i<numberline;i++)
16.     {
17.         //malloc *
18.         lines[i]=(char*)calloc(1,linelen);
19.
20.
21.         //gán giá trị để' demo thôi
22.         lines[i][0]='A'+i;
23.     }
24.
25.     //xem giá trị gán mẫu
26.     for (int i=0;i<numberline;i++)
27.     {
28.         printf("%s\n",lines[i]);
29.     }
30.
31.     //gia'i phóng
32.     //free *
33.     for (int i=0;i<numberline;i++) free(lines[i]);
34.     //free **
35.     free(lines);
36.
37.     getch();
38. }

```

trường hợp 2 hay gặp : Xử lý con trỏ mảng các chuỗi cấp phát bằng hàm

C Code:

Select All | Show/Hide

```

1. #include <stdio.h>
2. #include <conio.h>
3. #include <malloc.h>

```

```

4. void HamCapPhat(char*** lines,int numberline,int linesize)
5. {
6.     //malloc **
7.     (*lines)=(char**)calloc(1,sizeof(char*)*numberline);
8.
9.     for (int i=0;i<numberline;i++)
10.    {
11.        //malloc *
12.        (*lines)[i]=(char*)calloc(1,linesize);
13.
14.
15.        //gán giá trị để demo thôi
16.        (*lines)[i][0]='A'+i;
17.    }
18. }
19. void HamGiaiPhong(char** lines,int numberline)
20. {
21.     //free *
22.     for (int i=0;i<numberline;i++) free(lines[i]);
23.     //free **
24.     free(lines);
25. }
26. void main(void)
27. {
28.     char** lines;
29.     int numberline=10;
30.     int linelen=200;
31.     //cấp phát bằng hàm
32.     HamCapPhat(&lines,numberline,linelen);
33.
34.     //xem giá trị gán bằng hàm
35.     for (int i=0;i<numberline;i++)
36.     {
37.         printf("%s\n",lines[i]);
38.     }
39.
40.     //giai phóng bằng hàm
41.     HamGiaiPhong(lines,numberline);
42.
43.     getch();
44. }

```

trường hợp 3 hay gặp : Xử lý con trỏ int \*\* (float \*\*, double \*\*, ?????? \*\* tương tự nhé)

C Code:

[Select All](#) | [Show/Hide](#)

```

1. #include <stdio.h>
2. #include <conio.h>
3. #include <malloc.h>
4. void main(void)
5. {
6.     int** array;
7.     int sodong=10;
8.     int socot=200;
9.
10.    //câ'p phát bằng hàm
11.    //malloc **
12.    array=(int**)calloc(1,sizeof(int*)*sodong);
13.
14.    for (int i=0;i<sodong;i++)
15.    {
16.        //malloc *
17.        array[i]=(int*)calloc(1,socot);
18.    }
19.
20.    //gia'i phóng
21.    //free *
22.    for (int i=0;i<sodong;i++) free(array[i]);
23.    //free **
24.    free(array);
25.
26.    getch();
27. }

```

## Phụ lục 1. '\0' là gì?

Hôm nay update 1 câu hỏi rất hay, riêng cái này thì có nhiều người thật sự ko hiểu rõ bản chất của nó, vì vậy mình xin update đáp án như sau

- + '\0' là 1 **hằng kí tự** có mã là 0
- + '\0' được coi là null terminated character
- + là kí tự kết thúc chuỗi ANSI (char\* và các dẫn xuất của char\*)

Vậy còn NULL ??????????????

- + NULL là 1 **hằng số nguyên** : 0
- + Có sự chuyển đổi giữa mọi dạng con trỏ sang NULL



'\0' khác với NULL ở chỗ nào (nếu bạn tinh ý, chỉ cần đọc đoạn trên là bạn sẽ hiểu ra vấn đề ngay lập tức, nếu ko thì hãy check đoạn code sau nha)

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <conio.h>
3.
4. void main()
5. {
6.     printf("%d\n",sizeof('\0')); // in ra 1
7.     printf("%d\n",sizeof(NULL)); // in ra 4
8.     getch();
9. }
```

### III. Trích dẫn 1 câu đố vui về con trỏ ?

 Nguyên bản được gửi bởi ictrack 

Hôm nay phải đào mộ rồi, đáng lẽ mình không trở thành kẻ đào mộ nếu đã phát hiện ra đề tài này sớm hơn.  
Mình có câu đố khác cho mọi người về thao tác con trỏ nâng cao:

C++ Code:

Select All | Show/Hide

```
1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4.
5. char *(*(aa[10][10]))(char **as,int size);
6. char *hs[]{"Nguyen","Quang","Hien"};//một ma'ng các chuô~i
7. //khai báo pa, pf, tf ở đây
8. int main()
9. {
10.
11.
12.     for(pa=aa;pa<&aa[10];++pa)
13.         for(pf=*pa;pf<*pa+10;++pf)
14.             *pf = 0;
15.     aa[0][0]=&ts;
16.     printf("%s",aa[0][0])(hs,sizeof(hs)/sizeof(char*));//in ra một chuô~i gồm các kí tự đầu cu'a ma'ng hs
```

```
17.     return 0;
18. }
```

Trong đoạn mã có sử dụng các biến sau: *pa, pf, ts* nhưng chưa được khai báo. Hãy khai báo các biến này (cộng thêm định nghĩa hàm khác nếu cần) để sao cho đoạn mã trong hàm *main* có thể chạy được (**không dùng các chỉ thị tiền xử lý, không sửa gì ở main**), đồng thời kết quả của hàm *printf* ở phía cuối đoạn mã trên sẽ là chuỗi gồm các kí tự đầu của mảng *hs* (mảng *hs* là mảng bất kì), trong trường hợp cụ thể của đoạn mã trên thì đầu ra là

Output:

Code:

NQH

Giải thích khai báo này:

C++ Code:

[Select All](#) | [Show/Hide](#)



```
1. char *(*( (*aa[10][10])))(char **as, int size);
```


Giải thích tác dụng của đoạn mã này

C++ Code:

[Select All](#) | [Show/Hide](#)

```
1. for(pa=aa; pa<&aa[10]; ++pa)
2.     for(pf=*pa; pf<*pa+10; ++pf)
3.         *pf = 0;
```

 Nguyên bản được gửi bởi langman 

 cái cậu này, đi tao mộ này tết à :( nhưng mà cái này hay nên mình đồng tình với cậu.... hj hj

để mình dùng logic phân tích câu hỏi của cậu nhé

PHP Code:

```
char *(*( (*aa[10][10])))(char **as, int size);
```

**Bước 1** : đầu tiên, mình dịch đoạn code sau

PHP Code:

```
char *(*p);  
p=2;
```

nhận được thông báo như sau : cannot convert from 'int' to 'char \*\*'  
vậy có nghĩa là p ở đây là char\*\*  
ồ vậy là done 1 vấn đề

#### Bước 2:

PHP Code:

```
char *(*p)(char**,int );  
p=2;
```

nhận được thông báo như sau : cannot convert from 'int' to 'char \*(\_\_cdecl \*)(char \*\*,int)'  
vậy ta thấy 1 điều : đây là 1 con trỏ hàm viết đẹp, chuẩn hơn phải là

PHP Code:

```
char* (*p)(char**,int);
```

//cái \* đầu tiên phải đưa về gần char mới là chuẩn,  
//giống như là char\* gets(char\*)  
//char\* ở đây là kiểu dữ liệu trả về

#### Bước 3:

PHP Code:

```
char *(*p)(char**a,int b);  
p=2;
```

mà vẫn nhận được thông báo như thế , vậy có nghĩa là cái định danh thêm vô ko có ý nghĩa gì,

=> đã giải thích được cái (char \*\*a,int size)  
của bạn rồi nhé, đoạn này có ý nói là con trỏ hàm có 2 tham số truyền vào là  
(char \*\*,int)

viết lại là

PHP Code:

```
char* ((*aa[10][10]))(char **,int);
```

**Bước 4:** tiếp theo  
dịch thử

PHP Code:

```
char* ((*p))(char **,int);  
p="aa";
```

nhận được thông báo và tôi thấy nó hoàn toàn giống với

PHP Code:

```
char* (**p)(char **,int);  
p="aa";
```

ok? vấn đề đến đây lại dễ hiểu hơn

**Bước 5 :** quan trọng nhất

PHP Code:

```
void ham(int a)(int b,int c)  
{  
    cout<<a<<"la nhi";  
}  
void main()  
{  
    ham(2)(2,3);  
    system("pause");  
}
```

dịch code này, tôi nhận được dòng báo lỗi : function returns function  
ồ, lại 1 vấn đề nữa được thắc mắc, đó là 2 cái ngoặc này của bạn  
(\*aa[[]])()  
tôi đã hiểu,

char\* ((\*p)))(char \*\*as,int size);  
cả cái đoạn tôi đóng đó đó sẽ trả về 1 tên hàm hoặc con trỏ hàm (chưa bàn chi tiết, chi tiết ở bước 7) và trở thành

PHP Code:

```
char* hàm(char**,int)
```

ok.vấn đề đã giải quyết 60%

**Bước 6 :**

PHP Code:

```
char* (*( (*aa[10][10])))(char **,int);
```

chỉ khác bước 5 ở chỗ đây là 1 mảng các con trỏ như bước 5

**Bước 7 :**

```
char* (*( (*aa)))(char **,int);
```

giả sử ((\*aa)) trả về tên hàm là **hamx**



vậy ta có

```
char* (hamx)(char **,int)
```

vậy đến đây vấn đề đã được giải quyết 99% rồi,

..... ok?????

vậy đơn giản hóa vấn đề, ta có đây là khai báo 1 mảng con trỏ... hj hj hj....

 Nguyên bản được gửi bởi **icttrack** 

cái này đúng là ts đó langman. Ở yêu cầu đề bài mình có nói rằng

Bài này cũng rất khoai, mình xin giải đáp một phần bài tập này.

C++ Code:

[Select All](#) | [Show/Hide](#)

```
1. char *(*( (*aa[10][10])))(char **as,int size);
```

Khai báo trên có nghĩa là khai báo **một mảng của mảng con trỏ hàm trả về con trỏ hàm trả về con trỏ kiểu char.**

Đây là mã giải đáp đề bài của mình, biên dịch dưới dạng mã C, chạy được trên VC++ và GNU GCC

C Code:

[Select All](#) | [Show/Hide](#)

```
1. #include <stdio.h>
2. #include <stdlib.h>
```



```

3. #include <string.h>
4.
5. char* firstLetter(char **as, int size)
6. {
7.     static char* s=0;
8.     int i;
9.     if(s!=0)
10.         free(s);
11.     //minh hoa cho bai tap, mac du co ro ri bo nho
12.     s=(char *)malloc(sizeof(char)*(size+1));
13.     for(i=0;i<size;++i)
14.         s[i]=as[i][0];
15.     s[i]='\0';
16.     return s;
17. }
18.
19. char *(*( ts() ))(char **as, int size)
20. {
21.     return &firstLetter;
22. }
23.
24.
25. int main()
26. {
27.
28.     char*((*((*pa)[10])))(char **as,int size);
29.     char *(*( (*aa[10][10])() ))(char **as,int size);
30.     char* (*( (**pf)() ))(char **as,int size);
31.     char *hs[]={ "Nguyen", "Quang", "Hien" };
32.     for(pa=aa;pa<&aa[10];++pa)
33.         for(pf=*pa;pf<*pa+10;++pf)
34.             *pf = 0;
35.     aa[0][0]=&ts;
36.     printf("%s",aa[0][0])(hs,sizeof(hs)/sizeof(char*));
37.     return 0;
38. }

```

Với đoạn mã đầy đủ đây rồi, mọi người có thể giải thích các câu hỏi mình đã nêu không?

(chưa viết xong)

Đã được chỉnh sửa lần cuối bởi langman : 23-11-2013 lúc 03:15 PM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 04:56 PM

#9



**langman**  
Thành viên mới  
★★★★★

Ngày gia nhập: 06 2007  
Nơi ở:  
C:\WINDOWS\system32\dlcache\  
Bài viết: 3,006

### Chap IX : C++0x và con trỏ thông minh (smart pointer in C++0x)

update sau

Đã được chỉnh sửa lần cuối bởi langman : 13-04-2013 lúc 02:53 AM.

^\_^

Tổng hợp các câu chuyện hài hước vui nhộn, sử dụng Speech Synthesis để đọc : <https://www.youtube.com/channel/UCLk...Tjrg/playlists>

Bùi Tấn Quang



09-12-2010, 07:14 PM

#10



**zstar**  
XCoworker Member  
★★★★★

Ngày gia nhập: 04 2009  
Nơi ở: Gầm cầu  
Bài viết: 2,230

ủng hộ langman , mong cậu tiếp tục để mọi người có dịp học hỏi  
bài này rất tốt cho newbie  
thank !

▼ Trang 1 trên tổng số 18 **1** 2 3 11 ... ► Cuối cùng ►►

Quick Navigation ▼ Thủ thuật, Tutorials và Mã nguồn C/C++/C++0x Top

« Đề tài liên trước | Đề tài liên sau »

#### Các đề tài tương tự

[MS SQL Lỗi SQL Server 2005 express chạy chậm khi chạy report?](#)

Gửi bởi dongtrien trong diễn đàn Thắc mắc Microsoft SQL Server & Microsoft Access

Trả lời: 1

Bài viết cuối: 26-03-2013, 09:12 PM

[Q-Smart S15 thiết kế chắc chắn chạy android giá rẻ](#)

Gửi bởi 16thang4 trong diễn đàn Giới thiệu website, sản phẩm của bạn

Trả lời: 0

Bài viết cuối: 22-08-2012, 04:07 PM

[Làm sao để khi chạy trực tiếp app này thì không chạy mà phải lấy app khác gọi nó mới chạy??](#)

Gửi bởi pimpim\_kute trong diễn đàn Thắc mắc lập trình C#

Trả lời: 4

Bài viết cuối: 28-06-2012, 03:35 PM

#### Quyền hạn của bạn



Bạn **không thể** gửi đề tài mới

**BB code:** On

Bạn **không thể** gửi bài trả lời

**Mặt cười:** On

Bạn **không thể** gửi các đính kèm

**[IMG]** code: On

Bạn **không thể** chỉnh sửa bài viết của bạn

**[VIDEO]** code is On

**Tìm hiểu luật lệ tham gia diễn đàn**

**HTML code:** Off

-- Computer Style ▼

[Liên hệ chúng tôi](#) [Cộng đồng C Việt](#) [Archive](#) [Top](#)

Toàn bộ thời gian tính theo múi GMT +7. Bây giờ là **11:20 PM**.

- Sáng lập bởi Kevin Hoang @2006
- Nguồn đã được cung cấp bởi vBulletin® 4.2.2
- Bản quyền nguồn ©2018 vBulletin Solutions, Inc
- Sử dụng và phát triển bởi Cộng đồng C Việt® : 2006 - 2014