

真 OO无双 之 真乱舞书

寫程式是很快樂的一件事 Since Sep.15,2006

(原創) 為什麼將二維陣列傳入函數時，還要傳入column數? (C/C++) (C) (C#)

Abstract

C語言的二維陣列有很多較難理解之處，其中一個就是當將二維陣列傳入函數時，竟然還要傳入column數，難到C compiler不能自己做嗎?也一併討論其他二維陣列相關的議題。

Introduction

使用環境: Visual Studio 2010 Ultimate

(我曾經在Mar.24,2008年討論過這個題目(原創) 為什麼將二維陣列傳入函數時，還要傳入column數? (C/C++) (C)，不過由於當時純文字討論居多，且缺少圖片解釋，3年後又對C的pointer有更深刻的體驗，所以決定以過去這篇博文為基礎，繼續深入討論)

本文將討論以下主題：

- 1.C#如何將二維陣列傳入函數?
- 2.C如何使用array style將二維陣列傳入函數?
- 3.C如何使用一維陣列的pointer將二維陣列傳入函數?
- 4.C如何使用一維陣列中的陣列的pointer將二維陣列傳入函數?
- 5.Conclusion

首先要了解C語言的風格，C語言的風格是較少的syntax sugar，能讓你在語言中就知道compiler在幹什麼，但缺點也是語法較不人性化且較低階，C++雖較人性化較高階，充斥著syntax sugar，但相對的黑箱作業較多，所以才有Inside the C++ Object Model這本書專門研究compiler在背後到底做了些什麼。

不過令人欣慰的是，C語言雖然語法較不人性化，但卻很有邏輯，也就是說若能確實的了解其語意，就能理解當初為什麼K&R會這樣去設計C語言，若今天時空轉換，或許你也會採用K&R的方式。

C語言有很多地方被人詬病，一個是function pointer語法，另外一個就是二維陣列傳入陣列時的語法，若能理解他的原理，就能習慣這些語法。

C#如何將二維陣列傳入函數?

先來看看C#怎麼將二維陣列傳入函數

array_2_dim_pass_to_function.cs / C#

```
1 /*
2 (C) OOMusou 2007 http://oomusou.cnblogs.com
3
4 Filename : array_2_dim_pass_to_function.cs
5 Compiler : Visual Studio 2005 / C# 2.0
6 Description : Demo how to pass 2 dim array to function
7 Release : 03/24/2008 1.0
8 */
9 using System;
10
11 class Client {
12     static void func(int [,] ia) {
13         int i,j;
14         for(i = 0; i < ia.GetLength(0); ++i) {
15             for(j = 0; j < ia.GetLength(1); ++j) {
16                 Console.Write("{0} ", ia[i,j]);
17             }
18             Console.WriteLine();
19         }
20     }
21
22     static void Main() {
23         int [,]ia = {{1, 2}, {3, 4}, {5, 6}};
24         func(ia);
25     }
26 }
```

執行結果

```
1 2
3 4
5 6
```

12行

```
static void func(int [,] ia) {
```

只須宣告ia為二維陣列int [,]，並不需指定row size與column size，當然也不需事先宣告macro。

14行


```
for(i = 0; i < ia.GetLength(0); ++i) {  
    for(j = 0; j < ia.GetLength(1); ++j) {  
        Console.Write("{0} ", ia[i,j]);  
    }  
    Console.WriteLine();  
}
```

為什麼不需傳入row size與column size呢?因為C#二維陣列自帶GetLength(), 可傳回row size與column size。


C#目前看起來都很直觀，語法也很漂亮，我們試著將以上程式改成C語言。

C如何使用array style將二維陣列傳入函數?

array_2_dim_pass_to_function_full_subscript.c / C



```
1 /*  
2 (C) OOMusou 2008 http://oomusou.cnblogs.com  
3  
4 Filename      : array_2_dim_pass_to_function_full_subscript.c  
5 Compiler      : Visual C++ 8.0  
6 Description   : Pass 2 dim array to function by subscript  
7 Release      : 03/16/2007 1.0  
8 */  
9 #include <stdio.h>  
10  
11 #define ROWSIZE 3  
12 #define COLSIZE 2  
13  
14 void func(int ia[][]) {  
15     int i,j;  
16     for(i = 0; i < ROWSIZE; ++i) {  
17         for(j = 0; j < COLSIZE; ++j) {  
18             printf("%d ", ia[i][j]);  
19         }  
20         printf("\n");  
21     }  
22 }  
23  
24 int main() {  
25     int ia[][] = {{1, 2}, {3, 4}, {5,6}};  
26     func(ia);  
27 }
```



很遺憾的，以上的C程式並無法編譯成功，主要問題在

25行

```
int ia[][] = {{1, 2}, {3, 4}, {5,6}};
```

與14行


```
void func(int ia[][COLSIZE]) {
```

並沒有提供row size與column size，這也顯示了C語言與C#在陣列方面明顯地不同。在C語言必須提供row size與column size，**或者不提供row size，只提供column size**後，就可以成功compile了。

Solution 1:

同時提供row size與column size

array_2_dim_pass_to_function_full_subscript_row_column_size.c / C



```
1 /*  
2 (C) OOMusou 2008 http://oomusou.cnblogs.com  
3  
4 Filename      : array_2_dim_pass_to_function_full_subscript_row_column_size.c  
5 Compiler      : Visual C++ 10.0  
6 Description   : Pass 2 dim array to function by subscript  
7 Release      : 03/16/2007 1.0  
8 */  
9 #include <stdio.h>  
10  
11 #define ROWSIZE 3  
12 #define COLSIZE 2  
13  
14 void func(int ia[ROWSIZE][COLSIZE]) {  
15     int i,j;  
16     for(i = 0; i < ROWSIZE; ++i) {  
17         for(j = 0; j < COLSIZE; ++j) {  
18             printf("%d ", ia[i][j]);  
19         }  
20         printf("\n");  
21     }  
22 }
```

```

22 }
23
24 int main() {
25     int ia[ROWSIZE][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
26     func(ia);
27 }

```

25行

```
int ia[ROWSIZE][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
```

在宣告與設定array時同時告知row size與column size給C compiler。

14行

```
void func(int ia[ROWSIZE][COLSIZE]) {
```

在傳入function時也告知row size與column size給C compiler。

這種寫法唯一的遺憾是，若將來陣列大小改變，還必須手動改變ROWSIZE與COLSIZE macro，無法如C#那樣自動反應陣列大小。

Solution 2:

只提供column size

array_2_dim_pass_to_function_full_subscript_only_column_size.c / C

```

1 /*
2 (C) OOMusou 2008 http://oomusou.cnblogs.com
3
4 Filename      : array_2_dim_pass_to_function_full_subscript_only_column_size.c
5 Compiler      : Visual C++ 10.0
6 Description    : Pass 2 dim array to function by subscript
7 Release       : 03/16/2007 1.0
8 */
9 #include <stdio.h>
10
11 #define ROWSIZE 3
12 #define COLSIZE 2
13
14 void func(int ia[][COLSIZE]) {
15     int i, j;
16     for(i = 0; i < ROWSIZE; i++) {
17         for(j = 0; j < COLSIZE; j++) {
18             printf("%d ", ia[i][j]);
19         }
20         printf("\n");
21     }
22 }
23
24 int main() {
25     int ia[][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
26     func(ia);
27 }

```

25行

```
int ia[][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
```

14行

```
void func(int ia[][COLSIZE]) {
```

都只提供column size而已，而省略了row size，為什麼C compiler允許我們這樣子寫呢？這裡先賣個關子，稍後會解釋。

C如何使用一維陣列的pointer將二維陣列傳入函數？

就我們對記憶體的認知，記憶體都是一維線性的，也就是二維陣列純粹是在我們腦中邏輯上的結構，事實上在記憶體中並非如我們想像中的方式存放著。

邏輯上的二維陣列

1	2
3	4
5	6

記憶體中所實現的二維陣列

1	2	3	4	5	6
---	---	---	---	---	---

在此我們可以得到一個結論：

C語言是用一維陣列去實現二維陣列。

C如何使用一維陣列的pointer將二維陣列傳入函數？

既然C語言是用一維陣列來實現二維陣列，也就是我們可以用傳遞一維陣列的方式來傳遞二維陣列，基於傳遞一維陣列的經驗，我們會傳遞一維陣列第一個element的pointer進函數，現在我們把二維陣列當成一維陣列傳遞進函數。

array_2_dim_pass_to_func_by_1_dim_ptr.c / C

```
1 /*
2 (C) OOMusou 2008 http://oomusou.cnblogs.com
3
4 Filename : array_2_dim_pass_to_func_by_1_dim_ptr.c
5 Compiler : Visual C++ 10.0
6 Description : Pass 2 dim array to function by 1 dim pointer
7 Release : 03/16/2007 1.0
8 */
9 #include <stdio.h>
10
11 #define ROWSIZE 3
12 #define COLSIZE 2
13
14 void func(int *pia) {
15     int i;
16     for(i = 0; i < ROWSIZE * COLSIZE; i++) {
17         printf("%d ", *(pia+i));
18     }
19 }
20
21 int main() {
22     int ia[ROWSIZE][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
23     func((int *)ia);
24 }
```

執行結果

1 2 3 4 5 6

23行

```
func((int *)ia);
```

將陣列型態的ia轉成int *，若不加這個轉型，在Visual Studio 2010會有以下的warning，但仍可編譯成功。

warning C4047: 'function' : 'int *' differs in levels of indirection from 'int [3][2]

warning C4024: 'func' : different types for formal and actual parameter 1

IntelliSense: argument of type "int (*)[2]" is incompatible with parameter of type "int *"

也就是說int [][]與int *不同，必須強制轉型，C compiler所認可的相容pointer是int (*)[2]，這也是下一個要討論的主題。

14行

```
void func(int *pia) {
```

使用pointer來接收傳進來的記憶體位址，如同一維陣列一樣。

這種寫法的缺點，就是你在函數內，只能使用純pointer的操作方式，或者如使用一維陣列subscript方式，如ai[2]這樣，因為基本上你已經將二維陣列當成一維陣列使用，但是這種方式也不是完全沒有用處，有的時候二維陣列用一維陣列處理反而比較方便，比如說在處理演算法階段使用二維陣列比較直覺，但要送進I/O時，因為data bus是serial transfer，必須每個clk送32 bit進data bus，這時使用一維陣列處理反而比較方便。

C語言可以將二維陣列用pointer如傳遞一維陣列方式傳入函數，但傳入後只能用pointer操作或者當成一維陣列使用。

C如何使用一維陣列中的陣列的pointer將二維陣列傳入函數？

在上一個主題中，Visual Studio 2010的warning已經暗示int *不是相容的pointer，真正C compiler所認可的pointer是int (*)[2]，我們試著用這種寫法寫寫看：

array_2_dim_pass_to_func_by_2_dim_ptr_0.c / C

```
1 /*
2 (C) OOMusou 2008 http://oomusou.cnblogs.com
```

```
3
4 Filename      : array_2_dim_pass_to_func_by_2_dim_ptr_0.c
5 Compiler      : Visual C++ 10.0
6 Description   : Pass 2 dim array to function by 2 dim pointer
7 Release       : 03/16/2007 1.0
8 */
9 #include <stdio.h>
10
11 #define ROWSIZE 3
12 #define COLSIZE 2
13
14 void func(int (*pia)[COLSIZE]) {
15     int i, j;
16     for(i = 0; i < ROWSIZE; i++) {
17         for(j = 0; j < COLSIZE; j++) {
18             printf("%d ", pia[i][j]);
19         }
20         printf("\n");
21     }
22 }
23
24 int main() {
25     int ia[ROWSIZE][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
26     func(ia);
27 }
```

14行

```
void func(int (*pia)[COLSIZE]) {
```

這裡出現了兩個奇怪的語法：

- 1.為什麼*ia要用括號()刮起來？
- 2.為什麼還需傳入column size？

總不能將這個語法背起來吧，用背的一定很容易忘。

18行

```
printf("%d ", pia[i][j]);
```

儘管我們傳過去的是一個pointer，卻可以使用`pia[i][j]`這種好用的subscripting寫法，類似操縱陣列一樣，由此可見這種寫法才是C compiler所真正認可的pointer寫法，不過這也是一種syntax sugar，所以看不出C compiler在背後到底是如何存取這個二維陣列，我們再試著用pointer全面改寫。

array_2_dim_pass_to_func_by_2_dim_ptr_1.c / C

```
1 /*
2 (C) OOMusou 2008 http://oomusou.cnblogs.com
3
4 Filename      : array_2_dim_pass_to_func_by_2_dim_ptr_1.c
5 Compiler      : Visual C++ 10.0
6 Description   : Pass 2 dim array to function by 2 dim pointer
7 Release       : 03/16/2007 1.0
8 */
9 #include <stdio.h>
10
11 #define ROWSIZE 3
12 #define COLSIZE 2
13
14 void func(int (*pia)[COLSIZE]) {
15     int i, j;
16     for(i = 0; i < ROWSIZE; i++) {
17         for(j = 0; j < COLSIZE; j++) {
18             printf("%d ", (*(pia + i) + j));
19         }
20         printf("\n");
21     }
22 }
23
24 int main() {
25     int ia[ROWSIZE][COLSIZE] = {{1, 2}, {3, 4}, {5, 6}};
26     func(ia);
27 }
```

18行

```
printf("%d ", (*(pia + i) + j));
```

改成了完全pointer操作方式，當然這又出現了另外一個問題，為什麼pointer要這樣運算才能得出值？

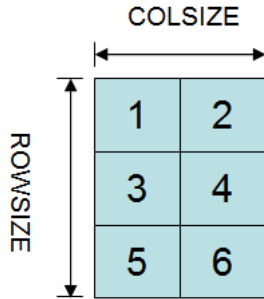
所以目前一共有三個疑問：

- 1.為什麼*ia要用括號()刮起來？
- 2.為什麼還需傳入column size？

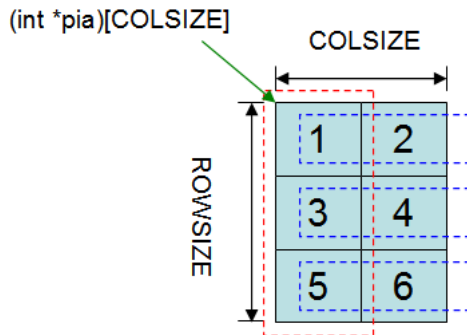
3. 用pointer對二維陣列取值時，為什麼可以這樣寫？

要解釋這些疑問，首先要了解C compiler是如何去處理二維陣列。

在邏輯上，我們認為的二維陣列是長這樣子



但是C compiler所認為的二維陣列是長這樣子

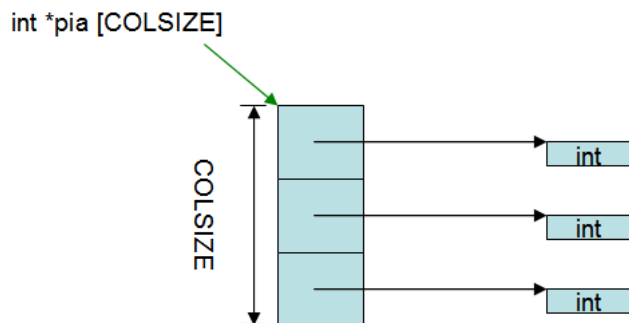


紅色部分相當於一個有ROWSIZE element的一維陣列，而每個element的大小為有COLSIZE element的一維陣列，所以若我們以pointer方式傳進函數時，是傳進一個pointer指向紅色的一維陣列，而該陣列的每個element又包含一個藍色的一維陣列，所以該pointer的表示法是 `(int *pia)[COLSIZE]`。

關於這個奇怪的寫法，若拿掉括號成為

```
int *pia[COLSIZE]
```

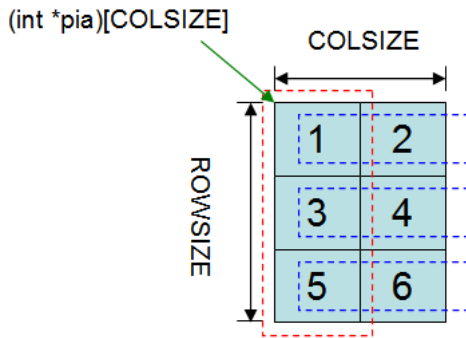
根據C語言operator運算優先順序，會先取[]再取*，也就是以上表示相當於一個有COLSIZE的一維陣列，每個element皆為int *型別。



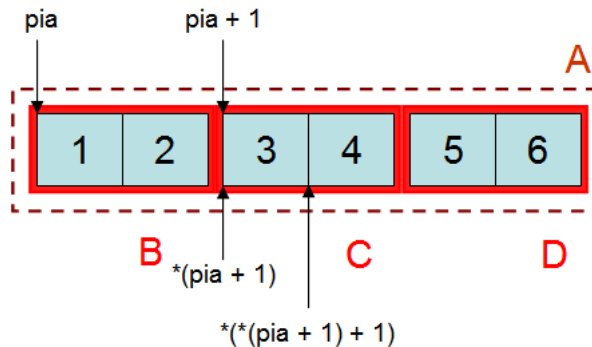
所以只有加上括號後

```
(int *pia)[COLSIZE]
```

由於先取*，所以pia會是一個pointer，表示指向一個一維陣列，且每個element都是int [COLSIZE]，我承認這裏的語法很怪，很類似function pointer那邊一樣是個很怪的語法，不過最少能表現出背後所代表的意義。這解釋了第1個問題。



我們知道陣列本身並不是C語言的原生型態，而是一個底層透過指標來操作的型態，因為記憶體本身是一維線性的，所以我們必須了解二維陣列到底在記憶體中是如何儲存，才能夠透過指標運算存取二維陣列。



如上圖所示，二維陣列在記憶體中，事實上也是如一維陣列般，依序的儲存在記憶體中，先儲存完第1列，接著第2列，配合著C compiler邏輯的陣列，A是一個一維ROWSIZE大的一維陣列，而每個element各自為B、C、D有COLSIZE大的int陣列。

pia為指向A這個一維陣列第一個element的起始位址，也就是1這個值的位址，而pia + 1呢？理論上應該加上1 * sizeof(B)，但問題B並不是int或者char，有固定的size大小，B是另外一個陣列。如是int的話，C compiler就知道+1是4 byte，若是char的話，+1就是1 byte，但是B是一個陣列，所以會根據B陣列的COLSIZE * sizeof(int)而定，這也是為什麼當指標傳入函數時，還要傳入COLSIZE的原因，因為要告訴C compiler，每次+1時，到底要移動多少個byte。這解釋了第2個問題。

假設我們想存取ia[1][1]，到底C compiler是如何得到4這個值呢？

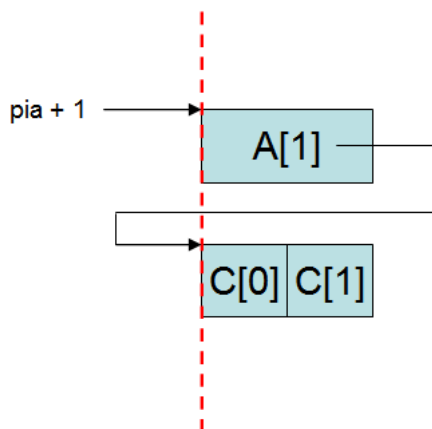
我們知道ia相當於pia，所以指向第一個element，也就是1個位址，不過別忘記了，因為pia的型別是(int *) [COLSIZE]，也就是指向的是以ROWSIZE為element的A陣列，所以當我們動用pointer運算時，pia + 1並不是指向2，而是指向3，因為A陣列每一個element為B陣列，所以+1相當於加上了1 * sizeof(B)，也就是1 * COLSIZE * sizeof(int)，所以是1 * 2 * 4 byte = 8 byte，因此會指向3。

或許你會說：『既然已經指向3，若我要的就是ia[1][0]，那可以直接取值了嗎？』，在C語言，要對pointer取值，不單只是address要對，且type也要對，目前pia+1的型別是(int *) [COLSIZE]，因為型別不對，要型別int才會取到3。

事實上，*(pia+1)所存的值也是指向3，如上圖所示，也就是C陣列第一個element的起始位址，而且這次的pointer型態已經是int，因此若要取ia[1][0]，已經可以直接對*(pia+1)取值，也就是*(*(pia+1)+0)。

若我們要取的是ia[1][1]，則還要以*(pia+1)為基礎再加上1，然後再取值，也就是*(*(pia+1)+1)。

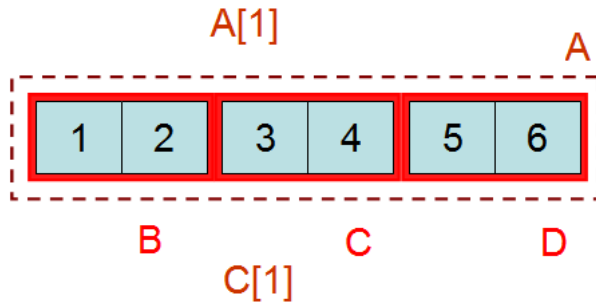
我知道這個結果很多人無法接受，不相信的人可以自己印出pia + 1與*(pia + 1)的值做驗證。



我特別畫了一個示意圖表示以上的關係，pia+1指向A[1]，而A[1]也就是*(pia + 1)所指的就是C[0]的起始位置。

(pia+1)的型別是(int *) [COLSIZE]，所以*(pia+1)的型別是int array [COLSIZE]，因為array與pointer可以互換，所以*(pia+1)的型別也相當於int *，所以*(*(pia+1))的型別是int。

也可以用代數的方法加以證明：



ia[1][1]可視為A[1]中的C[1]

Step 1:

取A[1]

```
= &A[0] + 1 * sizeof(C)
```

```
= *(pia + 1)
```

```
= C
```

Step 2:

取C[1]

```
= &C[0] + 1 * sizeof(int)
```

```
= *(C + 1)
```

將Step 1結果代進Step 2得

```
ia[1][1] = (*(pia + 1) + 1)
```

這解釋了第3個問題。

完整檔案下載

[array_2_dim_pass_to_function.7z](#)

[array_2_dim_pass_to_func_by_1_dim_ptr.7z](#)

[array_2_dim_pass_to_function_full_subscript_row_column_size.7z](#)

[array_2_dim_pass_to_function_full_subscript_only_column_size.7z](#)

[array_2_dim_pass_to_func_by_2_dim_ptr_0.7z](#)

[array_2_dim_pass_to_func_by_2_dim_ptr_1.7z](#)

Conclusion

這是C語言比較難懂的地方，也是我第二次試圖用很淺顯的方式去解釋，並加上了很多圖片幫助了解，或許你可以參考我三年前寫的這篇(原創) 為什麼將二維陣列傳入函數時，還要傳入column數? (C/C++) (C)，搞不好這篇的風格比較合你的胃口。

- 1.C語言可以使用array style的方式將二維陣列傳入函數，可以同時提供row size與column size，也可省略row size只提供column size。
- 2.C語言可以將二維陣列用pointer如傳遞一維陣列方式傳入函數，在某些場合使用一維陣列操作方式可能更加方便。
- 3.當C語言的二維陣列以pointer型態傳入函數時，必須要使用包含column size的一維陣列的pointer。

See Also

(原創) 為什麼將二維陣列傳入函數時，還要傳入column數? (C/C++) (C)

全文完。

分类: C, C#, C/C++



真 OO无双
关注 - 0
粉丝 - 1103

+加关注

1 0

« 上一篇: (原創) 從事硬體開發後所學到的4件事 (SOC) (Misc)

» 下一篇: (筆記) 如何在Debussy / Verdi顯示FSM的state名稱? (SOC) (Debussy) (Verdi)

posted on 2011-06-12 08:32 真 OO无双 阅读(20128) 评论(7) 编辑 收藏

评论

#1楼 2011-06-12 09:17 tuoluofu

无双大最近又有好几篇新博啦，呵呵！学习。。。

支持(0) 反对(0)

#2楼 2011-06-12 23:46 oic

很有趣的討論！一直以來從你的blog裡學到很多東西。我想發表一下我的看法。(pia+1)的型別是pointer to an array of 2 int.當deference時，*(pia+1)的型別就變成array of 2 int.在C語言中，array所表示的意義其實就是pointer (例如，傳遞array給函數與傳遞pointer無異)。所以*(pia+1)的型別就相等於pointer to int.這時它的值是位址，而不是3跟4(8 bytes)。有錯請指正，謝謝！

支持(0) 反对(0)

#3楼 2011-06-13 06:55 oomusou

@ oic

你的想法跟我用代數證明的方法一樣
也就是 $*(pia+1) = C = \&C[0]$

這完全是因為在C中有一個很詭異的等式
 $arr = \&arr[0]$
而這個等式就是C compiler所動的手腳依據
也就是你說的array事實上是個pointer
也就是強迫將array轉型成pointer
強迫將 $pia+1$ 等於 $*(pia+1)$

只是這裡若你真的去dump記憶體連續的內容
還是會發現不太合理

不過可能你是對的
只是我還沒想通這一點
這部分我想了十幾年
可能三年後我又有新的領悟

[支持\(0\)](#) [反对\(0\)](#)**#4楼 2011-06-13 08:59 _安德鲁**

也可省略row size只提供column size。这句话比较经典。
不过我经常是在字库检索的时候会用到数组。若是二维的话，一般再拆成两个元素，用结构体来表示，个人感觉结构比较简单。反正就是尽量转换为一维数组来做。

[支持\(0\)](#) [反对\(0\)](#)**#5楼 [楼主] 2011-06-14 07:45 真 OO无双**[引用](#)

oic: 很有趣的討論! 一直以來從你的blog裡學到很多東西。我想發表一下我的看法。(pia+1)的型別是pointer to an array of 2 int. 當deference時, *(pia+1)的型別就變成array of 2 int.在C語言中, array所表示的意義其實就是pointer (例如, 傳遞array給函數與傳遞pointer無異)。所以*(pia+1)的型別就相等於pointer to int. 這時它的值是位址, 而不是3跟4(8 bytes)。
有錯請指正, 謝謝!

經過一天的思考
我覺得你的指正是對的
我會修改文中的寫法
謝謝

[支持\(0\)](#) [反对\(0\)](#)**#6楼 [楼主] 2011-06-14 07:46 真 OO无双**[引用](#)

.COM 缺氧®; 也可省略row size只提供column size。这句话比较经典。
不过我经常是在字库检索的时候会用到数组。若是二维的话，一般再拆成两个元素，用结构体来表示，个人感觉结构比较简单。反正就是尽量转换为一维数组来做。

一維array + struct也是個好方法

[支持\(0\)](#) [反对\(0\)](#)**#7楼 2013-04-30 00:14 adern9**

我是正在學C的菜鳥，您的文章讓我受益良多，謝謝

[支持\(0\)](#) [反对\(0\)](#)[刷新评论](#) [刷新页面](#) [返回顶部](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问](#) 网站首页。

【推荐】超50万行VC++源码：大型组态工控、电力仿真CAD与GIS源码库

【活动】腾讯云服务器推出云产品采购季 1核2G首年仅需99元

【推荐】这6种编码方法，你掌握了几个？

【推荐】2019热门技术盛会400则演讲资料全收录

相关博文：

- (筆記) struct對function可以call by value嗎?可以return一個struct嗎? (C/C++)
- (原創) Function Pointer、Delegate和Function Object (C/C++) (template) (.NET) (C#)
- (筆記) 如何對一變數指定某一個bit的值? (SOC) (C/C++) (Verilog)
- (翻譯) 為什麼C#不提供默認參數(default parameter)? (.NET) (C#) (C++/CLI) (C/C++)