指標 Pointer

by 多拉A夢

在講指標之前....

談談『變數』

5 , ‘c’, 3.4123...

『變數』

```
int a = 10;

char c = ‘z’;

float x = 3.1415926;
```

我們雖然知道...

我們雖然知道...

包包有名稱....

我們雖然知道...

包包有名稱....

包包也可以塞東西...

那包包的位置在哪呢？

記憶體位置

記憶體位置

可以想成包包的位置

我們今天不用擔心....

要如何拿到包包的位置:D

oh ya ~

我們來看看變數是啥米...

變數

資料形態

變數的值

變數的位置

摁.... 超級多的專有名詞很難懂嗎:D

包包名稱

包包的種類

包包裡面的東西

包包的位置

舉例

int x = 10 ;

x =

int, char, float

10, ‘c’, 3.14159

變數的位置?

舉例

int x = 10 ;

x =

int, char, float

10, ‘c’, 3.14159

變數的位置?

舉例

int x = 10 ;

int, char, float

x =

10, ‘c’, 3.14159

那這裡？ → **變數的位置?**

來做個簡單的實驗...

```cpp
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>

using namespace std;

int main(){

    int x = 10;
    cout << "Value of x = " << x << endl;
    cout << "Position of x = " << &x << endl;

    return 0;
}
```

注意這裡有 ‘&’拿來抓取位置用的

我跑出來的結果：D

```cpp
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>

using namespace std;

int main(){

    int x = 10;
    cout << "Value of x = " << x << endl;
    cout << "Position of x = " << &x << endl;

    return 0;
}
```

```
Value of x = 10
Position of x = 0x7fff56e4e698
```

舉例

int x = 10 ;

int, char, float

x =

10, ‘c’, 3.14159

實驗結果

既然有了位置的觀念....

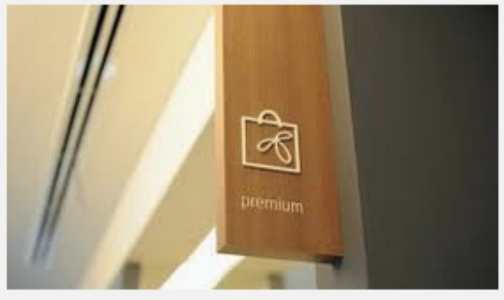
正式進入指標：D

What is 指標？

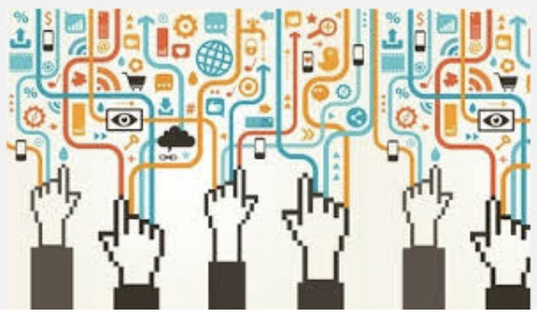
來Google一下圖片 ~







就是你想的那樣...

指標可以指著別人（X

他的概念是這樣的...

指標

這裡面放

包包的位置

小明的包包

小說

小華的包包

數學課本

小美的包包

彩色書刊

他的概念是這樣的...

小明的包包

小說

指標

小明的

包包的位置

小華的包包

數學課本

小美的包包

彩色書刊

他的概念是這樣的...

小明的包包

小說

指標

小華的

包包的位置

小華的包包

數學課本

小美的包包

彩色書刊

他的概念是這樣的...

小明的包包

小說

小華的包包

數學課本

小美的包包

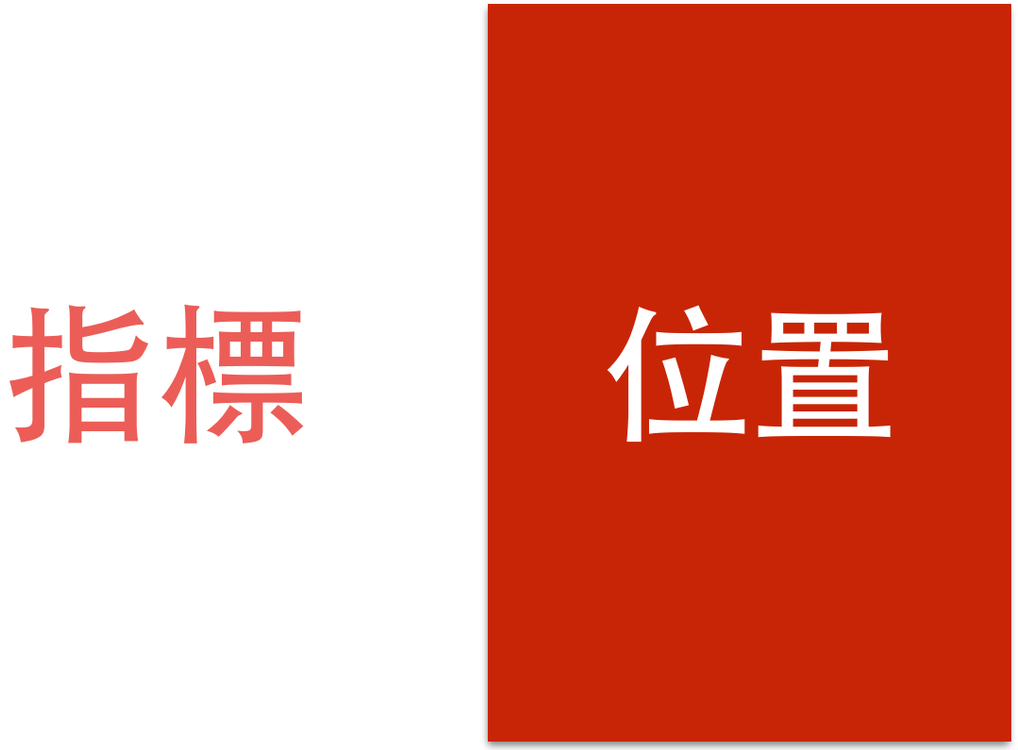
彩色書刊

指標

小美的

包包的位置

指標裡面裝的是位置



因此.....

指標本身也是變數

裡面放的是位置

簡稱『指標變數』

~~有講跟沒講一樣~~

要指著別人....

要先知道別人在哪裡

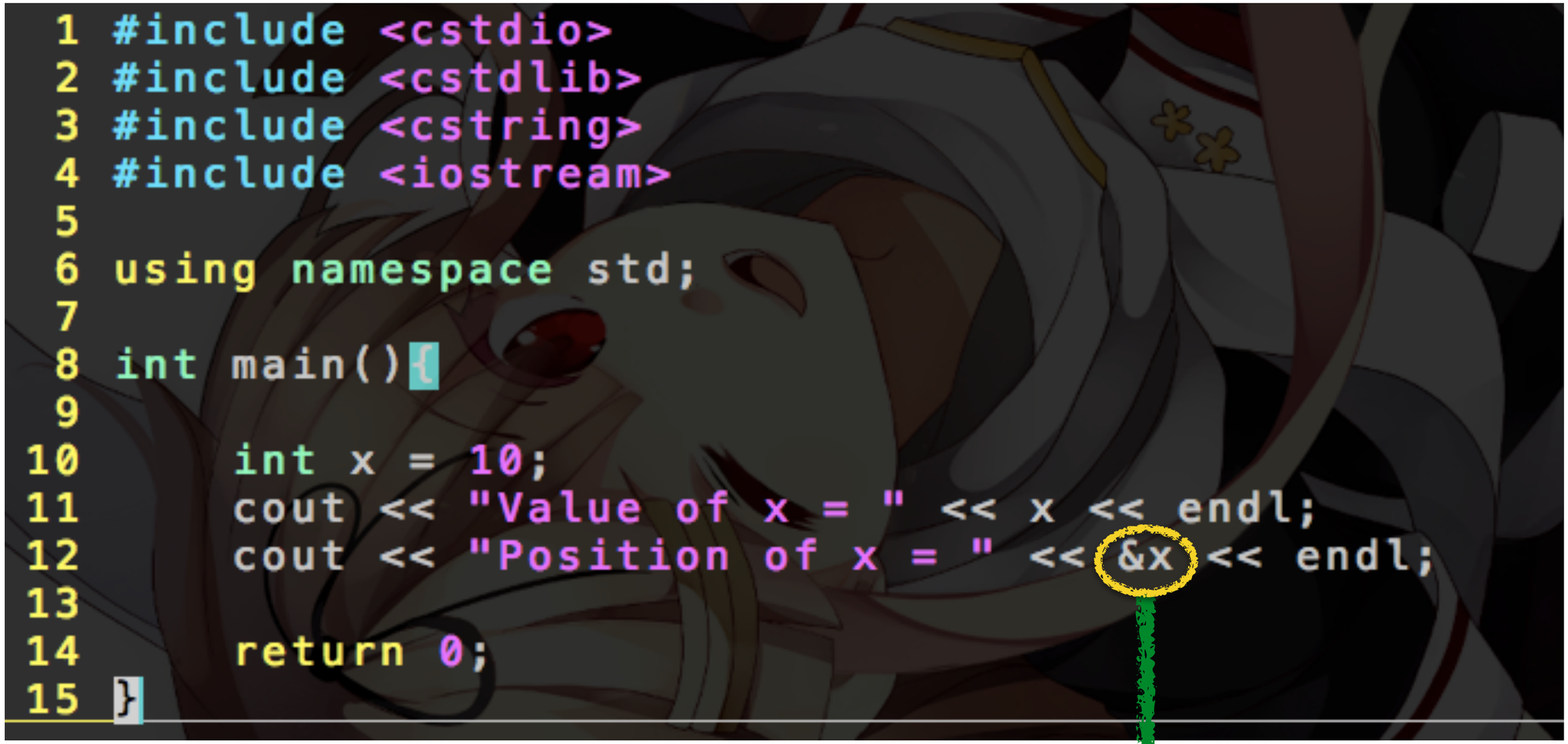
那要如何得知呢？

來看看剛剛的實驗...

```cpp
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>

using namespace std;

int main(){

    int x = 10;
    cout << "Value of x = " << x << endl;
    cout << "Position of x = " << &x << endl;

    return 0;
}
```

注意這裡有 ‘&’拿來抓取位置用的

這樣就知道位置了：D



```
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <iostream>

using namespace std;

int main(){

    int x = 10;
    cout << "Value of x = " << x << endl;
    cout << "Position of x = " << &x << endl;

    return 0;
}
```

注意這裡有‘&’拿來抓取位置用的

指標的宣告

- 宣告的方式：資料形態　* 指標變數名稱
  - 資料形態：int, float, char . . .
  - * ⟶ 星星符號，代表指標
    - 指標變數名稱：看你想取什麼綽號...
- 舉例：
  - `int *ptr;`
  - `char *ptr;`

指標的運用

- 我們分開討論指標的用法
- 『宣告』還有『不是宣告』的時候使用上有點不同

指標的運用

- 在宣告時assign value是這樣用的.....

```
6  using namespace std;
7
8  int main(){
9      int x = 10;
10     int *ptr = &x;
11
12     cout << "value of x = " << x << endl;
13     cout << "address of x = " << &x << endl;
14     cout << "valure of ptr = " << ptr << endl;
15     cout << "value which ptr point to = " << *ptr << endl;
```

指標的運用

- 在『不是在宣告』時assign value是這樣用的.....

```
5
6  using namespace std;
7
8  int main(){
9      int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

宣告ptr 指標變數

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

把 x 的位置傳給 ptr

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "value of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
value of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

x 的位置

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
valure of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

ptr 這個『指標變數』裡面的值

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "value of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
value of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

你會發現，『ptr的值』跟『x 的位置一樣』

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "value of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
value of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

因為我們做了 ptr = &x ~~~

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
valure of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

*ptr：『指標變數』指到的值（汗

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
valure of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

*ptr：『指標變數』指到的值（汗

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
valure of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

——>這個位置的值

*ptr：『指標變數』指到的值（汗

指標的運用

```
 5
 6 using namespace std;
 7
 8 int main(){
 9     int x = 10;
10     int *ptr;
11     ptr = &x;
12
13     cout << "value of x = " << x << endl;
14     cout << "address of x = " << &x << endl;
15     cout << "valure of ptr = " << ptr << endl;
16     cout << "value which ptr point to = " << *ptr << endl;
```

```
value of x = 10
address of x = 0x7fff5fb84678
valure of ptr = 0x7fff5fb84678
value which ptr point to = 10
```

——>這個位置的值

*ptr：『指標變數』指到的值（汗

如果還是很難理解...

來看看圖形～～

int i = 100;

Address 1585640876

Variable

Value 100

int *ptr;

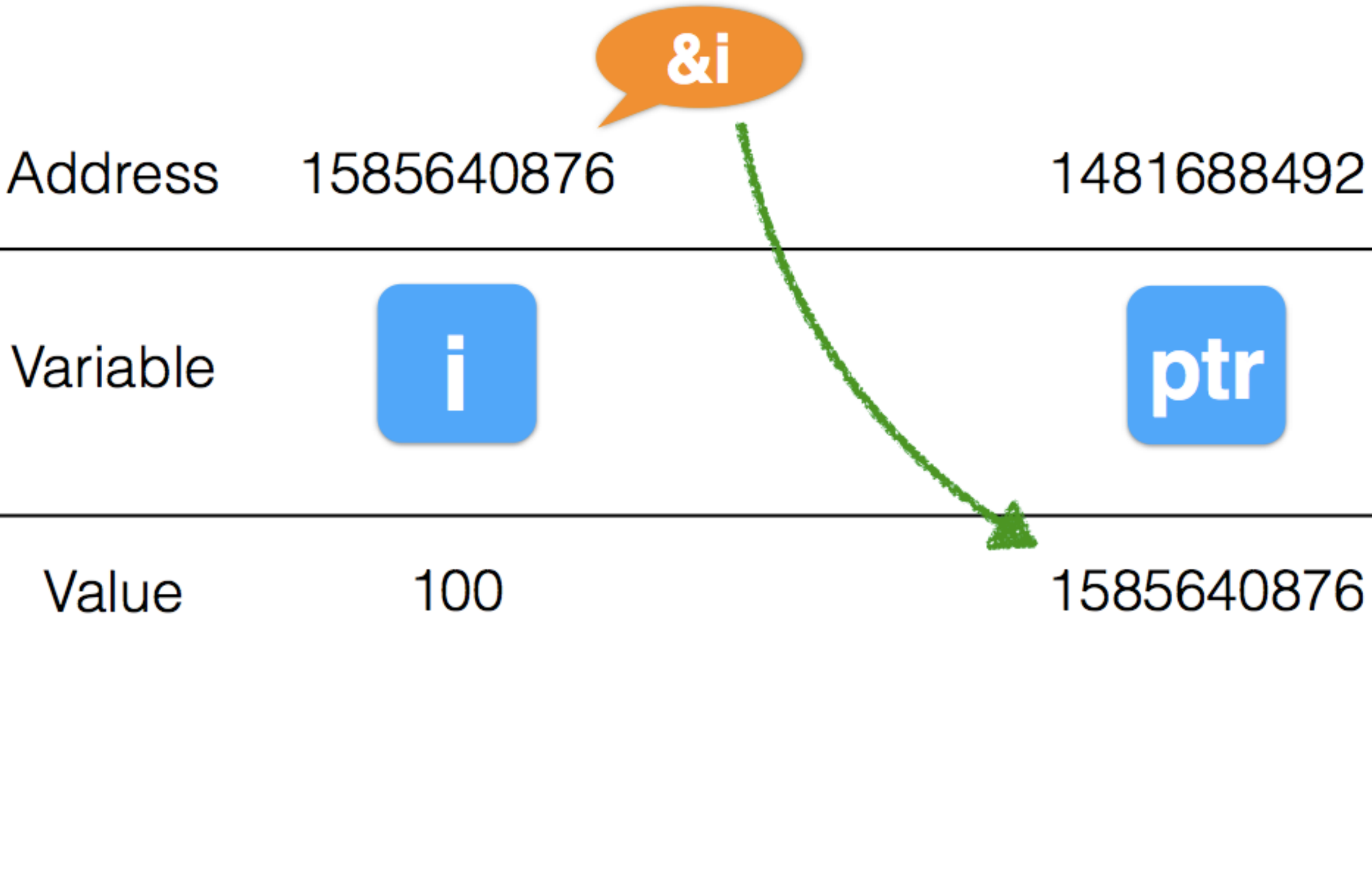
Address	1585640876	1481688492
Variable	i	ptr
Value	100	???

&i ——> 取得變數 i 的位址

&i

Address	1585640876	1481688492
Variable	i	ptr
Value	100	???

ptr = &i;


&i
Address
1585640876
1481688492
Variable
i
ptr
Value
100
1585640876

Address	1585640876	1481688492
Variable		
Value	100	1585640876

整理一下用法

- x 是一個變數, ptr是一個指標變數
- x => 變數的值
- &x => 取得變數的位址
- ptr => 『指標變數』的值
- *ptr => 取得『指標變數』指到的值

我們順藤摸瓜一下...

- 又是來這個例子：int x = 10; int *ptr = &x;

- 我們知道... ptr指到x的位置，***ptr**　的值等於10

我們順藤摸瓜一下...

- 又是來這個例子：int x = 10; int *ptr = &x;
- 我們知道... ptr指到x的位置，***ptr** 的值等於10
- 如果我們做了這樣的事情：***ptr = 1000;**
- **請問 x 會變成多少？**

改一下剛剛的code

```
8  int main(){
9      int x = 10;
0      int *ptr;
1      ptr = &x;
2
3      cout << "value of x = " << x << endl;
4      cout << "address of x = " << &x << endl;
5      cout << "valure of ptr = " << ptr << endl;
6      cout << "value which ptr point to = " << *ptr << endl;
7
8      *ptr = 100;
9
0      cout << "value of x = " << x << endl;
1      cout << "address of x = " << &x << endl;
2      cout << "valure of ptr = " << ptr << endl;
3      cout << "value which ptr point to = " << *ptr << endl;
```

實驗結果

```
./a.out
value of x = 10
address of x = 0x7fff58bef638
valure of ptr = 0x7fff58bef638
value which ptr point to = 10

value of x = 100
address of x = 0x7fff58bef638
valure of ptr = 0x7fff58bef638
value which ptr point to = 100
```

比較一下...

```
value of x = 10
address of x = 0x7fff58bef638
value of ptr = 0x7fff58bef638
value which ptr point to = 10

value of x = 100
address of x = 0x7fff58bef638
value of ptr = 0x7fff58bef638
value which ptr point to = 100
```

我們順藤摸瓜一下...

- 又是來這個例子：int x = 10; int *ptr = &x;
- 我們知道... ptr指到x的位置，***ptr** 的值等於10
- 如果我們做了這樣的事情：***ptr = 1000;**
- **請問 x 會變成多少？**
- **答：因為ptr指到的是x的位置，*ptr 的值是 x的**
  - **更改 *ptr 的值 等同 更改x的值，所以x = 1000**

一個非常不正規的記憶法

一個非常不正規的記憶法

全家就是你家

一個非常不正規的記憶法

全家就是你家

全家的東西被偷了

一個非常不正規的記憶法

全家就是你家

全家的東西被偷了

等於我家的東西被偷了

~~亂七八糟~~

指標：指向陣列

既然一般的變數都可以了

指標：指向陣列

既然一般的變數都可以了

那陣列呢？

指標：指向陣列

```
 8
 9 int main(){
10     int s[] = {3,2,1,4,5};
11     int *ptr;
12     ptr = &s[0];
13     for(int i = 0; i < 5; ++i)
14         cout << i << " : " << ptr[i] << endl;
15
16     return 0;
17 }
```

- assign：用法1 ptr = &s[0];
- ptr 當作陣列來使用（今天先教這個）

```
0 : 3
1 : 2
2 : 1
3 : 4
4 : 5
```

指標：指向陣列

```
 8 int main(){
 9     int s[] = {3,2,1,4,5};
10     int *ptr;
11     ptr = s;  // same as 'ptr = &s[0]'
12     for(int i = 0; i < 5; ++i)
13         cout << i << " : " << ptr[i] << endl;
14
15     return 0;
16 }
```

- assign：用法2 ptr = s;
- 這個用法跟 ptr = &s[0] 一樣

```
0 : 3
1 : 2
2 : 1
3 : 4
4 : 5
```

指標：指向陣列

- 如果跟之前一樣，亂改東西呢？

```
 9 int main(){
10     int s[] = {3,2,1,4,5};
11     int *ptr;
12     ptr = &s[0];
13     for(int i = 0; i < 5; ++i)
14         cout << i << " : " << ptr[i] << endl;
15
16     ptr[2] = 10000;
17     for(int i = 0; i < 5; ++i)
18         cout << i << " : " << ptr[i] << endl;
19
```

- 大家應該都猜的到啦： D

指標：指標的陣列

- 既然有『指標變數』這種東西...
- 假設我一次想要開1000個指標變數...
- 難道我要 int *ptr1, *ptr2, *ptr3……, *ptr100？
- 上面這種寫法看起來不怎麼友善.....
- 所以..... 直接開指標陣列啦！！！

指標：指標的陣列（1）

```
 9 int main(){
10
11     int x = 10, y = 100;
12     int *ptr[100];  //指標陣列
13                     //每一個元素可以放位置
14
15     ptr[0] = &x;    //第0個指標指向x
16     ptr[1] = &y;    //第1個指標指向y
17
18     for(int i = 0; i < 2; ++i)  //輸出0~1指標的內容
19         cout << i << " : " << ptr[i] << " : " << *ptr[i] << endl;
20             //^^^^^^^          ^^^^^^^^^              ^^^^^^^^
21             // 第i個      第i個指標指的位置    第i個指標指的位置的值
22
23     cout << " x position = " << &x << endl; //檢查
24     cout << " y position = " << &y << endl; //檢查
25
26     return 0;
27 }
```

指標：指標的陣列（2）

```
 9 int main(){
10     int arr[5] = {9, 10, 123, 21, 1};
11     int *ptr[5];   //指標陣列
12                    //每一個元素可以放位置
13
14     for(int i = 0; i < 5; ++i)
15         ptr[i] = &arr[i]; // 第i個ptr指標陣列的值
16                           // 等於第i個arr陣列的值
17
18     for(int i = 0; i < 5; ++i)  //輸出0~4指標的內容
19         cout << i << " : " << ptr[i] << " : " << *ptr[i] << endl;
20             //^^^^^^^          ^^^^^^^^^              ^^^^^^^^
21             // 第i個      第i個指標指的位置    第i個指標指的位置的值
22
23     for(int i = 0; i < 5; ++i)  // 這裡用來檢查
24         cout << i << " : " << &arr[i] << endl;
25         //      ^^^^              ^^^^^^^^
26         //      第i個          第i個arr的位置
27     return 0;
28 }
```