Introduction to C++ (Season 1)

Unit 3: More than C

第3单元:更上一层楼 – 超越C的语法

Section 5 : Const

第5节： 常量

Named Constants (命名常量➔ §2.5)

❖ const datatype CONSTANTNAME = VALUE;

```
const double PI = 3.14159;
const int SIZE = 3;
int const X = 5;
```

5. Named constants (including enumeration values) must be all uppercase using underscore to separate words.
5. 符号常量(包括枚举值)必须全部大写并用下划线分隔单词
例如： MAX_ITERATIONS, COLOR_RED, PI

❖ const in C vs in C++

- in C (C89), const means "ReadOnly Variable" (只读变量)
- in C++, const means "Constant" (常量)

```
const int ARRAY_SIZE = 10;
int arr[ARRAY_SIZE]; //OK in C++, Error in C
```

const and pointers (常量与指针)

- two features of a pointer(指针的两个属性):
  - pointer variable (指针变量本身)
  - data that the pointer points to (指针变量所指向的数据)
- const + pointer

variable	variable
variable	constant
constant	variable
constant	constant

A variable pointer to a constant value

- Shortly: Pointer to Constant (常量指针/常指针)
- 特征：指针所指向的内容不可以通过指针的间接引用(*p)来改变。

```
const int* p1;
const int x = 1;
p1 = &x;        //指针 p1的类型是  (const int*)

*p1 = 10;       // Error!
```

An invariable pointer to a variable value

- Pointer Constant (指针常量)
  - 指针本身的内容是个常量，不可以改变。

```
int x = 1, y = 1;
int* const p2 = &x; //常量 p2的类型是  (int*)

*p2 = 10;       // Okay! → x=10
p2 = &y;        // Error! p2 is a constant
```

- 数组名就是数组的首地址的别名。现在可以说：数组名就是一个指针常量。

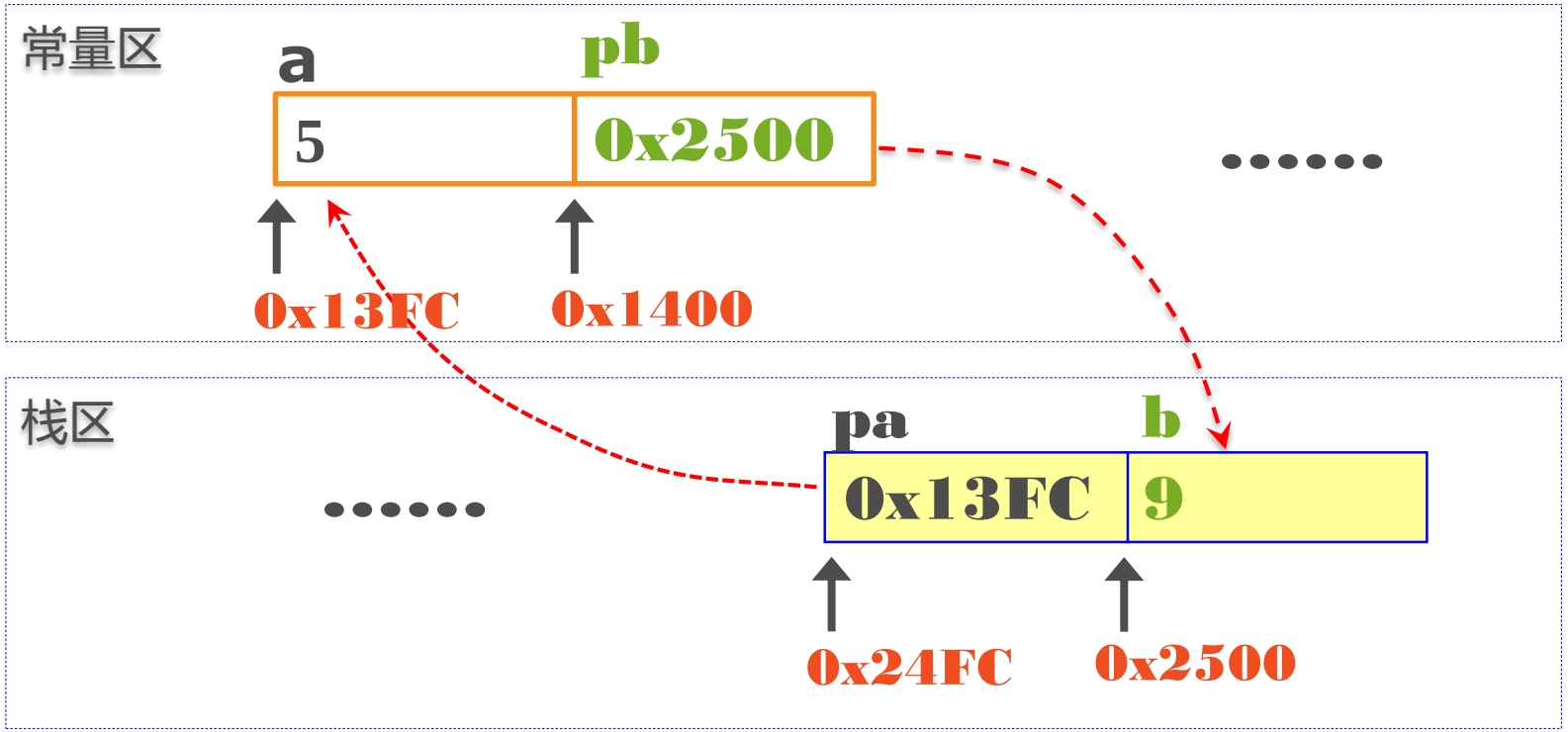
下面的代码是否正确?

```
int a[]={1,2,3};
a=a+1;
```

Memory location of pointers (指针的内存布局)

指针布局实例：（以下代码在某函数内部）

```
const int a = 5;
int b = 9;
const int* pa = &a; //pointer to constant
int* const pb = &b; //pointer constant
```



变量、常量与常指针

```
int i=10;
int const * pi=&i;
```

只是说明 pi 指向的位置中所存的内容，不能通过 *pi 的方式被改变

```
const int ci=10;
const int *pci=&ci;
```

ci存放于常量区，不可被改变，pci指向的地址中所存的内容，不能通过 *pci 的方式被改变

```
const int ci=10;
int* pi=&ci;
```

ci存放于常量区，不可被改变，也不能通过指向ci的指针改变ci，因此*指向ci的指针必须是常指针*

参见 ISO C++ 标准2003版 8.3.1节

Summary (总结)

```
const int * x
int * const y
```

- 在前先读，在前不变
  - *（指针）和 const（常量）谁在前先读谁；
  - *代表被指的数据，名字代表指针地址
  - const在谁前面谁就不允许改变。