June 20th, 2013

Pointers:

A pointer is declared by putting an asterisk (\*) in front of the variable name in the declaration statement:

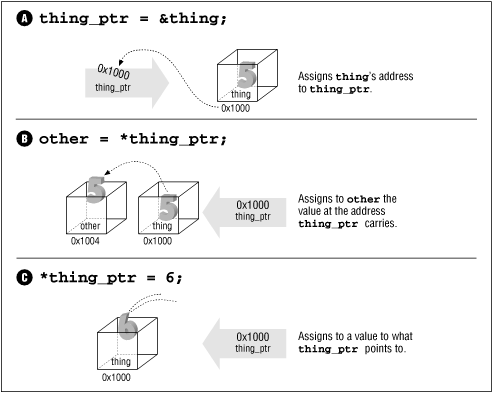
int thing; /\* define a thing \*/

int \*thing\_ptr; /\* define a pointer to a thing \*/

将thing\_ptr所指向的位置的那个内容替换成5. \*加星号表示取出后面这个pointer指向位置的内容。

\*thing\_ptr = 5;

也就等价于thing = 5;



#include <stdio.h>

void inc\_count(int \*count\_ptr)

{

(\*count\_ptr)++;

}

int main()

{

int count = 0; /\* number of times through \*/

while (count < 10)

inc\_count(&count);

return (0);

}

**Const Pointers:**

1. const char \*answer\_ptr = "Forty-Two";

const放前面说明answer\_ptr指向的地方里面存的内容是const

1. char \*const name\_ptr = "Test";

const放在\*的后面说明这个指针的值是固定的，也就是他总是指向同一个地方，这个值不能改变。

char array[5];

char \*array\_ptr = &array[0];

In this example, \*array\_ptr is the same as array[0], \*(array\_ptr+1) is the same as array[1].

**Array:**

Rather than writing:

array\_ptr = &array[0];

we can write:

array\_ptr = array;

**Command-Line Arguments:**

The procedure main actually takes two arguments.

main(int argc, char \*argv[])

The parameter argc is the number of arguments on the command line (including the program name). The array argv contains the actual arguments.

Iterator:

It is an object that pointing to some element in a range of elements, and has the ability to iterate through the elements of the range using a set of operators.就像指针一样，他每一次取到的值都是这个队列里面的一个元素的指针，或者叫做引用。

**Iterator categories:**

1. Input and output iterators: they can perform sequential single-pass input or output operations.
2. Forward iterators: they are limited to one direction in which to iterate through a range forward.
3. Bidirectional iterators: like forward iterators but can also be iterated through backwards.
4. Random-access iterators: have the ability to access ranges non-sequentially; distant elements can be accessed directly by applying an offset value to an iterator without iterating through all the elements.