在C语言中，static的作用有三条：一是隐藏功能，二是保持持久性功能，三是默认初始化为0。

在C语言中，static的字面意思很容易把我们导入歧途，其实它的作用有三条，分别是：

一是隐藏功能，对于static修饰的函数和全局变量而言

二是保持持久性功能，对于static修饰的局部变量而言。

三是因为存放在静态区，全局和局部的static修饰的变量，都默认初始化为0

下面我逐一给大家介绍：

（1）先来介绍它的第一条也是最重要的一条：隐藏。

当我们同时编译多个文件时，所有未加static前缀的全局变量和函数都具有全局可见性。为理解这句话，我举例来说明。我们要同时编译两个源文件，一个是a.c，另一个是main.c。

下面是a.c的内容

|  |  |
| --- | --- |
| 1  2  3  4  5 | **char** a = 'A'; // global variable  void msg()  {  **printf**("Hello\n");  } |

下面是main.c的内容

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **int** main(void)  {   extern **char** a; // extern variable must be declared before use  **printf**("%c ", a);   (void)msg();   return 0;  } |

程序的运行结果是：

A Hello

你可能会问：为什么在a.c中定义的全局变量a和函数msg能在main.c中使用？前面说过，所有未加static前缀的全局变量和函数都具有全局可见性，其它的源文件也能访问。此例中，a是全局变量，msg是函数，并且都没有加static前缀，因此对于另外的源文件main.c是可见的。

如果加了static，就会对其它源文件隐藏。例如在a和msg的定义前加上static，main.c就看不到它们了。利用这一特性可以在不同的文件中定义同名函数和同名变量，而不必担心命名冲突。Static可以用作函数和变量的前缀，对于函数来讲，static的作用仅限于隐藏，而对于变量，static还有下面两个作用。

（2）static的第二个作用是保持变量内容的持久。存储在静态数据区的变量会在程序刚开始运行时就完成初始化，也是唯一的一次初始化。共有两种变量存储在静态存储区：全局变量和static变量，只不过和全局变量比起来，static可以控制变量的可见范围，说到底static还是用来隐藏的。虽然这种用法不常见，但我还是举一个例子。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | #include <stdio.h>  **int** fun(void){   static **int** count = 10; // 事实上此赋值语句从来没有执行过   return count--;  }  **int** count = 1;  **int** main(void)  {  **printf**("global\t\tlocal static\n");   for(; count <= 10; ++count)  **printf**("%d\t\t%d\n", count, fun());   return 0;  } |

 程序的运行结果是：

global          local static

1               10

2               9

3               8

4               7

5               6

6               5

7               4

8               3

9               2

10              1

（3）static的第三个作用是默认初始化为0。其实全局变量也具备这一属性，因为全局变量也存储在静态数据区。在静态数据区，内存中所有的字节默认值都是0x00，某些时候这一特点可以减少程序员的工作量。比如初始化一个稀疏矩阵，我们可以一个一个地把所有元素都置0，然后把不是0的几个元素赋值。如果定义成静态的，就省去了一开始置0的操作。再比如要把一个字符数组当字符串来用，但又觉得每次在字符数组末尾加'\0'太麻烦。如果把字符串定义成静态的，就省去了这个麻烦，因为那里本来就是'\0'。不妨做个小实验验证一下。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #include <stdio.h>  **int** a;  **int** main(void)  {  **int** i;   static **char** str[10];  **printf**("integer: %d; string: (begin)%s(end)", a, str);   return 0;  } |

程序的运行结果如下

integer: 0; string: (begin)(end)

最后对static的三条作用做一句话总结。首先static的最主要功能是隐藏，其次因为static变量存放在静态存储区，所以它具备持久性和默认值0。

以上内容是关于c语言中static的作用。

下面给大家介绍C语言中使用静态函数的好处。

 C语言中使用静态函数的好处：

静态函数会被自动分配在一个一直使用的存储区，直到退出应用程序实例，避免了调用函数时压栈出栈，速度快很多。  
关键字“static”，译成中文就是“静态的”，所以内部函数又称静态函数。但此处“static”的含义不是指存储方式，而是指对函数的作用域仅局限于本文件。 使用内部函数的好处是：不同的人编写不同的函数时，不用担心自己定义的函数，是否会与其它文件中的函数同名，因为同名也没有关系。

c语言中static的语义1.static变量:1).局部a.静态局部变量在函数内定义,生存期为整个源程序，但作用域与自动变量相同，只能在定义该变量的函数内使用。退出该函数后， 尽管该变量还继续存在，但不能使用它。b.对基本类型的静态局部变量若在说明时未赋以初值，则系统自动赋予0值。而对自动变量不赋初值，则其值是不定的。2).全局全局变量本身就是静态存储方式， 静态全局变量当然也是静态存储方式。但是他们的作用域，非静态全局 变量的作用域是整个源程序（多个源文件可以共同使用）； 而静态全局变量则限制了其作用域， 即只在定义该变量的源文件内有效， 在同一源程序的其它源文件中不能使用它。2.static函数（也叫内部函数）只能被本文件中的函数调用，而不能被同一程序其它文件中的函数调用。区别于一般的非静态函数（外部函数） static在c里面可以用来修饰变量，也可以用来修饰函数。 先看用来修饰变量的时候。变量在c里面可分为存在全局数据区、栈和堆里。其实我们平时所说的堆栈是栈而不包含对，不要弄混。