static在C语言中的作用

static修饰变量

按照作用范围的不同，变量分为局部变量和全局变量。如果用static修饰变量，不论这个变量是全局的还是局部的都是存储在**全局/静态数据区**。

局部变量

- **普通局部变量**
  在任何一个函数内部定义的变量（不加static修饰符）都属于这个范畴。编译器一般不对普通局部变量进行初始化，也就是说它的值在初始时是不确定的，除非对其显式赋值。
  普通局部变量存储于进程栈空间，使用完毕会立即释放。

- **静态局部变量**
  使用static修饰符定义的局部变量，即使在声明时未赋初值，编译器也会把它初始化为0。
  静态局部变量存储于进程的静态数据区，即使函数返回，它的值也会保持不变。

具体程序如下

```
#include <stdio.h>

void example(void)
{
        int n=10;
        static int m=10;

        printf("n=%d\n", n);
        n++;
    printf("n++=%d\n", n);

        printf("m=%d\n", m);
        m++;
    printf("m++=%d\n", m);
}
int main(void)
{
        example();
        printf("--------------------\n");
        example();
        printf("--------------------\n");
        example();
```

```
        return 0;
}
```

运行结果如下

```
n=10
n++=11
m=10
m++=11
--------------------
n=10
n++=11
m=11
m++=12
--------------------
n=10
n++=11
m=12
m++=13
请按任意键继续. . .
```

全局变量

全局变量定义在函数体外部，在全局数据区分配存储空间，且编译器会自动对其初始化。

- 普通全局变量： 对整个工程可见，其他文件可以使用extern外部声明后直接使用。也就是说其他文件不能再定义一个与其相同名字的变量了（否则编译器会认为它们是同一个变量）。
- 静态全局变量：仅对当前文件可见，其他文件不可访问，其他文件可以定义与其同名的变量，两者互不影响。在定义不需要与其他文件共享的全局变量时，加上static关键字能够有效地降低程序模块之间的耦合，避免不同文件同名变量的冲突，且不会误使用。

static修饰函数

函数的使用方式与全局变量类似，在函数的返回类型前加上static，就是静态函数。其特性如下：

- 静态函数只能在声明它的文件中可见，其他文件不能引用该函数。
- 不同的文件可以使用相同名字的静态函数，互不影响。

static修饰类的成员

会**一直存在于程序的整个**生命周期中，不会像普通变量一样随着对象的存在而存在，消亡而消亡
静态成员变量的声明。

静态成员变量属于类，**该类的所有对象共享静态成员变量**。

示例如下

```
#include<iostream>
using namespace std;
class Data
{
    public:
        int n;
        static int m;     //静态成员变量 类内声明
};
int Data::m=0;          //类外初始化


int main()
{
    Data d1,d2;
    return 0;
}

// d1.n 与 d2.n 是两个不同的变量

// d1.m 与 d2.m 是同一个变量
```

静态变量成员的初始化

类的static变量**必须在类内声明，类外初始化**

静态成员变量的初始化语句为：
数据类型 类名::静态成员变量 = 初始值;