第五十七节： static 的重要作用。

【57.1　变量前加入 static 后发生的“化学反应”。】

有两类变量，一类是全局变量，一类是局部变量。定义时，在任何一类变量前面加入 static 关键词，变量原有的特性都会发生某些变化，因此，static 像化学的催化剂，具有神奇的功能。加 static 关键词的书写格式如下:

```
static unsigned char a;       //这是在全局变量前加的 static 关键词
void HanShu(void)
{
    static unsigned char i;   //这是在局部变量前加的 static 关键词
}
```

【57.2　在全局变量前加 static。】

static 读作“静态”，全局变量前加 static，称为静态全局变量。静态全局变量和普通全局变量的功能大体相同，仅在有效范围(作用域)方面有差异。假设整个工程有多个文件组成，普通全局变量的有效范围能覆盖全部文件，在任何一个文件里，以及跨文件与文件之间，在传递信息的层面上都畅通无阻。而静态全局变量只能在当前定义的那个文件里起作用，活动范围完全被限定在一个文件，彷佛被加了紧箍咒，由不得你任性，在传递信息的层面上仅仅局限于定义变量时所在的那一个文件。这部分的内容有个大致印象就可以，暂时不用深入研究，等以后学到“多文件编程”时再关注，因为我当前的程序例子只有一个源文件，还没涉及“多文件编程”。

【57.3　在局部变量前加 static。】

这是本节重点。我常把局部变量比喻宾馆的客房，客人入住时被分配在哪间客房是随机临时安排的，第二天退房时宾馆会把客房收回继续分配给下一位其他的客人，是临时公共区。而加入 static 后的局部变量，发生了哪些变化？加入 static 后的局部变量，称为静态局部变量。静态局部变量就像宾馆的 VIP 客户，VIP 客户财大气粗，把宾馆分配的客房永远包了下来，永远不许再给其它客人入住。总结了静态局部变量的两个重要特性：

第一个，静态局部变量不会在函数调用时被初始化，它只在单片机刚上电时被初始化了一次，因为它的内存模型不是分配在“栈”，而是跟全局变量一样放在“全局数据区”，拥有自己唯一的地址。因此，静态局部变量的数值跟全局变量一样，具有“记忆”功能，你每次调用某个函数，函数内部的静态局部变量的数值是维持最后一次被更改的数值，不会被“清零”的。但是跟全局变量又有差别，全局变量的有效范围（作用域）是整个工程，而静态局部变量毕竟是“局部”，在传递信息的层面仅局限于当前函数内。而普通局部变量，众所周知，每次被函数调用时，都会被重新初始化，会被“清零”的，没有“记忆”功能的。

第二个，每次函数调用时，静态局部变量比普通局部变量少开销一条潜在的“初始化语句”，原因是普通局部变量每次被函数调用时都要重新初始化，而静态局部变量不用进行这个操作。也就是说，静态局部变量比普通局部变量的效率高一点，虽然这个“点”的时间开销微不足道，但是写程序时不能忽略这个“点”。静态局部变量用到好处之时，能体现一个工程师的功力。

【57.4　静态局部变量的应用场合。】

静态局部变量适用在那些“频繁调用”的函数，比如 main 函数主循环 while(1)里直接调用的所有函数，

还有以后讲到的定时器中断函数，等等。因为静态局部变量每次被调用都不会被重新初始化，用在这类函数时就省去了每次初始化语句的时间。还有一类用途，就是那些规定不能被函数初始化的场合，比如在很多用 switch 搭建程序框架的函数里，这类 switch 程序框架俗称为状态机思路。

【57.5　能用全局变量替代静态局部变量吗？】

能用全局变量替代静态局部变量吗？能。哪怕在整个程序里全部用全局变量都可以。全局变量是一把牛刀，什么场合都用牛刀虽然也能解决问题，但是显得鲁莽没有条理。尽量把全局变量，普通局部变量，静态局部变量各自优势充分发挥出来才是编程之道。能用局部变量的尽量用局部变量，这样可以减少全局变量的使用。当局部变量帮分担一部分工作时，最后全局变量只起到一个作用，那就是在各函数之间传递信息。局部变量与全局变量的分工定位明确了，程序代码阅读起来就没有那么凌乱，思路也清晰很多。

【57.6　例程练习和分析。】

现在编写一个程序来熟悉 static 的性能。

```
/*---C 语言学习区域的开始。-----------------------------------------------*/

//函数的声明。
unsigned char HanShu(void);
unsigned char HanShu_static(void);

//变量的定义。
unsigned char a; //用来接收函数返回的结果。
unsigned char b;
unsigned char c;
unsigned char d;
unsigned char e;
unsigned char f;

//函数的定义。
unsigned char HanShu(void)
{
   unsigned char i=0;   //普通局部变量，每次函数调用都被初始化为 0.
   i++;  //i 自加 1
   return i;
}

unsigned char HanShu_static(void)
{
   static unsigned char i=0;   //静态局部变量，只在上电是此初始化语句才起作用。
   i++;  //i 自加 1
   return i;
}
```

```
void main() //主函数
{
    //下面函数内的 i 是普通局部变量，每次调用都会被重新初始化。
    a=HanShu();  //函数内的 i 每次重新初始化为 0，再自加 1，所以 a 等于 1。
    b=HanShu();  //函数内的 i 每次重新初始化为 0，再自加 1，所以 b 等于 1。
    c=HanShu();  //函数内的 i 每次重新初始化为 0，再自加 1，所以 c 等于 1。

    //下面函数内的 i 是静态局部变量，第一次上电后默认为 0，就不会再被初始化，
    d=HanShu_static(); //d 由 0 自加 1 后等于 1。
    e=HanShu_static(); //e 由 1 自加 1 后等于 2。
    f=HanShu_static(); //f 由 2 自加 1 后等于 3。

    View(a);  //把第 1 个数 a 发送到电脑端的串口助手软件上观察。
    View(b);  //把第 2 个数 b 发送到电脑端的串口助手软件上观察。
    View(c);  //把第 3 个数 c 发送到电脑端的串口助手软件上观察。
    View(d);  //把第 4 个数 d 发送到电脑端的串口助手软件上观察。
    View(e);  //把第 5 个数 e 发送到电脑端的串口助手软件上观察。
    View(f);  //把第 6 个数 f 发送到电脑端的串口助手软件上观察。

    while(1)
    {
    }
}
/*---C 语言学习区域的结束。----------------------------------------------*/
```

在电脑串口助手软件上观察到的程序执行现象如下：

```
开始...

第 1 个数
十进制:1
十六进制:1
二进制:1

第 2 个数
十进制:1
十六进制:1
二进制:1

第 3 个数
十进制:1
十六进制:1
二进制:1
```

```
第 4 个数
十进制:1
十六进制:1
二进制:1

第 5 个数
十进制:2
十六进制:2
二进制:10

第 6 个数
十进制:3
十六进制:3
二进制:11
```

分析：

变量 a 为 1。
变量 b 为 1。
变量 c 为 1。

变量 d 为 1。
变量 e 为 2。
变量 f 为 3。

【57.7　如何在单片机上练习本章节 C 语言程序？】

直接复制前面章节中第十一节的模板程序，练习代码时只需要更改“C 语言学习区域”的代码就可以了，其它部分的代码不要动。编译后，把程序下载进带串口的 51 学习板，通过电脑端的串口助手软件就可以观察到不同的变量数值，详细方法请看第十一节内容。