第五十一节： for 和 while 的循环嵌套。

【51.1　循环的嵌套。】

大循环的内部又包含了小循环，称为循环嵌套。生活中，循环嵌套的现象很常见，一年 12 个月，假设每个月都是 30 天（仅仅假设而已），1 月份 30 天，2 月份 30 天.....11 月份 30 天，12 月份 30，这里的年就是大循环，年内部的月就是小循环。一年 12 个月，大循环就是 12 次。一个月 30 天，小循环就是 30 次。用 for 语句来表达，大意如下：

```
for(m=1;m<=12;m++)  //大循环。一年 12 个月。这里的 m 看作月，代表一年 12 个月的大循环。
{
    for(d=1;d<=30;d++) //内嵌小循环。一月 30 天。这里的 d 看作天，代表一个月 30 天的小循环。
    {

    }
}
```

【51.2　循环嵌套的执行顺序。】

例子如下：

```
for(i=0;i<2;i++)  //大循环
{
    语句 1;
    for(k=0;k<3;k++) //内嵌的小循环
    {
        语句 2;
    }
    语句 3;
}
```

上述例子中，带 i 的 for 称为大循环，带 k 的 for 称为小循环，单片机从大循环入口进来，由上往下执行，执行第 1 次大循环，先执行 1 次“语句 1”，接着进入小循环，小循环要连续循环执行 3 次“语句 2”才跳出小循环，之后执行 1 次“语句 3”，然后再返回到大循环入口判断 i 条件是否满足，此时条件满足，继续执行第 2 次大循环，1 次“语句 1”，3 次“语句 2”，1 次“语句 3”，第 2 次循环结束后又返回到大循环入口判断 i 条件，此时 i 已经等于 2 不再小于 2 了，因此条件不满足，结束整个循环嵌套。上述执行的语句顺序如下：

```
语句 1;   //第 1 次大循环开始
语句 2;
语句 2;
语句 2;
语句 3;
语句 1;   //第 2 次大循环开始
语句 2;
语句 2;
语句 2;
语句 3;
```

根据此顺序，再看一个具体的程序例子：

```
a=0;
b=0;
for(i=0;i<2;i++)  //大循环
{
     a=a+1;  //被执行了 2 次
     for(k=0;k<3;k++) //内嵌的小循环
     {
           b= b+1;  //被执行了 6 次
     }
}
```

上述例子中，执行完程序后，a 的值变成了 2，b 的值变成了 6。重点分析 b 的变化，“b=b+1”在内嵌循环体里被执行了 6 次，6 次从何而来？就是 i 乘以 k 等于 6。这个乘法次数是循环嵌套一个很重要的特性。上述程序如果用 while 语句来实现，等效如下：

```
a=0;
b=0;
i=0;  //控制大循环的变量初始化
while(i<2)  //大循环
{
     a=a+1;  //被执行了 2 次
     k=0;    //控制小循环的变量初始化
     while(k<3) //内嵌的小循环
     {
         b= b+1;  //被执行了 6 次
         k=k+1;
    }
     i=i+1;
}
```

【51.3　循环嵌套的常见用途---二维数组的应用。】

二维数组 a[2][3]，它有 6 个变量，在没有学 for 语句之前，如果要依次把每个元素单独赋值清零真不容易，要写 6 次赋值语句如下：

```
a[0][0]=0;
a[0][1]=0;
a[0][2]=0;
a[1][0]=0;
a[1][1]=0;
a[1][2]=0;
```

自从懂了 for 嵌套语句之后，可以让同样功能的代码简洁许多。上述代码等效于如下：

```
for(i=0;i<2;i++)  //大循环
{
     for(k=0;k<3;k++) //内嵌的小循环
```

```
        {
            a[i][k]=0;
         }
    }
```

【51.4　循环嵌套的常见用途---大延时。】

单片机项目会经常会用到 delay 这个延时函数，大部分都是利用 for 循环来实现，小延时的函数往往不用嵌套，直接如下编写：

for(k=0;k<N;k++);

上述的 N 是控制循环次数，每次循环都要消耗单片机一点时间，如果 N 越大需要消耗的时间就越多，起到延时的作用。但是 N 所能取的最大值受它所定义的类型所限制，比如 unsigned char 类型最大范围是 255，unsigned int 类型最大范围是 65535，unsigned long 类型最大范围是 4294967295。如果要实现更大的延时怎么办？就可以用 for 的循环嵌套，利用循环嵌套可以使得循环总次数进行乘法翻倍的放大，很容易编写大延时的函数。比如：

```
    for(i=0;i<M;i++)  //大循环
    {
          for(k=0;k<N;k++); //内嵌的小循环
    }
```

此时循环的次数是 N 乘以 M 的乘积。如果 N 和 M 都是 unsigned long 类型，就意味着最大循环次数是 4294967295 的平方，次数大到惊人。

【51.5　例程练习和分析。】

现在编写一个循环嵌套的练习程序。

程序代码如下：

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

      unsigned char a=0;  //观察这个数最后的变化
      unsigned char b=0;  //观察这个数最后的变化
      unsigned char c=0;  //观察这个数最后的变化

      unsigned char i;  //控制大循环体的条件判断变量
      unsigned char k;  //控制内嵌小循环体的条件判断变量
void main() //主函数
{
      for(i=0;i<2;i++)  //大循环
      {
         a=a+1;    //被执行了 2 次
         for(k=0;k<3;k++)  //内嵌小循环
         {
              b=b+1;  //被执行了 6 次，也就是 i 乘以 k，2 乘以 3 等于 6.
         }
```

```
        c=c+1;    //被执行了 2 次
    }

    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。

    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:2
十六进制:2
二进制:10

第 2 个数
十进制:6
十六进制:6
二进制:110

第 3 个数
十进制:2
十六进制:2
二进制:10
```

分析：

变量 a 为 2。

变量 b 为 6。

变量 c 为 2。

【51.6　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。