第五十六节： return 在函数中的作用以及四个容易被忽略的功能。

【56.1　return 深入讲解。】

return 在英语单词中有“返回”的意思，上一节提到，凡是“有输出”的函数，函数内部必须有一个“return+变量或者常量”与之配套，表示返回的结果给外部调用者接收，这个知识点很容易理解，但是容易被忽略的是另外四个功能：

第一个是 return 语句隐含了立即退出的功能。退出哪？退出当前函数。只要执行到 return 语句，就马上退出当前函数。即使 return 语句身陷多层 while 或者 for 的循环中，它也毫不犹豫立即退出当前函数。

第二个是 return 语句可以出现在函数内的任何位置。可以出现在第一行代码，也可以出现在中间的某行代码，也可以出现在最后一行的代码，它的位置不受限制。很多初学者有个错觉，以为 return 只能出现在最后一行，这是错的。

第三个是 return 语句不仅仅可以用在“有输出”的函数，也可以用在“无输出”的函数，也就是可以用在前缀是 void 的函数里。回顾上一节，在“有输出”的函数里，return 后面紧跟一个变量或者常量，表示返回的数，但是在“无输出”的函数里，因为是“无输出”，此时 return 后面不用跟任何变量或者常量，这种写法也是合法的，表示返回的是空的。此时 return 主要起到立即退出当前函数的作用。

第四个是 return 语句可以在一个函数里出现 N 多次，次数不受限制，不一定必须只能一次。不管一个函数内有多少个 return 语句，只要任何一个 return 语句被单片机执行到，就立即退出当前函数。

【56.2　中途立即退出的功能。】

下面的书写格式是合法的：

```
void HanShu(void)  //“无输出”函数的定义。
{
    语句 1;
    return; //立即退出当前函数。对于这类“无输出”函数，return 后面没有跟任何变量或者常量。
    语句 2;
    return; //立即退出当前函数。对于这类“无输出”函数，return 后面没有跟任何变量或者常量。
    语句 3;
    return; //立即退出当前函数。对于这类“无输出”函数，return 后面没有跟任何变量或者常量。

}
```

分析：当 HanShu 此函数被调用时，单片机从“语句 1”往下执行，当遇到第一个 return 语句后，马上退出当前函数。后面的“语句 2”和“语句 3”等代码永远不会被执行到。多说一句，大家仔细看看 return 后面跟了什么数没有？什么都没有。因为此函数的前缀是 void 的，是“无输出”的。

【56.3　身陷多层 while 或者 for 的循环时的惊人表现。】

下面的书写格式是合法的：

```
void HanShu(void)  //“无输出”函数的定义。
{
```

```
    语句 1;
    while(1)  //第一个循环
    {
        while(1)  //第二个循环中的循环
       {
          return; //立即退出当前函数。
       }
       语句 2;
       return; //立即退出当前函数。
    }
    语句 3;
    return; //立即退出当前函数。
}
```

分析：当 HanShu 此函数被调用时，单片机从“语句 1”往下执行，先进入第一个循环，接着进入第二个循环中的循环，然后遇到第一个 return 语句，于是马上退出当前函数。后面的“语句 2”和“语句 3”等代码永远不会被执行到。此函数中，虽然表面看起来有那么多可怕的循环约束着，但是一旦碰上 return 语句都是浮云，立刻退出当前函数。

【56.4　在“有输出”函数里的书写格式。】

把上面例子中“无输出”改成“有输出”的函数后：

```
unsigned char HanShu(void)  //“有输出”函数的定义。
{
    unsigned char a=9;
    语句 1;
    while(1)  //第一个循环
    {
        while(1)  //第二个循环中的循环
        {
          return a; //返回 a 变量的值，并且立即退出当前函数。
        }
        语句 2;
         return a; //返回 a 变量的值，并且立即退出当前函数。
    }
    语句 3;
    return a; //返回 a 变量的值，并且立即退出当前函数。
}
```

分析：因为此函数是“有输出”的函数，所以 return 语句后面必须配套一个变量或者常量，此例子中配套的是 a 变量。当 HanShu 函数被调用时，单片机从“语句 1”往下执行，先进入第一个循环，接着进入第二个循环中的循环，然后遇到第一个“return a”语句，马上退出当前函数。而后面的“语句 2”和“语句 3”等代码是永远不会被执行到的。再一次说明了，return 语句不仅有返回某数的功能，还有立即退出的重

要功能。

【56.5　项目中往往是跟 if 语句搭配使用。】

前面的例子只是为了解释 return 语句的执行顺序和功能，实际项目中，如果中间有多个 return 语句，中间的 return 语句不可能像前面的例子那样单独使用，它往往是跟 if 语句一起搭配使用，否则单独用 return 就没有什么意义。比如：

```
void HanShu(void)  //“无输出”函数的定义。
{
    语句 1;
    if(某条件满足)
    {
       return; //立即退出当前函数。
    }
    语句 2;
    if(某条件满足)
    {
       return; //立即退出当前函数。
    }
    语句 3;
}
```

分析：单片机从“语句 1”开始往下执行，至于在哪个“return”语句处退出当前函数，就要看哪个 if 的条件满不满足了，如果所有的 if 的条件都不满足，此函数会一直执行完最后的“语句 3”才退出当前函数。

【56.6　例程练习和分析。】

写一个简单的除法函数，在除法运算中，除数不能为 0，如果发现除数为 0，就立即退出当前函数，并且返回运算结果默认为 0。

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

//函数的声明。
unsigned int ChuFa(unsigned int BeiChuShu,unsigned int ChuShu);

//变量的定义。
unsigned int a;//此变量用来接收除法的运算结果。
unsigned int b;//此变量用来接收除法的运算结果。

//函数的定义。
unsigned int ChuFa(unsigned int BeiChuShu,unsigned int ChuShu)
{
    unsigned int Shang;  //返回的除法运算结果：商。
```

```
    if(0==ChuShu)   //如果除数等于 0，就立即退出当前函数，并返回 0
    {
        return 0; // 退出当前函数并且返回 0.此时后面的代码不会被执行。
    }

    Shang=BeiChuShu/ChuShu;  //除法运算的算法
    return Shang;  //返回最后的运算结果：商。并且退出当前函数。
}

void main() //主函数
{
    a=ChuFa(128,0);  //函数调用。128 除以 0，把商返回给 a 变量。
    b=ChuFa(128,2);  //函数调用。128 除以 2，把商返回给 b 变量。

    View(a);  //把 a 发送到电脑端的串口助手软件上观察。
    View(b);  //把 b 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:0
十六进制:0
二进制:0

第 2 个数
十进制:64
十六进制:40
二进制:1000000
```

分析：

变量 a 为 0。

变量 b 为 64。

【56.7　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。