第五十二节： 支撑程序框架的 switch 语句。

【52.1　switch 的重要性。】

switch 是非常重要的语句，我所有的单片机项目都是用 switch 搭建程序主框架。如果说 while 和 for 是一对孪生兄弟，那么“if-else if”和 switch 也是一对孪生兄弟，凡是用“if-else if”能实现的功能都可以用 switch 实现。switch 有条件分支的功能，当条件的分支超过 3 个以上时，switch 会比“if-else if”更加直观清晰。

【52.2　switch 的语法。】

switch 常见的格式如下:

```
switch(变量)     //根据变量的数值大小从对应的 case 入口进来
{
     case 0:   //入口 0
         语句 0;
         break; //switch 程序体的出口之一
     case 1:  //入口 1
         语句 1;
         break; //switch 程序体的出口之一
     case 2: //入口 2
         语句 2;
         break; //switch 程序体的出口之一
}    //最下面的花括号也是一个 switch 程序体的出口之一
```

分析：单片机从第一行的 switch(变量)进来，依次往下查询跟变量匹配的 case 入口，然后从匹配的 case 入口进来，往下执行语句，直到遇上 break 语句，或者 return 语句，或者“最下面的花括号”这三种情况之一，才跳出当前 switch 程序体。上述例子中，假如变量等于 3,单片机从 switch(变量)进来，往下查询跟 3 匹配的 case 入口，因为没有发现 case 3，最后遇到“最下面的花括号”于是结束 switch 程序体，像这种变量等于 3 的情况，就意味着 switch 里面的有效语句没有被执行到。多补充一句，在 case 2 选项中，“语句 2”后面紧跟的 break 可以省略，因为 case 2 是最后一个 case，即使没有遇到 break 也会遇到“最下面的花括号”而结束 switch 程序体。上述程序功能如果用“if-else if”语句来实现，等效于如下：

```
if(0==变量)
{
    语句 0;
}
else if(1==变量)
{
    语句 1;
}
else if(2==变量)
{
    语句 2;
}
```

【52.3　switch 的 break。】

刚才的例子中，可以看到三个关键字:switch,case,break。其实并不是每个 case 都必须要跟 break 配套,break 只是起到一个出口的功能。假如没有遇到 break,程序会一直往下执行,直到遇到 break 或者 switch “最下面的花括号”为止。比如:

```
switch(变量)     //根据变量的数值大小从对应的 case 入口进来
{
     case 0:   //入口 0
         语句 0;
         break;
     case 1:  //入口 1
         语句 1;
     case 2: //入口 2
         语句 2;
         break;
     case 3: //入口 3
         语句 3;
         break;
}   //最下面的花括号也是一个 switch 程序体的出口之一
```

分析：假如此时 switch(变量)的变量等于 1，单片机经过查询后，就从匹配的 case 1 入口进来，执行“语句 1”后，居然没有遇到 break 语句，于是紧接着碰到“case 2”入口的语句，现在问题来了，单片机此时是退出 switch 程序体还是忽略“case 2”入口语句而继续执行后面的“语句 2”？答案是：忽略“case 2”入口语句而继续执行后面的“语句 2”。这里有点像坐地铁，你只关注一个入口和一个出口，进入地铁内之后，你中途再遇到无数个入口都可以忽略而继续前进，直到你到达目的地的出口才结束整个乘车过程。继续刚才的分析，单片机执行“语句 2”之后，紧接着遇到 break 语句，这时才跳出整个 switch 程序体。回顾一下整个流程，本例子中 case 1 没有 break 语句，就继续往下执行下面 case2 里面的语句，直到遇到 break 或者“最下面的花括号”为止。

【52.4　case 的变量有顺序要求吗？】

switch 语句内部的 case 有规定顺序吗？必须连贯吗？switch 程序体内部可以写很多 case 入口，这些 case 入口是不是必须按从小到大的顺序？是不是规定必须 case 数字连贯？答案是：没有规定顺序，也没有规定 case 数字连贯。case 的数值只是代表入口，比如以下两种写法都是合法的：

第一种：case 不按从小到大的顺序（这种格式是合法的）:

```
switch(变量)
{
     case 2:
         语句 2;
         break;
     case 0:
         语句 0;
         break;
```

```
        case 1:
            语句1;
            break;
    }
```

第二种：case 的数字不连贯（这种格式也是合法的）：

```
    switch(变量)
    {
        case 0:
            语句0;
            break;
         case 3:
            语句3;
            break;
         case 9:
            语句9;
            break;
    }
```

【52.5　switch 的 default。】

default 是入口语句，它在 switch 语句中也不是必须的，应根据程序需要来选择。default 相当于“if-else if-else ”组合语句中的 else，也就是当 switch 的入口变量没有匹配的 case 入口时，就会默认进入 default 入口，就像“if-else if-else ”语句中当前面所有的条件不满足时，就进入 else 语句的程序体，比如：

```
    switch(变量)     //根据变量的数值大小从对应的 case 入口进来
    {
         case 0:   //入口 0
             语句0;
             break; //switch 程序体的出口之一
         case 1:  //入口 1
             语句1;
             break; //switch 程序体的出口之一
         case 2: //入口 2
             语句2;
             break; //switch 程序体的出口之一
         default:  //当所有的 case 不满足，就从 default 的入口进来
             语句3;
             break;
    }    //最下面的花括号也是一个 switch 程序体的出口之一
```

分析：假如 switch 的入口变量等于 35，单片机从上往下查询，因为没有找到 case 35，所以就会从默认的 default 入口进来执行” 语句 3”，然后遇到 break 语句才跳出 switch 程序体。上述程序功能如果用“if-else if-else”组合语句来实现等效于如下：

```
if(0==变量)
{
    语句 0;
}
else if(1==变量)
{
    语句 1;
}
else if(2==变量)
{
    语句 2;
}
else   //相当于 switch 中的 default
{
    语句 3;
}
```

【52.6　switch 中内嵌 switch。】

if 语句可以内嵌 if 语句，while 语句也可以内嵌 while 语句，switch 语句当然也可以内嵌 switch。比如:

```
switch(a)
{
   case 1:
        switch(b)  //内嵌的 switch
        {
             case 1:
                  Break;
             case 2:
                  Break;
        }
        Break;
   case 2:
        Break;
}
```

分析：上述这种 switch 内嵌 switch 语句也是合法的，而且在实际项目中也很常用，大家目前先有个大概的了解即可，暂时不深入讲解。

【52.7　例程练习和分析。】

现在编写一个 switch 的练习程序。

程序代码如下：

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

     unsigned char k;   //switch 的入口变量
     unsigned char a;   //观察此变量的变化来理解 switch 的执行顺序
void main() //主函数
{
      a=0;
      k=2;   //入口变量等于 2
      switch(k)
      {
         case 0:  //入口 0
           a++;
           break; //跳出 switch
         case 1:  //入口 1
           a++;
         case 2:  //入口 2,上述 k 等于 2 所以从这里进来
           a++;
         case 3:  //入口 3
           a++;
         case 4:  //入口 4
           a++;
           break;  //跳出 switch
         case 5:  //入口 5
           a++;
           break;  //跳出 switch
         default:  //当前面没有遇到匹配的 case 入口时，就从此 default 入口进来
           a++;
           break;  //跳出 switch
      }            //最后一个 switch 的花括号也是跳出 switch

      View(a);  //把第 1 个数 a 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:3
```

```
十六进制:3
二进制:11
```

分析：

变量 a 为 3。单片机从 case 2 入口进来，因为 case 2 和 case 3 都没有 break 语句，直到遇到 case 4 的 break 语句才结束 switch 程序体，因此整个过程遇到了 3 次“a++”语句，因此变量 a 的“自加一”执行了 3 次后从 0 变成了 3。

【52.8　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。