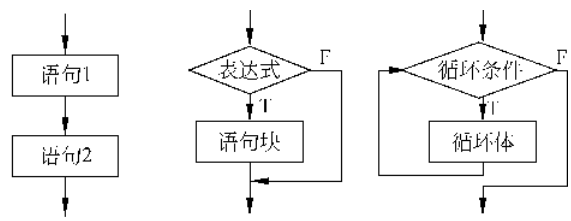
第3章 选择结构与循环结构设计

Visual Basic 的程序代码具有 3 种基本结构，即顺序结构、选择结构、循环结构。

顺序结构的程序段是一直从上往下逐行、逐个语句依次执行的，一般用于对事件进行顺序处理。顺序结构的程序在上一章中已有应用。顺序结构的流程图如图 3-1(a)所示。



(a) 顺序结构流程图　(b) 选择结构流程图　(c) 循环结构流程图

图 3-1　程序的 3 种基本结构

选择结构的程序段是由 If 或 Select Case 语句构成的，具有判断功能，一般用于对事件按不同情况选择不同的处理。选择结构的流程图如图 3-1(b)所示。

循环结构的程序段是由 For 或 Do 等循环语句构成的，具有重复执行某段程序的功能，一般用于对数据等进行累次处理。循环结构的流程图如图 3-1(c)所示。

在本章中，将对选择结构和循环结构进行讨论，全面学习程序的基本结构，掌握程序设计的基本方法，从而在求解实际应用中的问题时，能编写出相应的程序。

3.1　VB 布尔表达式

在 If 选择结构和 Do 循环结构中，均会对具体的条件进行判断，这就要用到 Visual Basic 的布尔表达式，即运算结果为 Boolean 类型的逻辑真值常量 True 或逻辑假值常量 False 的表达式，分为关系表达式和逻辑表达式。

1. 关系表达式

关系表达式是将同类型的数据用关系运算符连接而成的式子，用于对关系运算符两边的数据进行比较，常作为实际应用中的一个具体条件，写在 If、Do While 等语句中，实现某种判断。Visual Basic 中主要的关系运算符见表 3-1。

表 3-1　关系运算符及关系表达式示例

关系运算符	含　义	示例(设 x 值为 5、s 值为"AB")	运算结果
<	小于	x<5	False
<=	小于或等于	x<=5	True
=	等于	s="AB"	True
>=	大于或等于	x>=10	False
>	大于	x>4	True
<>	不等于	s<>"ABC"	True

注意：

以上关系运算符本身只能使用英文字符形式。

在关系运算中，涉及以下的比较规则。

(1) 若两操作数是数值型，则按其大小比较。

(2) 若两操作数是字符型，则首先比较两字符串的第一个字符的 ASCII 码值的大小，如果不相同，即比较两字符串的大小；如果相同，再比较第二个字符，依次类推。

(3) 关系运算符优先级相同，左边先运算。

但是，实际应用中一般是不应该出现连续的纯关系运算的，在 Visual Basic 中，这将导致很隐蔽的错误。例如，设 x 值为 5，那么 Visual Basic 对 0<x<4 的运算结果为 True！这是因为，Visual Basic 中实际存储 True 的值为－1，False 的值为 0。于是，0<x<4 先演算为－1<4 了。实际上，上式正确的写法要用到逻辑表达式：0<x And x<4。

2. 逻辑表达式

逻辑表达式是用逻辑运算符将关系表达式或具有逻辑值的数据连接而成的式子，常作为实际应用中的一种多因素条件，写在 If、Do While 等语句中，实现某种判断。

Visual Basic 中主要使用的逻辑运算符见表 3-2。

表 3-2　逻辑运算符及逻辑表达式示例

逻辑运算符	含　义	示例(设 x 值为 5、s 值为"AB")	运算结果
Not	逻辑非(取反)	Not s="AB"	False
And	逻辑与(并且)	0<x And x<4	False
Or	逻辑或(或者)	x>5 or s<>"ab"	True

值得注意的是，对于类似 0<x And x<4 这样的逻辑表达式，若写成 0<x And <4，是不合乎逻辑表达式的书写规则的，将出现编译错误，如图 3-2 所示。

图 3-2　Visual Basic 的"编译错误"消息框

逻辑运算的优先级顺序依次为 Not，And，Or。若一个较长的逻辑表达式中含有多种运算，则圆括号最优先，然后，先执行算术运算符、字符运算符，再执行关系运算符，最后执行逻辑运算符。

例如，要操作销量与单价之积大于 10 万的武汉及广州生产的某种商品，其逻辑表达式可形如：

```
销量 * 单价 > 100000 And (产地 = "武汉" Or 产地 = "广州")
```

其中具体的运算顺序请读者自己分析。

3.2　If 条件语句

先看看一个计算分段函数的数学问题：

$$y=\begin{cases} x & x\leqslant 0 \\ 2+x & x>0 \end{cases}$$

若编程求解 y 的计算结果，要根据输入的 x 的值来选取相应的计算式子，如果 x 小于等于 0，$y=x$；否则 $y=2+x$。

当编程所求解的应用问题要根据条件按不同情况作不同处理时，就要用到选择结构的程序，这经常利用 If 语句去实现。

3.2.1　单分支 If 语句

【例 3-1】 设两个变量 a 和 b 已输入了数值，比较它们的大小，若 a 小于 b，互换二者的值，使 a 大于 b。

分析：

（1）这是一个典型的单分支选择结构程序问题，实际要求是，当 a 小于 b 时，才互换二者的值，不满足此条件就无须互换，为此，程序中要用到单分支 If 语句；

（2）变量是内存中的单元，以高、低电位表示及存储数据，具有“存新冲旧”的特点，所以，a，b 两变量互换，不能直接写为 a＝b 和 b＝a，而要借助第 3 个变量。

相应程序代码如下：

```
Private Sub Command1_Click()
  Dim a As Integer, b As Integer
  Dim t As Integer
  a = Text1
  b = Text2
  If a < b Then
    t = a
    a = b
    b = t
  End If
  Label3 = "a = " & a & "      b = " & b
End Sub
```

运行该程序时，假设给 a 从 Text1 中输入 5，给 b 从 Text2 中输入 7，单击 Command1 触发程序的执行，a 与 b 将互换，如图 3-3 所示。

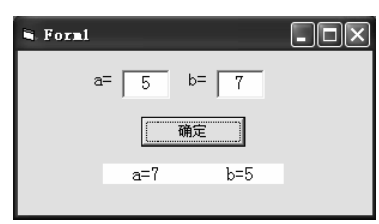


图 3-3　a，b 两变量互换

若给 a 输入 9，给 b 输入 7，执行程序时，用于互换

的 3 个赋值语句将不会被执行。亦即,写在程序中的这一分支的 3 个语句执行与否,是有选择的。

单分支 If 语句一般形式如下:

```
If <表达式> Then
    <语句块>
End If
```

其中表达式一般为关系表达式、逻辑表达式,也可为算术表达式。表达式值按非零为 True、零为 False 进行判断。

单分支 If 语句的执行过程是,当表达式的值为 True 时,执行 Then 下面语句块中的若干语句,再往 End If 之下执行;若表达式值为 False 则越过语句块,直接跳到 End If 之下执行。其流程如图 3-4 所示。

【例 3-2】 在文本框 Text1 的 KeyUp 事件中编写程序,使 Text1 中只接受数字输入,当输入非数字字符时,程序能自动选定这一字符,并弹出消息框要求重输。

分析:

显然,本问题中适宜用单分支 If 语句,另外,之所以利用 Text1 的 KeyUp 事件,是因为这样就对问题的处理时机更为准确,感兴趣的读者可将其中程序代码移到 Text1 的其他事件中,比较效果的相异之处。

程序代码如下:

```
Private Sub Text1_KeyUp(KeyCode As Integer, Shift As Integer)
  Dim test As Boolean
  'Print KeyCode                          '用于观察键码参数的值,现以前缀单撇号屏蔽
  test = KeyCode >= 48 And KeyCode <= 57                '大键盘区数字键
  test = test Or (KeyCode >= 96 And KeyCode <= 105)     '小键盘区数字键
  If Not test Then
    Text1.SetFocus
    Text1.SelStart = Len(Text1) - 1
    Text1.SelLength = Len(Text1)    '这三行实现字符的选定
    MsgBox "刚键入的不是数字!请重输。"
  End If
End Sub
```

程序运行效果如图 3-5 所示。

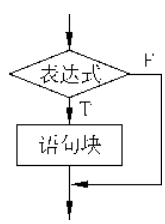


图 3-4 单分支选择结构程序流程图

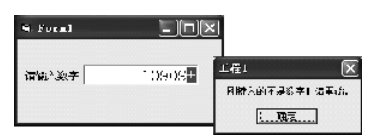


图 3-5 文本框输入控制程序运行效果

3.2.2　双分支 If 语句

【例 3-3】 简单的密码检验程序。根据输入的密码是否为“qwertyuiop”，分别作两种不同的处理。下面介绍其具体设计要求。

运行程序时，分别在两个文本框中输入用户名与密码，单击“确定”命令按钮，判断密码，如果密码字符为“qwertyuiop”，就在一个标签中显示出用户名及“你好，欢迎你!”的字样。

否则，清空该标签，再用消息框提示“密码错误！重新输入。”

分析：

(1) 对于输入的密码是否正确要进行判断，然后作不同的处理，为此，要在程序中运用双分支 If 语句；

(2) 本应用问题的界面设计要素见表 3-3，程序运行效果如图 3-6 所示。

表 3-3　密码检验程序窗体及各控件属性

控　　件	属　性　名	属　性　值
Form1	Caption	密码检验
Label1	Caption	请输入用户名：
Text1	Text	空
Label2	Caption	请输入密码：
Text2	PasswordChar	*
	Text	空
Command1	Caption	确定
Label3	BackColor	突出显示文本
	Caption	空

相应程序代码如下：

```
Private Sub Command1_Click()
  If Text2 = "qwertyuiop" Then
      Label3 = Text1 & "你好,欢迎你!"
  Else
      Label3 = ""
      MsgBox "密码错误!重新输入。"
  End If
End Sub
```

双分支 If 语句一般形式如下：

```
If <表达式> Then
    <语句块 1>
Else
     <语句块 2>
End If
```

其中表达式一般为关系表达式、逻辑表达式，也可为算术表达式。表达式值按非零为True、零为False进行判断。

在应用中，若要根据一个条件是否满足而作两种不同的处理时，应当使用双分支If语句。双分支If语句的执行过程是：

先计算If之右的表达式，当表达式的值为True时，选择Then下面语句块1执行其中的若干个语句，然后跳出到End If的下一行；否则，选择Else下面语句块2执行其中的若干个语句，再到End If的下一行。

其流程如图3-7所示。

图3-6　简单的密码检验程序

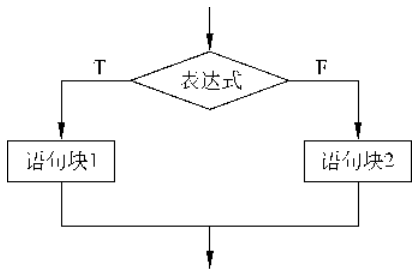


图3-7　双分支选择结构程序流程图

【例3-4】 编程求解本节开头的一个分段函数计算问题。本例请读者自行完成。

【例3-5】 以对一个标签可见性的控制，示范设计一个具有“开关”作用的命令按钮。程序运行效果先参见图3-8所示。

(a)　(b)

图3-8　具有“开关”作用的命令按钮

分析：

(1) 程序运行时，要求达到的具体功能是，单击命令按钮奇数次，标签隐去，成不可见的，命令按钮表面的字符变成“显示标签”，单击命令按钮偶数次，标签显出，成可见的，命令按钮表面的字符又变成“隐去标签”；

(2) 于是，问题关键在于命令按钮单击次数的记载及其奇、偶判断，为此，在窗体模块的通用段声明一个整形变量k，可在命令按钮的单击事件过程中首先赋值为k = k + 1，则每当该命令按钮被单击时，k值增1，记下了单击次数；

(3) 理论上，为确保整形变量k的值不致溢出，写为k = k Mod 2 + 1更好；

(4) 用双分支If语句，判断k是否为奇数，据此在双分支的选择结构中编写出两种处理的语句块。

程序代码如图 3-9 所示。

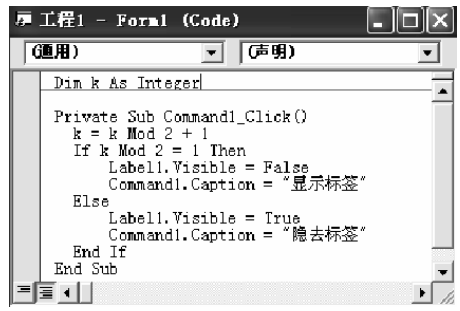


图 3-9　例 3-5 的程序代码窗口

3.2.3　多分支 If 语句

多分支 If 语句一般形式如下：

```
If <表达式 1> Then
     <语句块 1>
ElseIf <表达式 2> Then
     <语句块 2>
  …
ElseIf <表达式 n> Then
     <语句块 n>
[Else
     <语句块 n + 1> ]
End If
```

在实际应用中，利用该语句形成多分支选择结构，实现对多种不同的情况进行多种不同的处理。Visual Basic 判断其中各条件的顺序依次为表达式 1、表达式 2、……，一旦先遇到某表达式的值为 True，则执行该条件下的语句块，然后跳出到 End If 的下一行。其流程如图 3-10 所示。

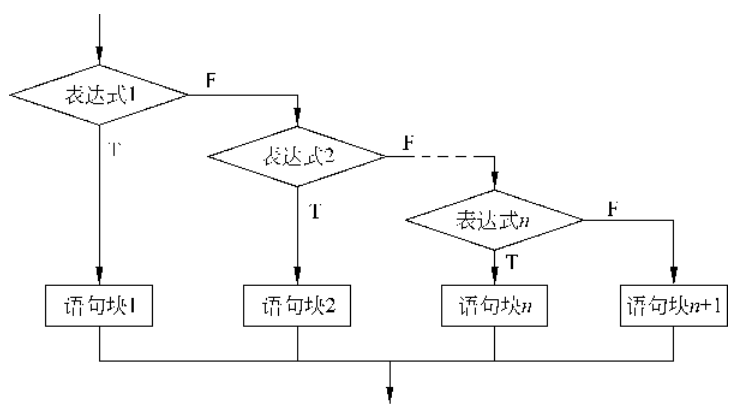


图 3-10　多分支选择结构程序流程图

【例 3-6】 “猜价”小游戏程序。随机产生 100 至 3000 中的一个整数赋给某变量，作为商品的价格，让游戏者从文本框中输入所猜的价格，确定后，程序经判断将表明猜价是“正确”还是“高了”、“低了”，并限定只给 10 次机会。若第 10 次仍没猜对，则不让输入猜价了，而将正确的价格公布出来。

分析：

(1) 本问题要涉及两个事件过程，一是单击“开始”命令按钮，象征性展示某商品，并产生一个限定范围的随机整数作为该商品的价格，赋给变量 x 存放在内存中；二是输入所猜的价格后，单击“确定”命令按钮，判断猜价与 x 比较的情况，显然，二者不应混合为一。

(2) 于是，变量 x 应为模块级的，另外，用于记载猜价次数的变量 n 也必须是模块级的。

(3) 判断猜价与 x 比较会有多种可能，程序段采用多分支 If 语句的选择结构，注意 ElseIf 不能写成 Else If。

程序运行效果及程序代码分别如图 3-11 和图 3-12 所示。

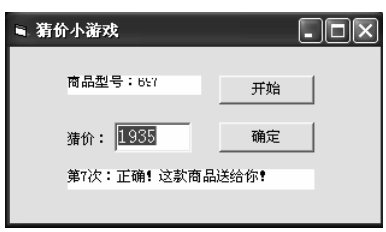


图 3-11　限次“猜价”小游戏程序

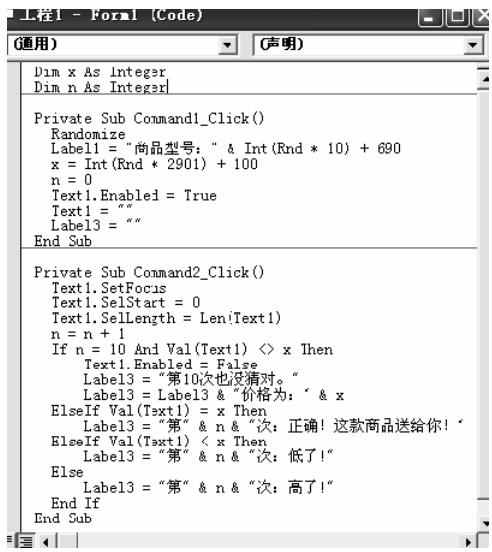


图 3-12　例 3-6 的程序代码窗口

如果读者有兴趣将这个小游戏设计为限时猜价的，待学习掌握了计时器控件的运用后，可以比较容易的实现。

3.2.4　If 语句的嵌套

If 语句嵌套，指的是在外层 If 语句的某分支的语句块中，又完整地包含着 If … End If 语句。

【例 3-7】 对于例 3-6 的判断猜价的过程作改进，如果猜对了，又判断若是第 3 次或更早即猜中，则在报告“正确”的字符信息的前面，加上惊叹的字符“哇!”。于是用到了 If 语句的嵌套。

程序段如下：

```
…
n = n + 1
If n = 10 And Val(Text1) <> x Then
    Text1.Enabled = False
    Label3 = "第 10 次也没猜对。"
    Label3 = Label3 & "价格为：" & x
ElseIf Val(Text1) = x Then
    Label3 = "第" & n & "次：正确!这款商品送给你!"
    If n <= 3 Then
        Label3 = "哇!" & Label3
    End If
ElseIf Val(Text1) < x Then
    Label3 = "第" & n & "次：低了!"
Else
    Label3 = "第" & n & "次：高了!"
End If
…
```

细心的读者可能会发现，在本章的程序中，If 语句各分支的语句块，左端均往右缩进了若干列，形成所谓的“锯齿形”。这是值得推荐的约定俗成的良好书写习惯，便于醒目地看清程序结构，利于读懂代码。在各种嵌套的程序结构中，更有必要遵守这种格式。

【例 3-8】 对于例 3-6 的判断猜价的过程，改为更高效、更实用的事件来触发，可利用 Text1 的 KeyPress 事件，当猜价输入完毕按回车键时，执行例 3-6 的判断猜价的程序，亦即，将原来“确定”命令按钮中的代码，移到本例事件的 If … Then 的下一行。如此，形成了 If 语句嵌套的应用。

该事件过程如下（花括号为帮助理解而加，不是程序中应有的）：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
  If KeyAscii = 13 Then   '若按了回车键
    Text1.SetFocus
    Text1.SelStart = 0
    Text1.SelLength = Len(Text1)
    n = n + 1
    If n = 10 And Val(Text1) <> x Then
        Text1.Enabled = False
        Label3 = "第 10 次也没猜对。"
        Label3 = Label3 & "价格为：" & x
    ElseIf Val(Text1) = x Then
        Label3 = "第" & n & "次：正确!这款商品送给你!"
    ElseIf Val(Text1) < x Then
        Label3 = "第" & n & "次：低了!"
    Else
        Label3 = "第" & n & "次：高了!"
    End If
  End If
End Sub
```

请注意该过程中两层 If 语句的完全包含关系，认清各层 If…Then 与相应 End If 的准确配对（如花括号所示）。

如果本例再结合例 3-7 的要求，则形成的是 3 层的 If 语句嵌套。虽然代码行数增加了，但结合实际去进行分析，应该能较清楚地理解程序的意图。

3.3 Select Case 多分支选择语句

Visual Basic 的程序设计中，还可以用 Select Case 语句组成多分支的选择结构。当多分支的各个情况的特征均体现为常量值时，采用此种结构编程就更为简明直观，程序可读性较强。

Select Case 语句的一般形式如下：

```
Select Case <变量或表达式>
  Case <表达式列表 1>
     <语句块 1>
  Case <表达式列表 2>
     <语句块 2>
  …
[ Case Else
     <语句块 n+1> ]
End Select
```

说明：

（1）＜变量或表达式＞可以是数值型或字符串表达式；

（2）＜表达式列表 i＞与句首的变量或表达式的类型相同，可以是下面 4 种形式之一，①表达式，②一组用逗号分隔的枚举值，③表达式 1 To 表达式 2，④Is 运算符表达式。

第一种形式与某个值比较，后 3 种形式与设定值的范围比较，4 种形式在数据类型相同的情况下，可以混合使用。例如：

```
Case  1 To 9                          '表示测试表达式的值在[1,9]的范围内
Case  "a", "e", "i", "o", "u"         '表示小写元音字母
Case  4,6,8,Is>10                     '表示测试表达式的值为 4,6,8 或大于 10
```

Select 语句的作用是根据句首＜变量或表达式＞的结果与各 Case 子句中＜表达式列表 i＞的值进行比较，决定执行哪一组语句块。若有多个 Case 子句中的值与测试值匹配，则根据自上而下判断次序，只执行第一个与之匹配的语句块。

【例 3-9】 “智报成语”。从文本框 Text1 中输入 1 到 10 之间的一个整数，程序报出一到十为特征的一个相应成语。

分析：

这些成语诸如一帆风顺、二龙戏珠、……、十全十美等。由于目前还没有讲解数组，所

以，在程序设计中，要写成多分支选择结构，根据输入的数字，使用 Select Case 语句，按情况在标签 Label2 中以字符常量显示相应成语。

程序代码如下：

```
Private Sub Command1_Click()
  Select Case Val(Text1)
    Case 1
      Label2 = "一帆风顺"
    Case 2
      Label2 = "二龙戏珠"
    Case 3 To 6
      Label2 = "三头六臂五湖四海"
    Case 7, 8
      Label2 = "七上八下"
    Case 9
      Label2 = "九霄云外"
    Case 10
      Label2 = "十全十美"
    Case Is > 10
      Label2 = "数不胜数"
    Case Else
      Label2 = ""
    MsgBox "无可奉告"
  End Select
End Sub
```

值得指出的是，当各个 Case 语句只能以关系或逻辑表达式才便于描述实际问题时，要将 Select Case 语句句首的表达式写为逻辑真值常量 True，于是其功能也就与多分支 If 语句等效了。

3.4　For 循环语句

【例 3-10】 编程计算 1 ＋ 3 ＋ 5 ＋ 7 ＋ … ＋ 101。

分析：

设计算结果用变量 s 表示。估计没有谁会在程序中直接写成 s＝1＋3＋5＋7＋ … ＋ 101 来求解，也不会以将各奇数都写全的方式来算出 s。怎么办呢？经典的算法是利用循环结构，让变量 i 从 1 到 101 依次取值，累次执行赋值语句 s ＝ s ＋ i，可称之为循环累加。

程序代码如下，其中用到 For 循环语句，实现对 s 的累加求解。

```
Private Sub Command1_Click()
  Dim s As Integer, i As Integer
  s = 0
```