

# 第 5 章 数 组

## 本章将介绍的内容

### 基础部分：

- 数组的定义和数组元素的引用。
- 控件数组的概念。
- 求平均值、最大(小)值；查找和排序算法。

### 提高部分：

- 数组的高级应用。

## 各例题知识要点

例 5.1 数组的概念；数组的定义；引用数组元素。

例 5.2 一维数组的应用：逆序输出。

例 5.3 一维数组的应用：求最大(小)值；顺序查找。

例 5.4 冒泡排序。

例 5.5 控件数组。

(以下为提高部分例题)

例 5.6 选择法排序。

例 5.7 二维数组。

例 5.8 过程调用时传递数组参数。

例 5.9 动态数组。

在计算机应用领域中,常常遇到需要对批量数据进行处理的情况,如统计大量的学生考试成绩和求平均值等。这类问题通常都具有数据处理量大、各数据间存在内部联系的特点。如果单纯采用简单变量处理这些数据,不但烦琐而且对于某些问题还可能根本无法解决。

在 VB 中,通常使用数组解决这类问题。所谓数组,就是由一组(若干个)类型相同的

相关变量结合在一起而构成的集合；而构成数组的每一个变量称为数组元素。作为同一数组中的元素，它们都使用统一的变量名称，只是通过不同的下标加以区分。只有一个下标的数组称为一维数组，有两个下标的数组称为二维数组。

## 5.1 一维数组

**【例 5.1】** 数组引例。窗体上添加 3 个标签和 2 个命令按钮，其中第 3 个标签的 AutoSize 属性为 True。程序运行时，随机产生 10 个两位整数显示在黄色标签中；单击“平均值”按钮，计算该 10 个数的平均值，显示在蓝色标签中，如图 5-1 所示；单击“大于平均值”按钮，找出该 10 个数中大于平均值的整数，显示在蓝色标签中，如图 5-2 所示。注意，只有在计算出平均值后，“大于平均值”按钮才有效。

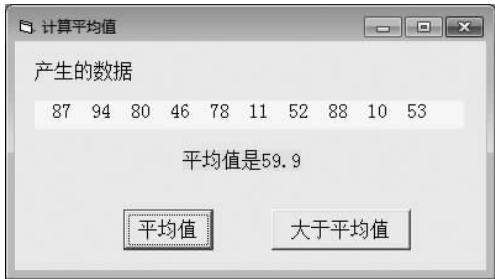


图 5-1 计算 10 个数据的平均值



图 5-2 显示大于平均值的数据

**【解】** 将标签 AutoSize 属性设置为 True 时，其大小可根据显示内容的长度自动进行调整。为了实现产生随机数据、计算平均值的功能，可编写代码如下：

```
Dim sum As Long                                '定义窗体级变量，存放 10 个整数的累加和
Private Sub Form_Load()
    Dim a As Long
    Dim i As Long
    Randomize                                    '产生随机种子
    For i=1 To 10
        a=Int(Rnd * 90)+10                      '产生一个 10 到 99 之间的整数 a
        lblData.Caption=lblData.Caption & " " & a    '连接显示 a
        sum=sum+a                                '将 a 累加到 sum 中
    Next
End Sub

Private Sub cmdAve_Click()                      '单击“平均值”按钮
    lblAve.Caption="平均值是" & sum / 10
    cmdLarge.Enabled=True                       '使“大于平均值”按钮可用
End Sub
```

在 Form\_Load 过程的 For 循环中，使用变量 a 依次存放 10 个随机数据，待循环结束

时 sum 中存放了 10 个数据的累加和,而 a 中仅保留了最后一个随机数据。

但是,为了进一步找出大于平均值的各个数据,就必须全部保存 10 个整数,以便在计算出平均值后,再通过比较判断,依次得到大于平均值的各数据。为此,不能只定义 1 个变量 a,而需要定义 10 个变量,如 a1,a2,⋯,a10。由于各变量相互独立,不能再使用 For-Next 语句产生数据。修改后的程序代码如下:

```
Dim sum As Long           '以下 11 个变量在两个事件中都要使用,定义为窗体级变量
Dim a1 As Long            '以下定义 10 个窗体级变量,分别存放 10 个随机数
Dim a2 As Long
    ⋮
Dim a10 As Long

Private Sub Form_Load()
    Randomize
    a1=Int(Rnd * 90)+10     '以下 10 条语句产生 10 个随机数并保存
    a2=Int(Rnd * 90)+10
        ⋮
    a10=Int(Rnd * 90)+10
    lblData.Caption=a1 & " " & a2 & ... & a10    '显示 10 个数据
    sum=a1+a2+...+a10      '累加 10 个数据
End Sub

Private Sub cmdAve_Click() '单击“平均值”按钮
    lblAve.Caption="平均值是" & sum / 10
    cmdLarge.Enabled=True
End Sub

Private Sub cmdLarge_Click() '单击“大于平均值”按钮
    Dim ave As Double
    ave=sum / 10
    lblAve.Caption=""
    If a1 > ave Then        '以下 10 个 If 语句,找出大于平均值的各数据
        lblAve.Caption=lblAve.Caption & a1 & " "
    End If
    If a2 > ave Then
        lblAve.Caption=lblAve.Caption & a2 & " "
    End If
        ⋮
    If a10 > ave Then
        lblAve.Caption=lblAve.Caption & a10
    End If
End Sub
```

这样烦琐的程序才仅对 10 个数据进行处理,若要求对 100 个、1000 个甚至更多的批

量数据进行同样的处理,则代码量激增,令人无法接受。使用本章介绍的数组解决此类批量处理问题,将使整个程序代码书写简洁、清晰。使用数组实现例 6.1 功能的程序代码如下:

```
Dim sum As Long
Dim a(1 To 10) As Long           '定义数组 a,包含 10 个 Long 类型的元素
Private Sub Form_Load()
    Dim i As Long
    Randomize
    For i=1 To 10
        a(i)=Int(Rnd * 90)+10      '产生随机数,存放在下标为 i 的元素中
        lblData.Caption=lblData.Caption & a(i) & " "
        sum=sum+a(i)              '将下标为 i 的元素值累加到 sum 中
    Next
End Sub

Private Sub CmdAve_Click()
    lblAve.Caption="平均值是" & sum / 10
    cmdLarge.Enabled=True
End Sub

Private Sub CmdLarge_Click()
    Dim ave As Double
    Dim i As Long
    ave=sum / 10
    lblAve.Caption=""
    For i=1 To 10
        If a(i) > ave Then          '判断下标为 i 的元素是否大于平均值 ave
            lblAve.Caption=lblAve.Caption & a(i) & " " '连接 a(i)
        End If
    Next
End Sub
```

程序说明:

(1) 采用数组后可以方便地使用同一个数组名代表逻辑上相关的一批变量(10 个),而为了表示不同的数组元素,只需要简单地指出该元素的下标即可。

(2) 语句 Dim a(1 To 10) As Long 定义了一个名为 a 的一维数组,它包含 10 个元素: a(1), a(2), ..., a(9), a(10), 其中每一个元素都是一个 Long 类型的变量,也就是说,在每个元素中只能存放整型数据。请注意: 在使用数组前,必须对其进行定义。VB 中定义一维数组的一般形式如下:

```
Dim 数组名 ([下界 To ]上界) As 数据类型
```

其中,上界和下界规定了数组元素下标的取值范围,它们的值不得超过 Long 数据类

型的范围。数组中所包含元素的个数为：上界一下界+1。若省略[下界 To]，则系统默认下界为 0。例如：

```
Dim b(-2 To 3) As Long
```

定义了一维数组 b，其中包含 6 个 Integer 型元素：b(-2)，b(-1)，b(0)，b(1)，b(2)，b(3)；

```
Dim x(4) As String
```

定义了一维数组 x，其中包含 5 个字符型元素 x(0)，x(1)，…，x(4)。

(3) 数组名的命名规则与变量名相同。

(4) 数组元素代表内存中的一个存储单元，它可以像普通变量一样使用，只不过数组元素用下标形式表示。引用一维数组元素的一般形式是

数组名(下标)

**注意：**下标的取值范围应在定义该数组时所限定的[下界，上界]范围内，不得越界。

(5) 系统为数组中的元素分配连续的内存单元。如图 5-3 所示为数组 a 在内存中的存储结构示意图，系统为其分配 10 个连续的存储单元(4B×10=40B)。

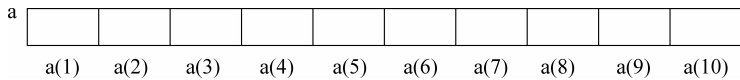


图 5-3 数组的存储结构示意图

(6) 由于在多个事件过程中都要用到变量 sum 和数组 a，所以将它们定义为窗体级。

**【例 5.2】** 窗体上添加 4 个标签和 1 个命令按钮。程序运行时，随机产生 20 个两位整数，显示在黄色标签中，如图 5-4 所示；单击“逆序输出”按钮，在绿色标签中逆序显示各个整数，如图 5-5 所示。

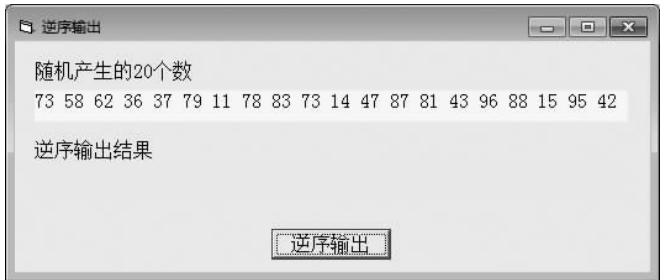


图 5-4 运行初始界面

**【解】** 为了逆序输出各数据，必须要全部保存 20 个整数，如果使用普通的变量存储，需定义 20 个不同的变量，处理起来十分烦琐。而且当处理的数据量增加到上百甚至上千时，采用普通的变量存储的方法便无法适用。因此，这种情况下，我们引入数组存储一组同类型的内部有联系的变量。

程序代码如下：

```
Option Base 1
```

'指定数组默认的元素下标下界



图 5-5 逆序输出

```

Dim a(1 to 20) As Long                                '定义窗体级数组 a, 包含 20 个整型元素

Private Sub Form_Load()
    Dim i As Long
    For i=1 To 20
        a(i)=Int(Rnd * 90)+10                        '产生随机数,存放在下标为 i 的元素中
        lblData.Caption=lblData.Caption & a(i) & " " '将 a(i) 连接到标签中
    Next
End Sub

Private Sub CmdRev_Click()
    Dim i As Long
    For i=20 To 1 Step -1                            '逆序输出
        lblOut.Caption=lblOut.Caption & a(i) & " "
    Next
End Sub

```

程序说明：

(1) 语句 Option Base 1 的作用是,在定义数组时若省略[下界 To],则默认元素的下标从 1 开始。因此,本例中语句 Dim a(10) As Long 等价于 Dim a(1 To 10) As Long。出现在 Option Base 后面的数字只能是 0 或 1,如果是 0,则元素下标从 0 开始(此时该语句没有实质意义)。

(2) Option Base 语句必须出现在数组定义之前,且位于所有事件过程的前面。其作用范围仅限于出现在同一代码窗口且在定义时未指出下标下界的数组。在一个代码窗口中,Option Base 语句只能出现一次。

(3) 通常情况下,数组操作总是借助于循环语句实现。在例 5.1 和例 5.2 中,正是使用了 For-Next 语句,巧妙地利用循环变量 i 与数组元素下标的一一对应关系实现了对数组元素的逐一引用,并对数组元素进行相应的处理。

现实生活中的许多问题都可以使用数组解决。为了更好地掌握数组的应用,应在平时的编程练习中注意多模仿、多实践,不断总结经验,提高独立编程能力。

**【例 5.3】** 窗体上添加 6 个标签和 1 个命令按钮。程序运行时,随机产生 10 个 1~100 的整数显示在黄色标签中,如图 5-6 所示;单击“查找”按钮,找出该 10 个数中的最大

值,显示在蓝色标签中,将最大值所在的位置显示在绿色标签中,如图 5-7 所示。

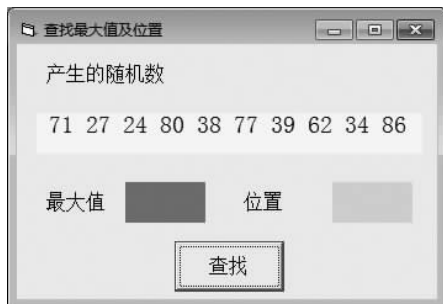


图 5-6 运行初始界面

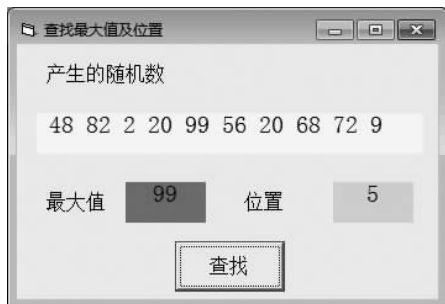


图 5-7 查找最大值及位置

**【解】** 题目要求查找一组数中的最大值及其所在位置,需要将这组数保存到数组中,并根据数组的下标确定数据元素所在的位置。一种简单的方法是定义数组时下标从 1 开始,这样数组元素下标就与位置完全对应。

程序代码如下:

```
Dim a(1 To 10) As Long
Private Sub CmdSearch_Click()
    Dim i As Long
    Dim j As Long
    Dim Max As Long
    Max=a(1)
    j=1
    For i=2 To 10
        If Max<a(i) Then
            Max=a(i)
            j=i
        End If
    Next
    LblMax.Caption=Max
    LblIndex.Caption=j
End Sub

Private Sub Form_Load()
    Dim i As Long
    Randomize
    For i=1 To 10
        a(i)=Int(Rnd * 100)+1
        LblData.Caption=LblData.Caption & " " & a(i)
    Next
End Sub
```

程序说明:

(1) Dim a(1 To 10) As Long 定义了窗体级的数组 a,且下标从 1 开始。Max 和 j 两个变量分别保存当前数组元素中的最大值和下标。查找中首先将两个变量分别赋值为第一个数组元素的值和下标。通过循环依次对后续的 9 个数组元素进行顺序查找。最后将这两个变量的值显示在界面对应的标签中。

(2) 如果定义数组时下界省略,则数组元素下标从 0 开始。此时,元素所在位置与下标之间差 1,界面上显示时就需要将下标+1。

**【例 5.4】** 冒泡排序。窗体上添加 4 个标签和 2 个命令按钮。程序运行时,单击“产生数据”按钮,随机产生 10 个 1~100 的整数显示在黄色标签中,如图 5-8 所示;单击“冒泡排序”按钮,用冒泡排序法对黄色标签中的整数按照从小到大的顺序排序,显示在绿色标签中,如图 5-9 所示。



图 5-8 产生数据



图 5-9 冒泡排序

**【解】** 题目要求将一组数用冒泡排序法从小到大排序。首先需要将这组数保存到数组中,然后按照冒泡排序的方法进行排序。冒泡排序的思想是:比较相邻的两个元素,如果第一个比第二个大,就交换它们两个。从前往后对每一对相邻元素做同样的工作,至此最后的元素应该是最大的数,第一轮比较结束。第二轮则对除最后一个元素外的其他元素重复上述步骤。后续每一轮都对持续减少的元素重复上述过程,直到没有任何一对元素需比较,则排序结束。

程序代码如下:

```
Dim a(9) As Long
Private Sub CmdRnd_Click()
    Dim i As Long
    Randomize
    For i=0 To 9
        a(i)=Int(Rnd * 90)+10
        LblBefore.Caption=LblBefore.Caption & " " & a(i)
    Next
End Sub

Private Sub CmdSort_Click()
```



```

Dim i As Long
Dim j As Long
Dim t As Long

For i=1 To 9
    For j=0 To 9 - i
        If a(j) > a(j+1) Then
            t=a(j)
            a(j)=a(j+1)
            a(j+1)=t
        End If
    Next
Next

For i=0 To 9
    LblAfter.Caption=LblAfter.Caption & " " & a(i)
Next
End Sub

```

程序说明：

(1) Dim a(9) As Long 定义了窗体级的数组 a, 由于下界省略, 因此下标为 0~9, 共 10 个元素。定义两个循环变量 i 和 j, i 表示冒泡排序的轮次, 共需 9 轮; j 则表示每轮中比较的元素下标范围。随着轮次增加, 需比较的元素持续减少。

(2) 如果排序要求是从大到小, 则只需要在比较时将判断语句改为  $a(j) > a(j + 1)$  时交换即可。

## 5.2 控件数组

**【例 5.5】** 利用单选按钮设置不同字体。要求当单击某个单选按钮时, 改变标签中文字的字体, 界面如图 5-10 所示, 执行界面如图 5-11 所示。

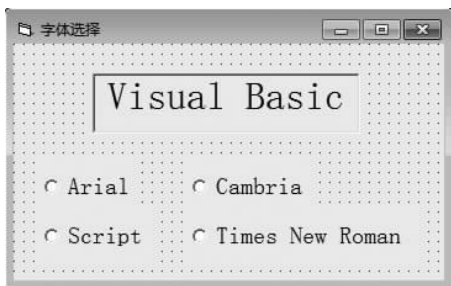


图 5-10 设计界面

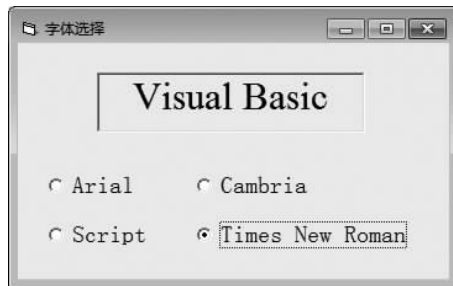


图 5-11 字体选择

**【解】** 本例中字体的选择采用的都是单选按钮, 类型相同, 而且这 4 个单选按钮的名

称就是字体名称。下面采用控件数组实现。

程序代码如下：

```
Private Sub OptFont_Click(Index As Integer)
    LblTitle.Font=OptFont(Index).Caption
End Sub
```

程序说明：

(1) 将具有相同类型和名称的一组控件称为控件数组。以本例题中的 4 个单选按钮为例，它们都属于同一控件类型——单选按钮，且具有相同的控件名 OptFont，因而这 4 个单选按钮就是一个控件数组。与前面介绍的数组类似，可通过索引号(Index 属性)标识和区分同一控件数组内的各个控件。

(2) 在设计阶段，添加第 1 个单选按钮后，其 Index 属性值是空的，但把第 1 个按钮复制 3 次并创建成单选按钮控件数组后，此时 4 个单选按钮的 Index 属性自动变为 0、1、2、3，这是系统默认设置的结果。实际上也可以通过 Index 属性人为指定索引号，其取值范围是 0~32 767，但对于同一控件数组中的各控件，索引号必须互不相同。

(3) 在设计时，使用控件数组比直接向窗体中添加相同类型的多个控件所消耗的资源要少；而且对于同一控件数组中的各控件，它们都共享相同的事件过程，从而可减少程序的编码量。

(4) 程序运行时无论单击哪一个单选按钮，都会触发 OptFont\_Click 事件过程。

当单击 Arial 按钮时，将调用 OptFont 的公共事件过程 OptFont\_Click，并将该按钮的索引号 0 赋予参数 Index，再根据 Index 的不同取值而执行相应操作。同理，单击 Times New Roman 按钮时，也将执行 OptFont\_Click 事件过程，并将索引号 3 传递给 Index 参数，即同一控件数组中的各控件共享同一事件过程。

(5) 引用控件数组中各控件对象的方法是：

控件数组名(索引号)

如 OptFont(0)代表 OptFont 数组中索引号为 0 的单选按钮，即 Arial 按钮；OptFont(0).Caption 则引用的是索引号为 0 的单选按钮的 Caption 属性。

## 5.3 提高部分

### 5.3.1 数组的高级应用

**【例 5.6】** 删除数组元素。窗体上添加 5 个标签、1 个文本框和 1 个命令按钮，如图 5-12 所示。程序运行时，随机产生 10 个两位整数显示在黄色标签中，在文本框中输入要删除数组元素的下标，单击“删除”按钮，将指定下标的数组元素删除，并将删除后的新数组显示在绿色标签中，如图 5-13 所示。