第 7 章　复合数据类型

📖 前几章介绍的都是属于基本数据类型或简单数据类型（字符串、整型、实型等）的数据，可以通过简单变量名来访问它们的元素。除基本数据类型外，Visual Basic 还提供了复合数据类型，包括数组和记录。复合数据类型是按照一定规则组成的元素类型的数据，元素类型又称基类型，它可以是简单数据类型，也可以是复合数据类型。对于复合数据类型来说，不能用一个简单变量名来访问它的某个元素。

本章将介绍 Visual Basic 提供的两种复合数据类型，即数组和记录。💻

7.1　数组

数组是有序的数据集合。在其他语言中，数组中的所有元素都属于同一个数据类型；而在 Visual Basic 中，一个数组中的元素可以是相同类型的数据，也可以是不同类型的数据。

7.1.1　数组的定义

数组必须先定义后使用。在计算机中，数组占据一块内存区域，数组名是这个区域的名称，区域的每个单元都有自己的地址，该地址用下标表示。定义数组的目的就是通知计算机为其留出所需要的空间。

在 Visual Basic 中，可以用 4 个语句来定义数组。这 4 个语句格式相同，但适用范围不一样，其中：

- Dim 或 Private 用在窗体模块或标准模块中，定义窗体或标准模块数组，也可用于过程中。
- ReDim 用在过程中。
- Static 用在过程中。
- Public 用在标准模块中，定义全局数组。

以上 4 个语句都可以用来定义数组，下面以 Dim 语句为例来说明数组定义的格式，当用其他语句定义数组时，其格式是一样的。

1. **第一种格式**

对于一维数组，格式如下：

```
Dim  数组名(下标上界)As 类型名称
```

对于二维数组，格式如下：

```
Dim  数组名(第一维下标上界,第二维下标上界)As 类型名称
```

例如：

```
Dim  Test(2,3)As Integer
```

在一般情况下，下标的下界默认为 0。如果希望下标从 1 开始，可以通过 Option Base 语句来设置，其格式如下：

```
Option  Base  n
```

Option Base 语句用来指定数组下标的默认下界。

格式中的 n 为数组下标的下界，只能是 0 或 1，如果不使用该语句，则默认值为 0。Option Base 语句只能出现在窗体层或标准模块层，不能出现在过程中，并且必须放在数组定义之前。此外，如果定义的是多维数组，则下标的默认下界对每一维都有效。

2. 第二种格式

用第一种格式定义的数组，其下标的下界只能是 0 或 1，而如果使用第二种格式，则可根据需要指定数组下标的下界。格式如下：

```
Dim 数组名([下界 To]上界[,[下界 To]上界]…)As 类型名称
```

可以看出，第二种格式实际上已包含了第一种格式，只要省略格式中的“下界 To”，即变为第一种格式。当下标为 0 或 1 时，可以省略“下界 To”。因此，如果不使用 Option Base 语句，则下述数组定义语句是等效的：

```
Dim A(8,3)
Dim A(0 To 8,0 To 3)
Dim A(8,0 To 3)
```

数组可以通过前面介绍的两种格式定义，无论用哪一种格式定义数组，下界都必须小于上界。有时可能需要知道数组的上界值和下界值，这可以通过 Lbound 和 Ubound 函数来测试，其格式如下：

```
Lbound(数组[,维])
Ubound(数组[,维])
```

这两个函数分别返回一个数组中指定维的下界和上界。其中“数组”是一个数组名，“维”是要测试的维。Lbound 函数返回“数组”某一“维”的下界值，而 Ubound 函数返回“数组”某一“维”的上界值，两个函数一起使用即可确定一个数组的大小。对于一维数组来说，参数“维”可以省略。如果要测试多维数组，则“维”不能省略。

7.1.2　动态数组

定义数组后，为了使用数组，必须为数组开辟所需要的内存区。根据内存区开辟时机的不同，可以把数组分为静态（Static）数组和动态（Dynamic）数组。通常把需要在编译时开辟内存区的数组叫做静态数组，而把需要在运行时开辟内存区的数组叫做动态数组。

1. 动态数组的定义

动态数组以变量作为下标值，在程序运行过程中完成定义，通常分为两步:首先在窗体层、标准模块或过程中用 Dim 或 Public 声明一个没有下标的数组（括号不能省略），然后在过程中用 ReDim 语句定义带下标的数组。例如：

```
Dim TV() As Integer          '在窗体层或标准模块中声明
Dim Size As Integer
Sub Form_Click()
   …
```

```
    Size = InputBox("Enter a value:", "Data", "12")
    ReDim TV(Val(Size))
    …
End Sub
```

该例先在窗体层或标准模块中用 Dim 语句声明了一个空数组 TV 和一个变量 Size，然后在过程中用 ReDim 语句定义该数组，下标 Size 在运行时输入。

ReDim 语句的格式如下：

```
ReDim [Preserve] 变量(下标)As 类型
```

该语句用来重新定义动态数组，按定义的上、下界重新分配存储单元，并可为定义的变量改变存储类型。当重新分配动态数组时，数组中的内容将被清除，但如果在 ReDim 语句中使用了 Preserve 选择项，则不清除数组内容。在 ReDim 语句中可以定义多个动态数组，但每个数组必须事先用“Dim Variable() As…”或“Public Variable() As…”这种形式进行声明，在括号中省略上、下界，在用 ReDim 语句重新定义时指定数组下标的上、下界。例如：

```
Dim StuName$(), Address$(), Cty$()
…
ReDim StuName$(Length), Address$(Addr), Cty$(ct)
```

ReDim 只能出现在事件过程或通用过程中，用它定义的数组是一个“临时”数组，即在执行数组所在的过程时为数组开辟一定的内存空间，当过程结束时，这部分内存即被释放。

在一个程序中，可以多次用 ReDim 语句定义同一个数组，随时修改数组中元素的个数，改变数组的维数。但不能用 ReDim 语句改变数组类型。

2. 数组的清除和重定义

数组一经定义，便在内存中分配了相应的存储空间，其大小是不能改变的。也就是说，在一个程序中，同一个数组只能定义一次。有时可能需要清除数组的内容或对数组重新定义，这可以用 Erase 语句来实现，其格式如下：

```
Erase 数组名[,数组名]……
```

Erase 语句用来重新初始化静态数组的元素，或者释放动态数组的存储空间。注意，在 Erase 语句中，只给出要刷新的数组名，不带括号和下标。例如：

```
Erase TV
```

当把 Erase 语句用于静态数组时，如果这个数组是数值数组，则把数组中的所有元素置为 0；如果是字符串数组，则把所有元素置为空字符串；如果是记录数组，则根据每个元素（包括定长字符串）的类型重新进行设置。而当把 Erase 语句用于动态数组时，将删除整个数组结构并释放该数组所占用的内存。也就是说，动态数组经 Erase 后即不复存在；而静态数组经 Erase 后仍然存在，只是其内容被清空。

7.2 数组的基本操作

建立一个数组之后，可以对数组或数组元素进行操作。数组的基本操作包括输入、输出及复制，这些操作都是对数组元素进行的。此外，Visual Basic 还提供了 For Each…Next 语句，可用于对数组的操作。

7.2.1　数组元素的输入、输出和复制

1. 数组的引用

数组的引用通常是指对数组元素的引用，其方法是在数组后面的括号中指定下标，例如：

```
x(8), y(2,3), z%(3)
```

一般来说，在程序中，凡是简单变量出现的地方，都可以用数组元素代替。数组元素可以参加表达式的运算，也可以被赋值。例如：

```
x(5)=x(2)+ x(4)
```

2. 数组元素的输入

数组元素一般通过 For 循环语句及 InputBox 函数输入。当数组较小或只需要对数组中的指定元素赋值时，可以用赋值语句来实现数组元素的输入。

多维数组元素的输入通过多重循环来实现。由于 Visual Basic 中的数组是按行存储的，因此，应把控制数组第一维的循环变量放在最外层循环中。

注意：当用 InputBox 函数输入数组元素时，如果要输入的数组元素是数值类型，则应显式定义数组的类型，或者把输入的元素转换为相应的数值，因为用 InputBox 函数输入的是字符串类型。

对于一维数组，可以用 Array 函数来为数组元素赋值，其格式如下：

```
数组变量名 = Array(数组元素值)
```

这里的“数组变量名”是预先定义的数组名，在“数组变量名”之后没有括号。之所以称为“数组变量”，是因为它作为数组使用，但作为变量定义，它既没有维数，也没有上、下界。“数组元素值”是需要赋给数组各元素的值，各值之间以逗号分开。例如：

```
Static Numbers As Variant
Numbers = Array(l,2,3,4,5)
```

将把 1、2、3、4、5 这 5 个数值赋给数组 Numbers 的各个元素，即 Numbers(0)=1，Numbers(l)=2，Numbers(2)=3，Numbers(3)=4，Numbers(4)=5。注意，在默认情况下，数组的下标从 0 开始，数组 Numbers 有 5 个元素。如果想使下标从 1 开始，则应执行

```
Option Base 1
```

对于字符串数组，其初始化操作相同。

注意：数组变量不能是具体的数据类型，只能是变体类型。

在一般情况下，数组元素的值通过赋值语句或 InputBox 函数读入数组，如果使用 Array 函数，则可使程序大为简化，例如：

```
Static  stName As Variant
StName = Array("王大明","李小芸","俗蓉","东方明","辛向荣")
```

3. 数组元素的输出

数组元素的输出可以用 Print 方法来实现。假定有一个 4 行 4 列的二维数组 a(4,4)，为了使数组中的数据仍按原来的 4 行 4 列输出，可以这样编写程序：

```
For i=l To 4
  For j = 1 To 4
    Print a(i,j); " ";
  Next j
```

```
    Print
Next i
```

4. 数组元素的复制

单个数组元素可以像简单变量一样从一个数组复制到另一个数组。例如：

```
Dim B(4,8),A(6,6)
…
B(2,3)=A(3,2)
```

二维数组中的元素可以复制到另一个二维数组中的某个元素，也可以复制到一个一维数组中的某个元素；反之亦然。例如：

```
Dim A(8),B(3,2)
…
A(3)=B(1,2)
B(2,1)=A(4)
```

为了复制整个数组，仍要使用 For 循环语句。

7.2.2 For Each…Next 语句

For Each…Next 语句类似于 For…Next 语句，两者都用来执行指定重复次数的一组操作。但 For Each…Next 语句专门用于数组或集合，其一般格式如下：

```
For Each 成员 In  数组
   循环体
   [Exit For]
   …
Next [成员]
```

这里的“成员”是一个变体变量，它是为循环提供的，并在 For Each…Next 结构中重复使用，它实际上代表的是数组中的每个元素。“数组”是一个数组名，没有括号和上下界。

用 For Each…Next 语句可以对数组元素进行处理，包括查询、显示或读取。它所重复执行的次数由数组中元素的个数确定，也就是说，数组中有多少个元素，就自动重复执行多少次。例如：

```
For Each x in stName
   Print x; " ";
Next x
```

执行结果为：王大明　李小芸　俗蓉　东方明　辛向荣。

将重复执行 5 次（因为数组 stName 有 5 个元素），每次输出数组的一个元素值。这里的 x 类似于 For…Next 循环中的循环控制变量，但不需要为其提供初值和终值，而是根据数组元素的个数确定执行循环体的次数。此外，x 的值处于不断的变化之中，开始执行时，x 是数组第一个元素的值，执行完一次循环体后，x 变为数组第二个元素的值……，当 x 为最后一个元素的值时，执行最后一次循环。x 是一个变体变量，它可以代表任何类型的数组元素。

可以看出，在数组操作中，For Each…Next 语句比 For…Next 语句更方便，因为它不需要指明结束循环的条件。请看下面的例子：

```
Dim arr(1 To 20)
Private Sub Form_Click()
```

```
    For i = 1 To 20
    arr(i)= Int(Rnd * 100)
    Next i
    For Each AE In arr
      If AE > 50 Then
        Print AE
        Sum = Sum+AE
      End If
      If AE > 95 Then Exit For
    Next AE
    Print Sum
End Sub
```

该例首先建立一个数组，并通过 Rnd 函数为每个数组元素赋给一个 0～99 之间的整数。然后用 For Each…Next 语句输出值大于 50 的元素，求出这些元素的和。如果遇到值大于 95 的元素，则退出循环。

注意：不能在 For Each…Next 语句中使用记录类型数组，因为 Variant 不能包含用户自定义类型。

【例 7.1】从键盘上输入 10 个整数，用冒泡排序法对这 10 个数从小到大排序。

排序是把一组数据按一定顺序排列的操作，它有很多种算法，其效率也有很大差别。冒泡排序是常用的一种排序方法。之所以称为“冒泡排序”，是因为值较小的或者说“较轻”的元素“浮到”作为继续排序的一组数的顶部。对于数值数据和字符串数据来说，其排序过程基本上相同，即从数据组的第一项开始，每一项（I）都与下一项（I+1）进行比较，如果下一项的值较小，就将这两项的位置交换，从而使值较小的数据项“升”到上面。这种操作反复进行，直到数组结束，然后再回到开头进行重复处理。当整个数组自始至终再也不出现项目交换时，全部数据项的排序即告结束。

首先在窗体上建立一个命令按钮，并把 Caption 属性设置为“排序”，然后编写如下的事件过程：

```
Private Sub Command1_Click()
   Static nmb(10)As Integer
   Print "排序前的数据:"
   For i = 1 To 10
      nmb(i)= Val(InputBox("输入要排序的数据"))
      Print nmb(i);
   Next i
   Print
   For i = 10 To 2 Step -1
     For j = 1 To i - 1
       If nmb(j)> nmb(j+1)Then
         t = nmb(j)
         nmb(j)= nmb(j+1)
         nmb(j+1)= t
       End If
     Next j
   Next i
   Print "排序后的数据:"
```

```
        For i = 1 To 10
           Print nmb(i);
        Next i
    End Sub
```

单击命令按钮后，结果如图 7.1 所示。

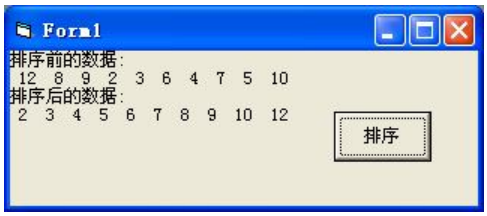


图 7.1　排序前后结果

上述过程首先定义一个一维数组，接着通过 For 循环用 InputBox 函数输入 10 个整数，然后用一个二重循环对输入的数进行排序，最后输出排序结果。在排序时，程序判断前一个数是否大于后一个数，如果大于，则交换两个数的下标，即交换两个数在数组中的位置。交换通过一个临时变量来进行。

7.3　控件数组

前面介绍了数值数组和字符串数组。在 Visual Basic 中，还可以使用控件数组，它为处理一组功能相近的控件提供了方便的途径。

控件数组是针对控件建立的，因此与普通数组的定义不一样。可以通过以下两种方法来建立控件数组。

第一种方法，步骤如下：

（1）在窗体上画出作为数组元素的各个控件。

（2）单击要包含到数组中的某个控件，将其激活。

（3）在属性窗口中选择“(名称)”属性，并输入控件的名称。

（4）对每个要加到数组中的控件重复第（2）步、第（3）步，输入与第（3）步中相同的名称。

当对第二个控件输入与第一个控件相同的名称后，Visual Basic 将显示一个对话框，如图 7.2 所示，询问是否要建立控件数组。单击“是”按钮将建立控件数组，单击“否”按钮则放弃建立操作。

图 7.2　建立控件数组

第二种方法，步骤如下：

（1）在窗体上画出一个控件，将其激活。

（2）执行“编辑”菜单中的“复制”命令（或按 Ctrl+C 组合键），将该控件放入剪贴板。

（3）执行“编辑”菜单中的“粘贴”命令（或按 Ctrl+V 组合键），将显示一个对话框，如图 7.2 所示，询问是否建立控件数组。

（4）单击对话框中的“是”按钮，窗体的左上角将出现一个控件，它就是控件数组的第二个元素。

（5）执行“编辑”菜单中的“粘贴”命令（或按 Ctrl+V 组合键），建立控件数组的其他元素。

控件数组建立后，只要改变一个控件的 Name 属性值，并把 Index 属性置为空（不是 0），就能把该控件从控件数组中删除。控件数组中的控件执行相同的事件过程，通过 Index 属性决定控件数组中的相应控件所执行的操作。

控件数组由一组相同类型的控件组成，这些控件共用一个相同的控件名字，具有同样的属性设置。数组中的每个控件都有唯一的索引号（Index Number），即下标，其所有元素的 Name 属性必须相同。

当有若干个控件执行大致相同的操作时，控件数组是很有用的，控件数组共享同样的事件过程。例如，假定一个控件数组含有 3 个命令按钮，则不管单击哪一个按钮，都会调用同一个 Click 过程。

控件数组的每个元素都有一个与之关联的下标，或称索引（Index），下标值由 Index 属性指定。由于一个控件数组中的各个元素共享 Name 属性，所以 Index 属性与控件数组中的某个元素有关。也就是说，控件数组的名字由 Name 属性指定，而数组中的每个元素则由 Index 属性指定。和普通数组一样，控件数组的下标也放在圆括号中，如 Option1(0)。

为了区分控件数组中的各个元素，Visual Basic 把下标值传送给一个过程。例如，假定在窗体上建立了两个命令按钮，将它们的 Name 属性都设置为 Comtest。设置完第一个按钮的 Name 属性后，如果再设置第二个按钮的属性，则 Visulal Basic 会弹出一个对话框，如图 7.3 所示，询问是否要建立控件数组。此时单击对话框中的“是”按钮，对话框消失，然后双击窗体上的第一个命令按钮，打开程序代码窗口，可以看到在事件过程中加入了一个下标（Index）参数，即：

```
Private Sub Comtest_Click(Index As Integer)
End Sub
```

图 7.3　建立名为 Comtest 控件数组

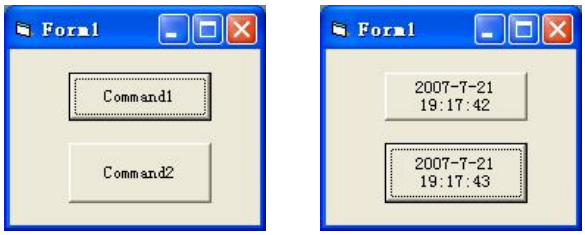
现在，不论单击哪一个命令按钮，都会调用这个事件过程，按钮的 Index 属性将传给过程，由它指明单击了哪一个按钮。

在建立控件数组时，Visual Basic 给每个元素赋一个下标值，通过属性窗口中的 Index 属性，可以知道这个下标值是多少。可以看到，第一个命令按钮的下标值为 0，第二个命令按钮的下标值为 l，依此类推。在设计阶段可以改变控件数组元素的 Index 属性，但在运行时不能改变。

控件数组元素通过数组名和括号中的下标来引用。例如：

```
Private Sub Comtest_Click(Index As Integer)
Comtest(Index).Caption = Now
End Sub
```

当单击某个命令按钮时，该按钮的 Caption（标题）属性将被设置为系统当前日期和时间，如图 7.4 所示。



单击前　　单击后

图 7.4　控件数组应用举例

【例 7.2】建立含有 3 个命令按钮的控件数组，当单击某个命令按钮时，分别执行不同的操作。按以下步骤建立：

（1）在窗体上建立一个命令按钮，并把其名称设置为 Comtest，然后用“编辑”菜单中的“复制”命令和“粘贴”命令复制两个命令按钮。

（2）把第 1、2、3 个命令按钮的 Caption 属性分别设置为“命令按钮 1”、“命令按钮 2”、“退出”。

（3）双击任意一个命令按钮，打开代码窗口，输入如下事件过程：

```
Private Sub Comtest_Click(Index As Integer)
  FontSize = 12
  If Index = 0 Then
     Print "单击第一个命令按钮"
  ElseIf Index = 1 Then
     Print "单击第二个命令按钮"
  Else
    End
End If
End Sub
```

上述过程根据 Index 的属性值决定在单击某个命令按钮时所执行的操作。所建立的控件数组包括 3 个命令按钮，其下标（Index 属性）分别为 0、1、2。第一个命令按钮的 Index 属性为 0，因此，当单击第 l 个命令按钮时，执行的是下标为 0 的那个控件数组元素的操作；而当单击第 2 个命令按钮时，执行的则是下标为 l 的那个控件数组元素的操作等。程序的运行情况如图 7.5 所示。

图 7.5　建立控件数组

控件数组可以在设计阶段通过设置相同的 Name 属性建立，可以通过改变数组中某个控件的 Name 和 Index 属性将其删除。此外，也可以在程序代码中通过 Load 方法建立控件数组，通过 Unload 方法删除数组中的某个控件。

7.4　记录

在 Visual Basic 中，可以根据需要定义自己的数据类型，它类似于 Pascal、Ada 语言中“记录类型”和 C 语言中“结构”类型的数据，因而通常称为记录类型。

7.4.1　记录类型和记录类型变量

1. 定义记录类型

记录类型通过 Type 语句来定义，其格式如下：

```
[Private|Public] Type  数据类型名
元素名[(下标)]  AS  类型名
元素名[(下标)]  AS  类型名
  …
End Type
```

记录类型的定义以 Type 开始，以 End Type 结束，格式中各部分的含义如下：

（1）Private：表示“私有”，所定义的记录类型只能在本模块中使用。当在窗体模块中定义记录类型时，必须使用 Private。

（2）Public：表示“公有”，所定义的记录类型可以在工程的任何地方使用。只有在标准模块中才能用 Public 定义记录类型（可以省略）。

（3）数据类型名：要定义的数据类型的名字，其命名规则与变量的命名规则相同。

（4）元素名：是记录类型中的一个成员，如果含有“(下标)”，则该成员是一个数组。

（5）类型名：可以是任何基本数据类型，也可以是记录类型。

记录类型的所有成员组成“成员表列”，也称为“域表”。

例如：

```
Private Type StudInfo
  intNo As Integer
  strName As String * 12
  strSex As String * 2
  sngMark(1 To 4)As Single
```

```
    sngTotal As Single
    sngAver As Single
End Type
```

这里的 StudInfo 是一个记录类型，用来定义与学生考试有关的信息，它由 6 个元素组成。其中 intNo（学号）是整型，strName（姓名）和 strSex（性别）是定长字符串，分别由 12 和 2 个字符组成，sngMark 是一个单精度型数组，用来存放 4 门课的考试成绩，sngTotal（总分）和 sngAver（平均分）是单精度型。

在使用 Type 语句时，应注意以下两点：

（1）记录类型中的元素可以是变长字符串，也可以是定长字符串。定长字符串的长度用类型名称加上一个星号和常数指明，一般格式如下：

```
String * 常数
```

这里的“常数”是字符个数，它指定定长字符串的长度，例如：

```
StrName AS String * l2
```

（2）在一般情况下，记录类型在标准模块中定义，其变量可以出现在工程的任何地方。当在标准模块中定义时，关键字 Type 前可以有 Public（默认）或 Private；而如果在窗体模块中定义时，则必须在前面加上关键字 Private。

2. 定义记录类型变量

记录类型变量的定义与基本数据类型变量的定义没有什么区别，但在引用时有所不同。例如，前面定义了一个名为 StudInfo 的记录类型，则可用下面的语句定义该类型的变量：

```
Dim Wang As StudInfo
```

以后就可以用“变量.元素”的格式引用记录中的各个成员。例如：

```
Wang.intNo
Wang.strName
Wang.strSex
Wang.sngMark
…
```

这种格式与“对象.属性”格式类似，要注意区分。

说明：

（1）记录类型与记录变量是不同的概念，定义一个记录类型并不意味着系统要分配一块内存单元来存放各记录成员，只是指定了这个类型的组织结构，即向编译程序“声明”由程序员自己所定义的记录有哪些成员、其类型是什么、长度是多少，反映了数据的抽象属性。只有用它定义了某个具体变量时，才分配存储空间。

（2）在同一个程序或同一个模块中，记录成员和记录变量可以同名，它们分别代表不同数据对象。例如：

```
Private Type StudInfo
  intNo As Integer
  …
End Type
Dim intNo As StudInfo
```

是允许的，虽然成员名 intNo 与变量名 intNo 相同，但它们的含义和引用方法不同。引用一个记录变量中的成员的值，需要指明记录变量名和成员名，如 intNo.intNo；而引用一个普通的

变量名，直接写出变量名（intNo）即可。编译时，它们被分配在不同的内存单元中。

7.4.2　记录变量的初始化及其引用

l. 记录变量的初始化

与普通变量一样，记录变量在使用前也必须具有确定的值。对于记录变量来说，只能用赋值语句或输入对话框对记录各个成员分别赋值。例如，定义了记录类型 StudInfo 的变量：

```
Dim Wang As StudInfo
```

之后就可以为记录中的各个成员赋值：

```
 Wang.intNo = 1001
Wang.strName = "张小红"
Wang.strSex = "女"
Wang.sngMark(1)= 83.5
Wang.sngMark(2)= 76
Wang.sngMark(3)= 74.5
Wang.sngMark(4)= 90
```

2. 记录变量的引用及操作

在定义了记录变量之后，就可以引用这个变量进行赋值、运算、输入和输出等操作。记录由不同类型的成员组成，而通常参加运算的是记录变量中的各个成员，引用时要在记录变量后面写上参加运算的成员名，一般格式如下：

```
记录变量.成员名
```

其中圆点符号“.”称为成员运算符，它的运算级别最高。例如，Wang.intNo 表示记录变量 Wang 中的 intNo 成员，对它的赋值可写成：

```
Wang.intNo = 1001
```

在 intNo 两侧有两个运算符，由于成员运算符“.”优先级高于赋值运算符“=”，因此，系统对上式的操作顺序是：先找到记录变量 Wang，然后从 Wang 所占的存储空间中找到成员 intNo，最后把 l001 放到为 Wang 分配给 intNo 的存储空间中。

【例 7.3】编写一个程序，记录和统计学生王小明的学习成绩。记录项包括学号（num）、姓名（name）、性别（sex）、年龄（age）、地址（address）和学习成绩（mark）。设王小明所学的 5 门课的成绩分别为 95 分、90 分、86 分、78 分、90 分。

程序如下：

```
Option Base 1
  Private Type student
  num As Integer
  name As String
  sex As String
  age As Integer
  address As String
End Type
Private Sub Form_Click()
  Dim lessons
  lessons = Array(95, 90, 86, 78, 90)
```

```
    Dim avg As Single
    Dim i As Integer
    Dim sum As Single
    Dim person As student
    sum = 0
    person.num = 201
    person.name = "王小明"
    person.sex = "男"
    person.age = 20
    person.address = "北京海淀区"
    For i = 1 To 5
       sum = sum+lessons(i)
    Next i
    avg = sum/5
    Print
    Print person.name
    Print "  学号" & "    " & "家住" & Space(10)& "五门课总分" &_
          "   " & "平均" & "分"
    Print " " & Str(person.num)& "     " & person.address & Space(5)&_
          Str(sum)& Space(8)& Str(avg)
End Sub
```

程序运行后，单击窗体，输出结果如图 7.6 所示。

图 7.6　输出结果

在这个程序中，定义了一个名为 student 的记录类型，记录学生的有关数据；为记录学生的学习成绩，又定义了一个名为 lessons 的数组。程序计算 lessons 数组元素的和及平均值，最后输出学生的有关情况及其 5 门功课的总分和平均分。

7.5　记录数组

一个记录变量中可以存放一组数据（如一个学生的学号、姓名、成绩等数据）。如果有 10 个学生的数据需要参加运算，显然应该使用数组，这就是记录数组。与数值型数组不同，记录数组的每个元素都是一个记录类型的数据，它们都分别包括各个成员（分量）项。

假定有一个学术团体，该团体有 100 个会员，见表 7.1。其中每个会员的有关情况可以定义为一个记录类型，把具有相同记录类型的变量组成一个数组，数组的每一个元素都是记录变量，这种数据类型称为记录数组。

表 7.1　100 个会员的有关信息

数组	编号（Num）	姓名（Name）	职称(Title)	地址（Addr）	邮编（Zip）	电话（Tel）
list(l)	1001	王大明	教授	北京	100084	62781712
list(2)	1002	张红	讲师	上海	200237	85372468
list(3)	1003	李洪华	副教授	天津	300110	27381578
…	…	…	…	…	…	…
list(100)						

记录数组与普通数组的定义基本相同，一般格式为：

```
Dim 数组名([下界] To 上界)As 记录名
```

例如，定义了下列常量和记录类型：

```
Const MAX_MEM = l00
Private Type mail          '定义会员通信录记录类型
  Num As Integer
  Name As String
  Title As String
  Addr As String
  Zip As Integer
  Tel As String
End Type
```

则可以用下面的语句定义一个记录数组：

```
Dim list(MAX_MEM)As mail
```

用上面的语句定义的记录数组名为 list，它的上界为 100，即存放 100 个数组元素，每个数组元素都是一个记录变量。

一个记录数组元素相当于一个记录变量，因此，前面介绍的关于记录变量的引用规则，同样适用于记录数组元素。而数组元素之间的关系和引用规则与以前介绍过的数值数组的规定相同。下面简单归纳几点，然后举例说明记录数组的引用方法和用途。

（1）引用某一记录数组元素的成员，用以下形式：

```
记录数组名(下标). 成员名
```

例如：

```
list(3).Num
```

引用的是数组 list 第三个元素的 Num（编号）成员。

（2）可以将一个记录数组元素赋给该记录数组中的另一个元素，或赋给同一类型的记录变量，例如：

```
list(2)= list(3)
```

是合法的，这两个数组元素都有同一记录类型，因此它符合记录的整体赋值规则。

（3）不能把记录数组元素作为一个整体直接输入输出。例如：

```
Print list(O)
```

是错误的。只能以单个成员为变量输入输出。例如：

```
Print list(O).Num
```

【例 7.4】编写程序，实现会员通信录的数据登录和显示输出操作。

为了简单起见，只输入 3 个会员的有关信息，然后输出。

程序如下：

```
Const MAX_MEM = 3
Private Type mail          '定义会员通信录记录类型
  Num As Integer
  Name As String
  Title As String
  Addr As String
  Zip As Integer
  Tel As String
End Type
Dim list(MAX_MEM)As mail
Private Sub Form_Click()
  Dim i As Integer
  Dim spa As String
  '从键盘上依次录入每个会员的各数据项的数据
  For i = 1 To MAX_MEM
     list(i).Num = InputBox("请输入编号 :")
     list(i).Name = InputBox("请输入姓名 :")
     list(i).Title = InputBox("请输入职称 :")
     list(i).Addr = InputBox("请输入地址 :")
     list(i).Zip = InputBox("请输入邮编 :")
     list(i).Tel = InputBox("请输入电话号码 :")
  Next i
  '输出数据
   Print
   Print "-----------------------------------------";
   Print "------------------"
   Print "   编号    姓名       职称     地址";
   Print "    邮政编码      电话号码"
   '依次显示已登录的数组元素的各成员值
   spa = "    "
   For i = 1 To MAX_MEM
     Print "-----------------------------------------";
     Print "------------------"
     Print "   " & list(i).Num & spa & list(i).Name;
     Print spa & list(i).Title & spa & list(i).Addr;
     Print "    " & list(i).Zip & spa & "    " & list(i).Tel
     Print
   Next
   Print "-----------------------------------------";
   Print "------------------"
End Sub
```

该程序有两个功能：一个是向记录数组中输入数据，另一个是列表显示数组中已有的全部数据。程序运行后，单击窗体，将显示输入对话框，根据提示在该对话框中输入 3 个会员的有关信息（表 7.1 中的前 3 项）。输入完成后，将在“输出”窗口中显示 3 个会员的信息，如图 7.7 所示。

图 7.7　记录数组应用举例

理论习题

一、选择题

1．下面关于控件数组与一般控件的叙述中，正确的是（　）。

A）控件数组一定由一个以上同类型的控件组成，而一般控件只有一个控件

B）控件数组的索引属性 Index 为 0，而一般控件的 Index 值为空

C）控件数组的 Index 为 1，而一般控件的 Index 为 0

D）数组的建立通过 Dim 语句声明，而一般控件数组不需要声明

2．用下面语句定义的数组元素数是（　）。

```
Dim A(-3 to 5)as Integer
```

A）6　　B）7　　C）8　　D）9

3．下面语句 Dim Counters(1 to 15)As Integer 定义了有（　）元素的数组。

A）14 个双精度整型数　　B）15 个双精度整型数

C）14 个整型数　　D）15 个整型数

4．下面语句 Static MatrixA(9,9)As Double 定义了有（　）元素的数组。

A）9 个　　B）18 个　　C）81 个　　D）100 个

5．语句 Options Base 1: Dim x(3,4)定义的数组所包含的数组元素的个数是（　）。

A）7　　B）12　　C）1　　D）20

6．设有数组声明：Dim x(-2 to 4,3 to 6),则下面引用数组元素正确的是（　）。

A）x(-2,3)　　B）x(5)　　C）x[-2,4]　　D）x(-1,7)

7．下列程序

```
Option Base 1
Dim x
Private Sub Form_Click()
   x = Array(1, 3, 5, 7, 9, 10, 8, 6, 4, 2)
   x(1)= 100
   For I = 10 To 1 Step -1
      Print x(I);
   Next I
End Sub
```

运行时，则输出为（ ）。

A）1 3 5 7 9 1 0 9 8 6 4 2　　B）1 3 5 7 9 10 8 6 4 100

C）2 4 6 8 10 9 7 5 3 1　　D）2 4 6 8 10 7 5 3 100

8．定义 10 个单精度实型一维数组，正确的语句是（ ）。

A）Dim a(9)as Single　　B）Option Base 1:Dim a(9)

C）Dim a#(9)　　D）Dim a(10)As Integer

9．下列程序

```
Option Base 1
Private Sub Form_Click()
   Dim x(10)
   For I= 1 to 10
      x(I)=10-I+1
   Next I
   For I=10 to 1 step-2
      Print x(I);
   Next I
End Sub
```

运行时输出的结果是（ ）。

A）9 7 5 3 1　　B）1 3 5 7 9

C）1 2 3 4 5 6 7 8 9 10　　D）10 9 8 7 6 5 4 3 2 1

10．下列程序

```
Option Base 1
Const a=10
Private Sub Form_Click()
   Dim x(a)As Integer
   k= -1
   For I=1 to a
      x(I)=I*K
      K=-K
   Next I
   For I=1 to 10
      Print x(i);
   Next I
End Sub
```

运行时输出的结果是（ ）。

A）1 3 5 7 9　　B）-1-3-5-7-9

C）-1 2 -3 4 -5 6 -7 8 -9 10　　D）1-2 3-4 5-6 7-8 9-10

11．下列程序

```
Private Sub Form_Click()
   Dim a
   a = Array("天天向上", "清化大学", "天上人间", "程序设计")
   For I=LBound(a,1)To UBound(a,1)
      If Left(a(I),1)="天" Then Print a(I);
```

```
    Next I
End Sub
```

运行时输出的结果是（　）。

A）天天向上　　B）天天向上天上人间

C）出错信息　　D）天天向上清华大学天上人间程序设计

12．下列程序

```
Option Base 1
Prvate Sub Form_Click()
   Dim x(10)As Integer, y(5)As Integer
   For I=1 to 10
      X(I)=10-I+1
   Next I
   For I=1 to 5
      Y(I)=x(2*I-1)+x(2*I)
   Next I
   For I=1 to 5
      Print Y(I);
   Next I
End Sub
```

运行时输出的结果是（　）。

A）3 7 11 45 19　　B）1 3 5 7

C）19 15 11 7 3　　D）不确定的值

13．下列程序

```
Option Base 1
Private Sub Form_Click()
  Const a=6
  Dim x(a)As Integer
   For I=1 to a
      x(i)=I^2
   Next I
   Print x(i)
End Sub
```

运行时输出的结果是（　）。

A）36　　B）25　　C）1　　D）出错信息

14．设执行以下程序段时依次输入 2、4、6，执行结果为（　）。

```
Private Sub Command1_Click()
   Dim a(4)As Integer, b(4)As Integer
   For k=0 To 2
      a(k+1)= Val(InputBox("Enter data:"))
      b(3-k)= a(k+1)
   Next k
   Print b(k)
End Sub
```

A）2　　B）4　　C）6　　D）0

二、填空题

1．下面程序实现矩阵的转置（即行列互换）。阅读程序并填空。

```
Option Base 1
Private Sub Form_Click()
   m=InputBox("输入行数:")
   n=InputBox("输入列数:")
   _________ a(m,n)As Integer, b(n, m)As Integer
   For i=1 To m
      For j=1 To n
         a(i,j)=Int(Rnd*90)+10
         Print a(i,j);
      Next j
      Print
   Next i
   For i=1 To m
      For j=1 To n
         b(j,i)= _________
         Print b(j,i)
      Next j
      Print
   Next i
End Sub
```

2．下面程序的功能是找出给定的 10 个数中最大的一个数，最后输出这个数以及它在原来 10 个数中的位置。请在下面画线处填入适当的内容，将程序补充完整。

```
Option Base1
Private SubForm_Click()
  Dim x
  x=Array(23,-5,17,38,-31,46,11,8,5,-4)
  Max=1
  k=1
10  k=k+1
  If x(k)>x(Max)Then
  _________
  End If
  If k<10 Then GoTo 10
  y = _________
  Print Max, y
End Sub
```

3．下面程序的功能是分别计算给定的 10 个数中正数之和与负数之和，最后输出这两个和数的绝对值之商。请在下面画线处填入适当内容，将程序补充完整。

```
Option Base 1
Private Sub Form_Click()
  Dim x
  x=Array(23,-5,17,38,-31,46,11,8,5,-4)
```

```
  s1 = 0
  s2 = 0
  For k=1 To 10
    If (x(k)>0)Then
      s1= ________
    Else
      s2= ________
    End If
  Next k
  y=s1/Abs(s2)
  Print y
End Sub
```

4．下列程序的运行结果为：_____。

```
Private Sub Command1_Click()
   Dim a(-1 To 6)
   For i=LBound(a,1)To UBound(a,1)
     a(i)=i
   Next i
   Print a(LBound(a,1)); a(UBound(a,1))
End Sub
```