第 6 章　数组

前面介绍过的数据类型均为基本类型，一个基本类型的变量可以用来存取和处理一个数值型数据或者字符串，对于成批的数据采用基本类型就需要用到数组和自定义类型。VB 语言提供的数组是一组相同类型变量的集合，自定义类型是由多个简单类型聚合而成，用来描述复杂数据。本章主要讲解数组的相关知识，包括一维数组、二维数组、动态数组和控件数组，介绍自定义数据类型和字符串的处理方法，以及列表框控件和组合框控件。

- 一维数组
- 二维数组
- 动态数组
- 自定义类型
- 列表框控件和组合框控件

6.1　一维数组

在应用中可能会碰到这样的问题：某班有 50 位学生，统计该班 VB 语言考试的平均成绩。如果用以前的方法来解决这个问题，先定义 50 个整型变量，再从键盘输入每一位学生的成绩，分别存放在这些变量里，然后利用循环结构计算成绩总和，再除以人数得到平均成绩。

显然，分别定义 50 个变量的工作十分繁琐，让人无法忍受，而且在语法上这些变量没有相关性，使用起来很不方便。

假设有变量 a1、a2、a3、…、a10，能否利用它们的共性和相关性重新构造一种新的数据类型？这个问题涉及了相同类型的相关数据的处理，在程序中用数组可以解决这个问题。

数组是具有相同类型的相关数据的集合，利用数组可以较为方便地解决大量数据处理的问题。数组按结构来划分，可以分为一维数组、二维数组和多维数组，其中一维数组是基础。

6.1.1　一维数组的定义

一维数组的定义方式如下：

Dim 数组名([下界 To]上界) As 类型

例如：

```
Dim a(1 To 10) As Integer
```

表示定义了一个有 10 个元素的整型数组 a，一个元素相当于一个普通的整型变量，每个元素可以存放一个整型数据。数组的元素在内存中按顺序存放，数组所占据的字节数是各元素所占字节数之和，显然数组 a 在内存占 10×2=20（字节）。

说明：

（1）数组名用合法的标识符命名，与变量的命名方法相同。

（2）数组中所有元素的数据类型都相同。

（3）下界和上界均为整型常量表达式，它们规定了元素下标的取值范围。下界最小可以是-32768，上界最大可以是 32767。要求下界≤上界，一维数组的长度即元素的个数等于上界-下界+1。

（4）对于没有赋初值的数组元素，数值型默认为 0，字符型默认为空字符串，逻辑型默认为 False。

（5）下界默认为 0，也可以使用 Option Base 语句设置数组下界的默认值，例如：

```
Option Base 1
Dim a(10) As Integer
```

定义了一个有 10 个元素的整型数组 a，它的上界是 10，下界则默认为 1。Option Base 语句的参数只能是 0 或 1，而且该语句在一个模块中只能出现一次。

6.1.2 数组元素的引用

引用元素必须要在定义数组之后，元素引用的形式如下：

数组名(下标)

例如：

```
a(3)=a(1)+a(2)
```

说明：在引用数组的元素时，应注意下标值不要超过数组的范围。假如某个数组的下界为 0，上界为 10，则其下标值的范围应该是 0～10。超过数组范围的现象称为下标越界，系统会予以报错。

下标从下界开始，到上界结束，它实际上是数组元素的序号，表示该元素在数组中的相对位置。例如数组 a 在内存中的存储结构如图 6-1 所示，图中 1000、1002 等是内存地址。

a(1)	1000
a(2)	1002
a(3)	1004
a(4)	1006
a(5)	

图 6-1 一维数组的存储结构

6.1.3 数组的应用

数组是相同类型相关数据的集合，由于引用数组元素中可以使用变化的下标，在程序中

可以结合循环结构来引用和处理数组中的元素。

数组元素的赋值可以采用以下方式：

```
a(1) = 100
a(2) = a(1) + 100
```

也可以在程序中用 InputBox 函数来为数组元素赋值，例如：

```
a(1) = Val(InputBox("请输入一个数:","输入")
```

对于 Variant 类型数组变量，可以用 Array 函数直接赋值，例如：

```
Dim a As Variant
a = Array(90,80,95,100,50)
```

【例 6.1】分别输入 5 位学生的 VB 语言成绩，计算并输出平均成绩。

分析：定义一个长度为 5 的整型数组，用来存放学生的成绩。采用 For-Next 语句进行处理，下标初始为 1，每次循环不断加 1，到 30 为止。在循环体中累加每一位学生的成绩，最后除以人数得到平均成绩。

```
Const N As Integer = 5
Private Sub Command1_Click()
    Dim a(1 To N) As Integer
    Dim i As Integer, total As Integer, average As Single
    total = 0
    For i = 1 To N          '输入学生成绩
        a(i) = Val(InputBox("请输入第" & i & "位学生的成绩"))
        total = total + a(i)
    Next i
    average = total / N '计算平均成绩
    Print "平均成绩是"; average
End Sub
```

运行程序，结果如图 6-2 所示。

图 6-2　例 6.1 的运行结果

说明：在程序的第一个 For-Next 语句中，反复调用 InputBox 函数输入学生成绩，并分别存放在数组 a 的各个元素中。为增加程序的通用性，在事件过程的外部定义了一个符号常量 N，表示数组的上界。如果需要修改数组的上界，则只需修改符号常量的初值即可。

【例 6.2】通过 Array 函数输入 10 位学生的成绩，输出最高分和最低分。

程序如下：

```
Option Base 1
Dim a As Variant
Private Sub Command1_Click()
    Dim max As Integer, min As Integer
    Dim i As Integer
```

```
        max = a(1)                    '设定初值
        min = a(1)
        For i = 2 To 10
            If max < a(i) Then        '找最高分
                max = a(i)
            End If
            If min > a(i) Then        '找最低分
                min = a(i)
            End If
        Next i
        Picture1.Print "最高分：" + Str(max) + _
        Chr(13) + "最低分：" + Str(min)
    End Sub

    Private Sub Form_Load()
        a = Array(78, 95, 85, 77, 89, 92, 73, 85, 75, 96)
    End Sub
```

程序运行结果如图 6-3 所示。

图 6-3　例 6.2 的运行结果

6.2　二维数组

一维数组主要用于存放一组相关数据，当遇到多组相关数据时，需要用到二维数组。

6.2.1　二维数组的定义

二维数组的定义方式如下：

Dim 数组名([下界 To]上界,[下界 To]上界) As 类型

例如：

```
Dim a(1 To 3,1 To 4) As Integer
```

表示定义了一个 3 行 4 列的二维整型数组 a，它的逻辑结构如表 6-1 所示。数组 a 有 3×4，即 12 个元素，每个元素可以存放一个整型数据。

表 6-1　数组 a 的逻辑结构

a(1,1)	a(1,2)	a(1,3)	a(1,4)
a(2,1)	a(2,2)	a(2,3)	a(2,4)
a(3,1)	a(3,2)	a(3,3)	a(3,4)

说明：

（1）通常把二维数组的第 1 个下标形象地称为行下标，第 2 个下标称为列下标。

（2）二维数组的元素个数为行的长度×列的长度，行或者列的长度为各自的上界-下界+1。

（3）类似地还可以定义多维数组。例如：

```
Dim a(1 To 3,1 To 4,1 To 5) As Integer    '共有 3×4×5，即 60 个元素的三维数组
```

表 6-1 所示的数组 a，实际上整合了 3 个一维数组：

a(1)、a(2)、a(3)、a(4)

b(1)、b(2)、b(3)、b(4)

c(1)、c(2)、c(3)、c(4)

整合过程如表 6-2 所示。

表 6-2　数组的整合

a(1) →a(1,1)	a(2) →a(1,2)	a(3) →a(1,3)	a(4) →a(1,4)
b(1) →a(2,1)	b(2) →a(2,2)	b(3) →a(2,3)	b(4) →a(2,4)
c(1) →a(3,1)	c(2) →a(3,2)	c(3) →a(3,3)	c(4) →a(3,4)

6.2.2　二维数组的应用

由于多了一维数据，二位数组可以处理更复杂的数据集合。

【例 6.3】设有一个 3×3 矩阵，其中元素是由计算机随机产生的小于 100 的整数，求：

（1）对角线上元素之和。

（2）方阵中最大的元素。

分析：矩阵可以用一个二维数组表示，利用单层循环可以求出对角线上的元素之和，利用双层循环可以找到最大数。

```
Dim a(3, 3) As Integer
Private Sub Command1_Click()
    Dim s As Integer, Max As Integer
    s = 0
    For i = 1 To 3
        s = s + a(i, i)
    Next
    Picture1.Print "对角线和是:"; s
     Max = a(1, 1)
     For i = 1 To 3
       For j = 1 To 3
           If a(i, j) > Max Then
               Max = a(i, j)
           End If
       Next j
    Next i
    Picture1.Print "最大数是:"; Max
```

```
End Sub

Private Sub Form_Load()
    Show
    Randomize
    For i = 1 To 3
        For j = 1 To 3
            a(i, j) = Int(Rnd * 99) + 1
            Picture1.Print a(i, j); Chr(9);
        Next j
        Picture1.Print
    Next i
End Sub
```

程序运行结果如图 6-4 所示。

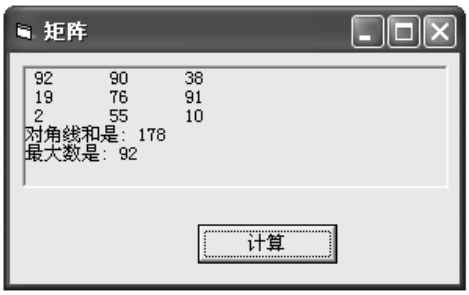

图 6-4　例 6.3 的运行结果

说明：通常把二维数组的行下标放在外层循环，而把列下标放在内层循环。

思考：三维数组的数据如何输入、输出及处理？

6.3　动态数组

前面介绍的数组都是指定长度和类型的数组，也可称为静态数组。如果无法预知元素的个数和数组的维数，VB 语言允许定义动态数组，以增强程序的灵活性，提高内存使用的效率。

动态数组在程序运行过程中才被分配存储空间，它的定义方式如下：

Dim 数组名() As 类型

例如：

```
Dim a() As Integer
```

表示定义了一个动态整型数组 a，数组的维数以及元素下标的下界和上界未知。可以用数组名赋值的方式把一个静态数组中全部元素的值依次赋给一个动态数组中的全部元素。例如：

```
Dim a(1 To 3) As Integer, b() As Integer, i%
For i = 1 To 3          '对静态数组 a 的所有元素赋值
    a(i) = i
Next i
b = a                   '数组名赋值
For Each x In b         '输出动态数组 b 中所有元素的值
```

```
    Print x
Next x
```

数组名赋值的方式自动确定了动态数组 b 的维数以及元素下标的下界和上界，它们均与静态数组 a 相同。也可以调用 LBound 和 UBound 函数，分别获得数组的下界和上界。这两个函数的格式如下：

LBound(a[,n])
UBound(a[,n])

说明：

（1）参数 a 是数组名。参数 n 表示数组 a 的第 n 维，如果省略，则默认是 1。

（2）LBound 函数返回数组 a 第 n 维的下界，UBound 函数返回数组 a 第 n 维的上界。

变体型数组的各个元素能够存放不同类型的数据，如果是动态变体型数组，则可以通过 Array 函数进行初始化，并自动确定动态数组中元素的个数。例如：

```
Dim b(), i%
b = Array(1, 2, 3)
For i = 0 To 2
    Print b(i)
Next i
```

定义数组 b 时，既未指定维数以及元素下标的下界和上界，也未指定数据类型，因此它是**动态变体型数组**。在 Array 函数中有 3 个参数，作为初值依次赋给了数组 b 的各个元素。由此确定了动态数组 b 中元素个数为 3，下界默认是 0，上界则为 2。

定义了一个动态数组之后，一旦需要即可在程序中使用 ReDim 语句，确定动态数组的维数以及元素下标的下界和上界。其一般形式如下：

ReDim [Preserve]数组名([下界 To]上界[,下界 To 上界,…]) [As 类型]

说明：

（1）可以多次使用 ReDim 语句对某个动态数组进行设置。

（2）数组的维数以及元素下标的下界和上界都能够改变，甚至下界和上界可以是有了确定值的变量，但是数组的类型不能改变。

（3）每次执行 ReDim 语句之后，数组中所有元素的值将会丢失。如果想保留数组元素的值，则可以使用关键字 Preserve。例如：

```
Dim a() As Integer
  …
ReDim a(2,3)                    '数组设置为 3 行 4 列
  …
ReDim Preserve a(2,4)           '数组设置为 3 行 5 列，并保留数组元素的值
```

在 ReDim 语句中使用关键字 Preserve 时，只能改变动态数组最后一维的上界。

【例 6.4】计算并输出 Fibonacci 数列的前 n 项。

Fibonacci 数列的特点是，前两个数为 1，1。从第 3 个数开始，每个数都是前面两个数的和。即：

F_1=1,F_2=1（n=1 或 2）

$F_n=F_{n-1}+F_{n-2}$（n>=3）

分析：由于 Fibonacci 数之间存在明显的位置关系，所以可以用数组来处理。定义一个动

态长整型数组 a，用来存放 Fibonacci 数列。从文本框接收用户输入的 n 值之后，采用 ReDim 语句将数组 a 的长度置为 n。

```
Private Sub Command1_Click()
    Dim F() As Long, n As Integer, i As Integer, j%
    Picture1.Cls                        '清除上次的输出
    n = Int(Val(InputBox("请输入 n", "提示")))
    ReDim F(1 To n)                     '设置动态数组的长度
    F(1) = 1
    F(2) = 1
    For i = 3 To n
        F(i) = F(i - 1) + F(i - 2)      '每一项是前两项之和
    Next i
    j = 0
    For i = 1 To n
        Picture1.Print Tab(j * 8); F(i);
        j = j + 1
        If i Mod 5 = 0 Then
            Picture1.Print
            j = 0
        End If
    Next i
End Sub
```

运行程序，结果如图 6-5 所示。

图 6-5　例 6.4 的运行结果

在程序中有时需要对数组重新进行初始化，则可以使用 Erase 语句达到目的，其一般形式如下：

Erase 数组名

说明：执行 Erase 语句之后，对于静态数组，系统会清除数组中的原有数据，并自动进行初始化；对于动态数组，系统会删除数组的结构，并释放数组所占的内存空间。如果以后想再次使用该动态数组，就必须采用 ReDim 语句重新对其进行设置。例如：

```
Private Sub Command1_Click()
  Dim a(1 To 5) As Integer, i%
  Dim b() As Integer
  For i = 1 To 5
       a(i) = i
  Next i
```

```
b = a
Picture1.Print "数组 a"

For i = 1 To 5                    '输出数组 a 所有元素的值
    Picture1.Print a(i);
Next i
Picture1.Print

Picture1.Print "数组 b"
For i = 1 To 5                    '输出数组 b 所有元素的值
    Picture1.Print b(i);
Next i
Picture1.Print

Erase a '对数组 a 重新初始化

Picture1.Print "Erase 后的数组 a"
For i = 1 To 5                    '再次输出数组 a 所有元素的值
    Picture1.Print a(i);
Next i
Picture1.Print

Erase b

ReDim b(1 To 5) As Integer
Picture1.Print "重新 Dim 的数组 b"
For i = 1 To 5                    '输出数组 b 所有元素的值
    Picture1.Print b(i);
Next i
Print
```

该程序段的运行结果如图 6-6 所示。从图中可以看到，执行 Erase 语句之后，数组 a 所有元素的值都重新初始化为 0。

图 6-6　Erase 语句的效果

6.4　控件数组

一组相同类型的相关数据可以用数组来描述和管理，那么一组功能相似的同类控件是否也能够用数组进行组织？回答是肯定的，这样的数组称为控件数组。控件数组由一组同属于

一类的控件组成，它们共用一个对象名，依靠索引（Index）属性彼此区分。

如何创建控件数组？主要有以下几种方法：

（1）复制现有的控件，然后粘贴在窗体中。第一次进行粘贴操作时，系统会提示是否创建控件数组，选择“是”即可。此时大多数的可视属性，如颜色、高度和宽度等，将会从源控件即数组中的第一个控件，复制到目标控件即新控件中。例如复制并粘贴一个命令按钮时，出现如图 6-7 所示提示。

图 6-7　复制现有控件的提示

（2）为现有的同类控件取同一个对象名，一般与第一个控件的名字一致，例如 Text1、Command1 等。这时系统也会提示是否创建控件数组，选择“是”即可。用这种方法创建的控件数组，其控件元素的属性值只是名字（Name）相同，其他属性依然保留最初创建这些控件时的设置。

如何访问控件数组中的元素？利用控件的 Index 属性。与数组的下标相似，Index 表示控件在控件数组中的相对位置，默认从 0 开始，依次加 1。控件元素的访问方法与普通数组的元素基本相同，例如现有控件数组 Text1，要将其中 Index 属性值为 1 的文本框控件设置 Text 属性值为“VB6.0”，可以写为：

```
Text1(1).Text="VB6.0"
```

在程序中使用控件数组，不仅可以借助循环结构统一处理数组中的控件，而且可以共享同一个事件处理过程。例如设计一个计算器，在窗体中安排 4 个命令按钮，分别完成加减乘除四则运算。考虑到这些命令按钮实现的功能相似，可以创建一个有 4 个元素的命令按钮控件数组，其中每一个命令按钮对应一个控件数组的元素。然后为这个控件数组定义一个单击事件过程，只要用户任意单击 4 个命令按钮中的一个，就会调用这个事件过程。此外还可以在程序中调用 Load 方法，动态创建控件数组中的新元素，达到在程序运行时创建新控件的目的。

【例 6.5】用控件数组查找数组中的最大数和最小数。

```
Option Base 1
Dim a As Variant
Private Sub Command1_Click(Index As Integer)
    Dim m
    m = a(1)
    Select Case Index
        Case 0
            For i = 2 To 10
                If a(i) > m Then
                    m = a(i)
                End If
            Next
            Picture1.Print "最大值是"; m
```

```
            Case 1
                For i = 2 To 10
                    If a(i) < m Then
                        m = a(i)
                    End If
                Next
                Picture1.Print "最小值是"; m
        End Select
    End Sub

    Private Sub Form_Load()
        a = Array(78, 95, 85, 77, 89, 92, 73, 85, 75, 96)
    End Sub
```

程序运行结果如图 6-8 所示。

图 6-8　例 6.5 运行结果

思考：按照这个思路设计一个计算器。

6.5　自定义类型

数组只能处理相同类型的数据，对于不同类型数据组成的实体，例如一个学生的数据实体包括学号、姓名、性别、年龄和成绩等数据项，可以用自定义数据类型来处理，这种自定义的类型又称为记录类型，它由一些基本类型的成员组成。定义记录类型的关键字是 Type，其一般形式如下：

```
Type <记录类型名>
  成员表列
End Type
```

说明：

（1）对成员表列中的所有成员都应进行类型声明。成员声明的形式如下：

```
成员名 As 类型
```

（2）记录类型只是构造了一个数据结构的模型，并没有定义实例，也不要求分配实际的内存空间。在程序中使用记录类型时，必须定义记录变量。

例如，学生信息可以用记录类型描述为：

```
Type Student
    no As Long          '学号
    name As String      '姓名
    sex As String       '性别
```

```
    score As Integer          '成绩
End Type
```

Student 类型有 4 个成员，分别表示学生的学号、姓名、性别和成绩，这些成员的类型可以不相同。需要指出的是，通常在程序的标准模块（.bas）中定义记录类型。在“工程”菜单中选择“添加模块”命令，即可创建标准模块。如果在窗体模块中定义记录类型，则必须用关键字 Private 进行声明。

先定义记录类型，再定义记录变量。记录变量所占内存空间的长度，是其各个成员所占空间的长度之和。定义记录变量的方法与定义普通变量基本相同，只不过数据类型是记录类型。例如：

```
Dim stud1 As Student,stud2 As Student        '定义两个 Student 类型的变量
```

访问一个记录变量的目的通常是引用它的成员，例如登记学生的姓名、统计学生的成绩等。引用记录变量成员的形式如下：

记录变量名.成员名

stud1.no 表示引用记录变量 stud1 中的 no 成员，它可以像普通变量一样使用，能够进行赋值等合法的运算。例如：

```
studl.no=20100101        '将 20100101 赋给 studl 变量的成员 no
```

自定义类型也可以定义该类型的数组，也称作记录数组。例如：

```
Dim MyClass(1 To 60) As Student
```

定义了一个数组 MyClass，它有 60 个元素，每个元素都相当于一个 Student 类型的记录变量。访问记录数组元素的成员的一般形式如下：

记录数组名(下标).成员名

例如：

```
MyClass(1).no=20100101
```

【例 6.6】演示用记录类型处理 5 位学生的成绩。

分析：在标准模块中定义 Student 记录类型，成员有姓名和成绩，成员的类型分别为字符串和整型。

```
Type Student
    No As Long
    name As String
    score As Integer
End Type
```

在窗体模块中定义一个长度为 5 的 Student 型数组，用来存放全班学生的姓名和成绩。采用 For-Next 语句进行处理，具体代码如下：

```
Private Sub Command1_Click()
  Dim MyClass(5) As Student, Max As Integer
  MyClass(1).no = 20100101
  MyClass(1).name = "张平"
  MyClass(2).no = 20100102
  MyClass(2).name = "李智"
  MyClass(3).no = 20100103
  MyClass(3).name = "孙梅"
  MyClass(4).no = 20100104
  MyClass(4).name = "倪红"
  MyClass(5).no = 20100105
```

```
    MyClass(5).name = "陈好"
    For i = 1 To 5
       With MyClass(i)
        .score = Val(InputBox("请输入" & .name & "的成绩", "提示"))
       End With
    Next
    For i = 1 To 5
       Picture1.Print MyClass(i).no, MyClass(i).name, MyClass(i).score
    Next
    Max = MyClass(1).score
     j = 1
     For i = 2 To 5
        If MyClass(i).score > MyClass(j).score Then
            j = i
        End If
     Next i
     Picture1.Print "最高分:"; MyClass(j).No,
          MyClass(j).name, MyClass(j).score
End Sub
```

运行程序，结果如图 6-9 所示。

图 6-9　例 6.6 的运行结果

说明：在程序中使用了 With 语句，以简化语句的书写。如果用 With 语句对某个记录变量作出声明，则在该语句的作用域中访问这个记录变量的成员时，可以省略记录变量名。

6.6　字符串的处理

日常生活中很多数据都是文本类型，VB 语言以字符串表示文本数据，在程序中除了可以用运算符进行字符串的连接和比较等基本操作之外，还可以调用内部函数完成字符串的查找、截取等一些高级操作。下面是对一些常用字符串处理函数的详细介绍。

1. 格式转换

字符串的格式转换是指字符串与数值的相互转换，以及字符与编码的相互转换等操作。

（1）Val 函数和 Str 函数。格式分别如下：

Val(s)

和

Str(n)

说明：Val 函数的功能是把字符串 s 转换为一个数值，例如 Val("12345")得到的值是 12345。Str 函数正好相反，该函数的功能是把数值 n 转换为一个字符串，例如 Str(12345)得到的值是"12345"。

（2）Asc 函数和 Chr 函数。格式分别如下：

Asc(s)

和

Chr(n)

说明：Asc 函数的功能是把字符串 s 中的第一个字符转换为相应的 ASCII 码，例如 Asc("ab")得到的值是 97，即字符 a 的 ASCII 码；Chr 函数正好相反，该函数的功能是把数值 n 即 ASCII

码转换为所对应的字符。例如 Chr(97)得到的值是"a"，而 Chr(Asc("a")-32)得到的值是"A"，因为小写字母的 ASCII 码比相应大写字母的 ASCII 码大 32。

（3）UCase 函数和 LCase 函数。格式分别如下：

UCase(s)

和

LCase(s)

说明：UCase 函数的功能是把字符串 s 中的小写字母转换为大写形式，例如 UCase("aBCdE")得到的值是"ABCDE"；LCase 函数正好相反，该函数的功能是把字符串 s 中的大写字母转换为小写形式，例如 LCase("AbCdE")得到的值是"abcde"。

2. 统计长度

函数 Len 用于统计字符串的长度即所包含字符的个数，其格式如下：

Len(s)

例如，Len("12345 汉字")得到的值是 7，注意每个汉字的长度计为 1。

3. 删除空格

函数 LTrim 删除字符串中前面的空格，函数 RTrim 删除字符串中后面的空格，函数 Trim 则删除字符串中前后两边的空格。它们的格式分别如下：

LTrim(s)
RTrim(s)
Trim(s)

例如，LTrim("　字符串　")得到的值是"字符串　"，RTrim("　字符串　")得到的值是"　字符串"，而 Trim("　字符串　")得到的值是"字符串"。

4. 生成字符串

字符串的生成是指按照要求产生一个字符串。

（1）String 函数。格式如下：

String(m,s 或 n)

说明：该函数的功能是产生一个由 m 个重复的字符组成的字符串。该字符是字符串 s 中的第一个字符，或者是 ASCII 码为数值 n 的字符。例如 String(5,"ABCD")和 String(5,65)得到的值均为"AAAAA"。

（2）Space 函数。格式如下：

Space(n)

说明：该函数的功能是产生一个由 n 个空格组成的字符串。例如，"Visual"+Space(1)+"Basic"+Space(2)+"6.0"得到的值是"Visual Basic　6.0"。

5. 查找和替换

字符串的查找是指在一个字符串中查找另一个指定的字符串，字符串的替换是指在一个字符串中把一个子串替换为另一个子串。

（1）InStr 函数。InStr 函数的格式如下：

InStr([n,]s1,s2)

说明：该函数的功能是在字符串 s1 中查找字符串 s2 首次出现的位置。参数 n 是字符串 s1 的起始查找位置，如果省略则默认值是 1，表示从头开始查找。如果找到，函数的返回值是字符串 s2 在字符串 s1 中第一次出现的位置；如果未找到，则函数的返回值是 0。例如：

InStr("我是好学生中的好学生","学生")返回：4

InStr(6,"我是好学生中的好学生","学生")返回：9

InStr(10,"我是好学生中的好学生","学生")返回：0

（2）Replace 函数。Replace 函数的格式如下：

Replace(s1,s2,s3[,m][,n][,…])

说明：该函数的功能是在字符串 s1 中把子串 s2 替换为子串 s3。参数 m 是字符串 s1 的起始查找位置，此时在函数的返回值中会删除位置 m 前的字符。如果省略 m 则默认值是 1，表示从头开始查找。参数 n 是进行替换操作的最大次数，如果省略 n 则默认值是-1，表示替换所有符合条件的子串。例如：

Replace("我是好学生中的好学生", "学生", "孩子")返回："我是好孩子中的好孩子"

Replace("我是好学生中的好学生", "学生", "孩子", 4) 返回："学生中的好孩子"

Replace("我是好学生中的好学生", "学生", "孩子", 1, 1) 返回："我是好孩子中的好学生"

Replace("我是好学生中的好学生", "学生", "孩子", , 1) 返回："我是好孩子中的好学生"

6. 截取子串

字符串的截取是指从字符串中连续抽取若干个字符，组成一个新的字符串即子串。

（1）Left 函数。格式如下：

Left(s,n)

说明：该函数的功能是从字符串 s 的左边取出 n 个字符，组成一个子串。例如 Left("我是好学生中的好学生",5)得到的值是"我是好学生"。

（2）Right 函数。Right 函数的格式如下：

Right(s,n)

说明：该函数的功能是从字符串 s 的右边取出 n 个字符，组成一个子串。例如 Right("我是好学生中的好学生",3)得到的值是"好学生"。

（3）Mid 函数。Mid 函数的格式如下：

Mid(s,m[,n])

说明：该函数的功能是从字符串 s 的第 m 个字符开始，取出 n 个字符，组成一个子串。如果省略 n，则表示取出从第 m 个字符开始的所有字符。例如：

Mid("我是好学生中的好学生",3,3)返回："好学生"

Mid("我是好学生中的好学生",3)返回："好学生中的好学生"

实际上：

Mid(s,1,n)的作用等价于 Left(s,n)

Mid(s,n)的作用等价于 Right(s,Len(s)-n+1)

Mid(s,Len(s)-n+1)的作用等价于 Right(s,n)

（4）Split 函数。Split 函数的格式如下：

Split(s[,d][,n][,…])

说明：该函数的功能是从字符串 s 中取出 n 个子串，子串之间的分隔符是参数 d。如果省略 d，默认分隔符是空格；如果省略 n 则默认值是-1，表示取出所有的子串。通常把函数的返回值赋给一个动态字符串数组，数组的每一个元素依次存放一个子串。例如：

```
Dim a() As String
a = Split("我是好学生中的好学生", "好")
```

```
Print a(0), a(1), a(2)
```

输出的结果是：

```
我是        学生中的        学生
```

以前介绍的数组输入方法，是采用循环结构反复调用 InputBox 函数，这种方法未免有些单调。Split 函数可以用来一次性地给一个数组赋初值，提高数据输入的效率。例如用户先在文本框中输入所有的数据，数据之间以事先约定的字符进行分隔，然后调用 Split 函数取出所有的数据，依次存入数组的各个元素。在这种情况下，可以调用 LBound 函数得到数组的下界，调用 UBound 函数得到数组的上界。

【例 6.7】演示利用 Split 函数改进的数据输入方法。

分析：在窗体中安排一个文本框，用于输入部分城市的名称。定义一个动态字符串数组 a，调用 Split 函数，从文本框中读取城市名，存放到数组 a 中。具体代码如下：

```
Private Sub Command1_Click()
  Dim a() As String, i%, j%, flag As Boolean, name$
  a = Split(Text1.Text)                          '输入要查找的城市名称
  Do
    city = InputBox("请输入要查询的城市名称")
    flag = False
    For i = LBound(a) To UBound(a)               'LBound 和 UBound 分别获得数组 a 的下界和上界
      If a(i) = city Then
        flag = True                              '找到，改变标志
        Exit For
      End If
    Next i
    If flag = True Then
      Picture1.Print "找到城市:"; city
    Else
      Picture1.Print "没有找到城市"; city
    End If
    MsgBoxValue = MsgBox("还要继续查询吗？", vbYesNo + vbquestin)
  Loop While MsgBoxValue = 6
End Sub
Private Sub Form_Load()
  Text1.Text = ""
End Sub
```

运行程序，结果如图 6-10 所示。

图 6-10　例 6.7 的运行结果

【例 6.8】统计一个字符串中数字、字母以及其他字符的个数。

分析：定义一个 3 元素的数组来分别记录数字、字母以及其他字符的个数。在循环语句中调用 Mid 函数，依次取出各个字符，用 If 语句的 ElseIf 结构判断字符的特征，相应进行计数。

```
Option Base 1
Private Sub Command1_Click()
    Dim s As String, c As String, i%
    Dim num(1 To 3) As Integer
    Dim t() As String
    t = Split("字母,数字,其他字符", ",")
    s = Text1.Text
    For i = 1 To Len(s)
        c = Mid(s, i, 1)                    '取出字符串中第 i 个字符
        If c >= "A" And c <= "Z" Or c >= "a" And c <= "z" Then
            num(1) = num(1) + 1             '字母的个数增 1
        ElseIf c >= "0" And c <= "9" Then
            num(2) = num(2) + 1             '数字的个数增 1
        Else
            num(3) = num(3) + 1             '其他字符的个数增 1
        End If
    Next i
    For i = 1 To 3
        Picture1.Print t(i - 1); "的个数为"; num(i)
    Next i
End Sub

Private Sub Form_Load()
    Text1.Text = ""
End Sub
```

运行程序，结果如图 6-11 所示。

图 6-11　例 6.8 的运行结果

说明：Mid(s, i, 1)的作用是从字符串 a 的第 i 个位置开始，取出 1 个字符，即得到第 i 个字符。随着 i 不断加 1，就可以得到字符串 a 的每一个字符。

6.7 列表框

如果可供用户选择的项目较少，一般采用一组单选按钮或复选框。但是如果存在大量的选项，采用单选按钮和复选框就显得十分繁琐，而采用列表框或者组合框则不失为一个较好的解决方案。列表框（ListBox）控件能够显示一个项目列表，用户可以从中选择一个或多个项目。如果项目列表中的项目过多而无法一次全部显示，则列表框将自动出现滚动条。在 VB 的工具箱中，列表框控件的图标如图 6-12 所示。

图 6-12 列表框图标

1. 属性

表 6-3 列出了列表框控件的常用属性。

表 6-3 列表框控件的常用属性

属性	作用
Name	设置列表框的对象名
Text	确定用户当前所选的项目，该属性不能在属性窗口中设置，只能在程序中设置或引用
List	设置列表框所显示的项目列表
ListCount	确定列表框中项目的总数，该属性只能在程序中设置或引用
ListIndex	确定当前选中的项目在项目列表中的索引值，该属性只能在程序中设置或引用
Selected	确定项目列表中某个项目是否被选中，该属性只能在程序中设置或引用
MultiSelect	确定列表框是否允许多选
Style	设置列表框的外观，默认值是 0，表示标准方式；如果是 1，则项目的左边有复选框

说明：

（1）程序第 1 个列表框控件的默认对象名是 List1，第 2 个列表框控件的默认对象名是 List2，依此类推。

（2）List 是列表框控件最重要的属性之一，其属性值是一个字符串数组，每一个元素存放项目列表其中的一个项目。List 数组的下标从 0 开始，例如输出列表框 List1 的第 2 个项目，则可以写为：

```
Print List1.List(1)
```

向列表框中添加项目有两种方法。第 1 种方法是在属性窗口中选中列表框的 List 属性，单击下拉按钮，输入一个项目后按 Ctrl+回车键，在下一行继续输入新项目。第 2 种方法是在程序中调用 AddItem 方法，在列表框中添加项目。

（3）在程序中 ListIndex 和 ListCount 往往与 List 属性配合使用。如果用户未选择任何项目，ListIndex 的值是-1；如果用户选中项目列表中的第 1 项，ListIndex 的值是 0；如果用户选中项目列表中的最后一项，则 ListIndex 的值是 ListCount-1。

（4）Selected 的属性值是一个逻辑型数组，其每一个元素与项目列表中的每一个项目一一对应。如果某个项目被用户选中，Selected 数组相应元素的值是 True；如果未被选中，则相应元素的值是 False。例如 List1.Selected(2)的值是 True，表示列表框 List1 的第 3 个项目被选中。

（5）MultiSelect 的属性值有 3 个，默认值是 0，如表 6-4 所示。

表 6-4　MultiSelect 属性值

常量	值	含义
None	0	不允许多选
Simple	1	简单多选，可以用鼠标单击或按空格键进行选择
Extended	2	扩展多选，可以借助 Shift 键或 Ctrl 键进行选择

2. 事件

列表框控件能够响应 Click 和 DblClick 等事件。在实际编程中经常针对列表框编写 DblClick 事件过程，使得双击列表框中的某个选项之后，可以对该选项进行相应的操作。例如在“文件”对话框的文件列表框中双击某个文件名，即可直接打开该文件。

3. 方法

列表框的常用方法如表 6-5 所示。

表 6-5　列表框的常用方法

方法	功能
AddItem	向列表框中添加一个项目
RemoveItem	从列表框中删除一个项目
Clear	清除列表框中所有项目

说明：

（1）AddItem 方法的调用形式如下：

对象.AddItem Item[,Index]

参数 Item 表示被添加到列表框中的字符串，即新项目。参数 Index 表示新项目在列表框中的索引值，即插入到项目列表中的位置。如果该参数被省略，则将把新项目插入到项目列表的末尾。

（2）RemoveItem 方法的调用形式如下：

对象.RemoveItem Index

参数 Index 表示被删除的项目在列表框中的索引值。例如删除列表框 List1 中的第 1 个项目，可以写为：

```
List1.RemoveItem 0
```

6.8　组合框

组合框（ComboBox）控件组合了文本框和列表框的特性，用户既可以在它的文本框部分输入文本以选择项目，也可以在它的列表框部分选择项目。当用户在列表框部分选定某个项目之后，该项目会自动出现在文本框部分。列表框将用户的选择限制在项目列表之内，而组合框则允许用户选择项目列表中所没有的项目。在 VB 的工具箱中，组合框控件的图标如图 6-13 所示。

图 6-13　组合框图标

1. 属性

组合框控件的大部分属性与列表框控件相同，此外还有一些与文本框相同的属性。表 6-6 列出了组合框控件的常用属性。

表 6-6　组合框控件的常用属性

属性	作用
Name	设置组合框的对象名
Text	确定用户当前选择的项目或者在文本框部分输入的项目
List	设置组合框所显示的项目列表
ListCount	确定组合框中项目的总数
ListIndex	确定当前选中的项目在项目列表中的索引值
Selected	确定项目列表中某个项目是否被选中
Style	设置组合框的类型

说明：

（1）程序第一个组合框控件的默认对象名是 Combo1，第二个组合框控件的默认对象名是 Combo2，依此类推。

（2）Style 的属性值有 3 个，默认值是 0，如表 6-7 所示。

表 6-7　Style 属性值

常量	值	含义
Dropdown Combo	0	下拉式组合框
Simple Combo	1	简单组合框
Dropdown List	2	下拉式列表框

下拉式组合框如图 6-14 所示。它将文本框和下拉式列表框组合在一起，用户可以直接用键盘在文本框中输入项目，也可以单击下拉按钮，打开列表框进行选择。

简单组合框如图 6-15 所示。它将文本框和列表框简单地组合在一起，列表框的项目列表直接显示在窗体上。

下拉式列表框如图 6-16 所示。它的功能与下拉式组合框相似，但是用户只能从列表框中进行选择，而不能直接在文本框中输入项目。

图 6-14　下拉式组合框

图 6-15　简单组合框

图 6-16　下拉式列表框

思考：当组合框为下拉式列表框类型时，用户能否选择项目列表中所没有的项目？

2. 事件

根据类型的不同，组合框控件能够响应的事件也有所不同。所有类型的组合框都能够响应 Click 事件，但是只有简单组合框（Style 的属性值为 1）才能响应 DblClick 事件。此外下拉式组合框和简单组合框还可以响应 Change 事件。

3. 方法

AddItem、RemoveItem 和 Clear 等方法也同样适用于组合框控件。例如在组合框 Combo1 中添加一个项目“中国”，可以写为：

```
Combo1.AddItem "中国"
```

例如清空组合框 Combo1 中的所有项目，可以写为：

```
Combo1.Clear
```

【例 6.9】演示列表框和组合框的应用。

```
Private Sub Form_Load()
    With Combo1
        .AddItem "学士"
        .AddItem "硕士"
        .AddItem "博士"
        .Text = ""
    End With

    With Combo2
        .AddItem "男"
        .AddItem "女"
        .Text = ""
    End With

    With List1
        .AddItem "体育"
        .AddItem "音乐"
        .AddItem "美术"
        .AddItem "文学"
        .AddItem "交际"
        .Text = ""
    End With
    Text1.Text = ""
End Sub
Private Sub Command1_Click()
    Dim s As String, i As Integer
    s = s + "姓名：" + Text1.Text + vbCr
    s = s + "年龄：" + Combo2.Text + vbCr
    s = s + "学位：" + Combo1.Text + vbCr
    s = s + "爱好：" + vbCr
    For i = 0 To List2.ListCount - 1              '得到所有的特长
      s = s + List2.List(i) + Space(2)
```

```
    Next i
    MsgBox (s)
End Sub
Private Sub List1_DblClick()
    Dim s As String
    s = List1.Text                          '从 List1 得到选中的项目
    List2.AddItem s                         '将该项目添加到 List2 中
    List1.RemoveItem List1.ListIndex        '删除已经选中的项目，防止重复选
End Sub
Private Sub List2_DblClick()
    List2.RemoveItem List2.ListIndex        '删除选中的项目
End Sub
```

运行程序，结果如图 6-17 所示。

图 6-17　例 6.9 的运行结果

说明：程序运行时，用户可以在组合框的列表框部分选择学生的学位，也可以在文本框部分输入项目列表中未列出的学位。如果用户在“特长”下面的列表框中双击一个项目，则上面的列表框中将自动出现该项目，表示用户选中了某个特长。如果用户在上面的“已选特长”列表框中双击一个项目，则该项目将自动消失，表示用户放弃了对某个特长的选择。

思考：如果只安排一个“特长”列表框，并且把列表框控件的 MultiSelect 属性值设置为 1，即允许多选，此时应如何编写程序，使得可以显示用户在“特长”列表框中选择的多个项目？

6.9　程序举例

【例 6.10】采用冒泡排序法对 n 个整数按升序排序。

分析：冒泡排序（BubbleSort）的基本概念是：依次比较相邻的两个数，将小数放在前面，大数放在后面。即首先比较第 1 个和第 2 个数，将小数放前，大数放后。然后比较第 2 个数和第 3 个数，将小数放前，大数放后，如此继续，直至比较最后两个数，将小数放前，大数放后。重复以上过程，仍从第 1 对数开始比较（因为可能由于第 2 个数和第 3 个数的交换，使得第 1 个数不再小于第 2 个数），将小数放前，大数放后，一直比较到最大数前的一对相邻数，将小数放前，大数放后，第 2 趟结束，在倒数第 2 个数中得到一个新的最大数。如此下去，直至最终完成排序。

由于在排序过程中总是小数往前放，大数往后放，相当于气泡往上升，所以称作冒泡排序。

算法可以用两层循环实现，外循环变量设为 i，内循环变量设为 j。外循环重复 n-1 次，内循环依次重复 n-1，n-2，…，1 次。每次进行比较的两个元素都是与内循环 j 有关的，它们可以分别用 a[j]和 a[j+1]标识，i 的值依次为 1，2，…，n-1，对于每一个 i，j 的值依次为 1，2，…n-i。

图 6-18 说明了该算法。

		a(1)	a(2)	a(3)	a(4)	a(5)	
初始数据		**165**	**100**	102	150	110	
第一次排序	第一次比较后	100	**165**	**102**	150	110	1,2 交换
	第二次比较后	100	102	**165**	**150**	110	2,3 交换
	第三次比较后	100	102	150	**165**	**110**	3,4 交换
	第四次比较后	**100**	**102**	150	110	165	4,5 交换
第二次排序	第一次比较后	100	**102**	**150**	110	165	1,2 不交换
	第二次比较后	100	102	**150**	**110**	165	2,3 不交换
	第三次比较后	**100**	**102**	110	150	165	3,4 交换
第三次排序	第一次比较后	100	**102**	**110**	115	165	1,2 不交换
	第二次比较后	100	102	110	115	165	2,3 不交换
第四次排序	第一次比较后	100	102	110	115	165	1,2 不交换
最后数据		100	102	110	115	165	

图 6-18　冒泡排序示例

图中加黑标注的是下次交换要比较的两个数，加底纹的表示不在排序范围内。具体代码如下：

```
Private Sub Command1_Click()
    Dim a(1 To 5) As Integer, i%, j%, t%
    Dim b() As Variant
    b = Array(165, 100, 102, 150, 110)
    n = 5
    For i = 1 To n
        a(i) = b(i - 1)
    Next i
    Picture1.Print "输出原数列"
    For i = 1 To n
      Picture1.Print a(i);
    Next i
    Picture1.Print
    '开始排序
    For i = 1 To n - 1
      For j = 1 To n - i
          If a(j) > a(j + 1) Then
              t = a(j)            'a(j)与 a(j+1)交换
              a(j) = a(j + 1)
              a(j + 1) = t
          End If
```

```
            Next j
        Next i
        Picture1.Print "输出排序之后的数列"
        For i = 1 To n
            Picture1.Print a(i);
        Next i
        Picture1.Print
    End Sub
```

运行程序，结果如图 6-19 所示。

图 6-19　例 6.10 的运行结果

说明：程序中为了从 1 开始，将 Array 函数生成的数组 b 逐个复制到 a 中，a 的下标从 1 开始，其实也可以将程序改成从下标 0 开始，具体代码如下：

```
Private Sub Command1_Click()
    Dim i%, j%, t%
    Dim a() As Variant
    a = Array(165, 100, 102, 150, 110)
    n = 5
    Picture1.Print "输出原数列"
    For i = 0 To n - 1
        Picture1.Print a(i);
    Next i
    Picture1.Print
    '开始排序
    For i = 0 To n - 2
        For j = 0 To n - i - 2
            If a(j) > a(j + 1) Then
                t = a(j)            'a(j)与 a(j+1)交换
                a(j) = a(j + 1)
                a(j + 1) = t
            End If
        Next j
    Next i
    Picture1.Print "输出排序之后的数列"
    For i = 0 To n - 1
        Picture1.Print a(i);
    Next i
    Picture1.Print
End Sub
```

【例 6.11】判断用户输入的文本是否为回文。如果一个文本的逆序与原文完全相同，这样的文本就称为回文，例如“abcba”、“B2B”、“16861”等。

分析：定义一个字符串变量 a，存放输入的文本，利用 Split 函数将其分隔到数组 b 中，用 s 代替 b 的每个元素。其次定义 2 个指示器 left 和 right，left 初始指示文本的第一个字符，right 初始指示文本的最后一个字符。在循环结构中反复判断 left 和 right 各自指示的字符是否相同，如果不同，显然不是回文；如果相同，则 left 不断加 1 向右移动，而 right 不断减 1 向左移动。

```
Private Sub Command1_Click()
    Dim a As String, i%, left%, right%, flag As Boolean
    Dim b() As String
    Dim s As String

    a = Trim(Text1.Text)
    b = Split(a)
    For i = LBound(b) To UBound(b)
      s = b(i)
      left = 1
      right = Len(s)
      flag = True
      Do While left < right
        If Mid(s, left, 1) <> Mid(s, right, 1) Then
            flag = False
            Exit Do
        End If
        left = left + 1
        right = right - 1
    Loop
    If flag = True Then
      Picture1.Print s + "是回文"
    Else
      Picture1.Print s + "不是回文"
    End If
  Next i
End Sub

Private Sub Form_Load()
    Text1.Text = ""
End Sub
```

运行程序，结果如图 6-20 所示。

图 6-20　例 6.11 的运行结果

说明：第 7 章将介绍判断回文的递归解法。

【例 6.12】编写程序，输出 n 行杨辉三角形。

分析：杨辉三角形中的各个元素实际上是二项式$(a+b)^n$的展开式中各项的系数，例如：

$(x+y)^1$展开后：$x+y$

$(x+y)^2$展开后：$x^2+2xy+y^2$

$(x+y)^3$展开后：$x^3+3x^2y+3xy^2+y^3$

$(x+y)^4$展开后：$x^4+4x^3y+6x^2y^2+4xy^3+y^4$

将多项式系数排列可以得到下面的图形：

```
1
1   1
1   2   1
1   3   3   1
1   4   6   4   1
```

其中第 m 项为：

$$C_n^m = \frac{n!}{m!(n-m)!}$$

杨辉三角形的规律如下：

（1）第一列及对角线元素均为 1。

（2）其他元素为其所在位置的上一行对应列和上一行前一列元素之和，如图 6-21 所示三角形中标注的三个数 4、6、10。

```
1
1    1
1    2    1
1    3    3    1
1    4    6    4    1
1    5   10   10    5    1
1    6   15   20   15    6    1
1    7   21   35   35   21    7    1
```

图 6-21　杨辉三角形分析示意图

程序代码如下：

```
Private Sub Command1_Click()
    Dim a() As Integer, n%, i%, j%, k%
    n = 10
    ReDim a(1 To n, 1 To n)
    For i = 1 To n
        a(i, 1) = 1                 '第一列元素置为 1
        a(i, i) = 1                 '对角线元素置为 1
    Next i

    For i = 3 To n                  '从第三行开始
        For j = 2 To i - 1          '从第二列开始，到对角线为止
            a(i, j) = a(i - 1, j - 1) + a(i - 1, j)
        Next j
```

```
        Next i

        For i = 1 To n
            k = 0
            For j = 1 To i
                Picture1.Print Tab(k * 6); a(i, j);
                k = k + 1
            Next j
            Picture1.Print
        Next i
    End Sub
```

运行程序，结果如图 6-22 所示。

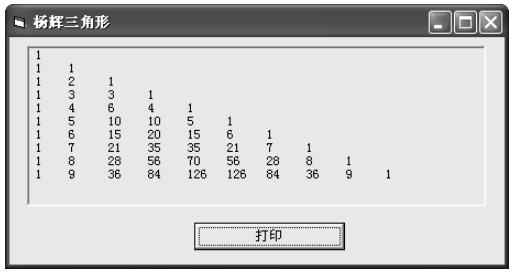

图 6-22　例 6.12 的运行结果

【例 6.13】编写程序删除一个字符串中的所有数字字符。

实现该算法的方法很多，下面是其中较简单的一种：

```
    Private Sub Command1_Click()
        Dim s As String, t As String, i%, j%
        s = Text1.Text
        t = ""
        For i = 1 To Len(s)
            If Not (Mid(s, i, 1) >= "0" And Mid(s, i, 1) <= "9") Then
               t = t + Mid(s, i, 1)
            End If
        Next i
        Picture1.Print t
    End Sub

    Private Sub Form_Load()
        Text1.Text = ""
    End Sub
```

运行程序，结果如图 6-23 所示。

图 6-23　例 6.13 的运行结果

如果要求不使用中间变量 t，直接在 s 上进行删除操作呢？下面的程序可以做到这一点：

```
Private Sub Command1_Click()
    Dim s As String, t As String, i%, j%
    s = Text1.Text
    t = ""
    j = Len(s)
    i = 1
    Do While i <= j
        If Mid(s, i, 1) >= "0" And Mid(s, i, 1) <= "9" Then
            s = Left(s, i - 1) + Right(s, j - i)
            j = Len(s)          '重新计算新串的长度
        Else
            i = i + 1           '非数字字符时才将 i 加 1，使得 i 指向下一个字符
        End If
    Loop
    Picture1.Print s
End Sub

Private Sub Form_Load()
    Text1.Text = ""
End Sub
```

习题

1．写出单击窗体后，下列程序段的运行结果。

```
Private Sub Form_Load()
    Show
    Dim a(3, 5) As Integer, i As Integer, j As Integer
    For i = 1 To 3
        For j = 1 To 5
            a(i, j) = a(i - 1, j) + j
        Next j
    Next i
    Print a(3, 4)
End Sub
```

2．将一个长度为 10 的数组逆序存储。

3．完成一个 3×3 矩阵的转置（即行列互换）。

4．编写程序，求下列矩阵各行元素之和及各列元素之和。

1	2	3	4	5
2	3	4	5	6
3	4	5	6	7
4	5	6	7	8

5．把两个已按升序排列的数列合并为一个新数列，该数列仍按升序排列。例如数列 a 是[1,3,6,7,9]，数列 b 是[2,4,5,8,10]，合并之后的新数列是[1,2,3,4,5,6,7,8,9,10]。

6．把一个数插入到一个已按升序排列的数列 a 中，并使该数列仍按升序排列。例如数列 a 是[1,3,6,8,9,10]，要插入的数是 4，合并之后的新数列是[1,3,4,6,8,9,10]。

7．某班有学生 60 人，学习语文、数学、英语和计算机 4 门课程。输入所有学生各门课程的成绩，输出单科成绩的最高分以及该班每门课的平均成绩。