三、Visual Basic程序设计

# 1.集成开发环境（IDE）及其主要窗口的使用；

# 2.应用程序的建立、编辑、调试、运行和保存；

# 3.窗体的常用属性、方法和事件；

属性

（1）Caption 属性

设置窗体的标题内容。标题内容应概括说明本窗体作用。

（2）MaxButton 属性和 MinButton 属性

MaxButton为True，窗体右上角有最大化按钮；为False时，无最大化按钮。

MinButton 属性为True，窗体右上角有最小化按钮；为False时，无最小化按钮。

（3）BorderStyle 属性

决定窗体边框式样及窗体是否能调整大小。

（4）BackColor属性和ForeColor 属性

BackColor属性设置窗体的背景颜色；ForColor 属性设置窗体的前景颜色。窗体的前景颜色是执行Print方法时所显示文本的颜色。

（5）Height属性和Width属性

设置窗体的初始高度和宽度。其单位为Twip。1Twip=1/20点=1/1440英寸=1/567厘米。

（6）Left属性和Top属性

用于设置窗体左边框距屏幕左边界的距离和窗体顶边距屏幕顶端的距离。其单位为Twip。

（7）Name（名称）属性

设置窗体的名称，在程序代码中用这个名称引用该窗体。新建工程时，窗体的名称缺省为 Form1；添加第二个窗体，其名称缺省为 Form2，以此类推。为了便于识别，用户通常给 Name 属性设置一个有实际意义的名称。

（8）Enabled属性

属性值为True或False，设置对象是否能够对用户产生的事件做出反应。一般在程序中设置，用于临时屏蔽对窗体或其他控件的的控制。

（9）Moveable属性

属性值为True或False，设置是否可以移动窗体。

（10）Visible属性

属性值为True或False，设置窗体是否被显示。用户可用该属性在程序代码中控制窗体的隐现。

（11）Picture属性

设置在窗体中显示的图片。单击Picture属性右边的按钮，弹出“加载图片”对话框，用户可选择一个图片文件作为窗体的背景图片。若在程序中设置该属性的值，需要使用LoadPicture函数。

（12）WindowState属性

设置窗体启动后的大小状态。它有三个可选值：

0-Normal: 窗体大小由Height和Width属性决定。

1-Minimized: 窗体最小化成图标。

2-Maximized: 窗体最大化，充满整个屏幕。

方法

（1）Click事件

程序运行后，单击窗体触发该事件。

（2）DbClick事件

程序运行后，双击窗体触发该事件。

（3）Load事件

Load事件是窗体被装入内存工作区时触发的事件。如果这个事件过程存在，就马上执行它。Load事件过程通常用于启动程序时对属性、变量的初始化，装载数据等。

# 4.基本数据类型和运算符；

VB中的标准数据类型

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据类型 | 关键字 | 类型符 | 占内存字节数 | 范围 |
| 字符串型 | String | $ | 与字符长度有关 | 最多65535个字符 |
| 整型 | Integer | % | 2 | -32768~32767 |
| 长整型 | Long | & | 4 | -2147483648~2147483647 |
| 字节型 | Byte | 无 | 1 | 0~255 |
| 单精度型 | Single | ! | 4 | 1.401298E-45≥|x|≤3.402823E38 |
| 双精度型 | Double | # | 8 | 4.94065645841247E-324≥|x|  ≤1.79769313486232E308 |
| 货币型 | Currency | @ | 8 | -922337203685477.5808~922337203685477.5807 |
| 逻辑型 | Boolean | 无 | 2 | True或False |
| 日期型 | Date | 无 | 8 | 100.01.01~9999.12.31 |
| 变体型 | Variant | 无 | 根据需要 |  |

详情见ppt

算术运算符

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 说明 | 示例 | 优先级 |
| ^ | 乘方 | x^y | 1 |
| \*、/ | 乘、除 | x\*y ，x/y | 2 |
| \ | 整除 | x\y | 3 |
| Mod | 取模 | x  Mod y | 4 |
| +、- | 加、减 | x+y，x-y | 5 |

字符串运算符

字符串只有连接运算，在VB中可以用“+”或“&”。建议尽量使用“&”，使程序看起来更明了。使用“&”运算符时应注意前后加空格，否则VB会当作长整数型的类型符来处理。

关系运算符

关系运算符用作两个数值或字符串的比较，返回值是逻辑值True或False。

VB中的关系运算符及使用示例。

|  |  |  |  |
| --- | --- | --- | --- |
| 运算符 | 意义 | 示例 | 返回值 |
| = | 等于 | "ABC"="ABF" | False |
| > | 大于 | "ABC">"AF" | False |
| >= | 大于等于 | "f" >= "Fgh" | True |
| < | 小于 | 25<45.5 | True |
| <= | 小于等于 | 23<=23 | True |
| <> | 不等于 | "XYZ"<>"xyz" | True |
| Like | 使用通配符匹配比较 | "WXYZ" Like "\*X\*" | True |
| Is | 引用对象比较 | Is>0 | 由对象当前值决定 |

关系运算符的优先级相同

逻辑运算符

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 运算符 | 意义 | 优先级 | 说明 | 示例 | 返回值 |
| Not | 取反 | 1 | 操作数为假时，结果为真 | Not true | False |
| And | 与 | 2 | 两个操作数均为真时，结果才为真。其余为假 | False And True  True And True | False  True |
| Or | 或 | 3 | 两个操作数只要有一个为真，结果为真 | False Or True  True Or True | True  True |
| Xor | 异或 | 3 | 两个操作数为一真一假时，结果为真 | False Xor True  True  Xor True | True  False |
| Eqv | 等价 | 4 | 两个操作数同为真或假时，结果为真 | False Eqv True  False Eqv False | False  True |
| Imp | 蕴含 | 5 | 第一个操作数为真，第二个操作数为假时，结果为假，其余情况都为真 | True Imp False  False Imp True  True  Imp True | False  True  True |

# 5.常量、变量、表达式的定义和使用；

常量

在程序执行的过程中保持不变的数据称为常量。在VB中，常量分为两种，文字常量和符号常量。符号常量又分为用户自定义和系统定义两种。

1．文字常量

文字常量直接出现在代码中，也称为字面常量或直接常量，文字常量的表示形式决定它的类型和值。

例如：

字符型："I am a student" 放在一对引号当中。

数值型：3.14159、56、8.432E-15

日期型：#3 jan,98# 放在一对#当中。

逻辑型：True 、False

2．符号常量

符号常量就是用标识符来表示一个常量，例如：我们把3.14定义为pi，在程序代码中，我们就可以在使用圆周率的地方使用pi。使用符号常量的好处主要在于，当我们要修改该常量时，只需要修改定义该常量的一个语句即可。

定义常量的方法如下：

格式：const 常量名 [as类型]=表达式

说明：常量名的命名规则与标识符相同。[as类型]用以说明常量的数据类型。

表达式可以是文字常量，也可以是运算符连接文字常量构成的表达式。在一行中说明多个常量时用逗号分开。例如：

Const mystr as string="Visual"+"Basic"

Const num = 85, pi as double=3.1415926

在VB中，系统定义了一系列常量。

例如：form1.Windowstate=vbMinimized 意义为将窗口最小化。

其中vbMinimized就是一个系统定义的常量，值为1。和form1.Windowstate=1相比较，form1.Windowstate=vbMinimized更明确的表达了语句的功能。

系统定义的常量在对象库中，可以在对象浏览器中通过不同的对象库查找它们的符号及取值。

变量

在程序执行过程中，其值可以改变的量称为变量。

在 Visual Basic 中执行应用程序期间，用变量临时存储数据。

变量代表内存中指定的存储单元，变量以标识符命名。

每个变量都有相应的类型，类型决定了该变量的取值范围和可以执行的运算操作。

1．变量的说明

在代码中，使用变量前，应首先定义所用到的变量（包括变量名和类型），使系统分配相应的内存空间，并确定该空间可存储的数据类型。所有变量都具有名字和类型。在VB中可以用类型说明语句或隐式说明来定义变量。

（1）类型说明语句

格式：说明符 变量名 [As 类型]

说明：

说明符是说明语句的关键字，它可以是Dim，Private，Public，Static。

Private 定义的是局部变量。

Public 定义的是全局变量。

Static 定义的是静态变量。

这三个关键字的用法将在后面的章节详细介绍。本章主要介绍Dim语句。

例如：

Dim a as integer

Dim b as long

Dim c as single

把变量a定义成整数型，变量b定义成长整数型，c定义成单精度型。

在一个说明语句中可以用逗号隔开说明多个变量，上面的三个语句可以写为：

Dim a as integer ，b as long，c as single

也可以用类型符来定义变量，例如上面语句写成：

Dim a% ，b&，c! 作用是一样的。

缺省 [As类型]的为可变类型。

另外，若把多个变量都定义成同一类型，例如：把X，Y，Z都定义成双精度型，必须写成：Dim X as double, Y as double, Z as double

如果写成：Dim x,y,z as double

则x，y定义成可变类型，z定义成双精度型。

对于字符型变量，VB中分为定长和变长两种。例如：

Dim Name as string，Id as string\*10

定义Name为变长字符型，其长度由接受的值决定。Id为定长字符型，长度为10个字符。注意，在VB中，汉字与字母长度相同。定长字符型接收数据时，不够指定长度的用空格补齐，右边超过指定长度的超出部分无效。

说明：Dim语句定义的变量其作用范围由Dim语句所在的位置决定。Dim语句出现在窗体代码的声明部分时，则窗体以及窗体中各控件的事件过程都可以使用这些变量。这种变量称为窗体级变量；在过程内部用Dim语句声明的变量，只在该过程内有效。这种变量称为局部变量。

表达式

表达式由常量、变量、函数、运算符、（）按照一定的规则组成，不管表达式的形式如何，都会计算出一个结果，该结果的类型由参与运算的数据和运算符决定。

* 表达式的书写规则

（1）表达式中的每个字符没有高低、大小的区别。

（2）只能使用圆括号，可以多重使用，圆括号必须成对出现。

（3）VB表达式中的乘号“\*”不能省略。

（4）能用系统函数的地方尽量使用系统函数。

例如：数学公式  写成VB表达式为：

(-b+sqr(b^2-4\*a\*c))/(2\*a)

只有算术运算符的表达式也称为算术表达式。

* 关系表达式和逻辑表达式

当使用关系运算符或逻辑运算符时，表达式又称为关系表达式或逻辑表达式。

关系运算一般表示一个简单的条件。

例如：age>20 、score>80、x+y>z等。

逻辑表达式表示较复杂的条件。

例如：数学中的0<x<5，写成VB表达式应为0<x And x<5。

* 结果类型
* 优先级

圆括号>算术运算符>关系运算符>逻辑运算符。

在复杂的表达式中，可以增加圆括号使表达式的运算次序更清晰。

算术表达式中，不同类型的数据计算时结果转化成精度高的类型。

关系表达式和逻辑表达式的结果是逻辑值：True、False。

# 6.常用内部函数的使用；

VB提供了大量的内部函数供用户调用。在本节中我们分类介绍一些常用的内部函数。

函数的一般调用格式：

格式：函数名（[参数表]）

说明：参数表可以有一个参数或逗号隔开的多个参数，多数参数都可以使用表达式。函数一般作为表达式的组成部分调用。

Visual Basic 6.0中部分常用函数

|  |  |  |
| --- | --- | --- |
| 类别 | 函数名 | 作用 |
| 类型转换函数 | Cint（x） | 将X的值的小数部分四舍五入转换为整型 |
| CLng（x） | 将X的值的小数部分四舍五入转换为长整型 |
| CSng（x） | 将X的值转换为单精度浮点型 |
| CDbl（x） | 将X的值转换为双精度浮点型 |
| CStr（x） | 将X的值转换为字符型 |
| CBool（x） | 将X的值转换为布尔型 |
| Cvar（x） | 将X的值转换为变体型 |
| Val（C） | 将代表值的字符串转换数值型数据 |
| Str$（N） | 将数值型数据换成代表数值的字符型 |
| Asc（C）） | 返回字符串C首字符的ASCII值 |
| Chr（N） | 返回ASCII值为N的字符 |
| 数  学  函  数 | Abs（x） | 返回X的值的绝对值 |
| Spr（x） | 返回X的值的平方根 |
| Fix（x） | 若x的值是正数，则返回该数的整数总数；若是负数，则返回一个不小于参数的最小整数 |
| Int（x） | 若x的值是正数，则返回该数的整数部分；若是负数，则返回一个不大于参数的最大整数 |
| Sgn（x） | 返回x的符号，x为正数时返回1，为0时返回0，x为负数时返回-1 |
| Exp（x） | 返回以e为底的x的指数值 |
| Log（x） | 返回以e为底的x的对数值 |
| Sin（x） | 返回x的正弦值 |
| Cos（x） | 返回x的余弦值 |
| Tan（x） | 返回x的正切值 |
| Atn（x） | 返回x的余切值 |
| Rnd | 返回一个0~1的单精度随机数 |
| 字符串函数 | Trim（C） | 去掉字符串C两端的空格 |
| Left（C，n） | 截取C最左边的n个字符 |
| Right（C，n） | 截取C最右边的n个字符 |
| Mid（C，m，n） | 截取C中从第m个字符开始的n个字符 |
| Len（C） | 返回C包含的字符数，汉子空格都算一个字符 |
| Ucase（C） | 将C中的小写字母转化成大写字母 |
| Lcase（C） | 将C中的大写字母转化为小写字母 |
| 日期与时间函数 | Time[$][0] | 返回系统当前时间 |
| Date[$][0] | 返回系统当前日期 |
| Now[0] | 返回系统当前日期和时间 |
| Day(C/N) | 返回数据中当月第几天 |
| Month(C/N) | 返回数据中当年第几月 |
| WeekDay（C/N） | 返回数据当天是星期几 |

# 7.顺序程序设计；

# 8.赋值语句、结束语句的使用；

赋值语句是VB程序最常用、最基本的语句，它能为变量提供数据。另外，若要在程序代码中设置对象的属性，也是使用赋值语句。

赋值语句有两种格式：

格式1： 变量名=表达式

格式2： [对象名.]属性名=表达式

赋值语句的作用是：首先计算“=”右边表达式的值，然后将该值赋给“=”左边的变量或对象的属性。在格式2中，若对象名省略，则默认对象为当前窗体。

例如：

x=2 ' 把2赋给x

y=x\*3 ' 计算x\*3的值，得6，把6赋给y

x=x+1 ' 计算x+1的值，得3，把3赋给x

a$= "Hello" ' 把“Hello”赋给a$

Text1.Text="你好！" ' 把“你好！”赋给Text1的Text属性

这里还要说明一下赋值语句中的类型问题。通常，在使用时应使表达式值的类型与变量（或对象的属性）类型相同，以避免出现“类型不匹配”的错误。不过，某些情况下，VB容许“=”两边类型不相同。

例如：

dim x!,y%,z$

x = “12.5”‘ 把数值形式的字符串赋给

数值变量

y=34.24 ‘ 把单精度数赋给整型变

量，此时截去小数部分

z=6 ‘ 把整数赋给实型变量

以上三种情况都是赋值相容的。在赋值相容前提下，当“=”两边类型不相同时，VB自动将表达式值的类型转换成“=”左边变量（或对象的属性）的类型，这种方式称为“向左看齐”。上例中，实际赋给x、y、z的值分别是12.5、34、6.0。

结束语句

格式：End

End语句用来结束程序的执行，并关闭已打开的文件。

例如：

Private Sub Command3\_Click()

End

End Sub

# 9.Print语句的使用；

Print是输出数据的一种重要方法。

格式：[对象名.]Print [表达式列表]。

功能：在对象上输出表达式的值。

（1）对象名：

可以是Form（窗体）、Debug（立即窗口）、Picture（图片框）、Printer（打印机）。省略此项，表示在当前窗体上输出。

例如：

Print "23\*2="；23\*2 ' 在当前窗体上输出23\*2= 46

Picture1.Print "Good " ' 在图片框Picture1上输出Good

Printer.Print "Morning" ' 在打印机上输出Morning

（2）表达式列表：

是一个或多个表达式，若为多个表达式，则各表达式之间“，”或“；”隔开。省略此项，则输出一空行。

（3）用“，”分隔各表达式时，各项在以14个字符位置为单位划分出的区段中输出，每个区段输出一项；用“；”分隔各表达式时，各项按紧凑格式输出。

（4）如果在语句行末尾有“；”，则下一个Print输出的内容，将紧跟在当前Print输出内容后面；

如果在语句行末尾有“，”，则下一个Print输出的内容，将在当前Print输出内容的下一区段输出；

如果在语句行末尾无分隔符，则输出完本语句内容后换行，即在新的一行输出下一个Print的内容。

例如：

Print 1；2；3

Print 4，5，

Print 6

Print 7，8

Print

Print 9，10

输出结果为：

1 2 3

4 5 6

7 8

9 10

（4）定位输出

Print方法中，可以使用Tab函数对输出项进行定位。

例如：

Print Tab(10) ; "姓名" ; Tab(25) ; "年龄"

则“姓名”和“年龄”分别从当前行的第10列和第25列开始输出。

输出结果如下：

姓名 年龄

在使用Tab函数时，要将输出的内容放在Tab函数的后面，并用“；”隔开。

Tab函数的格式为Tab(n)，其中n为整数表达式，用它来指定输出的起始位置。

Print方法中，还可以使用Spc函数，例如：

Print "后面有8个空格"；Spc (8) ; "前面有8个空格"

输出结果如下：

后面有8个空格 前面有8个空格

Spc函数格式为Spc(n)，其中n为整数表达式，表示在下一个输出项之前插入的空格数，Spc函数与各输出项之间必须用“；”隔开。

# 10.InputBox输入函数的使用；

我们知道，文本框可以接收用户的输入。输入框也可以接受用户的输入，但是其风格和用法有别于文本框。VB提供的InputBox函数可生成输入框。

例如：p$=InputBox（"请输入密码" ，"密码框" ）

执行该语句后，屏幕上显示如图3-6所示的输入框。



1. 标题
2. 文本区
3. 提示
4. **图3-6 输入框**

用户可在文本区输入数据，然后按“确定”按钮，函数返回值是用户在文本区输入数据，其类型为字符型。如果用户然后按“取消”按钮，则函数返回值是空字符串。

每执行一次InputBox函数，用户只能输入一个数据，另外，输入框的样式是固定的，用户不能改变。用户能改变的是输入框的“提示”和“标题”的内容，“提示”和“标题”都是字符串表达式。

InputBox函数的一般格式：

InputBox（提示[，标题][，缺省值][，x坐标位置][，y坐标位置]）

其中各参数的含义如下：

“提示”：必选项。字符串表达式，在对话框中作为提示信息。若要在多行显示提示信息，则可以在各行之间用vbNewLine来分隔，vbNewLine是代表换行的常量。

例如：InputBox ("第一行" & vbNewLine & "第二行")

“标题”：字符串表达式，在对话框中标题区显示，若省略，则标题为应用程序名。

“缺省值”：字符串表达式，在没有其它输入时作为缺省值。

“x坐标位置”、“y坐标位置”：整数表达式。坐标确定对话框左上角在屏幕上的位置，屏幕左上角为坐标原点，单位为Twip。1Twip==1/567厘米。

需要注意的是：各项参数次序必须一一对应，除了“提示”不能省略外，其余各项均可省略，但省略部分也要用逗号占位符跳过。

例如：f$ = InputBox("第一行" & vbNewLine & "第二行", , "ddd", 200, 200)

# 11.消息对话框MsgBox函数的使用；

执行VB提供的MsgBox函数，可以在屏幕上出现一个消息框，消息框通知用户消息并等待用户来选择消息框中的按钮，MsgBox函数返回一个与用户所选按钮相对应的整数。

MsgBox函数的格式：

MsgBox（提示，[，按钮数值][，标题]）

例如：inta=MsgBox（“密码错”，21，“密码核对” ）

执行该语句后，屏幕上显示如图3-10所示的消息框。







在MsgBox函数格式中，“提示”和“标题”的含义同InputBox函数。

“按钮数值”是三个数值之和，这三个数值分别代表钮的数目及类型，使用的图标样式，及缺省按钮是什么。

表5-4 “ buttons”参数值

|  |  |  |  |
| --- | --- | --- | --- |
|  | 数值 | 常数 | 含义 |
| 设置按钮的类型 | 0 | vbOKOnly | 添加[确定]按钮 |
| 1 | vbOKCancel | 添加【确定】、【取消】按钮 |
| 2 | vbAbortRetrylgnore | 添加【终止】、【重试】、【忽略】按钮 |
| 3 | vb YesNoCancel | 添加是【是】、【否】、【取消】按钮 |
| 4 | vb YesNo | 添加【是】、【否】按钮 |
| 5 | vbRetryCancel | 添加试【重试】、【取消】按钮 |
| 设置图标的样式 | 16 | vbCritical | 添加【×】图标 |
| 32 | vbQuestion | 添加【？】图标 |
| 48 | vbExclamation | 添加【！】图标 |
| 64 | vbInFormation | 添加【i】图标 |
| 设置默认按钮 | 0 | vbDefaultButton1 | 第一个按钮为默认按钮 |
| 256 | vbDefaultButton2 | 第二个按钮为默认按钮 |
| 512 | vbDefaultButton3 | 第三个按钮为默认按钮 |

MsgBox函数返回的是被单击的按钮,由“ buttons”参数值来决定,具体的返回值，如表5-5所示。当【消息】对话框含有多个按钮时,用户还可以根据 Msg Box函数返回值的不同,来执行不同的操作。

表5-5 MsgBox函数返回值

|  |  |  |
| --- | --- | --- |
| 返回值 | 常量 | 说明 |
| 1 | vbOK | 单击【确定】按钮 |
| 2 | vbCancel | 单击【取消】按钮 |
| 3 | vbAbort | 单击【终止】按钮 |
| 4 | vbRetry | 单击【重试】按钮 |
| 5 | vblgnore | 单击【忽略】按钮 |
| 6 | vbYes | 单击【是】按钮 |
| 7 | vbNo | 单击【否】按钮 |

MsgBox也可以写成语句形式，例如： MsgBox "密码错", , "密码核对

执行此语句也产生一个消息框，如图3-11所示。



MsgBox语句没有返回值，因此常用于比较简单的信息提示。

# 12.选择结构程序设计；

* 行if语句

格式：If 条件 then 语句1 [Else 语句2]

功能：当条件满足时，执行语句1；条件不满足时，执行语句2。

【例4.6】任意输入三个数，找出其中的最大值。

程序代码如下：

Private Sub Command1\_Click()

Dim a As Single, b As Single, c As Single, max As Single

a = Text1.Text

b = Text2.Text

c = Text3.Text

max = a

If b > max Then max = b

If c > max Then max = c

Text4.Text = max

End Sub

* 块if语句

格式：

If 条件 then

语句块1

[ Else

语句块2]

End If

说明：

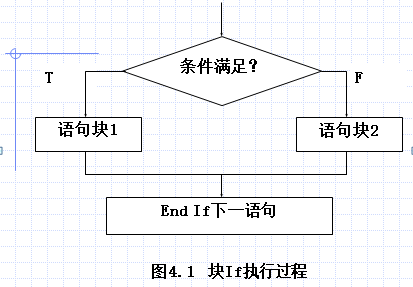
1．“条件”一般为关系表达式或逻辑表达式。

通常把关系表达式或逻辑表达式的值为真时，称为条件满足；值为假时，称为条件不满足。反之亦然。

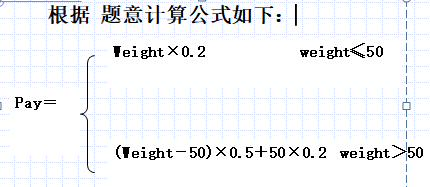
2．语句块1、语句块2分别是“条件”满足或不满足时，处理方法的描述，可以是若干个语句。

3．If…Then 、Else、End If 是VB的保留字。

4.块If首先判断“条件”，其值为真时，执行语句块1；为假时执行语句块2。当缺省[Else…]中的内容时，该选择结构只对条件满足的情况进行处理。



【例4.1】火车站行李费的收费标准是50公斤以内（包括50公斤）0.20元/公斤，超过部分0.50元/公斤。编写程序，要求根据输入的任意重量，计算出应付的行李费。



Private Sub Command1\_Click()

Dim weight as single,pay as single

weight= Text1.Text

If weight > 50 Then

pay = (weight - 50) \* 0.5 + 50 \* 0.2

Else

pay = weight \* 0.2

End If

Text2.Text = pay

End Sub

Private Sub Command2\_Click()

Text1.Text = ""

Text2.Text = ""

End Sub

Private Sub Command3\_Click()

End

End Sub

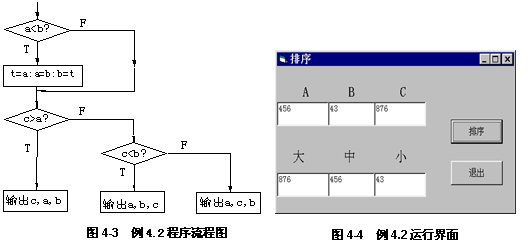
* 块if的嵌套

所谓块If语句的嵌套，就是在语句块1或语句块2中又包含块If语句。

【例4.2】任意输入三个数，按照从大到小的顺序输出。

算法分析：排序的基本方法，就是比较大小，然后根据比较的结果分别加以处理。

本例把三个数分别放在A，B，C中。处理过程为：若A<B为真， 交换A，B的值；否则不做处理。这样就保证了A≥B；然后再用C去比较。



Private Sub Command1\_Click()

Dim a As Single, c As Single, b As Single

a = Text1.Text

b = Text2.Text

c = Text3.Text '三个文本框的数据赋值给变量

If a < b Then

t = a

a = b

b = t 'a<b时交换a、b的值

End If '保证A>B

If c > a Then '用C去比较

Text4.Text = c 'C>A成立，C最大

Text5.Text = a

Text6.Text = b

Else

If c < b Then

Text4.Text = a 'C<B成立，C最小

Text5.Text = b

Text6.Text = c

Else

Text4.Text = a 'C处于中间

Text5.Text = c

Text6.Text = b

End If

End If

End Sub

Private Sub Command2\_Click()

End

End Sub

注意：

1．保持块If结构的完整，不要漏掉End If。

2．尽量采用缩进式书写格式，使结构清晰。

3．尽量选择恰当的条件，使程序简单明了。

* 多分支结构

当需要进行多路分支处理时，例如，根据一个学生的分数，判定他的等级（优、良、中、及格、不及格），由于条件比较多，会出现较繁琐的嵌套。而利用ElseIf语句或Select Case语句则可以方便解决这类问题。

格式：

If 条件1 then

语句块1

ElseIf 条件2 then

语句块2

……

[Else

语句块n+1]

End If

格式：

If 条件1 then

语句块1

ElseIf 条件2 then

语句块2

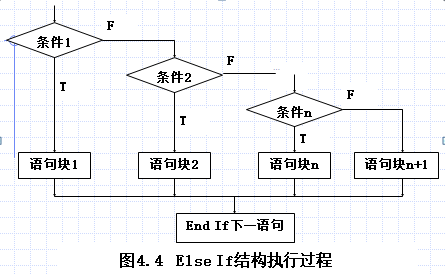
……

[Else

语句块n+1]

End If

2．如果所列出的条件都不满足，则执行Else语句后面的语句块；如果所列出的条件都不满足，又没有Else子句，则直接跳过End If，不执行任何语句块。图4.4 Else If结构执行过程



【例4.4】 输入一个学生的一门课分数X（百分制），当x≥90时，输出“优秀”；当80≤x＜90 时，输出“良好”；当 70≤x＜80 时，输出“中”；当60≤x＜70 时，输出“及格”，当 x＜60 时，输出“不及格”。

算法分析：本例适合用多路分支结构来解决。运行界面如图4-7所示。

Private Sub Command1\_Click()

Dim score!

score = Text1.Text

If score >= 90 Then

Text2.Text = "优秀"

ElseIf score >= 80 Then

Text2.Text = "良好"

ElseIf score >= 70 Then

Text2.Text = "中"

ElseIf score >= 60 Then

Text2.Text = "及格"

Else

Text2.Text = "不及格"

End If

End Sub

' “退出”按钮代码

Private Sub Command2\_Click()

End

End Sub

ElseIf语句，实际完成的是块If的嵌套，它和块If嵌套的格式上有很大区别， ElseIf结构只有一对If和End If语句。另外应注意：

【Else

If】

和【 ElseIf】的区别。

* Select case语句

当对一个表达式的不同取值情况作不同处理时，用ElseIf语句程序结构显得较为杂乱，而用Select Case语句将使程序的结构更清晰，Select Case语句又称为情况语句。

格式：

格式：

Select Case 测试表达式

Case 表达式列表1

语句块1

Case 表达式列表2

语句块2

…

Case 表达式列表n

语句块n

[Case Else

语句块n+1]

End Select

功能：根据“测试表达式”的值，选择第一个符合条件的语句块执行。

Select Case语句的执行过程是：

先求“测试表达式”的值，然后顺序测试该值符合哪一个Case子句中情况，如果找到了，则执行该Case子句下面的语句块，然后执行End Select下面的语句；如果没找到，则执行Case Else下面的语句块，然后执行End Select下面的语句。

说明：

1．“测试表达式”可以是数值型或字符串型表达式；

2．“表达式列表”形式有以下三种：

（1）一个表达式或用逗号隔开的若干表达式

例如：Case 2,4,6,8

表示的情况是：测试表达式的值等于2，4，6，8之一。

（2）表达式1 To表达式2

例如：Case 80 To 90

表示的情况是：80≤测试表达式≤90。

（3）Is 关系运算符 表达式

例如：Case Is>x^2

表示的情况是：测试表达式> x^2。

【例4.7】把例4.3中 “鉴定”按钮的代码用Select Case语句改写。

Private Sub Command1\_Click()

Dim score!

score = Text1.Text

Select Case score

Case Is >= 90

Text2.Text = "优秀"

Case Is >= 80

Text2.Text = "良好"

Case Is >= 70

Text2.Text = "中"

Case Is >= 60

Text2.Text = "及格"

Case Else

Text2.Text = "不及格"

End Select

End Sub

# 13.逻辑表达式的正确书写；

参见运算符与表达式部分

# 14.If语句和Select Case语句的使用；

参见if语句部分

# 15.循环结构程序设计；

所谓循环就是重复地执行一组语句。

在实际工作中，常遇到一些操作过程不复杂但需要反复进行相同处理的问题，这些问题如果单纯用顺序结构来处理，将得到一个非常乏味且冗长的程序。

例如，计算1~100所有奇数的平方和，如果用顺序结构来解决这个问题，我们就会给出下面的程序：

Private Sub Form\_Click()

Dim s&, x%

s = 0

x = 1

s = s + x ^2

x = x + 2

s = s + x ^2

x = x + 2

s = s + x ^2

……

x = x +2 'x的值累加到99

s = s + x ^2

Print "1~100之间所有奇数的平方和="; s

End Sub

我们用循环语句解决上面的问题，程序非常简短：

Private Sub Form\_Click()

Dim s&, x%

s = 0

For x = 1 To 99 Step 2

s = s + x ^2

Next x

Print "1~100所有奇数的平方和="; s

End Sub

# 16.For、Do和While循环语句的使用；

For循环的一般格式如下：

For 循环变量=初值 To 终值 [Step 步长]

[循环体]

Next [循环变量]

说明：

1．格式中各项的说明

（1）循环变量：

亦称为循环控制变量，必须为数值型变量。

（2）初值、终值：

都是数值型，可以是数值表达式。

（3）步长：

循环变量的增量，是一个数值表达式。一般来说，其值为正，初值应小于终值；若为负，初值应大于终值。但步长不能是0。如果步长是1， Step 1可略去不写。

（4）循环体：

在For语句和Next语句之间的语句序列。

（5）Next后面的循环变量与For语句中的循环变量必须相同。

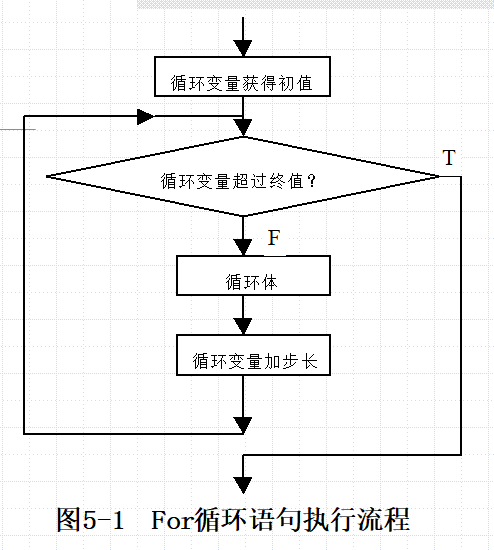
2．执行过程

For循环语句的执行过程如下：

（1）系统将初值赋给循环变量，并自动记下终值和步长。

（2）检查循环变量的值是否超过终值。如果超过就结束循环，执行Next后面的语句；否则，执行一次循环体。

（3）执行Next语句，将循环变量增加一个步长值再赋给循环变量，转到（2）继续执行。



For n=1 To 10 Step 3

Print n,

Next n

3．其它说明

（1）循环次数计算公式为：

循环次数=Int（（终值-初值）/步长）+1

（2）循环控制变量通常用整型数，也可以用单精度数或双精度数。值得注意的是，无论初值、终值和步长值是什么数值类型，最后都要转换成循环控制变量的类型。

【例5.1】求N！（N为自然数）。

分析：由阶乘的定义，我们可以得出N！=1\*2\*…\*(N-2)\*(N-1)\*N=(N-1)！\*N，也就是说，一个自然数的阶乘，等于该自然数与前一个自然数阶乘的乘积，

程序如下：

Private Sub Form\_Click()

Dim I%, f#, n%

n= InputBox("输入一个自然数：", "输入提示", "10")

f = 1

For I = 1 To n

f = f \* I

Next I

Print n; "!="; f

End Sub

【例5.2】判断用户输入的数是否为素数。

分析：素数的特征是只能被1和它自身整除。假设用户输入的正整数为N，我们只需确定在大于1小于等于的正整数中是否存在能整除N的数。如果有，则N就不是素数；如果没有，则N就是素数。

程序如下：

Private Sub Form\_Click()

Dim n%, flag%, I%, k%

n = InputBox("请输入一个正整数（≥3）")

k = Int(Sqr(n))

flag = 0

For I = 2 To k

If n Mod I = 0 Then flag = 1

Next I

If flag = 0 Then

Print n; "是一个素数"

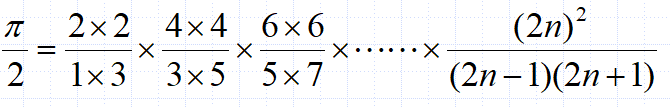
Else

Print n; "不是素数"

End If

End Sub

【例5.3】求π值。计算公式如下：



分析：不难看出，结果由n项分式相乘得到，只要给定了n值，用For-Next语句可以非常容易地实现。注意，n值越大，结果越接近π值。

Private Sub Form\_Click()

Dim I%, n%, p#

n = InputBox("请输入n的值（1-32767）")

p = 1

For I = 1 To n

p = p \* (4 \* I \* I) / ((2 \* I - 1) \* (2 \* I + 1))

Next I

p = 2 \* p

Print "n="; n; "时", "π="; p

End Sub

While循环

前面我们介绍了For…Next循环，它适合于解决循环次数事先能够确定的问题。对于只知道控制条件，但不能预先确定需要执行多少次循环体的情况，我们可以使用While循环。

语句格式如下：

While 条件

[循环体]

Wend

1．While语句说明

“条件”可以是关系表达式或逻辑表达式。While循环就是当给定的“条件”为True时，执行循环体，为False时不执行循环体。因此While循环也叫当型循环。

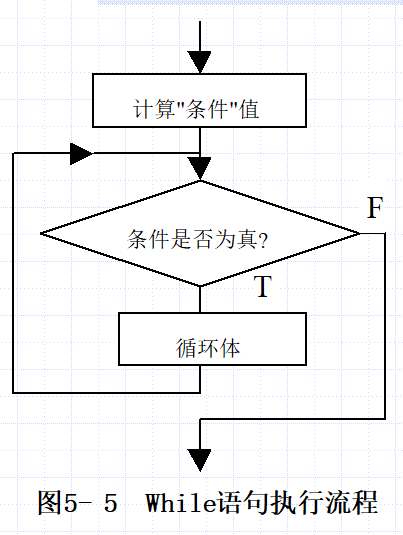
While循环的执行过程(如图5-5所示)：

（1）执行While语句，判断条件是否成立。

（2）如果条件成立，就执行循环体；否则，转到（4）执行。

（3）执行Wend语句，转到（1）执行。

（4）执行Wend语句下面的语句。



结合下面的程序段，我们做进一步的说明：

x=1

While x<5

Print x,

x=x+1

Wend

上面的程序就是在x<5的条件下，重复执行语句print x。每次执行循环之前，都要计算条件表达式的值。如果条件求值的结果为True，则执行循环体，然后再对条件进行计算判断，从而确定是否再次执行循环体；如果结果为False，则结束循环，执行Wend下面语句。

3．While循环的几点说明

（1）While 循环语句本身不能修改循环条件，所以必须在循环体内设置相应语句，使得整个循环趋于结束，以避免死循环。

（2）While循环语句先对条件进行判断，然后才决定是否执行循环体。如果开始条件就不成立，则循环体一次也不执行。

（3）凡是用For…Next循环编写的程序，我们都可以用While…Wend语句实现。反之，则不然。

【例5.4】找出一个最大正整数N，使N！<1000。

分析：该题就是要找到一个正整数，使它的阶乘最接近1000但又不超过1000。因此，我们应该将从1开始的自然数累乘，当积第一次大于1000时结束循环，累乘的最后一个数的前一个数即为所求。

程序如下：

Private Sub Form\_Click()

Dim I%, p%, n%

I = 0

p = 1

While p < 1000

I = I + 1

p = p \* I

wend

p = p / I

n = I - 1

Print "N="; n, n; "!="; p

End Sub

【例5.5】假设我国现有人口12亿，若年增长率为1.5%，试计算多少年后我国人口增加到或超过20亿。

人口计算公式为：p=y(1+r)n

y为人口初值，r为年增长率，n为年数。

程序如下：

Private Sub Form\_Click()

Dim p!, r!, I%

p = 12

r = 0.015

I = 0

While p < 20

p = p \* (1 + r)

I = I + 1

Wend

Print I; "年后，我国人口将达到"; p; "亿"

End Sub

Do循环

与前面介绍的While循环相比，Do循环具有更强的灵活性，它可以根据需要决定是条件满足时执行循环体，还是一直执行循环体直到条件满足。Do循环有两种语法形式。

格式1：

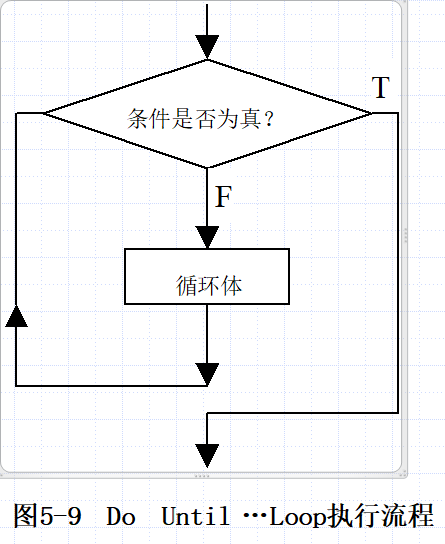
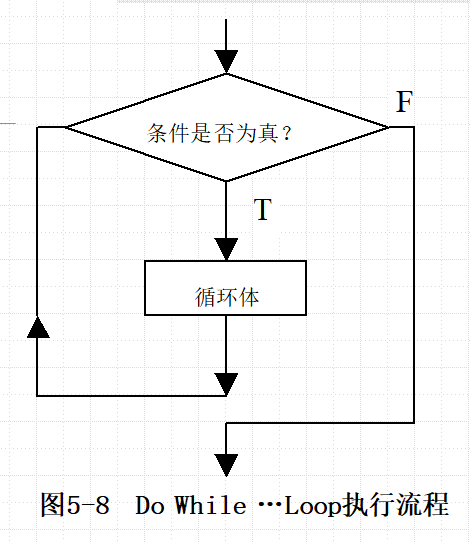
Do {while|until} <条件>

[<循环体>]

Loop

说明：

格式1是先判断，后执行。



格式2：

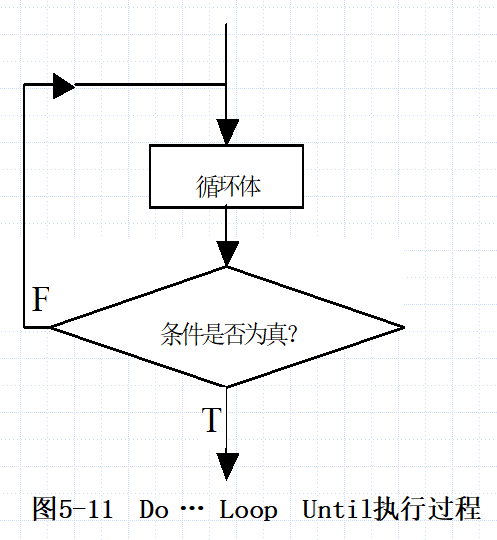
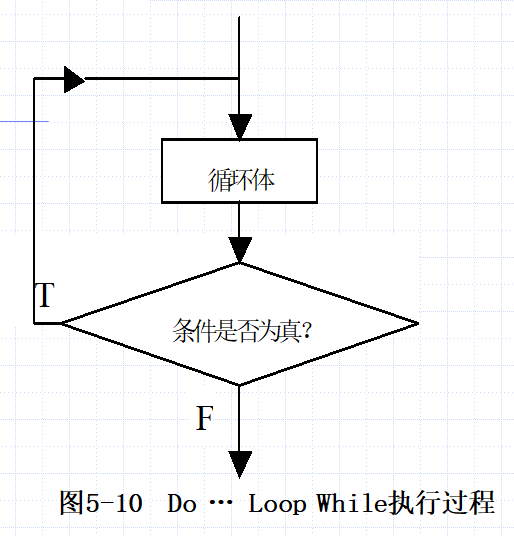
Do

[<循环体>]

Loop {while|until} <条件>

说明：

格式2是先执行，后判断。



值得注意的是，关键字While 用于指明条件成立时执行循环体，直到条件不成立时结束循环（如图5-8、5-10所示）；而Until则正好相反，条件不成立时执行循环体，直到条件满足才退出循环(如图5-9、5-11所示）。

【例5.6】求1+2+3+…+100。

我们用格式1实现，程序如下：

Private Sub Form\_Click()

Dim s%, I%

I = 1

s = 0

Do While I <= 100

s = s + I

I = I + 1

Loop

Print "s="; s

End Sub

【例5.7】求两自然数m，n的最大公约数。

设计思想：（1）m除以n得到余数r；

（2）若r=0，则n为要求的最大公约数，算法结束；否则执行（3）；

（3）n→m，r→n，再转到（1）执行。

程序如下：

Private Sub command1\_Click()

Dim m%, n%, r%

m = Val(Text1.Text)

n = Val(Text2.Text)

r = m Mod n

Do Until r = 0

m = n

n = r

r = m Mod n

Loop

Print " 它们的最大公约数是"; n

End Sub

循环的嵌套

在一个循环体内又包含了一个完整的循环，这样的结构称为多重循环或循环的嵌套。

在程序设计时，许多问题要用二重或多重循环才能解决。

我们前面学过的For循环、While循环、Do循环都可以互相嵌套。

二重循环的执行过程是外循环执行一次，内循环执行一遍，在内循环结束后，再进行下一次外循环，如此反复，直到外循环结束。

【例5.8】打印九九乘法表。

分析：打印九九乘法表，只要利用循环变量作为乘数和被乘数就可以方便的解决。

Private Sub Form\_Click()

Dim i%, j%, str$

Print Tab(35); "九九乘法表"

For i = 1 To 9

For j = 1 To 9

str = i & "×" & j & "=" & i \* j

Print Tab((j - 1) \* 9 + 1); str;

Next j

Print

Next i

End Sub



对于循环的嵌套，要注意以下事项：

（1）在多重循环中，各层循环的循环控制变量不能同名。但并列循环的循环控制变量名可以相同，也可以不同。

（2）外循环必须完全包含内循环，不能交叉。

下面的程序段都是错误的：

For I=1 To 100 For I=1 To 100

For J=1 To For I=1 To 10

…… ……

Next I Next I

Next J Next I

（a）内外循环交叉 （b）内外循环控制

变量同名

下面的程序段是正确的：

For i=1 to 100

For j=1 to 10

……

Next j

……

For j=100 to 120

……

Next j

Next i

（c）内循环并列的情况

【例 5.9】编写程序，输出100~1000之间所有的素数。

前面已经介绍过判断一个正整数是否为素数的方法。要找出100~1000所有素数，将这些数逐个用前面的方法测试就可以了。为了减少循环次数，可以将那些肯定不是素数的偶数排除。

程序如下：

Private Sub Form\_Click()

Dim i%, m%, flag%, n%

m = 101

n = 0

While m < 1000

flag = 0

For i = 2 To Sqr(m)

If m Mod i = 0 Then flag = 1

Next i

If flag = 0 Then

Print m;

n = n + 1

If n Mod 10 = 0 Then Print

End If

m = m + 2

Wend

End Sub

【例5.10】求



观察多项式就会发现，奇数项为正，偶数项为负，各项分子的指数与分母的阶乘数相同，各相邻指数相差为2.因此，可以设计一个二重循环，内层循环实现每项的计算，外层循环完成对各项的求和。

Private Sub command1\_Click()

Dim x#, n&, s#, i%, j%, k#, p#, f%

x = Val(Text1.Text)

n = Val(Text2.Text)

For i = 1 To n

p = 1: k = 1

For j = 1 To 2 \* i - 1

p = p \* j

k = k \* x

Next j

f = f \* (-1)

s = s + f \* k / p

Next i

Print "sin("; x; ")="; s

End Sub

循环的退出

在某些情况下，为了减少循环次数或便于程序调试，可能需要提前强制退出循环。VB为For…Next和Do…Loop循环语句提供了相应的强制退出循环的语句。

1．Exit For

用于For…Next循环，在循环体中可以出现一次或多次。当系统执行到该语句时，就强制退出当前循环。常见的使用方式是：

If 条件 Then Exit For

即当循环执行过程中满足某个条件时，就执行循环退出语句结束循环。

前面曾学习了一个判断用户输入的数是否为素数的例子(例5.2)，在这个例题中，无论用户输入的是否是素数，循环都是在循环变量超过终值后正常结束循环。然而仔细观察一下就会发现:如果用户输入的不是素数，往往没有必要把整个循环执行完。如用户输入15, int（sqr(5))=4, 循环变量终值为4,但当循环变量变化到3时，15 mod 3=0，由此就得知15为非素数，没有必要再将循环执行下去浪费时间，所以就可以在此时结束循环。而如果用户输入的数是素数，必然不会存在某个循环变量值能整除该数的情况，循环变量超过终值后正常结束循环。所以，可以在循环后通过对循环变量值的判断来断定用户输入的数是否为素数。

程序代码如下：

Private sb form\_click()

Dim n%,i%,k%

N=inputbox(“请输入一个正整数（≥3）”)

K=int(sqr(n))

For i=2 to k

If n mod i=0 then exit for

Next i

If i>k then

Print n;”是素数”

Else

Print n;”不是素数”

End if

End sub

2．Exit Do

用于Do…Loop循环，具体用法同Exit For一样。例如，我们要在1000~10000之间找一个既能被3整除又能被7整除的数，则可用下面的程序实现。

Private Sub Form\_Click()

Dim n%

n = 1000

Do While n <= 10000

If n Mod 3 = 0 And n Mod 7 = 0 Then

Print n

Exit Do

End If

n = n + 1

Loop

End Sub

# 17.数组的定义、初始化及基本应用；

在实际应用中，常常需要处理相同类型的一批数据。例如，为了处理100个员工的工资，可以用S(1)，S(2)，…，S(100)来分别代表每个员工的工资，其中S(1)代表第一个员工的工资，S(2)代表第二个员工的工资……。

在VB中，把一组相互关系密切的数据放在一起并用一个统一的名字作为标志，这就是数组。

数组中的每一个数据称为数组元素，用数组名和该数据在数组中的序号来标识。序号又称为下标，数组元素又称为下标变量。

例如，S(2)是一个数组元素，其中的S称为数组名，2是下标。在使用数组元素时，必须把下标放在一对紧跟在数组名之后的括号中。S(3)是一个数组元素，而S3是一个简单变量。

如果只用一个下标就能确定某个数组元素在数组中的位置，这样的数组称为一维数组。

如果用两个或多个下标才能确定某个数组元素在数组中的位置，则数组分别称为二维数组或多维数组。

一维数组的定义

数组应当先定义后使用，数组的定义又称为数组的声明或说明。

对于固定大小的一维数组，用如下格式进行定义：

说明符 数组名(下标) [As 类型]

例如：Dim y(5) As Integer

定义了一个一维数组，该数组的名字为y，类型为Integer，占据6个（0～5）整型变量的空间。

说明:

（1）“说明符”为保留字，可以为Dim，Public，Private，Static中的任意一个。在使用过程中可以根据实际情况进行选用。

定义数组后，数值数组中的全部元素都初始化为0，字符串数组中的全部元素都初始化为空字符串。

（2）“数组名”的命名遵守标识符规则。

（3）“下标”的一般形式为“[下界 to] 上界”。上界、下界为整数，并且下界应该小于上界。如果缺省“下界” ，下界默认为0。

例如：

Dim a(5) As Integer '定义a数组，含6个元素，下标值从0到5

Dim b(2 to 5) As Single '定义b数组，含4个元素，下标值从2到5

如果希望下界默认为1，则可以通过语句Option Base 1来设置。

Option Base 1语句只能出现在窗体级或模块级，不能出现在过程中，并且必须放在数组定义之前。

(4)要注意区分“可以使用的最大下标值”和“元素个数”。“可以使用的最大下标值”指的是下标值的上界，而“元素个数”则是指数组中成员的个数。

例如，在Dim a(5)中，数组可以使用的最大下标值是5。数组中的元素为：a(0)，a(1)，a(2)，a(3)，a(4)，a(5)，共有 6个元素。

（5）“As类型”用来说明“数组元素”的类型，可以是Integer，Long，Single，Double，Currency，String（定长或变长）等基本类型或用户定义的类型，也可以是Variant类型。如果省略“As类型”，则数组为Variant类型。

（6）在同一个过程中，数组名不能与变量名同名，否则会出错。

（7）可以通过类型说明符来指定数组的类型。

例如：Dim A%（5），B！（3 To 5），C#（12）

一维数组的引用

数组的引用通常是对数组元素的引用。

一维数组元素的表示形式为：

数组名(下标)

说明：

(1)下标可以是整型常量或整型表达

式。

（2）引用数组元素时，下标值应在数组声明的范围之内。否则将会出错。

（3）一般通过循环语句及InputBox函数给数组输入数据。数组的输出一般用Print方法、标签或文本框实现。

【例7.1】对输入的20个整数按每行5个元素格式输出。

Private Sub Command1\_Click()

Dim b(20) As Integer, i%

For i = 1 To 20

b(i) = InputBox("请输入一个整型数")

Next i

For i = 1 To 20

Print b(i);

If i Mod 5 = 0 Then Print

Next i

End Sub

【例7.2】编写程序，把输入的10个整数按逆序输出。

Private Sub Command1\_Click()

Dim a(10) As Integer, i%

Print "输入的数据为"

For i = 1 To 10

a(i) = InputBox("请输入一个整型数")

Print a(i),

Next i

Print

Print "逆序输出为"

For i = 10 To 1 step -1

Print a(i),

Next i

End Sub

一维数组的应用举例

【例7.3】从键盘上输入40人的考试成绩，输出高于平均成绩的分数。

分析：该问题可分三部分处理：一是输入40个人的成绩；二是求平均分；三是把这40个分数逐一和平均成绩进行比较，若高于平均成绩，则输出。

程序如下：

Private Sub Command1\_Click()

Dim score(40) As Single, aver!, i%

aver = 0

For i = 1 To 40

score(i) = InputBox("请输入成绩")

aver = aver + score(i)

Next i

aver = aver / 40

For i = 1 To 40

If score(i) > aver Then Print score(i)

Next i

End Sub

【例7.4】从键盘上输入10个整数，把这些数按由小到大的顺序排序输出。

排序的方法很多，如选择法、冒泡法等。这里介绍最容易理解的“比较排序法”。

设10个数存放在A数组中，分别为：A(1)、A(2)、A(3)、A(4)、A(5)、A(6)、A(7)、A(8)、A(9)、A(10)。

第1轮：先将A(1)与A(2)比较，若A(1)＞A(2)，则将A(1)、A(2)的值互换，否则，不作交换；这样处理后，A(1)一定是A(1)、A(2)中的较小者。

再将A(1)分别与A(3)、…、A(10)比较，并且依次作出同样的处理。最后，10个数中的最小者放入了A(1)中。第2轮：将A(2)分别与A(3)、…、A(10)比较，并依次作出同第1轮一样的处理。最后，第1轮余下的9个数中的最小者放入A(2)中，亦即A(2)是10个数中的第二小的数。

照此方法，继续进行第3轮…

…

直到第9轮后，余下的A(10)是10个数中的最大者。

至此，10个数已从小到大顺序存放在A(1)～A(10)中。

Private Sub Command1\_Click()

Dim t%, i%, j%, a(10) As Integer

For i = 1 To 10

a(i) = InputBox("输入一个整数")

Next i

Print "输入的10个整数为"

For i = 1 To 10

Print a(i),

Next i

Print

For i = 1 To 9

For j = i + 1 To 10

If a(i) > a(j) Then t = a(i): a(i) = a(j): a(j) = t

Next j

Next i

Print "排序后的结果为"

For i = 1 To 10

Print a(i),

Next i

End Sub

【例7.5】随机产生10个两位整数，找出其中最大值、最小值。

分析：该问题可以分为两部分处理：一是产生10个随机整数，并保存到一维数组中；二是对这10个整数求最大、最小值。

程序代码为：

Private Sub Command1\_Click()

Dim min%, max%, i%, a(10) As Integer

Randomize

For i = 1 To 10

a(i) = Int(Rnd \* 90) + 10

Next i

Print "产生的随机数为"

For i = 1 To 10

Print a(i),

Next i

Print

min = a(1): max = a(1)

For i = 2 To 10

If a(i) > max Then max = a(i)

If a(i) < min Then min = a(i)

Next i

Print "最大值为"

Print max

Print "最小值为"

Print min

End Sub

二维数组

假如有30个学生，每个学生有五门考试成绩，如何来表示这些数据呢？VB中可以用有两个下标的数组来表示，如第i个学生第j门课的成绩可以用S(i，j)表示。其中i表示学生号，称为行下标(i＝1，2，…，30）；j表示课程号，称为列下标(j＝1，2，3，4，5）。有两个下标的数组称为二维数组。

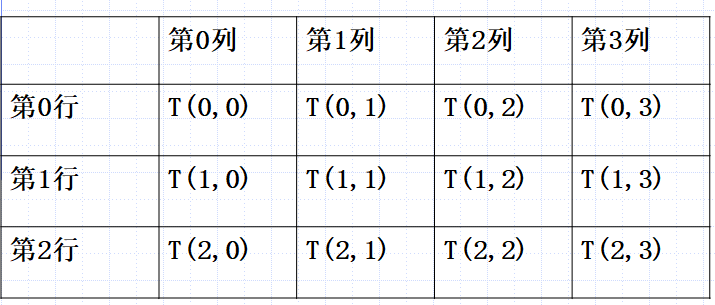
二维数组的定义

对于固定大小的二维数组，可以用如下格式进行定义：

说明符 数组名([下界to]上界，[下界to]上界) [As 类型]

例如： Dim T(2，3) As Integer

定义了一个二维数组，名字为T，类型为Integer，该数组有3行（0～2）4列（0～3），占据12（3×4）个整型变量的空间，如图7-3所示。



说明：

（1）可以将二维数组的定义方法推广至多维数组的定义。

（2）在实际使用时，可能需要数组的上界值和下界值，这可以通过Lbound函数和Ubound函数来求得。

其格式为：

Lbound(数组名[，维])

Ubound(数组名[，维])

这两个函数分别返回一个数组中指定维的下界和上界。

说明：

① 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。

② 对于一维数组来说，参数“维”可以省略。对于多维数组，则“维”不能省略。

例如： Dim A(1 To 10，0 To 5，-1 To 4）

定义了一个三维数组，用下面的语句可以得到该数组各维的上下界：

Print Lbound(A，1），Ubound(A，l）

Print Lbound(A，2），Ubound(A，2）

Print Lbound(A，3），Ubound(A，3）

输出结果为：

1 10

0 5

-1 4

二维数组的引用和一维数组基本相同，格式为：

数组名(下标1,下标2)

【例7.7】用二维数组输出数字方阵。

程序如下：

Private Sub Form\_Click()

Dim a(4, 4) As Integer, i%, j%

For i = 1 To 4

For j = 1 To 4

If i=j Then

a(i,j)=1

Else

a(i,j)=2

Endif

Next j

Next i

End Sub

For i = 1 To 4

For j = 1 To 4

Print a(i, j);

Next j

Print

Next i

End Sub

【例7.8】打印4名同学的英语、数学、法律3门课的考试成绩，并计算出每个同学的平均成绩。

分析：把4名同学的姓名及各科的考试分数分别存入一个一维字符串数组xm(4)和一个二维数值数组a(4,3)中，然后对数组（主要是二维数组）进行处理。

程序如下：

Private Sub Command1\_Click()

Dim a(4, 3) As Single, xm(4) As String \* 10, i%, j%, aver!

Print Tab(25); "成绩表"

Print

Print "姓名"; Tab(15); "英语"; Tab(25); "数学";

Print Tab(35); "法律"; Tab(45); "平均分"

Print

For i = 1 To 4

aver = 0

xm(i) = InputBox("输入姓名")

Print xm(i);

For j = 1 To 3

a(i, j) = InputBox("输入" & xm(i) & "的一个成绩 ")

aver = aver + a(i, j)

Next j

aver = aver / 3

Print Tab(15); a(i, 1); Tab(25); a(i, 2);

Print Tab(35); a(i, 3); Tab(45); aver

Print

Next i

End Sub

可调数组

在实际应用中，有时事先无法确定到底需要多大的数组，如果定义的数组“足够大”，显然又会造成内存空间的浪费。

可调数组提供了一种灵活有效的管理内存机制，能够在程序运行期间可以根据用户的需要随时改变数组的大小。

可调数组的定义分为两步：

第一步：声明一个没有下标参数的数组。

格式为：

说明符 数组名( ) [As 类型]

第二步：引用数组前用 ReDim语句重新定义。

格式为：

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

例如：

Private Sub Command1\_Click()

Dim a() As Integer

Dim n%

……

n = Val(InputBox("input n"))

ReDim a(n)

……

End Sub

说明：

（1）格式中的“说明符”、“ 数组名”、“ 类型”等说明同一维数组的定义。

（2）下界、上界可以是常量，也可以是有了确定值的变量。

（3）ReDim语句用来重新定义数组，能改变数组的维数及上、下界，但不能用其改变可调数组的数据类型，除非可调数组被声明为Variant类型。

（4） 每次使用ReDim语句都会使原来数组中的值丢失，可以在ReDim后加Preserve参数来保留数组中的数据，但Preserve只能用改变多维数组中最后一维的大小，前几维的大小不能改变。

（5）ReDim语句只能出现在过程中。

【例7.10】编程输出Fibonacci数列：l，1，2，3，5，8，……的前n项。

分析：在例7.6中，是输出Fibonacci数列的前20项，我们使用了固定数组；本例要求输出前n项，n是一个变量，因此，应该使用可调数组。

Private Sub Command1\_Click()

Dim Fib() , i%, n% ‘ 避免溢出，定义数组为Variant类型

n = InputBox("输入n的值(n>1)")

ReDim Fib(n)

Fib(1) = 1: Fib(2) = 1

For i = 3 To n

Fib(i) = Fib(i - 1) + Fib(i - 2)

Next i

For i = 1 To n

Print Fib(i),

If i Mod 5 = 0 Then Print '每行输出5个数

Next i

End Sub

控件数组

控件数组中每一个元素都是控件，它的定义方式与普通数组不同。可以通过以下两种方法建立控件数组。

方法一：复制已有的控件并将其粘贴到窗体上。

方法二：将窗体上已有的类型相同的多个控件的Name属性设置为同一值。

【例7.11】建立含有三个命令按钮的控件数组，单击第一个按钮，在窗体上画圆；单击第二个按钮，在窗体上画矩形；单击第三个按钮则退出。

设计界面：

在窗体上建立三个按钮，三个按钮的Name属性值均设置为“Command1”，Caption 属性分别设置为“画圆形”、“画矩形”、“退出”。

程序代码如下：

Private Sub Command1\_Click(Index As Integer)

If Index = 0 Then

Circle (1500, 1500), 800 ' 画圆形

ElseIf Index = 1 Then

Line (500, 500)-Step(1000, 1000), , B ' 画矩形

Else

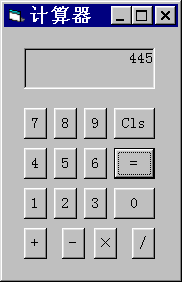
End ' 退出

End If

End Sub

上述程序根据Index的属性值决定在单击某个按钮时所执行的操作。

【例7.12】设计一个简易计算器，能进行整数的加、减、乘、除运算。其运行界面如图7-8所示。



界面设计：

一个标签用于计算器输出；

数字按钮控件数组Number；

操作符控件数组Operator；

一个“=”按钮用于计算结果；

一个“Cls”按钮用于清屏。

界面设计的详细情况见表7-1。



程序如下：

' 窗体级变量声明

Dim op1 As Byte ' 用来记录前面输入的操作符

Dim ops1&, ops2& ' 两个操作数

Dim res As Boolean ' 用来表示是否已算出结果

Private Sub clear\_Click()

dataout.Caption = ""

End Sub

Private Sub Form\_Load()

res = False

End Sub

按下数字键0~9的事件过程

Private Sub number\_Click(i1 As Integer)

If Not res Then

dataout.Caption = dataout.Caption & i1

Else

dataout.Caption = i1

res = False

End If

End Sub

按下操作键+、－、×、/的事件过程

Private Sub operator\_Click(i2 As Integer)

ops1 = dataout.Caption

op1 = i2 ' 记下对应的操作符

dataout.Caption = ""

End Sub

' 按下=键的事件过程

Private Sub result\_Click()

ops2 =dataout.Caption

Select Case op1

Case 0

dataout.Caption = ops1 + ops2

Case 1

dataout.Caption = ops1 - ops2

Case 2

dataout.Caption = ops1 \* ops2

Case 3

dataout.Caption = ops1 / ops2

End Select

res = True ' 已算出结果

End Sub

# 18.标签框、文本框、命令按钮、复选框、单选框、列表框、计时器等常用组件的应用；

一、窗体的主要属性

1.Caption 属性

设置窗体的标题内容。标题内容应概括说明本窗体作用。

2.MaxButton 属性和 MinButton 属性

MaxButton为True，窗体右上角有最大化按钮；为False时，无最大化按钮。

MinButton 属性为True，窗体右上角有最小化按钮；为False时，无最小化按钮。

3.BorderStyle 属性

决定窗体边框式样及窗体是否能调整大小。

4.BackColor属性和ForeColor 属性

BackColor属性设置窗体的背景颜色；ForColor 属性设置窗体的前景颜色。窗体的前景颜色是执行Print方法时所显示文本的颜色。

5.Height属性和Width属性

设置窗体的初始高度和宽度。其单位为Twip。1Twip=1/20点=1/1440英寸=1/567厘米。

6.Left属性和Top属性

用于设置窗体左边框距屏幕左边界的距离和窗体顶边距屏幕顶端的距离。其单位为Twip。

7.Name（名称）属性

设置窗体的名称，在程序代码中用这个名称引用该窗体。新建工程时，窗体的名称缺省为 Form1；添加第二个窗体，其名称缺省为 Form2，以此类推。为了便于识别，用户通常给 Name 属性设置一个有实际意义的名称。

8.Enabled属性

属性值为True或False，设置对象是否能够对用户产生的事件做出反应。一般在程序中设置，用于临时屏蔽对窗体或其他控件的的控制。

9.Moveable属性

属性值为True或False，设置是否可以移动窗体。

10.Visible属性

属性值为True或False，设置窗体是否被显示。用户可用该属性在程序代码中控制窗体的隐现。

11.Picture属性

设置在窗体中显示的图片。单击Picture属性右边的按钮，弹出“加载图片”对话框，用户可选择一个图片文件作为窗体的背景图片。若在程序中设置该属性的值，需要使用LoadPicture函数。

12.WindowState属性

设置窗体启动后的大小状态。它有三个可选值：

0-Normal: 窗体大小由Height和Width属性决定。

1-Minimized: 窗体最小化成图标。

2-Maximized: 窗体最大化，充满整个屏幕。

二、窗体常用事件

窗体最常用的事件有三种：Click（单击）、DbClick（双击）、Load（装入）。

1.Click事件

程序运行后，单击窗体触发该事件。

2.DbClick事件

程序运行后，双击窗体触发该事件。

3.Load事件

Load事件是窗体被装入内存工作区时触发的事件。如果这个事件过程存在，就马上执行它。Load事件过程通常用于启动程序时对属性、变量的初始化，装载数据等。

命令按钮

在VB应用程序中，命令按钮是使用的最多的对象之一，常常用它接受用户的操作信息，触发相应的事件过程，以实现指定的功能。

一、属性

1.Caption属性

设定命令按钮上显示的文本。

2.Default属性

该属性用于设置默认命令按钮。当Default属性设置为True时，按ENTER键相当于用鼠标单击了该按钮。

3.Style 属性和Picture属性

命令按钮上除了可以显示文字外，还可以显示图形。

若要显示图形，首先应将Style 属性设置为1，然后在Picture属性中设置要显示的图形文件。类似的，若要设置命令按钮的BackColor（背景色），也应将Style 属性设置为1。

Style 属性可设置为：

0-Standard： 标准的，命令按钮上不能显示图形。

1-Graphical：图形的，命令按钮上可以显示图形，也可以显示文字。

4.Value属性

该属性只能在程序运行期间引用或设置。True表示被按下，False（默认）表示未被按下。在代码中可通过设置Value属性为True，来触发命令按钮的Click事件。例如，利用下面代码，可通过程序来选择命令按钮，并触发命令按钮的Click事件。

Command2.Value=True

二、事件

最常用的事件是Click事件。

标签

标签（Label）主要用于显示不需要用户修改的文本。所以，标签可以用来标示窗体及窗体上的对象，如为文本框、列表框等添加描述性的文字，或者作为窗体的说明文字。

一、属性

1.Caption属性

设置标签要显示的内容。它是标签的主要属性。

2.BorderStyle属性

默认值为0，标签无边框；设置为1时，标签有立体边框。

3.Autosize属性

该属性用于设置标签是否自动改变尺寸以适应其内容。设置为True时，随着Caption的内容变化，自动调整标签的大小，并且不换行；设置为False时，标签保持设计时的大小，这时如果内容太长，只能显示一部分。默认值为False。

4.Alignment属性

确定标签中内容的对齐方式，有三种可选值：

0-Left Juseify：默认值，左对齐。

1-Right Juseify：右对齐。

2-Center：居中对齐。

5.BackStyle属性

该属性用于设置背景是否透明。默认值为1，不透明；设为0时，透明。所谓透明，是指无背景色。

二、事件

常用Click，DbClick事件。

文本框

文本框（TextBox）在窗体中为用户提供一个既能显示文本又能编辑文本的区域

一、属性

1.Text属性

设置文本框中显示的内容。它是文本框最主要的属性。

2.Locked属性

设置文本框中的内容是否可编辑。

默认值为False，表示可编辑，当设置为True时，不可编辑，此时文本框的作用相当于标签。

3.Maxlength属性

设置文本框中允许输入的最大字符数。

如果输入的字符数超过Maxlength设定的数目后，系统将不接受超出部分的字符，并发出嘟嘟声。该属性默认值为0，表示无限制。

4.MultiLine属性

决定文本框是否允许接收多行文本。

若设置为True，文本框可接收多行文本，当输入的文本超出文本框的边界时，会自动换行。默认值为False，文本框中只能输入一行文本。

5.PassWordChar属性

当MultiLine为False时，该属性可设置显示在文本框中的替代符。

例如，PassWordChar设置为“\*”，那么无论用户输入什么字符，文本框中显示的只是“\*”，但文本框接收的还是用户实际输入的字符。设置该属性主要用于输入口令。

6.ScrollBars属性

决定文本框中是否有滚动条。

只有当MultiLine属性为True时，文本框才能加滚动条。

二、事件

文本框除支持Click，DbClick事件，常用的还有Change，LostFocus事件。

1. Change事件

当用户输入新内容，或程序对文本框的Text属性重新赋值，从而改变文本框的Text属性时触发该事件。

2. LostFocus事件

当用户按下<Tab>键时光标离开文本框，或用鼠标选择其他对象时触发该事件，称为“失去焦点”事件。

焦点是对象接收用户鼠标或键盘输入的能力。当对象具有焦点时，可接收用户的输入。通常用该事件过程对文本框中的内容进行检查和确认。

三、方法

文本框最常用的方法是SetFocus，使用该方法可把光标移到指定的文本框中，使之获得焦点。当使用多个文本框时，用该方法可把光标移到所需要的的文本框中。

其使用格式为：

对象.SetFocus

【框架】控件属性Frame

和窗体一样，框架也可以作为一种容器类控件，可以向框架中添加其他控件，具体的添加方法是：先在工具箱中单击控件图标，然后在框架上按住，拖动鼠标，使可以向框架中添加控件。如果按住鼠标左键的位置不在框架上，则是向窗体中添加控件。

向框架添加控件之后，框架中的控件随着框架的移动而移动，如果框架被删除，则框架中的控件也被删除。

【框架】控件除了可以盛装控件，还起标识的作用，所标识的文本是通过【Caption】属性来设置的。

除了【Caption】、【Font】等共有属性外，【框架】控件还有【BorderStyle】属性。

【BorderStyle】属性

功能：返回或设置【标签】控件的边框样式。

说明：【BorderStyle】属性有两个取值0或1. 【BorderStyle】属性为0时（默认值），表示【框架】控件无边框；【BorderStyle】属性为1时，表示【框架】控件有固定的单线边框。

由于【框架】控件主要起标识作用，因此在设计程序时，很少为其添加事件。

单选按钮OptionButton和复选框CheckBox

单选按钮和复选框组出现，用来向用户提供选择。在一组【单选按钮】控件中，用户只能选其中的一个单选按钮，但在一组【复选框】控件中，用户可以同时选中多个复选框。

单击某个单选按钮，则该单选按钮被选中；单击其他单选按钮，则该按钮不被选中；单击某个复选框，则该复选框被选中；再次单击复选框，则该复选框不被选中。

【单选按钮】控件上所显示的文本是由【Caption】属性来设置的，所处的状态是由【Value】属性来获得的。【Value】属性主要用来设置或返回【单选按钮】控件的状态，它有两个取值True或False。【Value】属性为True时，表示该单选按钮被选中；【Value】属性为False时，表示该单选按钮未被选中。

和单选按钮一样，复选框上所显示的文本由【Caption】属性来设定，【复选框】控件的状态由【Value】属性来返回或设置，有3个取值0、1或2【Value】属性为0时，表示为2时，表示复选框不可用，此时复选框呈灰色显示。

在窗体上添加多个【单选按钮】控件时，所有的单选按钮都被归为1组，因此用户只能选中多个单选按钮中的1个。如果想同时选中多个单选按钮，则必须使用【框架】控件将单选按钮分组。单选按钮被分组后，用户便可以在每组中选中1个单选按钮。

多窗体

对于一些简单的程序而言，单一的一个窗体便能满足所有的要求。但在编写具体的应用程序时，单一的一个窗体是不能满足要求的，势必要用到1个以上的窗体。例如，一个学生信息管理系统，除了有用于显示学生信息的对话框之外，还应有学生基本信息输入的对话框。

多窗体是指包含多个窗体界面，这些窗体之间没有绝对的从属关系，每个窗体的地位都是平等的，它们之间只存在相互调用的关系，各个窗体出现的顺序也有所不同。

在多窗体应用程序中,启动窗体在程序运行时会自动被显示出来,而其他的窗体的显示必须通过Show方法来实现。用Show方法来显示窗体的语法结构如下

窗体名.Show [style]

其中“ style”参数用来指定窗体的模式,为可选参数,它的值为0或1。为1时,表示窗体是模态的,即用户只能操作所显示的窗体,而不能操作其他窗体;为0时,表示窗体是非模态的,即用户既能操作所显示的窗体,又可以操作其他窗体。例如,可以通过下面的代码来显示窗体名为“Form2”的窗体。

Form2.Show 1

在多窗体中,要将某个窗体隐藏起来,可以通过Hide方法或 Unload方法。用Hide方法和 Unload方法隐藏窗体的语法结构分别为:

窗体名.Hide

Unload 窗体名

【组合框】控件属性ComboBox

【列表框】控件和【组合框】控件都是列表选择型控件，主要是用来向用户提供一系列的列表项目，用户可以从这些列表项目中选择自己所需的项目。

【组合框】控件以下拉列表或组合列表的形式向用户提供一系列项目，它兼有【列表框】控件和【文本框】控件的功能。用户可以在列表框部分选择所需的项目，也可以在文本框部分输入所需的项目。在工具箱中，【组合框】控件的图标为

。

【组合框】控件的常用属性如下。

 【Style】属性

功能：返回或设置组合框的样式。

说明：Style属性有3种取值0，1或2。Style属性值为0时，表示组合框的样式为组合下拉式，如图5-13所示，用户通常是看不到所有列表项的，只有通过单击右端的箭头才可以看到全部的列表项，在这种样式下用户既可以在文本框部分输入列表项，也可以在下拉列表框部分选择列表项；【style】属性值为 1 时，表示组合框的样式为组合式，如图5-14所示，用户既可以在文本框中输入列表项，也可以在列表框中选则列表项；【style】属性值为2时，表示组合框的样式为简单下拉式，如图5-15所示，在样式上和组合式没什么区别，但在这种样式下，用户不能在文本框中输入列表项。

 【Text】属性

功能：返回或设置组合框被选中的列表项。

说明：如果列表项是在文本框中直接输入的，则【Text】属性返回的是在文本框中输入的列表项；如果列表项是列表框中选择的，则【Text】属性返回的是在列表项中选定的列表项。

 【List】属性

功能：返回或设置列表框中某一列表项的内容。

说明：【List】属性是一个字符串类型的数组，列表框中所有的列表项都被保存在该数组中，因此要访问或设置列表框中的某一项时，必须按以下语法结构来访问：

列表框控件名.List(列表项的索引值)[=字符串表达式]

设置【List】属性时，每输入一个列表项，按Enter键之后，才能输入下一个列表项。

【列表框】控件属性ListBox

【列表框】控件以列表的形式向用户提供一系列项目，如图5-12所示，用户可以从中选择一个或多个项目。列表项的输入同样也是通过【List】属性来完成。用户在列表框中单击某一个项目时，该项目会以蓝色光条的形式显示，表示该项目被选中。当列表框中的项目超出了列表框所能表示的范围时，在列表框中便会自动增加1个垂直滚动条，便于用户进行上下翻动。控件图标为 。

列表框的其他常用属性如下:

 【Text】

功能:返回【列表框】控件中最后被选中的列表项。

说明:【Text】属性是不显示在【属性】窗口里的，用户只能通过代码来访问该属性， 其语法结构如下:

[字符串变量]=列表框控件名.Text

 【MultiSelect】属性

功能:设置用户是否可以在【列表框】控件中选择多个列表项。

说明:【MultiSelect】属性有3个取值0、1或2。【MultiSelect】属性为0时，表示只能选 择一项;【MultiSelect】属性为1时，表示允许用户进行多项选择，在进行多项选择时，用户 只需单击所要选择的各个项即可，如果某项已被选中，再单击该项时，该项将被取消选中;【MultiSelect】属性为2时，也表示允许用户可以进行多项选择，但用户在进行多项选择时， 必须同时按住Shift键，同样如果某项已被选中，再单击该项时，则该项将被取消选中。

由于组合框、列表框都是向用户提供选择列表项的控件，除了共有【List】属性之外， 还有以下与列表有关的属性。

 【ListCount】属性

功能:返回【列表框】控件中所有项目的个数。

说明:【ListCount】属性不显示在【属性】窗口里，用户只能通过代码来访问该属性， 其语法结构如下:

[整型变量=]列表框控件名.ListCount

 【ListIndex】属性

功能:返回或设置当前被选中列表项的索引值。

说明:【ListIndex】属性不显示在【属性】窗口里，用户只能通过代码来访问或设置该属性，其语法结构如下：

列表控件名.ListIndex[=索引值]

 【Sorted】属性

功能：返回或设置【列表框】控件的列表项是否按字母升序来排列。

说明：【Sorted】属性有两个取值True或False。【Sorted】属性值为True时，表示按字母升序排列列表项；【Sorted】属性值为False时（默认值），表示按列表加入时的默认顺序排列列表。

列表框、组合框中的列表项除了可以通过设置【List】属性来完成外,还可以使用AddItem方法向列表框、组合框中增加列表项,其语法结构如下：

列表框控件名或组合框控件名.AddItem 字符串变量或表达式,[索引值]

如果省略“索引值”,则列表项总是添加到列表框或组合框的最后；如果指定“索引值”,则在所指定的位置添加列表项,并将该位置以后的列表项都向后移动1个位置。

如果想删除列表框、组合框中的列表项,则可以使用 RemoveItem方法,其语法结构如下：

列表框控件名.RemoveItem列表项索引值

单击列表框中的某一个列表项时,便会激发【列表框】控件的Click常用事件。组合框能响应的事件与【style】属性有关,当【Style】属性值为0或1时,如果直接在文本框中输入列表项或通过代码设置了【Text】属性,便会激发 Change事件；而当【 Style】属性值为2时,则不能响应 Change事件；当【Style】属性值为1时,如果在列表框中双击列表项,则会激发 DblClick事件,而在另外两种样式下,组合框不能响应 DblClick事件；当【style】属性值为0或2时,如果单击下拉箭头,则会激发 DropDown事件,而 Style属性值为1时,不能响应该事件。

焦点事件

焦点事件是Visual basic中常用的共有事件。焦点用于表示控件具有接受输入的能力。只有当控件具有焦点时,控件才可以被激活,才可以响应事件。在活动的窗体中,任何时刻都只能有1个控件具有焦点,并且只有当控件的【 Enabled】和【 Visible】属性都为True时,才能获得焦点。当控件获得焦点之后,便会以特殊的外观显示出来。例如,当命令按钮、单选按钮或复选框获得焦点之后,便会在控件之上显示1个虚的方框；当文本框获得焦点之后,便会在文本框中出现1个闪动的光标。使控件获得焦点最简单的方式便是直接单击该控件,除了这种方式之外,控件还可以通过按Tab键来获得焦点。连续按Tab键,焦点就会按控件添加的默认顺序在各个控件之间切换。

当控件获得焦点之后,便会激发 GotFocus事件；反之,当控件失去焦点便会激发LostFocus事件。焦点事件的语法结构如下。

 GotFocus事件

Private sub 控件名\_GotFocus()

End Sub

 LostFocus事件

Private sub 控件名\_LostFocus()

End Sub

滚动条控件

滚动条一般附在对话框的边缘,用来滚动对话框,以方便查看数据。在Vsual Basic6.0中,除了这项功能之外,滚动条还常常用来进行数据的输入。 Visual basic6.0为用户提供了两种样式的滚动条,一种为水平滚动条,另一种为垂直滚动条。

滚动条常用属性如下。

【Max】属性、【Min】属性

功能:返回或设置滚动条所能表示的范围,【Max】属性用于设置最大值,【Min】属性用于设置最小值。

说明:【Max】属性表示的是当滚动框处于最右端或最下端时,滚动条所对应的值；【Min】属性表示的是当滚动框处于最左端或最上端时,滚动条所对应的值,并且【Mn】属性值不能大于【Max】属性值。

【 SmallChange】属性、【 LargeChange】属性

功能:返回或设置滚动框每次滚动的范围,即滚动框每次移动的距离。

说明:【 SmallChange】属性用于设置滚动框每次移动的最小距离；【 Large Change】属性用于设置滚动框每次移动的最大距离。当单击滚动条两端的箭头时,滚动框便按【 SmallChange】属性所设定的值滚动；当单击滚动条的空白处时,滚动框便按【 Large Change】属性所设定的值滚动。

当滚动框处于不同的位置时,滚动条所表示的值也不一样,具体所表示的值由【value】属性返回或设置。当滚动框处于最左端或最上端时,【value】属性的值为【Mim】属性所设定的最小值；当滚动框处于最右端或最下端时,【 value】属性值为【Max】属性所设定的最大值；当滚动框处于其他位置时,【 Value】属性值介于最大值和最小值之间。当单击滚动条两端的箭头时,【value】属性值便按【 Smallchange】属性所设定的最小改变量递增或递减；当单击滚动条的空白处时,【 Value】属性值便按【 Large Change】属性所设定的最大改变量递增或递减。

定时器控件

【定时器】控件是专门用于计时的控件,它的大小是不可以改变的,并且在程序运行时是不可见的。【定时器】控件的属性只有7种,其中最为重要的属性为【 Interval】(事件间隔)属性。【 Interval】属性以毫秒(即0.001s)为单位,它决定着每隔多长时间激发1次计时事件。当【 Interval】属性所设定的时间(大于0的值)一旦达到,且定时器的【 Enable】属性为True,系统便会自动激发定时器的 Timer唯一事件；如果【 Interval】属性被设为0则定时器不起作用,因此也就不能激发定时器的 Timer事件了。

# 19.常用算法的实现：累加、累乘、求平均、求最大/最小值、条件计数、选择排序、冒泡排序、直接排序、顺序查找等。