第 3 章　VB 语言基础

上一章介绍了如何建立一个简单的 VB 应用程序。但是如果想编写稍微复杂的程序，还需要进一步学习 VB 语言的基础知识。本章将介绍 VB 语言的基础知识，这些知识比较零散、琐碎，但它们是今后编写程序代码的基础，因此需要投入一些时间来学习本章内容。为了系统化地掌握、灵活地运用 VB 语言的基础知识，本章将主要阐述数据类型、常量和变量、常用的内部函数、运算符表达式和编码规则等内容。

3.1　数 据 类 型

在使用 VB 程序处理问题时，会遇到各种不同类型的数据，例如数值型数据、字符型数据、货币型数据等。不同类型数据的取值范围、所应用的运算不同，在内存中所分配的存储单元数目也不同。因此，正确地区分和使用不同类型的数据，可以减少程序运行时所占用的内存，从而提高程序运行时的正确性和可靠性。VB 将数据分成两大类：标准数据类型和用户自定义数据类型。

3.1.1　标准数据类型

标准数据类型是 VB 系统定义的数据类型。VB 提供的标准数据类型共有七大类：字节型、数值型、字符型、布尔型、日期型、对象型和变体型。不同类型的数据有不同的表示方法、操作方式和取值范围。VB 中各种标准数据类型所占存储空间大小与取值范围的说明如表 3.1 所示。

对于 VB 中的数据，首选应该确定以下几个问题：

- 数据为何种类型；
- 数据的取值范围；
- 数据在内存中的存储形式、占用字节数；
- 数据能参与的运算方式。

1. 字节型

字节型(Byte)数据用于保存程序中的二进制数据信息，数据范围为 0～255。当使用字节型变量进行整数运算时，应注意计算结果的范围，采用适当的数据类型以避免溢出错误的发生。字节型变量通常使用布尔操作符进行运算，如与(AND)、或(OR)、非(NOT)和异或(XOR)等。字节型数据通常用于文件加密。

表 3.1 VB 的标准数据类型表

数据类型	关键字	占用字节数	类型符	前缀	范　围
字节型	Byte	1		Byt	0～255
逻辑型	Boolean	2		Bln	True 或 False
整型	Integer	2	%	Int	－32 768～32 767
长整型	Long	4	&	Lng	－2 147 483 648～2 147 483 647
单精度型	Single	4	!	Sng	－3.402823E38～－1.4011298E－45；1.401298E－45～3.402823E38
双精度型	Double	8	#	Dbl	±4.94D－324～±1.79D308
货币型	Currency	8	@	Cur	－922 337 203 685 477.5808～922 337 203 685 477.5807
日期型	Date	8		Dtm	1/1/100～12/31/9999
对象型	Object	4		Obj	任何对象引用
字符型	String	与字符串长度有关	$	Str	定长：0～65 535 个字符
					变长：0～约 20 亿个字符
变体型	Variant	按需分配		Vnt	上述有效范围之一

2. 数值型数据

数值型数据用于表示保存程序中所使用的数值信息，可细分为整型数、实型数和货币型数。其中整型数又包括整型(Integer)和长整型(Long)；实型数也称为浮点型数，包括单精度型(Single)和双精度型(Double)。

(1) 整型(Integer)和长整型(Long)

整型数据和长整型数据都是指不带有小数部分的数，它们可以表示正整数、负整数和零。整型数据和长整型数据的区别在于：整型数据占用的内存数少，表示的数值范围小。其优势在于运算速度快、精确。

(2) 单精度型(Single)和双精度型(Double)

单精度型数据和双精度型数据都可以表示带有小数部分的数，这两种数据类型的差别在于：单精度型数据可以精确到 7 位有效数字，而双精度型数据可以精确到 15 位有效数字。由此可见双精度型数据表示数的范围大、误差小，但缺点是占用内存字节数大、运算速度慢。

单精度型数据可用指数形式(科学计数法)来表示，即写成以 10 为底的指数形式，数值与指数之间使用符号(E 或 e)，例如，4.35×10^{9} 表示为 4.35E＋9，5.29×10^{-10} 表示为 5.29E－10。

双精度型数据也可用指数形式(科学计数法)来表示，数值与指数之间使用符号(D 或 d)，例如，4.17×10^{23} 表示为 4.17D＋23，-5.689×10^{-13} 表示为－5.689D－13。

(3) 货币型(Currency)

货币型数据是一种专门为处理货币而设计的数据类型，用于表示定点实数，最多保留

小数点左边 15 位数字和小数点右边 4 位数字。

所有数值型的数据都有一个有效的范围值，程序中的数据如果超出规定的范围，就会出现“溢出”。如果小于范围的下限值，系统将按 0 处理；如果大于范围的上限值，则系统只按上限值处理，并显示出错误信息。

3. 字符型数据

字符型(String)数据用于保存程序中使用的文本信息。这些信息是指用双引号("")括起来的一组字符串，因此字符型数据也称为字符串，例如"Sony"、"Visual Basic"。

字符串中包含的字符个数称为字符串的长度。不含任何字符(长度为 0)的字符串称为空字符串。例如""表示空字符串，而" "表示有一个空格的字符串。

在 VB 中，字符串分为变长字符串和定长字符串。变长字符串的长度不固定，随着对字符串变量的赋值，字符串的长度可变。变长字符串最多可以包含 2^{31} 个字符。一个字符串如果没有定义成固定长度的，默认为变长字符串。定长字符串的长度保持不变，如果赋值给字符串的字符数少于字符串的长度，则用空格填满不足部分；若超过字符串的长度，超出部分字符被截去。定长字符串最多可包含 65535 个字符。

4. 布尔型数据

布尔型数据(Boolean)也称为逻辑型数据，它只有 True 与 False 两个值，在程序中常用于表示逻辑判断的结果。当布尔型数据转换成数值型数据时，True 转换为－1，False 转换为 0。当数值型数据转换成逻辑型数据时，非 0 转换为 True，0 转换为 False。

布尔型数据和字节型数据一样都可使用布尔运算符，例如，与(AND)、或(OR)、非(NOT)和异或(XOR)等来进行运算。

5. 日期型数据

日期型(Date)数据用于保存程序中的日期和时间，通常采用两个＃符号把表示日期和时间的字符串括起来，就像字符串需要用双引号括起来一样。例如，＃2012-4-11＃和＃16:30:52＃分别表示日期和时间。日期型数据可以表示的日期范围为：公元 100 年 1 月 1 日到公元 9999 年 12 月 31 日；时间范围为：00:00:00 到 23:59:59。VB 可以接受多种表示形式的日期和时间，只要任何字面上可被认作日期和时间的字符都是合法的。赋值时如果输入的日期或时间是非法的或不存在的，系统将提示出错。

例如，以下赋值语句都是正确的：

```
Dim TestDate As Date
TestDate=#10/29/2012#
TestDate=#2012-10-29#
TestDate=#10/29/2012 9:47:29 pm#
```

6. 对象型数据

对象型数据(Object)可用来引用应用程序中的对象。使用前需要使用 Set 语句指定一个被声明为 Object 的变量，然后才能去引用应用程序所识别的任何实际对象。例如：

```
Dim objDb As Object
```

```
Set objDb=OpenDatabase("c:\Vb6\student.mdb")
```

7. 变体型数据

变体型数据(Variant)是一种可变的数据类型,可以存放任何类型的数据。当指定变量为 Variant 变量时,不必在数据类型之间转换,VB 会自动完成任何必要的转换。在程序中如不特别说明数据的类型时,VB 会默认它为变体型数据。它为 VB 的数据处理增加了智能性,是所有未定义的变量的默认数据类型,它对数据的处理取决于程序上下文的需要。

例如:

```
x="40"                  'x 的值是"40"(包含两个字符的字符串)
x=x-1                   '转换为数值运算,x 的值是 39
x=#01-07-2012#          '赋值一个日期
```

注意:

(1) 不同类型的数据所占的内存空间是不同的,因此选择使用合适的数据类型可以节省内存空间,而且还可以提高运行速度。例如,表示人的年龄(0～200),可以使用整型数据(Integer),这样比起采用长整型、单精度型或双精度型占用内存较少,可提高运算速度。

(2) 可在数据之后加一个类型符(见表 3.1)来标识该数据的类型,例如 52%、12.34@、87.5!等。

3.1.2 用户自定义数据类型

VB 系统除了提供标准数据类型之外,还允许用户根据不同需求自定义数据类型,这种数据类型是由一个或者多个标准数据类型的数据元素组成。

例如,在一个管理学生的软件系统中,一个学生通常要有许多信息,如学生的“姓名”、“性别”、“年龄”、“电话”、“所在院系”、“出生日期”等。为了有效地管理这些数据,常常需要把这些数据定义成一个新的数据类型。在 VB 中,可以用 Type 语句来让用户自己定义数据类型。这部分内容在第 5 章将进行详细讲解。

3.2 常量和变量

前一节介绍了 VB 中所使用的数据类型,这些数据类型既可以以常量的形式出现,也可以通过变量的形式出现。常量和变量都可以实现应用程序中数据的存储与交换,常量用来保存固定的数据,这些数据是程序要经常使用的,不会随着程序的运行而发生变化;变量则用于存储临时性的数据,这些数据可以在程序运行的过程中发生改变。

3.2.1 常量

在 VB 程序运行过程中,其值不能被改变的量称为常量。通过声明和使用常量的标识符,代替一个在程序执行时不会改变的值,能增强程序的可读性,使程序的维护变得简

单。在 VB 中有三类常量：普通常量、符号常量和系统常量。

1. 普通常量

普通常量也称为直接常量，它是在程序代码中直接给出的数据。普通常量的数据类型有：字符串常量、数值常量、日期/时间常量和逻辑常量。

(1) 字符串常量

字符串常量由一组字符串组成，是由除回车符和双引号以外的任何 ASCII 码字符和汉字组成。其内容必须用双引号括起来，如"A"、"12.3"、""、"True"、"10/08/2010"、"Visual Basic"等。

(2) 数值常量

数值常量是由 0～9、小数点以及符号位组成的常数。例如，－8、123456、0、3.141593、123.45、－100.05、7.23E＋10、－5.643D＋10、0.5E－24、－0.53E＋8。其中后 4 个例子是用科学计数法表示的。

在 VB 中除了十进制数外，还允许使用八进制数和十六进制数。八进制数由数字 0～7 组成，并以 &O 开头。十六进制数由数字 0～9 和字母 A～F 组成，并以 &H 开头。例如，&O123、&O345、&H6E、&HFFDC 等。

(3) 日期常量

日期常量要用两个＃号把表示日期和时间的值界定起来，例如，＃07/01/1997＃和＃2/11/2007 10:10:00 AM＃。

(4) 逻辑常量

逻辑常量只有两个值：True 和 False。将逻辑数据转换成整型数据时：True 为－1，False 为 0；反过来，其他数据转换成逻辑数据时：非 0 为 True，0 为 False。

2. 符号常量

符号常量是指在程序中用一个标识符来表示一个常量。符号常量又分为两种：用户定义常量和系统内部定义常量。

(1) 用户定义常量

用户定义常量是用户使用常量的声明语句(Const 语句)把一个字符串(称之为符号或者常量名)与一个常数值关联起来。声明常量的语法格式为：

```
Const 符号常量名 [As<数据类型>]=表达式
```

例如，以下均为正确的用户定义常量：

```
Const PI=3.1415926
Const IMIN As Integer=10
Const IDATE=#10/30/2012#
Const S%=12
```

注意：

- 符号常量名的命名规则与变量命名规则相同，为了便于程序的阅读，符号常量名习惯采用大写字母表示；

- 在使用类型说明符声明常量时，常量名与类型说明符之间不要有空格；
- 表达式由数值常量、字符串常量及运算符组成，但不能使用函数调用；
- 常量一旦声明，只能引用而不能改变，即不能对符号常量赋新值。

(2) 系统内部定义常量

系统内部定义的常量是 VB 系统为应用程序和控件定义的常量，存放于 VB 系统的对象库中。用户可以在“对象浏览器”中查看内部常量。选择“视图”菜单中的“对象浏览器”，则打开“对象浏览器”窗口。在下拉列表框中选择 VB 或 VBA 对象库，然后在“类”列表框中选择常量组，右侧的成员列表中即显示预定义的常量，窗口底端的文本区域中将显示该常量的功能。这些系统内部定义的常量，在编程时可直接使用、无需再次声明。常量名称的前缀标志着该常量的来源。

注意：

- 来自 VB 库文件的常量以 vb 开头；
- 来自数据访问库文件的常量以 db 开头；
- 自定义的常量不使用任何前缀，以免发生错误。

在程序中使用系统内部定义的常量，可使程序易于阅读和编写。例如，文本框控件(TextBox)的 Alignment 属性表示文本对齐方式，有三种取值，如表 3.2 所示。

表 3.2 VB 中 Alignment 属性值

系统常量	数值	说明	系统常量	数值	说明
vbLeftJustify	0	文本左对齐	vbCenter	2	文本居中
vbRightJustify	1	文本右对齐			

在程序中使用下面两行代码，均可表示 TextBox1 控件的文本右对齐。显然第一行代码更容易被用户阅读和理解。

```
TextBox1. Alignment=vbRightJustify
TextBox1. Alignment=1
```

3.2.2 变量

在 VB 程序运行过程中，其值可以改变的量称为变量。一个变量必须要有一个变量名和相应的数据类型。这样通过变量名就可以使用该变量，而数据类型决定了该变量的存储方式以及在内存中占据存储单元的大小。变量名实际上是一个符号地址，而该变量的值存储在其地址所指向的内存单元里。所以程序从变量中取值，实际上是通过变量名找到相应的内存地址，从其存储单元中取得数据。

在大多数的编程语言中，要求变量“先声明，后使用”。所谓声明变量，就是在程序中指定变量的名称和它的数据类型，以便系统为它分配合适的存储单元。但是，在 VB 中使用一个变量时，可以不加任何声明而直接使用，这种使用称之为隐式声明。使用这种方法虽然简单，但却容易在发生错误时令系统产生误解。一般对于变量来说，最好遵循刚才提

到的“先声明，后使用”的原则。与上述隐式声明相对应的称之为显示声明。所谓显式声明，是指每个变量必须事先声明，才能够正常使用，否则会出现错误警告。

1. 显式声明变量

所谓显式声明，就是用一个声明语句来定义变量名和数据类型。声明变量的语句形式如下：

```
Dim 变量名 [As 数据类型]
```

其中，Dim 是定义变量所使用的关键字，其后跟随被定义的变量名。[As 数据类型]是被声明变量的类型。此项可以省略。若省略“As 数据类型”，则声明的变量默认为变体型。

为了方便定义，可在变量名后加类型符来代替“As 数据类型”。例如，“Dim 变量名[类型符]”。变量名与类型符之间不能有空格(类型符可参见表 3.1)。另外，一条 Dim 语句可以同时声明多个变量，但每个变量必须有自己的类型声明，类型声明不能共用。例如：

```
Dim iCount As Integer , sum As Single
```

等价于

```
Dim iCount%, sum!
```

对于字符串类型变量，如果其存放的字符串长度是固定的，那么可以把它定义为定长字符串，定义方法为：

```
Dim 字符串变量名 AS String * 字符数
```

例如：

```
Dim strAddress As String * 50        '声明固定长度字符串变量可最多放 50 个字符
```

2. 隐式声明

VB 允许用户在编写应用程序时不声明变量而直接使用，这就是变量的隐式声明。系统会为这些隐式变量临时分配存储空间并使用。所有隐式声明的变量都是 Variant 数据类型。Variant 数据类型很像一条变色龙，它可以在不同场合表示不同的数据类型。不同数据类型之间的转换不需要用户去实现，因为 VB 系统会自动完成各种必要的转换。

为了使程序具有更好的可读性，应尽量避免使用隐式声明的变量。可以在模块的声明中使用 Option Explicit 语句来强制声明变量。这样程序中如果有未声明的变量，VB 将产生一个“变量未定义”的编译错误。用户也可以在“工具”菜单项中选择“选项”菜单项，在如图 3.1 所示的对话框中选择“编辑器”选项卡，再将其中的“要求变量声明”选项前的复选标记选中，单击“确定”按钮。VB 会在以后生成的新模块中自动添加

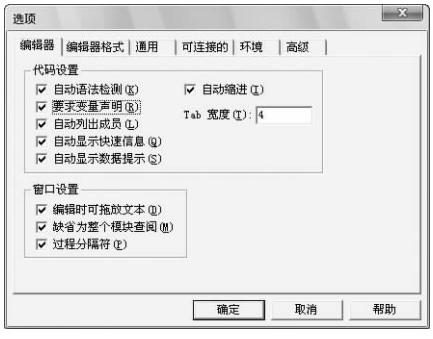

图 3.1　“选项”对话框中设置变量声明要求

Option Explicit 语句，对于已经存在的模块则不能做修改，需要用户自己手工添加。

3. 变量命名规则

VB 中变量的命名规则如下：

(1) 必须以字母或汉字开头，由字母、汉字、数字或下划线组成，不能含有小数点、空格等特殊字符。

例如，a * b、x $ y、? xy、ab. c、12sum、_a1 和 score%x 等都是错误的变量名。

(2) 变量名的长度不能超过 255 个字符。

(3) 不能使用 VB 中的关键字(语句名、函数名等)。

例如，CONST、Public、Print 等均为非法变量名。

(4) VB 中不区分变量名的大小写。

(5) 为了增加程序的可读性，可在变量名前加一个缩写的前缀来表明该变量的数据类型。

例如，可用 strName 表示字符串变量，iCount 表示整型变量，dblx 表示双精度变量，sngYz 表示单精度变量。

3.3 运算符和表达式

在 VB 中有 4 种运算符：算术运算符、连接运算符、关系运算符和逻辑运算符。由运算符、圆括号、常量、变量和函数组成的有意义的式子称为表达式。根据运算的结果，表达式也可以被分为 4 类：算术表达式、字符串表达式、关系表达式和逻辑表达式。

3.3.1 运算符

1. 算术运算符

算术运算符要求参与运算的数据为数值型，运算的结果也是数值型，除了负号运算符(－)是单目运算符(一个操作数)以外，其他的都是双目运算符(两个操作数)。当表达式中含有多个运算符时，需要按运算符的优先级来决定运算顺序，其中表 3.3 按优先级从高到低的顺序列出了 VB 的算术运算符。如果一个表达式中出现了多个同等级的运算符，运算顺序按从左到右顺序进行。

表 3.3 VB 中的算术运算符

运算符	说　明	例　子	运算结果
^	幂运算	4^2	16
－	取负数	－5	－5
* /	乘法和除法	4 * 5,25/3	20,8.3333
\	整除	25\3	8
Mod	取余	26 Mod 3	2
＋ －	加法和减法	1＋2,5－3	3,2

2. 字符串连接运算符

字符串连接运算符有两个：& 和＋，它们都可用来把两个字符串连接起来，同时合并成一个新字符串。字符串常量与运算符 & 之间需要加一个空格。因为，符号 & 也可以用来表示长整型类型符。

例如：

```
"Visual Basic"+"程序设计"              '结果为"Visual Basic 程序设计"
"Visual Basic" & "Program design"     '结果为"Visual Basic Program design"
```

连接符 & 和＋的区别：

＋：若两旁的操作数均为字符串型数据，其运算结果也是字符串型数据。若两个操作数中有一个为数值型数据，而另一个操作数是由数字符号组成的字符串，则自动将数字字符串转换成数值型的数据然后再进行算术加运算；若另一个操作数中含有非数字字符，则会出现“类型不匹配”错误。

例如：

```
"3"+5                                 '结果为 8
"3"+"5"                               '结果为"35"
"Text"+2                              '出错
"Text"+"2"                            '结果为"Text2"
```

&：无论两旁操作数是否为字符串型数据，在进行连接操作时，系统都会将两个操作数自动转换成字符型，再连接。

例如：

```
"3" & "5"                             '结果为"35"
3 & 5                                 '结果为"35"
3+"5" & 5                             '结果为"85"
```

3. 关系运算符

关系运算符都是双目运算符，其作用在于比较两个操作数或表达式值的大小。如果关系成立，运算结果为逻辑值真(True)，否则返回假(False)。关系运算符的优先级低于算术运算符，各个关系运算符的优先级是相同的，运算顺序为从左到右。表 3.4 列出了 VB 中的关系运算符。

表 3.4　VB 中的关系运算符

运算符	说　明	例　子	运算结果
＝	等于	"dog"＝"cat"	False
＞	大于	"a"＞"b"	False
＞＝	大于或者等于	"teacher"＞＝"教师"	False
＜	小于	35＜4	False

续表

运算符	说　明	例　子	运算结果
<=	小于或者等于	"35"<="4"	True
<>	不等于	"car"<>"jeep"	True
Like	字符串模式匹配	"database" like " * base"	True
Is	对象引用是否是同一对象	object1 Is object2	True 或 False

使用关系运算符比较时，应注意以下规则：

(1) 当两个操作数均为数值型数据时，按数值大小比较。

(2) 当两个操作数均为字符型数据时，需要从左到右依次比较它们所包含字符。如果待比较的字符相同，继续比较下一个字符；如果不同，则比较它们相应的 ASCII 码值。例如：

```
"ABOARD">"ABR"        结果为 False
```

(3) 数值型与可转换为数值型的数据比较。

例如，30>"89"，按数值比较，结果为 False。

(4) 数值型与不能转换成数值型的字符型比较。

例如，77>"sdcd"，不能比较，系统出错。

(5) 汉字字符大于西文字符。

(6) 关系运算符的优先级都是相同的，运算顺序从左到右。

(7) "Like"运算符是 VB 6.0 新增加的，可以与通配符 * 、?、# 、[字符列表]、[!字符列表]结合使用，用于在数据库的 SQL 语句中进行模糊查询。其中？表示任何单个字符，* 表示零个或多个字符，# 表示任何一个数字(0～9)，[字符列表]表示字符列表中的任何单个字符，[!字符列表]表示不在字符列表中的任何单个字符。

like 运算符的使用格式为：str1 Like str2，其中字符型变量 str1 和 str2 可以是字符常量，也可以是字符型的变量。例如，查找姓名变量中姓王的学生，表达式为：姓名 Like "王 * "；如果要查找姓名变量中没有王字的学生，则表达式为：姓名 Like [!王]

(8) Is 运算符确定两个对象引用是否引用同一个对象，它不执行值比较。如果 object1 和 object2 引用同一个对象实例，则为 True；如果它们不引用同一个对象，则为 False。

4. 逻辑运算符

在所有逻辑运算符中，除 Not 是单目运算符之外，其余都是双目运算符。其作用是将操作数用于逻辑运算，结果为逻辑值真(True)或假(False)。表 3.5 按优先级从高到低列出了 VB 中的逻辑运算符。

说明：

(1) 逻辑运算符的优先级低于关系运算符，各个逻辑运算符的优先级不相同，Not(逻辑非)最高，Imp(逻辑蕴含)最低。

表 3.5　VB 中的逻辑运算符

运算符	作用	优先级	说　明	实例
Not	取反	1	操作数为假时，结果为真，反之结果为假	Not F＝T Not T＝F
And	与	2	操作数均为真时，结果才为真	T And T＝T T And F＝F F And T＝F F And F＝F
Or	或	3	操作数中有一个为真时，结果为真	T Or T＝T T Or F＝T F Or T＝T F Or F＝F
Xor	异或	3	操作数相反时，结果才为真	T Xor F＝T T Xor T＝F F Xor F＝F
Eqv	等价	4	操作数相同时，结果才为真	T Eqv T＝T T Eqv F＝F
Imp	蕴含	5	第一个操作数为真，第二个操作数为假时，结果才为真，其余结果均为假	T Imp F＝T T Imp T＝F

(2) VB 中常用的逻辑运算符是 Not、And 和 Or。如果将它们用于表示复杂的表达式，需要将此表达式根据逻辑运算符，拆分成多个子表达式。例如：

数学上表示某个数在某个区域时用 $0 \leqslant X < 100$，在 VB 程序中应写成：

```
X>=0 And X<100
```

(3) 参与逻辑运算的量一般都应是逻辑型数据，如果参与逻辑运算的两个操作数是数值量，则以数值的二进制值逐位进行逻辑运算(0 为 False，1 为 True)。

例如，计算 10 And 7 时，需先将两个操作数转换成二进制数 1010 和 0111，然后按位进行与运算，最终结果为 $(0010)_2$，即 $(2)_{10}$。

3.3.2　表达式

1. 表达式的组成

表达式是由常量、变量、各种运算符、函数和圆括号按一定的规则组成的字符序列。表达式运算的结果是由参与运算的数据以及运算符所决定的。在表达式中，经常会出现操作数与运算符之间的类型不匹配，以及不同类型数据的混合运算。这时，需要将一些数据按一定的规则进行类型转换。转换方法有两种：系统自动转换和使用转换函数。

2. 表达式的分类

根据表达式中所包含的运算符类型，可以将基本表达式(只含有一种类型的运算符)进行如下分类。

(1) 算术表达式。由算术运算符、圆括号、数据和内部函数组成的一组字符串称为算

术表达式。例如，y＝x^2＋cos(x)－5，其中 cos()是余弦函数。

(2) 字符串表达式。由字符串运算符(& 或＋)、字符型常量和字符型变量组成的一组字符串称为字符串表达式。例如，"ABC" & "DE"＋str(y)，其中 str()是字符串转换函数。

(3) 关系表达式。由关系运算符(＞，＜，＝，＜＞等)和操作数组成的一组字符串称为关系表达式。它的作用是比较操作数之间的大小关系，如果关系成立，表达式返回逻辑值 True；否则返回逻辑值 False。例如，表达式 5＜3 返回 False。

(4) 逻辑表达式。由逻辑运算符和操作数组成的一组字符串称为逻辑表达式。此类表达式的返回结果也是逻辑值(True 或 False)。例如，表达式 Not(5) & 1 返回 False。

(5) 日期型表达式。在 VB 中没有特定的日期运算符，但是日期型数据可以进行两种运算(＋和－)。因此，日期型表达式是由日期型数据和运算符＋或－组成的一组字符串。例如，表达式 #05/08/2012#-#05/01/2012#，其结果为数值 7；表达式 #05/01/2012#＋7，其结果为日期型数据：#05/08/2012#。

3. 表达式中各种运算符的优先级

当一个表达式中出现多种运算符时，此时表达式需要按运算符的优先级来逐步计算出结果。一般情况下，运算按照括号、函数、算术运算、字符运算、关系运算和逻辑运算的顺序来进行。因此，可以总结得出 VB 中不同类型的运算符之间的优先级如下：

算术运算符＞字符串连接运算符＞关系运算符＞逻辑运算符

同类运算符之间的优先级可参考 3.3.1 节中的内容。在表达式中，可以用括号改变优先顺序，强制表达式的某些部分优先运行。在表达式中使用括号，可以使表达式的结构更清晰、易懂。但是在 VB 程序中只能使用圆括号，如果想用多层括号，则需要注意括号的匹配。例如，y＝((x＋5)^3)/6。

4. 书写表达式时需注意的问题

(1) 在算术表达式中，乘号(＊)不能省略。例如，x 乘以 y 不能写成 xy，需要写成 x＊y。

(2) 所有的二元运算符不能相邻，它们之间必须要有操作数。例如，a＋mod b 是错误的。

(3) 为了改变运算顺序，可以使用圆括号()来提高优先级低的运算符级别。在表达式中只能使用圆括号()，而不能使用中括号[]和大括号{}。圆括号可以嵌套使用，但必须成对出现。

(4) 使用 VB 的数学表达式规范来表示传统的数学公式。

例如，数学式 $3\times\{2x\div[5y(97+z)]\}$，需写成 VB 表达式为

3＊(2＊x/(5＊y＊(97＋z)))

例如，数学式 $\frac{x+y}{x-y}$，需写成 VB 表达式为

(x＋y)/(x－y)

例如，圆面积的数学表达式 πr^2，需写成 VB 表达式为 3.1415926＊r^2，或声明一个常量代替 π 值。

3.3.3　常用内部函数

常用内部函数是由 VB 系统提供的，每个函数可以完成某个特定的功能。在程序中可以通过调用某个函数来使用它的功能，函数调用的一般格式为：

```
函数名(参数 1,参数 2,…)
```

其中，参数需要放在括号里面，若有多个参数，则需要用逗号间隔。

按照功能划分，常用内部函数可分为数学函数、字符串函数、日期与时间函数、转换函数和格式输出函数等。在以下叙述中，使用 N 表示算术表达式、C 表示字符串表达式、D 表示日期表达式。函数名后出现 $ 符号，则表示返回值为字符串。用户可以通过帮助菜单获得所有内部函数的使用方法。

1. 数学函数

VB 中提供了大量的数学函数，其使用方法与数学中的定义基本上是一致的，例如，三角函数、指数函数、对数函数等。表 3.6 列出了常用的数学函数。

表 3.6　常用的数学函数

函数名	功　能	例　子	结　果
Abs(N)	取绝对值	Abs(−7.8)	7.8
Cos(N)	余弦函数	Cos(0)	1
Exp(N)	以 e 为底的指数函数，即 e^N	Exp(2)	7.389
Log(N)	以 e 为底的对数函数	Log(10)	2.303
Rnd[(N)]	产生随机数	Rnd	[0,1)之间的数
Sgn(N)	符号函数	Sgn(1) Sgn(0) Sgn(−1)	1 0 −1
Sin(N)	正弦函数	Sin(30 * 3.14/180)	0.499
Sqr(N)	平方根函数	Sqr(9)	3
Tan(N)	正切函数	Tan(60 * 3.14/180)	1.729

使用数学函数的注意事项：

(1) 所有数学函数的参数都是数值型，计算结果也是数值型。

(2) 三角函数的参数是以弧度为计算单位。所以在使用时，需要先将弧度值转换成角度值，公式为：角度值＝弧度值 * (3.14/180)。

(3) Rnd 函数可以随机产生一个 0～1 之间的双精度数。如果要随机产生一个[a,b]之间的数，可以采用公式：Int(Rnd * (b−a+1)+a)来实现。默认情况下，每次运行同一个应用程序时，Rnd 函数都会产生相同的随机数序列。如果想产生不同的随机数序列，可以在使用 Rnd 函数之前，调用语句 Randomize。

(4) 函数中的参数可以是常量、变量或者是表达式,甚至可以是函数(称为函数嵌套)。例如,Sqr(abs(－9))的计算结果为 3。

例 3.1 给定一个两位数,交换个位数和十位数的位置,把处理后的数显示在窗体上。

编写的窗体单击事件过程代码如下:

```
Private Sub Form_Click()
  Dim x As Integer, a As Integer
  Dim b As Integer, c As Integer
  x=36
  a=Int(x/10)
  b=x Mod 10
  c=b * 10+a
  Print "处理后的数: "; c
End Sub
```

运行程序后单击窗体,输出结果为

```
处理后的数: 63
```

2. 转换函数

转换函数多用于数据类型的转换,表 3.7 列出了常用的转换函数。

说明:

(1) 函数 Chr()和函数 Asc()互为反函数,例如,Asc("A ")＝65,Chr(122)＝"A "。

(2) 整函数 Fix(R)、Int(R)和 Round(R)都可以将实数转换成整数,但它们之间的差别在于:函数 Fix(R)直接去掉 R 的小数部分,只保留整数部分;函数 Int(R)返回小于 R 的最大整数;函数 Round(R)对 R 的小数部分进行四舍五入。

当 N＞0 时 Fix(N)与 Int(N)相同,当 N＜0 时,Int(N)与 Fix(N)－1 相等。例如:

Fix(9.59)＝9　　Int(9.59)＝9

Fix(－9.59)＝－9　　Int(－9.59)＝－10

(3) 利用 Int()函数可以对数据进行四舍五入。

对一个正数 x 舍去小数位时进行四舍五入操作可使用如下形式:Int(x＋0.5)。例如:

当 x＝5.4 时,Int(5.4＋0.5)＝5

当 x＝5.6 时,Int(5.6＋0.5)＝6

(4) Str()函数将非负数值转换成字符类型后,会在转换后的字符串左边增加空格(即数值的符号位)。例如:

Str(12.3)的结果是" 12.3"(第一个字符是空格),而不是"12.3"。

(5) Val()函数从左向右将数字字符转换为数值。一旦遇到非数字字符,转换结束并返回之前转换所得的数值。例如:

Val("a123")的值为 0,而 Val("4.2sa10")的值为 4.2。

(6) VB 中还有类型转换函数,例如 CInt()、CBool()、CSng()和 CStr()等,它们详细的使用说明可查阅帮助文档。表 3.7 给出了常用的转换函数。

表 3.7　常用的转换函数

函数名	功　能	例子	结果
Asc(C)	C 的首字符转换成 ASCII 码值	Asc("AB")	65
Chr$(N)	ASCII 码值转换成字符	Chr$(65)	"A"
Fix(N)	截去小数取整	Fix(9.8)	9
Hex[$](N)	十进制数转换成十六进制数	Hex[$](100)	64
Int(N)	取小于或等于 N 的最大整数	Int(9.8) Int(−9.8)	9 −10
Lcase$(C)	大写字母转换为小写字母	Lcase$("ABC")	"abc"
Oct[$](N)	十进制数转换成八进制数	Oct[$](100)	144
Round(N)	四舍五入取整	Round(9.8) Round(−9.8)	10 −10
Str$(N)	数值转换成字符串	Str$(123.456)	"123.456"
Ucase$(C)	小写字母转换为大写字母	Ucase$("abc")	"ABC"
Val(C)	数字字符串转换为数值	Val("123ABC")	123

3. 字符串操作函数

计算机的很多应用领域都会涉及大量的字符串处理操作，例如字符串的查找、比较和获取子串等操作。因此，VB 提供了大量的、用于字符串处理的函数。表 3.8 列出了常用的字符串函数。

表 3.8　常用的字符串函数

函　数　名	功　能	例　子	结　果
InStr([N1,]C1,C2[,M])	在 C1 中从 N1 开始找 C2，省略 N1 从头开始找，找不到为 0	InStr(3,"ABCDBCG","BC")	5
Left(C,N)	从字符串 C 的左边取 N 个字符	Left("ABCDEFGH",3)	"ABC"
Len(C)	字符串长度	Len("VB 程序设计")	6
LenB(C)	字符串占用字节数	LenB("VB 程序设计")	12
Ltrim(C)	去掉字符串左边空格	Ltrim(" ABC ")	"ABC"
Mid(C,N1,[,N2])	取子串，在 C 中从 N1 位开始向右取 N2 个字符，默认到 N2 结束	Mid("ABCDEFGH",3,3)	"CDE"
Right(C,N)	从字符串 C 的右边取 N 个字符	Right("ABCDEFGH",4)	"EFGH"
Rtrim(C)	去掉字符串右边空格	Rtrim("ABC ")	"ABC"
Space(N)	产生 N 个空格	Space(3)	" "
String(N,C)	生成 N 个字符，字符为 C 中首字符	String(2, "BCDE")	"BB"
StrReverse(C)	字符串逆序	StrReverse("BCDE")	"EDCB"
Trim(C)	去掉字符串 C 的左、右边空格	Trim(" ABC ")	"ABC"

说明：

(1) 在 Mid()函数中，若省略第三个参数 N2，则得到从 N1 开始到字符串结尾的所有字符，如 Mid("ABCDE", 2)的结果为"BCDE"。

(2) 在 String()函数中，参数 C 可以是 ASCII 码值，例如 String(3, "A")等价于 String(3, 65)。

(3) 在表 3.7 中，所有返回值为字符串型的函数，其函数名的尾部可加上字符 $，例如 Left("AB", 1)也可以写成 Left$("AB", 1)。

例 3.2 使用字符串函数示例。先从字符串 a 中找出某个指定字符(本例为空格)，再以此字符为界，将原有字符串拆成两个字符串。

编写的窗体单击事件过程代码如下：

```
Private Sub Form_Click()
    Dim a As String, b As String, c As String, n As Integer
    a="Visual Basic"
    n=InStr(a, " ")                    '查找空格的位置
    b=Left(a, n-1)                     '取空格左边部分的字符串
    c=Mid(a, n+1)                      '去右边部分的字符串
    Print b
    Print c
End Sub
```

程序运行后单击窗体，输出结果是：

```
Visual
Basic
```

4. 日期/时间函数

日期/时间函数用于进行日期和时间的处理，表 3.9 列出了 VB 中常用的日期/时间函数。

表 3.9 常用的日期/时间函数

函数名	功　能	例　子	结　果
Date[()]	返回系统日期	Date()	2012-5-10
DateSerial(年,月,日)	返回一个日期形式	DateSerial(12,5,10)	2012-5-10
DateValue(C)	同上，但自变量是字符串	DateValue("2012-5-10")	2012-5-10
Day(C|N)	返回日期代号(1～31)	Day("2012-5-10")	10
Hour(C|N)	返回小时(0～24)	Hour(#9:34:45AM #)	9
Minute(C|N)	返回分钟(0～59)	Minute(#9:34:56AM #)	34
Month(C|N)	返回月份代号(1～12)	Month("2012-5-10")	5
MonthName(N)	返回月份名	MonthName(5)	五月

续表

函数名	功　能	例　子	结　果
Now	返回系统日期和时间	Now	2012-5-10 9:34:45AM
Second(C|N)	返回秒(0～59)	Second(#9:34:45AM #)	45
Time[()]	返回系统时间	Time	9:34:45AM
WeekDay(C|N)	返回星期代号(1～7)星期日为1,星期一为2,依此类推	WeekDay("2012-5-10")	5
WeekDayName(N)	将星期代号(1～7)转换为星期名称	WeekDayName(1)	星期日
Year(C|N)	返回年代号(1753～2078)	Year("2012-5-10")	2012

说明：

(1) 函数 Date()、Now()、Time()都是返回系统日期或时间的函数，注意它们返回值的区别。

(2) 日期函数中自变量 C|N 可以是数值表达式，也可以是字符串表达式，其中 N 表示相对于 1899 年 12 月 31 日前后的天数。

5. 格式输出函数

VB 在显示数字的格式上比较灵活，对于数值、日期和字符串采用标准格式显示。使用 Format 函数，可以将数值转换成指定格式的字符串输出。使用格式如下：

```
Format$(表达式[,"格式字符串"])
```

例如：

```
Format(3.567, "##.0000")
```

其中，表达式可以是数值、日期或字符串型表达式。格式字符串表示转换后的格式，用双引号括起。格式字符串是由三种格式符构成的：

(1) 数值型数据格式化的有关格式符见表 3.10。

表 3.10　数值格式符

符号	功　能	数值表达式	格式字符串	结果
0	实际数字位数小于符号位数，数字前后加 0	123.456 123.456	"0000.0000" "000.00"	0123.4560 123.46
#	实际数字位数小于符号位数，数字前后不加 0	123.456 123.456	"#####.#####" "###.##"	123.456 123.46
.	加小数点	123	"000.00"	123.00
,	千分位	1234.5	"00,000. 00"	01,234.50
%	数值乘以 100，加百分号	123.456	"###.##%"	12345. 6%
$	在数字前加 $	123.456	"$###.##"	$123.46

续表

符号	功　能	数值表达式	格式字符串	结果
＋	在数字前加＋	123.456	"＋＃＃＃.＃＃"	＋123.46
－	在数字前加－	123.456	"－＃＃＃.＃＃"	－123.46
E＋	用指数表示	123.456	"0.000E＋00"	1.235E＋02
E－	用指数表示	0.001234	"0.000E－00"	1.234E－03

说明：对于符号 0 与＃，当数值表达式的整数部分位数比格式字符串的位数多时，数据将原样显示；若小数部分的位数比格式字符串的位数多时，系统将按四舍五入处理后显示。

（2）日期和时间型数据格式化的有关格式符见表 3.11。

表 3.11　日期和时间格式符

符号	功　能	符号	功　能
d	显示日期(1～31)，个位前不加 0	dd	显示日期(01～31)，个位前加 0
ddd	显示星期缩写(Sun～Sat)	dddd	显示星期全名(Sunday～Saturday)
ddddd	显示完整日期(yy/mm/dd)	dddddd	显示完整长日期(yyyy/mm/dd)
w	星期为数字(1～7，1 是星期日)	ww	一年中的星期数(1～53)
m	显示月份(1～12)，个位前不加 0	mm	显示月份(01～12)，个位前加 0
mmm	显示月份缩写(Jan～Dec)	mmmm	显示月份全名(January～December)
y	显示一年中的天(1～366)	yy	两位数显示年份(00～99)
yyyy	四位数显示年份(0100～9999)	q	季度数(1～4)
h	显示小时(0～23)，个位前不加 0	hh	显示小时(00～23)，个位前加 0
m	在 h 后显示分(0～59)，个位前不加 0	mm	在 h 后显示分(00～59)，个位前加 0
s	显示秒(0～59)，个位前不加 0	ss	显示秒(00～59)，个位前加 0
tttt	显示完整时间(小时、分和秒)，默认格式为 hh:mm:ss	AM/PM am/pm	12 小时的时钟，中午前为 AM 或 am，中午后为 PM 或 pm
A/P,a/p	12 小时的时钟，中午前为 A 或 a，中午后为 P 或 p		

说明：非格式说明符-、/、:等原样显示。

（3）字符串类型数据格式化的有关格式符见表 3.12。

说明：格式串内@号的个数决定了显示串的长度。如果要显示的字符串长度小于格式串的长度，字符串显示时右对齐；如果要显示的字符串长度大于格式串的长度，字符串原样显示。

表 3.12　字符串格式符

符号	功　能	字符串表达式	格式字符串	结果
<	强制以小写显示	123HELLO	"<"	123hello
>	强制以大写显示	Hello	">"	HELLO
@	实际字符位数小于符号位数,字符前加空格	12345 789	"@@@@@@@@" "@@@@@@@@"	12345 789
&	实际字符位数小于符号位数,字符前不加空格	hello hey	"&&&&&&&&" "&&&&&&&&"	hello hey

6. Shell 函数

在 VB 中不但提供了可调用的内部函数,还可以调用各种应用程序。通过 Shell 函数可以调用 DOS 或 Windows 下运行的可执行文件,如果调用成功返回值代表这个程序的进程 ID,若调用不成功,则会返回 0。

Shell 函数格式:

```
Shell(命令字符串,窗口类型)
```

其中,命令字符串表示要执行的应用程序名,包括路径。

窗口类型可以选择 0～4 或 6 的整数。一般取值为 1 表示正常窗口,默认值为 2,窗口最小化为图标。

例如,当程序在运行时执行 Windows 下的计算器,然后切换到 DOS 界面,代码如下:

```
i=Shell("c:\windows\calc.exe")
j=Shell("c:\command.com", 1)
```

3.4　编 码 规 则

VB 和其他程序设计语言一样,编写代码有一定的书写规则,主要规定如下。

1. VB 代码中不区分字母的大小写

为了提高程序的可读性,VB 对用户程序代码进行自动转换。

(1) 为了便于程序的阅读,系统自动将关键字的首字母转换成大写,其余字母转换成小写。

(2) 若关键字由多个英文单词组成,VB 自动将每个单词的首字母转换成大写。

(3) 对于用户自定义的变量、过程名,VB 以第一次定义的为准,以后输入的自动转换成首次定义的形式。

2. 语句书写自由

(1) 一行最多允许书写 255 个字符。

(2) 在同一行上可以书写一条或多条语句,若书写多条语句,语句间用冒号(:)分隔。

(3) 单行语句可分若干行书写,需在本行后加上续行符" _"(由一个空格字符和一个

下划线字符组成)。

3. 使用注释有利于程序的维护和调试

注释以 Rem 开头,或用单撇号(')引导注释内容。用单撇号(')引导注释内容,可以直接书写在语句的后面。例如:

```
'This is a VB
REM  This is a VB program
```

也可以使用“编辑”工具栏的“设置注释块”、“解除注释块”命令将选中的若干行语句或文字设置注释或取消注释。

4. 使用缩进格式

在编写程序代码时,为了使程序结构更具可读性,可以使用缩进格式反映代码的逻辑结构和嵌套关系。例如:

```
Private Sub Form_Click()
  X=5
  If X<0 Then
    Print "X 小于零。"
  Else
    Print "X 大于或等于零。"
  End If
End Sub
```

习 题 三

一、选择题

1. 下面(　　)是合法的变量名。

A. ForLoop　　B. Const　　C. 6abc　　D. b#x

2. 在 VB 中,变量定义语句为 Dim a as Integer, b,则变量 a 和 b 的类型分别为(　　)。

A. a,b 均为整型　　B. a 为整型,b 为变体类型

C. a,b 均为变体类型　　D. a 为变体类型,b 为整型

3. 在一行内写多条语句时,每个语句之间用(　　)符号分隔。

A. ,　　B. :　　C. 、　　D. ;

4. RND 函数不可能是下列(　　)值。

A. 1　　B. 0　　C. 0.123　　D. 0.0005

5. 获得系统时间的函数是(　　)。

A. data$　　B. time$　　C. date$　　D. gettime$

6. 表达式 16/4−2^5 * 8/4 mod 5\2 的值为(　　)。

A. 14　　B. 4　　C. 20　　D. 2

7. 与数学表达式 ab/4cd 对应,VB 的不正确表达式是(　　)。