第3章

数据类型、常量与变量

这里的数据指的是可以被计算机处理的信息。为了快速地对数据进行运算并有效地利用存储空间，Visual Basic 把各种不同的数据归纳为多种数据类型。每种数据类型都有类型名称，每种类型的数据占用一定数量的存储空间，可以表示的值也有一定的范围限制。

数据类型可以用来定义变量、数组、常量、自定义类型和过程的参数。

3.1 基本数据类型

3.1.1 数值型

数值型是数据类型中的一个大类，包括表 3.1 所示的 6 种具体类型。

表 3.1 数值型数据类型

序号	类型名称	中文名称	字节数	可表示值的范围
1	Byte	字节型	1	0～255 之间的整数
2	Integer	整型	2	－32768～32767 之间的整数
3	Long	长整型	4	－2147483648～2147483647 之间的整数
4	Single	单精度浮点型	4	-3.402823×10^{38}～3.402823×10^{38} 之间的实数，6～7 位有效数字
5	Double	双精度浮点型	8	$-1.79769313486232\times10^{308}$～$1.79769313486232\times10^{308}$ 之间的实数，14～15 位有效数字
6	Currency	货币型	8	－922337203685477.5808～922337203685477.5807，15 位整数和 4 位小数组成的定点数

这 6 种数据类型都是用来表示数值的。在实际编程时，应根据具体用途决定应该选用什么类型来表示数值。表示范围越大、精度越高的数据类型所占用的存储空间也越大、运算速度越慢。

3.1.2 String 型

字符串是指连续的字符序列。String(字符串)类型是专门用来存放文字信息的。字符串型又分为“定长字符串型”和“变长字符串型”两大类。

顾名思义，定长字符串数据能够包含字符的个数是一定的，这个长度是可以指定的，但是不能多于 64K(2^{16})个字符。而变长字符串可以包含的字符数是可变的，所占用的内存空

间也会变化,但总是要比实际的字符数多 10 个额外字节,最多不能超过 20 亿(2^{31})个字节(还受实际系统资源的限制)。

3.1.3 Boolean 型

Boolean(逻辑、布尔)类型的数据只可能有两个值:True(逻辑"真")和 False(逻辑"假"),用来表示"是"与"否"、"开"与"关"、"对"与"错"这类只有两种取值的情况。虽然"是"与"否"可以只用一个二进制位来表示,但是一个逻辑型数据却要占 2 个字节的存储空间。

3.1.4 Date 型

Date(日期时间)类型又称为日期型,这种类型的数据可以存放日期信息、时间信息或者同时存放日期与时间信息。Date 类型数据用 8 个字节来表示日期(公元 100 年 1 月 1 日～9999 年 12 月 31 日)和时间(12:00:00AM～11:59:59PM,即 0:00:00～23:59:59)。

除了上述这些类型之外,Visual Basic 的基本数据类型还包括 Object(对象型)和 Variant(变体类型),将分别在 3.3.6 节和 3.3.7 节中讲解。

3.2 直接常量

常量(Constant)是指在程序运行过程中其值始终保持不变的量。常量有两种:直接常量与符号常量。下面是各种数据类型直接常量的表示法。

3.2.1 整型常量

1. 十进制表示法

整型常量的十进制表示法与人们的日常书写方法相同,下面是一些十进制整型常量:

```
0    10    -12    2000    -1234
```

如果在一个整型常量之后加"&"字符会使其成为长整型常量。例如,10 是整型常量,但 10& 是长整型常量,二者数值大小相同,占用内存却不同。如果在一个超出整型表示范围的常量后面加"&",Visual Basic 会自动删除"&"号,因为它只可能是长整型常量。

下面是一些十进制长整型常量:

```
100000    10&    -100&
```

2. 八进制表示法

以"&O"(字母 O)开头,后面接由 0～7 组成的八进制数。如果要表示长整型数,末尾要加一个"&"号。下面是一些八进制常量:

```
&O11    &O123    &O7777    &O176340    &O176340&
```

分别等于十进制的 9、83、4095、－800、64736。

应该注意的是,&O176340 被理解为 Integer 类型常量,把它写为 16 位二进制形式,最高位是 1,表示负值,是十进制数－800 的补码形式。而 &O176340& 是 Long 类型常量,把

它写为32位二进制形式，最高位是0，表示正值64736。

3. 十六进制表示法

以“&H”开头，后面接由0～9、A～F组成的十六进制数。如果要表示长整数，末尾要加一个“&”号。下面是一些十六进制常量：

```
&H11    &HFF    &HFFFF    &HFFFF&    &HFF76340
```

分别等于十进制的17、255、－1、65535(长整数)、267871040。把一个表示负数的十六进制数转换为十进制数，要用到补码。

如果一个以八进制或十六进制表示的常量只可能是长整数时(超出整型的表示范围)，末尾可以不加“&”(如上面的 &HFF76340)。

3.2.2　浮点型常量

浮点型常量可以使用日常的书写方法。如果整数部分或小数部分为0，则可以省略这一部分，但要保留小数点。下面是几个浮点型常量：

```
3.14159    0.23    24.    -.45    -0.05
```

也可以用指数形式表示浮点型常量，用mEn来表示$m\times10^n$，其中的m是一个整数常量或实数常量，n必须是整数常量，m和n均不能省略。例如：

1E2表示1×10^2　　-.5E-2表示-0.5×10^{-2}　　13.22E0表示13.22×10^0

浮点常量中的“E”可以写为小写“e”，也可以用“D”或“d”来代替。

可以在浮点型常量后面加感叹号“!”或井号“#”来指明该常量是单精度浮点型常量还是双精度浮点型常量。否则，Visual Basic会根据数值大小自动识别。

3.2.3　字符串型常量

字符串常量必须使用英文的双引号“”把实际的文本括起来。双引号称为字符串的**“界定符”**，表示字符串的开始与结束。下面是几个字符串常量：

```
"你好!"  "␣"  "12.34"  "a0123"  "Let It Be"  ""
```

字符串常量中可以包括任何可输入的字符，包括汉字、中文标点、英文和英文符号。空格也是合法的字符，上面的第二个字符串就是由空格组成的。如果两个引号之间没有任何字符，表示一个**空字符串**(上面的最后一个)。空字符串是特殊的字符串，下面的语句使用空字符串常量清空文本框的内容：

```
Text1.Text = ""          '清空文本框内容
```

> 因为双引号当作了字符串常量的界定符，所以如果要在字符串中包含双引号，就要使用两个连续的双引号表示一个双引号，例如，下面的语句在文本框中只显示一个双引号。
>
> ```
> Text1.Text = "这是一个双引号："""
> ```
>
> 总之，一个字符串中的双引号应该成对使用。如果字符串中包括汉字的全角双引号，与普通字符相同，不必进行特殊处理。

3.2.4 逻辑型常量

逻辑型常量只有两个：True 和 False。注意，它们没有任何界定符。"True"与"False"不是逻辑值，而是字符串常量。

3.2.5 日期时间型常量

日期时间型常量既可以表示一个日期，也可以表示一个时间，或者同时表示日期与时间。日期时间型常量使用“#”号作界定符。一般可辨认的表示日期时间的文本都可以作为日期时间型常量。例如下面 4 个常量都表示同一个日期：“2008 年 1 月 2 日”：

```
#1/2/2008#    #2008-1-2#    #Jan 2,2008#    #January 2,2008#
```

下面是一些时间常量：

```
#12:00:00 PM#(中午 12 点)    #12:00:00 AM#(午夜 12 点)
#8:20:20 PM#                 #2:00:00 PM#                 #2:14:02#
```

下面是两个日期时间常量：

```
#1/2/2008 2:14:02 AM#        #11/12/2008 6:00:00 PM#
```

分别表示“2008 年 1 月 2 日凌晨 2 点 14 分零 2 秒”和“2008 年 11 月 12 日下午 6 点整”。

在“代码”窗口中输入日期时间常量时，Visual Basic 会自动转换为内部统一形式。1930—2029 年之间日期的年份，会被省略掉前两位数，如“08”表示“2008”年。时间值的表示采用 12 小时制(上午为“AM”、下午为“PM”)。

3.3 变　　量

变量(Variable)是程序运行过程中用来保存临时数据所占用的内存空间，所以也称为“内存变量”。程序通过变量名来操作变量，每个变量有一定的数据类型、作用范围，占用一定字节的内存空间。熟练地使用变量是学习编程的必经之路。

图 3.1 展示了一个变量从开辟内存开始，直到被从内存中清除为止，中间的这一段时间被反复赋值和取值的过程。

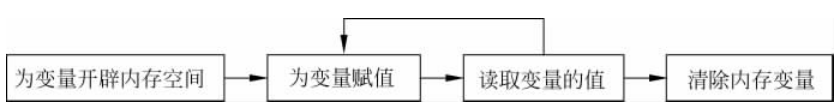

图 3.1　内存变量的生存期

3.3.1 变量命名规则

Visual Basic 对变量名有以下要求：

(1) 以字母开始，可以包括数字、字母和下划线。

(2) 不能包含标点符号。

(3) 不能多于 255 个字符。

(4) 不能与关键字重复。

(5) 在同一作用域中,变量名不能互相重复。

虽然变量名中可以包含汉字,但是不建议这样做。

下面列出的是一些合法的变量名:

```
Abc    Name    intAge    x12    My_var1    PI
```

下面是一些非法的变量名:

```
12ab    _x    ab.cd    $MyVar    Call    x[1]  a+b  :
```

当变量名不符合规则时,Visual Basic 编辑器会显示错误信息。

与对象名同理,给变量命名时也应该注意它的描述性。每一种数据类型都有一个约定前缀(见附录 C),可以使用前缀与英文单词或汉语拼音形成有意义的变量名。例如,可以使用 strUserName 作变量名来保存用户姓名。

3.3.2　定义变量

定义变量就是为变量分配内存空间,也称为"声明变量"。定义变量时需指定变量名、数据类型以及作用域。定义变量的统一语法格式如下:

```
Public|Private|Dim|Static␣变量名␣[As␣数据类型名␣[*字符串长度]]
```

其中的"Public|Private|Dim|Static"4 个关键字指定变量的作用范围(即作用域)。如果省略"As 数据类型名"部分,定义的是变体类型变量。当定义定长字符串变量时,需要"*字符串长度"部分指定字符串长度。

变量的作用域是指变量生效的范围,即能够对该变量赋值又能读取该变量值的代码范围。在 Visual Basic 中,变量有三种作用域:过程级、模块级和全局级。下面分别介绍三种作用域变量的定义方法。

1. 过程级变量

过程级变量又称为局部变量,它的作用域是定义它的过程(包括事件过程和第 6 章讲到的通用过程)。也就是说,它在哪个过程中定义就只能在这个过程中使用。Visual Basic 允许在过程中的任何位置定义过程级变量,但只有在定义之后的语句才能使用该变量。定义过程级变量的语句为:

```
Dim|Static␣变量名␣[As␣数据类型名␣[*字符串长度]]
```

使用 Dim 关键字定义的过程级变量的特点是当所在过程执行完毕,变量就会消失,释放所占用的内存。下一次执行该过程时,重新给变量分配内存空间。

使用 Static 关键字定义的过程级变量被称为"静态变量"。静态过程级变量在程序启动时即被分配内存空间,程序结束时清除,所以在每次过程执行完毕后变量的值仍被保留,下一次该过程被执行时变量的值仍然可用。

下面定义了三个不同类型的过程级变量:

```
Dim intMyVar1 As Integer          '定义整型变量 intMyVar1
Dim blnSex As Boolean             '定义逻辑型变量 blnSex
```

```
Static strName As String          '定义字符串型静态变量 strName
```

2. 模块级变量

定义模块级变量的语句必须放在模块开始的通用声明段中(位于“代码”窗口最顶部,所有过程的前面)。定义模块级变量的语法格式为:

Private|Dim␣变量名␣[As␣数据类型名␣[*字符串长度]]

其中,关键字 Private 和 Dim 是等效的。

模块级变量的作用域是所在的模块,也就是说,定义变量的这个模块中的所有过程都可以访问该变量。模块级变量在程序启动时被创建,程序结束时被清除。

3. 程序级变量

程序级变量也称为全局变量或公共变量,是指在程序的所有模块中都可以对其值进行存取的变量。

与模块级变量相同,全局变量必须在模块开头的通用声明段中定义,语法格式如下:

Public␣变量名␣[As␣数据类型名␣[*字符串长度]]

与模块级变量、静态过程级变量相同,全局变量也是在程序启动时创建,程序结束时被清除。

> 在窗体模块中不能定义全局定长字符串型变量。应在标准模块中定义全局定长字符串变量。
>
> 在模块顶部的通用声明段中,不能有赋值、过程调用等执行语句,只能进行变量、常量和数据类型的定义。

4. 变量的作用域

由前面的叙述可知,定义变量时使用的关键字和定义变量的位置决定了变量的作用域。图 3.2 形象地说明了全局变量、模块级变量和过程级变量作用域的覆盖范围。

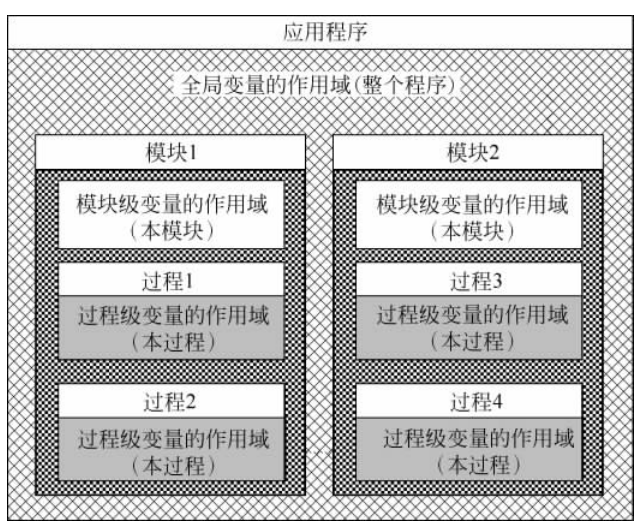

图 3.2　变量的作用域

程序中某个过程中的语句可以存取本过程中定义的过程级变量、所在模块定义的模块级变量、任意模块定义的全局变量，不能存取其他过程中定义的过程级变量和其他模块中定义的模块级变量。

访问变量　在过程中访问过程级变量、模块级变量、本模块定义的全局变量、标准模块中定义的全局变量时，可以直接使用变量名。如：

```
int1 = 1              '为变量赋值
```

访问其他窗体模块中的全局变量时，必须在变量名前加模块名和“.”加以限定。例如，在窗体模块 Form1 的过程中给窗体模块 Form2 中定义的全局变量 int2 赋值，必须使用如下的语句：

```
Form2.int2 = 2        '为其他窗体模块中定义的全局变量赋值
```

5. 一条语句定义多个变量

Visual Basic 允许使用一条语句同时定义多个变量。语法格式如下：

Public | Private | Dim | Static ␣ 变量 1 ␣ [As 类型 1][,变量 2 [As 类型 2]…]

语句中每个变量都要指明类型，否则被定义为变体类型。例如，下面语句定义了 3 个模块级变量，其中 a 和 b 是变体类型，c 是整型：

```
Private a, b, c As Integer
```

下面的语句定义的 3 个变量都是整型全局变量：

```
Public a As Integer, b As Integer, c As Integer
```

下面语句定义的 3 个变量中，a 是逻辑型变量，b 是变体类型变量，c 是日期型变量：

```
Dim a As Boolean, b, c As Date
```

变量的默认值　一个变量在被定义之后、被首次赋值之前的这一段时间中具有默认值。对于不同的数据类型，默认值不相同。

(1) 数值型变量的默认值为 0。

(2) 逻辑型变量的默认值为 False。

(3) 日期时间型变量的默认值为 #0:00:00# 。

(4) 变长字符串变量的默认值为空字符串" "。

(5) 定长字符串变量的默认值是全部由空格组成的字符串，空格个数等于定长字符串的字符个数。

(6) 对象型变量的默认值为 Nothing。

(7) 变体类型变量的默认值为 Empty。

6. 强制变量定义

默认情况下，Visual Basic 并不要求变量必须定义，未被定义的变量被认为是 Variant

类型的过程级变量。

但是,这种默认的设置很容易引起问题,因为如果把变量名拼写错误,会被 Visual Basic 当作另外的变量,不能被检查出来。为了避免上述问题,需要强制变量定义,方法是在模块的通用声明段中加上如下语句:

```
Option Explicit
```

这样,该模块中所有的变量必须先进行定义才能被存取(赋值或取值),否则会引起"变量未定义"的错误(如图 3.3 所示)。

可以改变默认的设置来强制变量定义,在"工具"菜单的"选项"对话框中选择"编辑器"选项卡,并选择"要求变量声明"复选框。这样,Visual Basic 会自动在新建模块的声明段中加上 Option Explicit 语句。已有的模块必须手工添加这一语句。

图 3.3　错误信息

3.3.3　变量的赋值与取值

变量是一个内存区域,程序代码通过变量名对其进行赋值和取值操作。给变量赋值使用的是如下的赋值语句:

[Let]␣变量名 = 表达式

上述赋值语句中"Let"是可以省略的关键字;"="是赋值号,而不是等号;"表达式"指的是用运算符连接对象属性、常量、变量、函数形成的具有一定值的算式(单个属性、常量、变量、函数是表达式的特例)。赋值语句把赋值号"="右边"表达式"的值写到"变量名"所代表的内存空间中,原来的值被覆盖,不复存在。如果表达式值的数据类型与变量的数据类型不同,在赋值时会进行数据类型转换。

下面是一条定义变量的语句和一条赋值语句:

```
Dim a As Integer          '定义变量 a,开辟 2 个字节内存空间,默认值为 0
a = 10                    '赋值语句,把整型常量 10 赋给变量 a,a 的值变为 10
```

要读取变量的值,只需将变量名写到表达式中。在表达式中,变量名即代表变量的当前值。读取变量的值时,并不会改变变量的值。

```
b = a                     '读取变量 a 的值赋给变量 b,变量 a 的值不变
a = b                     '读取变量 b 的值赋给变量 a,变量 b 的值不变
a = a + 1                 '把变量 a 的值与 1 相加后再赋给 a,即 a 的值增加 1
a = a + b - c             '把变量 a 的值与变量 b 的值相加再减去变量 c 的值,
                          '计算结果赋给变量 a,a 的值发生变化,b 和 c 的值不变
```

> 当给数值型变量赋一个超出其表示范围的值,会导致"溢出"错误。

值得注意的是,如果要把相同的值赋给两个变量,应该使用两个赋值语句,如:

```
a = 2: b = 2
```

不能写成这样:

```
a = b = 2          '错误,不能以这种方式将 2 同时赋给变量 a 和 b
```

上式也是一条赋值语句，但是意义不同，将在第 4 章讲解。

赋值号的左边只能是单个变量名或对象的属性名，不能是表达式。例如，下面的语句是错误的：

```
a + b = 2              '错误，赋值号左边不能是表达式，无法给表达式赋值
```

【例 3.1】 使用过程级变量。

创建工程，建立图 3.4 所示的窗体界面，窗体和两个按钮的对象名分别是 Form1、Command1 和 Command2。在“代码”窗口中编写如下两个按钮的事件过程：

```
Private Sub Command1_Click()
    Dim i As Integer          '动态过程级变量
    i = i + 1
    Command1.Caption = i
End Sub

Private Sub Command2_Click()
    Static i As Integer       '静态过程级变量
    i = i + 1
    Command2.Caption = i
End Sub
```

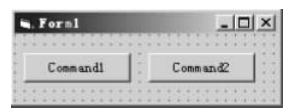

图 3.4　窗体界面（例 3.1）

启动程序，连续多次单击两个按钮，看看程序是如何执行的。

连续单击按钮 Command1，它上面显示的数字一直是“1”。因为每次单击按钮，都会引发 Click 事件，执行一次 Command1_Click 事件过程。首先，执行 Dim i As Integer 语句，为变量 i 开辟内存，这时它的初始值为 0；然后，执行 i=i+1 语句，变量 i 自增 1，它的值为 1；最后，执行 Command1.Caption=i 语句，按钮的 Caption 属性被赋值为 1（由整数 1 转换为字符串“1”），按钮表面上显示“1”。每次过程执行完毕，自动释放非静态过程级变量所占用的内存空间，下一次执行该过程的情况与上一次完全相同，因此按钮表面上的数字一直是“1”。

单击按钮 Command2 的情况就不同了。因为 Command2_Click 事件过程中使用了 Static 关键字，将变量 i 定义为静态过程级变量；第一次单击 Command2 之前变量 i 已被创建（默认值为 0），单击后增加 1，然后赋值给按钮 Command2 的 Caption 属性，显示在按钮表面；过程结束时变量 i 并不会被清除，下一次单击执行事件过程时不再重新开辟内存，它的值继续增加 1。所以，连续单击 Command2，会使它显示的数字以 1、2、3、…不断递增。整个程序结束时，静态过程级变量才会被从内存中清除。

注意，虽然 Command1_Click 和 Command2_Click 两个事件过程中都有名为 i 的过程级变量，但它们却是两个完全不同的变量（占用不同的内存空间），互不干扰，分别在所处的过程中起作用。

【例 3.2】 使用模块级变量。

创建与例 3.1 相同的窗体界面（如图 3.4 所示）。在“代码”窗口编写以下程序代码：

```
Dim i As Integer              '模块级变量

Private Sub Command1_Click()
  Static i As Integer         '静态过程级变量
  i = i + 1
  Command1.Caption = i
```

```
End Sub

Private Sub Command2_Click()
  Static i As Integer          '静态过程级变量
  i = i + 1
  Command2.Caption = i
End Sub

Private Sub Form_Click()
  i = i + 1
  Form1.Caption = i
End Sub
```

启动程序,连续多次单击窗体客户区和两个按钮,看看程序是如何执行的。

连续单击两个按钮控件,每个按钮上的数字都会以1、2、3、…不断递增。连续单击窗体客户区,标题栏上的数字也会以1、2、3、…不断递增。两个按钮的事件过程中分别定义了名为i的静态过程级变量,能够维持各自的递增。因为Form_Click事件过程中未定义任何过程级变量,所以过程中的i代表的是模块级变量i。每次单击窗体,Form_Click事件过程即被执行一次,模块级变量i会增加1,并显示到标题栏上。

【例3.3】 为模块级变量赋初值。

创建与例3.1相同的窗体界面(如图3.4所示)。在"代码"窗口编写以下程序代码:

```
Dim i As Integer             '模块级变量

Private Sub Command1_Click()
   i = i + 1
   Command1.Caption = i
End Sub

Private Sub Command2_Click()
   i = i + 1
   Command2.Caption = i
End Sub

Private Sub Form_Click()
   i = i + 1
   Form1.Caption = i
End Sub

Private Sub Form_Load()
   i = 10
End Sub
```

程序代码中有4个事件过程,都未定义局部变量,所以它们中的i都是指的同一个变量:即模块级变量i。程序启动时,首先执行窗体的Load事件过程,为变量i赋初值10。然后,无论单击的是按钮还是窗体,都会为i加1,然后由被单击的对象显示。单击窗体和按钮,显示的数值会相互攀升,而不是单独递增。

> 窗体的Load事件过程是为模块级变量、全局变量赋初值,为控件进行初始化的最佳位置。

3.3.4　变量的同名问题

1. 不允许同名的情况

一般情况下，在同一作用域内不能定义重名的变量。

(1) 同一个过程中不能定义两个或更多的同名过程级变量，即使类型不相同也不能同名。

(2) 同一个模块中不能定义同名的模块级变量。

(3) 同一个模块中不能定义同名的全局变量。

(4) 同一个模块中的模块级变量和全局变量不能同名。

2. 允许同名的情况

(1) 不同的过程中可以定义同名的过程级变量。

(2) 不同的模块中可以定义同名的模块级变量。

(3) 过程中可以定义与模块级变量同名的过程级变量。

(4) 过程中可以定义与全局变量同名的过程级变量。

(5) 模块中可以定义与其他模块定义的全局变量同名的模块级变量。

(6) 不同的模块中可以定义同名的全局变量。

3. 变量同名时的情况

(1) 不同作用域的变量同名时，作用域小的变量会屏蔽作用域大的变量，即过程级变量屏蔽模块级变量和全局变量，模块级变量屏蔽全局变量。例如，在例 3.2 中，按钮事件过程中的变量 i 屏蔽模块级变量 i，过程中被访问的 i 实际上是过程级变量。

(2) 如果不同模块中全局变量同名，访问其他模块中定义的全局变量时应添加模块名进行限定(形式为“模块名.变量名”)。访问本模块或标准模块中定义的全局变量时不必进行限定。如果本模块与标准模块中的全局变量同名，访问标准模块中的全局变量时也应加模块名进行限定。

(3) 当全局变量与过程级变量同名时，在过程中直接使用这个变量名时，指的是过程级变量。如果使用定义全局变量的模块名来限定变量名，则可访问该全局变量。

(4) 如果本模块中的模块级变量与其他模块中的全局变量同名，可以在变量名前加模块名来访问全局变量。

4. 变量与对象的同名情况

在窗体模块中，窗体的属性名、窗体的方法名、窗体模块中的事件过程名和通用过程名、窗体上的控件名与模块级定义的变量被认为是同一层次的，所以模块级变量名和全局变量名不能与前面提到的所有名称相重复。

例如，在窗体模块中，不能定义一个名为 Caption 或 Move 的模块级变量或全局变量，因为它们是窗体的属性名和方法名。

过程中的局部变量比过程要低一级，所以，与控件同名的过程级变量会屏蔽控件，如要访问该控件，要用窗体名来限定。

例如，在窗体 Form1 上有文本框 Text1 与按钮 Command1。如果在 Command1_Click 事件过程中定义了一个名为 Text1 的过程级变量。那么，在这个事件过程中，语句 Text1 = "您

好"是为过程级变量 Text1 赋值,而 Form1. Text1. Text = "您好"才是为控件 Text1 的 Text 属性赋值。语句 Text1. Text = "您好"会产生语法错误,因为 VB 把 Text1 理解为变量,而不是控件。

> 在定义变量时,尽量使用作用范围小的变量,也就是说:能使用过程级变量解决问题,就不使用模块级变量;能使用模块级变量完成任务,就不使用全局变量。

3.3.5 定长字符串与变长字符串变量

字符串类型(String)的变量分为两类:定长字符串与变长字符串。

变长字符串变量的定义语句为:

```
Public |Private |Dim |Static ␣ 变量名 ␣ As ␣ String
```

定长字符串变量的定义语句为:

```
Public |Private |Dim |Static ␣ 变量名 ␣ As ␣ String ␣ * ␣ 字符串长度
```

上面语句中的"字符串长度"应是正整数常量,表示的是字符数。例如:

```
Dim str1 As String              '定义变长字符串变量
str1 = "你好!"                  '为变长字符串变量赋值
Dim str2 As String * 4          '定义定长字符串变量
str2 = "你好吗?"                '为定长字符串变量赋值
str2 = "我今天很好!"            '会截尾,赋值后变量的值为:"我今天很"
```

定长字符串变量所占的内存空间是一定的,当其中的字符信息没达到这个长度时,所剩的空间用空格字符填充。如果给定长字符串变量赋一个超过其长度的字符串,会截掉多余部分,不会出现"溢出"错误。

> 注意,不能在窗体模块或类模块中定义全局定长字符串变量,全局定长字符串变量只能在标准模块中定义。

3.3.6 对象型变量

对象型(Object)变量占用 4 个字节的内存空间,保存的是对某个对象的引用,程序通过对象型变量可以间接地对它所引用对象进行操作。

定义对象型变量的语句是:

```
Public |Private |Dim |Static ␣ 变量名 ␣ As ␣ Object |Control |对象类型名
```

使用 Object 关键字定义的变量可以引用任何一种类型的对象。使用 Control 关键字定义的变量能够引用所有的控件对象。而使用一个具体的"对象类型名"(如 TextBox、Label 等)定义的对象型变量只能引用指定类型的对象。

给对象型变量赋值使用 Set 语句:

```
Set ␣ 对象型变量名 = 对象型表达式
```

赋值号“＝”右边的“对象型表达式”可以是对象名，也可以是另一个对象型变量，还可以是返回对象型值的方法或函数。

例如，假设有一个名为 cmdOK 的按钮，可以通过以下的 3 条语句来设置其 Caption 属性：

```
Dim objButton As Object                    '定义对象型变量
Set objButton = cmdOK                      '把按钮对象赋给对象型变量
objButton.Caption = "OK"                   '通过对象型变量设置按钮对象的 Caption 属性
```

定义特定类型对象型变量，要使用对象的类型名，如已学习过的 Form、TextBox、CommandButton 和 Label 等。特定类型对象型变量只能引用同一类型的对象。例如：

```
Dim objLabel As Label                      '定义 Label 标签类型变量
Set objLabel = cmdOK                       '错误！类型不匹配，cmdOK 为按钮对象
```

对象型变量在被定义之后、赋值之前的这段时间，它的值（即默认值）是一个特殊值：Nothing，表示未引用任何对象。也可以通过给对象型变量赋 Nothing 值，使之不引用任何对象。

```
Set objButton = Nothing                    '使对象型变量不再引用任何对象
```

3.3.7　变体数据类型

变体类型（Variant）是一种特殊的数据类型，该类型的变量可以存储几乎所有系统定义类型的数据（自定义类型除外），为其赋不同类型的值，就会变成相应的类型。变体变量在存放数值时（包括整型、浮点型等），占 16 个字节的内存；存放字符串时，占用内存量是字符串的实际长度与 22 个额外字节之和。

1. 变体变量的定义与赋值

定义变体类型变量的语法为：

Public | Private | Dim | Static ␣ 变量名 ␣ [As ␣ Variant]

因为变体类型是 Visual Basic 的默认类型，在定义变体变量时可以省略“As Variant”。变体变量可以引用对象，当为变体变量赋对象型值时，必须使用 Set 语句。

例如：

```
Dim vnt1 As Variant                        '定义变体类型变量
vnt1 = "17"                                '数据类型变为字符串型，值为"17"
vnt1 = 15                                  '数据类型变为数值型，值为 15
Set vnt1 = cmdOK                           '类型变为对象型，是对 cmdOK 控件的引用
```

变体类型占用内存较大，运算速度很慢，如果使用常规类型能够解决问题，不应使用此类型。

2. 变体类型的特殊值

除了其他类型数据允许的取值之外，Variant 变量还可以有两个特殊的值：Empty 和 Null。

（1）Empty。此值是变体变量的默认值（即定义之后、赋值之前，变体变量的值为 Empty），也可通过赋值语句将 Empty 值赋给变体变量。

（2）Null。表示数据未知或数据不确定（主要用于对数据库的操作）。Null 值有如下特点：①如果表达式的任何一部分是 Null，则整个表达式的值也为 Null（逻辑运算除外）；

②把 Null 值作为参数传递给一个函数,则函数的返回值为 Null;③可以给一个变体变量赋 Null 值;④Null 值与任何值都不相等,更不等于 0,甚至与其本身也不相等。

3.3.8 类型转换

类型转换是指把数据从一种类型转换为另一种类型。类型转换发生在以下 3 种情况:

- 为变量和属性赋值时。如果赋值号左边的变量或属性的类型与被赋的值类型不一致,会发生类型转换。
- 计算表达式时。表达式中,如果运算量的类型与运算符的要求不符,则进行类型转换。
- 参数传递时。在调用对象的方法或通用过程时,如果提供的参数与要求的类型不一致,则进行类型转换。

1. 隐式转换

隐式转换,也称为"默认转换",是指不使用专门的类型转换函数,按默认规则进行转换。

(1) 数值型之间的转换

不同的数值类型之间可以互相转换。把整数转换为浮点类型时,存储格式转换,数值的大小不变。当把浮点数转换为整型数时,小数部分要"四舍五入"为整数,如果小数部分恰好是 0.5,则要向最近的偶数靠拢。

例如,下面语句将浮点类型的常量 4.56 赋给整型变量 i,在赋值过程会发生默认类型转换,变量的值变为 5。

```
i = 4.56                          '浮点数转换为整数,变量 i 的值为 5
```

如果被赋的值为 4.5,则转换为整数 4。

```
i = 4.5                           '转换时向最近的偶数靠拢,变量 i 的值为 4
```

因为一种类型可以表示的值,另一种类型可能不能表示,所以在类型转换时,应注意避免出现"溢出"错误。

(2) 字符串类型与数值型之间的转换

所有的数值都可以转换为字符串类型,反之则不然,只有字符串内容全部是数值信息的才可以转换为数值型。

例如,下面的语句向字符串变量 s1 赋一个数值:

```
s1 = 12.34                        's1 的值为"12.34"
```

下面的语句向整型变量 j 赋字符串值:

```
j = "1.2e3"                       'j 的值为 1200
```

如果无法进行默认转换,则显示如图 3.5 所示的"类型不匹配"错误。例如,下面的语句将包含字母的字符串赋给数值型变量 k,会出错。

```
k = "abc123"        '出错! 无法转换,类型不匹配
```

图 3.5 "类型不匹配"错误

注意,空字符串不能赋给任何的数值类型,否则也会出现"类型不匹配"错误。如:

```
k = ""              '出错! 类型不匹配
```

(3) 逻辑型值的转换

由逻辑型转换为数值型的规则是：False 转换为 0，True 转换为－1。逻辑型转换为字节型时，False 转换为 0，True 转换为 255。由数值型转换为逻辑型时，0 转换为 False，非 0 值均转换为 True。

逻辑型值转换为字符串时，True 和 False 分别转换为"True"和"False"。

把字符串转换为逻辑型时，只有"True"和"False"(或其他大小写形式，如"TRUE"和"FALSE")可以分别被转换为 True 和 False。其他任何字符串都不能转换为逻辑值，会出现“类型不匹配”错误。

(4) 日期时间类型的转换

日期时间型值转换为字符串时，会按日期的短格式(可以在 Windows 控制面板的“区域设置”中设置)转换为相应的字符串。例如：

```
s2 = #2/1/99 8:20:00#          '字符串变量 s2 的值为"99-2-1 8:20:00"
```

表示有效时间的字符串可以转换为日期时间型值。例如：

```
d1 = "13:23:34"                '日期型变量 d1 的值为 #1:23:34 PM#
```

日期时间型数据转换为数值型时，日期部分转换为数值的整数部分，它的值为此日期距 1899 年 12 月 30 日的天数；时间部分转换为小数部分，从午夜零点到该时刻占一整天 24 小时的比例，中午 12:00:00 转换为 0.5。例如，下面的赋值语句将日期值赋给浮点型变量：

```
sng1 = #1/1/2000 6:00:00AM#    'sng1 的值为 36526.25
```

如果把日期时间型数据转换为整型，则时间部分会被忽略掉。把数值型转换为日期时间型是把日期时间型转换为数值型的逆过程。

变体类型的 Empty 值向数值型转换时变为 0，向字符串型转换时变为空字符串""。

2. 显式转换

显式转换是指使用 Visual Basic 提供的类型转换函数进行转换(见表 3.2)。使用这些类型转换函数可使程序的可读性增强，并且能进行强制类型转换，避免可能出现的歧义性。

表 3.2　类型转换函数

函数	转换为	函数	转换为	函数	转换为
CBool( )	Boolean	CByte( )	Byte	CInt( )	Integer
CDate( )	Date	CDbl( )	Double	CStr( )	String
CLng( )	Long	CSng( )	Single		
CVar( )	Variant	CCur( )	Currency		

使用这些函数的方法是，在括号中填入被转换的数据(可以是属性、常量、变量、函数以及由它们构成的表达式)，然后把转换函数放入表达式，则转换的结果就会参与表达式的计算。

显式转换的规则和隐式转换相同，隐式不能转换的，显式也不能转换。例如：

```
i3 = CInt(123.5)               '整型变量 i3 的值是 124
f1 = CDbl("x1.2")              '类型不匹配错误！以字母开头的字符串无法转换为数值
```

除了表 3.2 中列出的类型转换函数外，VB 还提供了其他的一些与类型转换有关的内部函数，如 Int、Fix、Val 和 Format 等，将在第 9 章中介绍。

3. 不能进行转换的情况

在数据类型默认转换过程中可能出错的情况归纳为以下 4 类：

(1) 包含非数值字符(字母和符号)的字符串向数值型转换时出现“类型不匹配”错误。

(2) 非"True"或"False"的字符串向逻辑型转换时，出现“类型不匹配”错误。

(3) 非日期内容的字符串向日期型转换时，出现“类型不匹配”错误。如试图将"abc"转换为日期型。

(4) 转换时超出目标类型的表示范围，出现“溢出”错误。如试图将字符串"－1"转换为 Byte 类型。

3.3.9 类型声明符*

除了前面讲的变量定义语句之外，还可以使用“类型声明符”来定义变量。有了类型声明符，就不必使用 Dim 等语句来指定变量的数据类型，只需在第一次使用时，在变量名的后面加一个类型声明符即可。Visual Basic 支持的类型声明符与相对应的数据类型列于表 3.3 中。

表 3.3 类型声明符

声明符	数据类型	声明符	数据类型
$	变长字符串型	!	单精度浮点型
%	整型	#	双精度浮点型
&	长整型	@	货币型

例如：

```
x% = 5                    '定义一个整型变量,并赋值
```

变量的类型确定之后，不能再改变它的类型。例如，下面的语句是错误的：

```
x% = 3: x& = 5            '错误
```

上述定义变量的方法只适用于过程级变量，并且要求模块声明段中没有使用 Option Explicit 语句。

另外，也可以在 Dim|Private|Public|Static 语句中使用类型声明符，但是要省略掉“As 类型名”部分。这种方法可以定义全局变量、模块变量和过程级变量，并且不受 Option Explicit 语句的制约。

例如，下面语句可以定义整型变量 int1。

```
Dim int1%
```

Visual Basic 中的类型声明符主要是为了与旧版本的 Basic 语言相兼容，在实际应用中应该使用 As 关键字定义变量。

3.3.10 DefType 语句*

默认情况下，当不强制变量定义时，未显式定义的变量被 Visual Basic 认为是变体类型变量，因为 Variant 是默认的变量类型。如果要更改默认变量类型，就要在模块的声明段中使用表 3.4 中所列的 DefType 语句族中的相应语句。

表 3.4　DefType 语句

语句	数据类型	语句	数据类型	语句	数据类型
DefBool	Boolean	DefDate	Date	DefSng	Single
DefInt	Integer	DefObj	Object	DefDec	Decimal
DefCur	Currency	DefByte	Byte	DefStr	String
DefDbl	Double	DefLng	Long	DefVar	Variant

表 3.4 列出的 DefType 语句族中所有语句的用法相同：

DefType␣字母范围 1[，字母范围 2[，字母范围 3…]]

上面的语法中，DefType 可以是表 3.4 中任一语句，“字母范围”指的是单个字母或者是使用连字符“－”(同减号)连接起来的两个字母。

例如，下面语句定义了以 A、D～F(即 D、E、F)、W～Z(即 W、X、Y、Z)为首字母的所有未经显式定义的变量为 Integer 型：

```
DefInt A,D-F,W-Z
```

多个 DefType 语句可以同时使用，但是所定义的字母范围不能相互重叠。

注意，这些 DefType 语句只能用在模块声明段，只在使用它的模块中起作用。它们只能在未使用 Option Explicit 的模块中使用时起作用，只对未显式定义的变量起作用。

3.4　符号常量

符号常量是常量的一种，区别于直接常量。符号常量与变量相似，属于某一数据类型并有名称，先定义后使用。定义时必须指定符号常量的值，在运行过程中它的值不能改变(即不能被赋值)。

符号常量与直接常量相比，优点有：①便于记忆与识别，可以使用具有描述性的名字替代一个抽象的值，如使用 PI 代表数值 3.1415926；②便于程序的修改，如果要改变常量所代表的值，只需在定义常量的位置修改一次即可。

符号常量也有作用域，由定义时使用的语句和位置决定。定义符号常量的方法如下：

(1) 过程级常量。在过程中定义，作用域为所在的过程。语法为：

Const␣常量名␣[As 数据类型名] = 表达式

(2) 模块级常量。在模块的通用声明段中定义，可在本模块的所有过程中使用。语法为：

[Private]␣Const␣常量名␣[As␣数据类型名] = 表达式

(3) 全局常量。在标准模块的声明段中定义，程序的所有模块均可使用。语法为：

Public␣Const␣常量名␣[As␣数据类型名] = 表达式

如果省略“As 类型名”部分，定义的是变体类型的常量。常量的命名规则与变量相同，一般可为常量名加上“con”前缀，或使用大写字母以示区别。

> 注意,与全局变量不同,全局常量只能在标准模块中定义,不能在窗体模块和类模块中定义。

例如,下面语句定义了模块级常量 PI 来代替数值 3.141593,并用来计算圆的面积:

```
Private Const PI As Single = 3.141593     '定义单精度浮点类型符号常量 PI
s = PI * r * r                            '通过半径 r 计算圆的面积,然后赋给变量 s
```

定义符号常量时,可以使用不包括函数和变量的常量表达式来赋值(表达式中可以使用除 Is 外所有的算术运算符与逻辑运算符),也可以使用其他已定义的常量来赋值。例如:

```
Const con1 As Integer = PI + 2.5          '正确,如果 PI 是已定义的符号常量
Const con2 = i + Sin(j)                   '错误! 不能使用函数 Sin 和变量 i、j
```

可以使用一条语句定义多个符号常量,方法与定义变量相似。

除了自定义的符号常量外,Visual Basic 中还有大量预定义的符号常量,由系统提供,一般以“vb”为前缀。这些预定义的符号常量可以直接使用。

习 题 3

一、选择题

1. Integer 类型的变量可存放的最大整数为_____。

(A) 255　　(B) 256　　(C) 32768　　(D) 32767

2. 下面的 4 对数据类型中,_____所占的内存字节数相等。

(A) Integer 和 Boolean　　(B) Integer 和 Single

(C) Date 和 Single　　(D) Long 和 Double

3. 下列数据类型中,占用内存最小的是_____。

(A) Boolean　　(B) Byte　　(C) Integer　　(D) Single

4. 使用 Public Const 语句定义全局常量,该语句可以放在_____。

(A) 过程中　　(B) 窗体模块的声明段中

(C) 标准模块的声明段中　　(D) 窗体模块或标准模块的声明段中

5. 在窗体模块的声明段中定义变量时,不能使用关键字_____。

(A) Dim　　(B) Private　　(C) Public　　(D) Static

6. _____数据类型的变量不能存放负值。

(A) Integer　　(B) Single　　(C) Byte　　(D) Long

7. _____不是字符串常量。

(A) "你好"　　(B) " "　　(C) "True"　　(D) #False#

8. 下面列出的语句中,没有错误的是_____。

(A) txt1. Text + txt2. Text = txt3. Text

(B) cmdAdd. Name = cmdSub

(C) 12Label. Caption= 1234

(D) frmFirst. Move 1000,1000,2000,1200

9. 变量名最多不能超过的字符个数为_____。

(A) 10　　(B) 12　　(C) 40　　(D) 255

10. _____是日期型常量。

(A) "2/1/99"　　(B) 2/1/99　　(C) #2/1/99#　　(D) {2/1/99}

11. 下面赋值语句中，_____不能使字节型变量 byt1 在内存中的二进制位成为 00001111。

(A) byt1=15　　(B) byt1=1111　　(C) byt1=&HF　　(D) byt1=&O17

12. 下列语句中，_____会产生错误。

(A) Dim int1 As Integer: int1=True

(B) Dim str1 As string * 10: str1="123.4.5"

(C) Dim int1 As Integer: int1="123.4"

(D) Dim bln1 As Boolean: bln1="Yes"

二、填空题

1. 下列数据类型的变量各占多少字节的内存：

Byte：__(1)__；Integer：__(2)__；Long：__(3)__；Single：__(4)__；Double：__(5)__。

2. 把整型数 1 赋给一个逻辑型变量，则逻辑型变量的值为__(6)__。

3. 刚被定义尚未赋值的日期型变量的值为__(7)__；逻辑型变量的值为__(8)__；对象型变量的值为__(9)__；变体变量的值为__(10)__。

4. 对象型变量可以引用一个对象。使用 Dim objFirst As Object 语句定义一个对象型变量，如果要把名称为 cmdFirst 的命令按钮赋予它，应使用__(11)__语句。

5. 在一条 Dim 语句中可以定义多个变量，如 Dim strVar, intVar, sngVar As Integer，则 strVar、intVar 与 sngVar 的数据类型分别是__(12)__、__(13)__和__(14)__。

6. 如果 int1 是整型变量，则执行 int1="2"+3 语句之后，int1 的值为__(15)__；执行 int1="2"+"3"语句之后，int1 的值为__(16)__。

7. 把逻辑值 True 赋给整型变量之后，此变量的值会变为__(17)__。

8. 默认情况下，所有未经显式定义的变量均被视为__(18)__类型。如果要强制变量的定义，应在模块的声明段使用__(19)__语句。

9. 如果要在文本框 Text1 中显示“He said, "Good morning!".”(注：不包括外层的中文双引号，内层是英文双引号)，则应使用以下的赋值语句：Text1.Text = __(20)__。

10. 新建工程，建立如图 3.6 所示的窗体界面，文本框和命令按钮的对象名分别是 Text1 和 Command1。

在“代码”窗口中输入以下程序行：

```
1  Public int1 As Integer          '①
2  Dim int1 As Integer             '②
3  Private int1 As Integer         '③
4  Private Sub Command1_Click()
5      Dim int1 As Integer         '④
6      Static int1 As Integer      '⑤
```

图 3.6　填空第 10 题

```
7       int1 = int1 + 1
8       Text1.Text = int1
9   End Sub
```

其中语句①～⑤同时只用一条。如果运行程序,连续单击 Command1 按钮三次,则使用语句①时文本框中显示的是__(21)__;使用语句②时显示的是__(22)__;使用语句③时显示的是__(23)__;使用语句④时显示的是__(24)__;使用语句⑤时显示的是__(25)__。

三、判断题

1. Variant 是一种特殊的数据类型,除了定长字符串数据及自定义类型外,可以保存任何类型的数据。Variant 还可以保存 Empty 和 Null 等特殊值。

2. 使用 Dim 语句定义了一个变量之后,还可以使用 ReDim 语句把此变量重新定义为其他的类型。

3. 使用 Static 语句定义的静态过程级变量,能在该过程的多次调用之间保持它的值,并且其他的过程也可以使用这个变量的值。

4. 在定义符号常量的语句中可以先不赋值,在以后赋值;但是,一旦被赋值便不能再赋新值。

5. 定义符号常量时给常量赋值可以使用表达式,但不能包含变量和函数调用。

6. 因为 Single 类型的变量可表示的范围大于 Long 类型的变量,所以 Single 类型占用内存空间大于 Long 类型。

7. 日期时间型变量既可以只保存日期值,也可以只保存时间值,但不能同时保存日期和时间值。

8. 在同一个过程中不能定义同名的变量;在过程中不能定义与同一模块的模块级变量同名的静态过程级变量。

9. 给长度为 4 的定长字符串变量赋一个长度为 8 的字符串会产生“溢出”错误。

10. 一个变量在刚被定义尚未被赋值之前没有值。

11. 一个应用程序的不同模块可以定义同名的全局变量,但是在一个模块中存取另一个模块中的全局变量时,应在变量前加模块名来限定。使用本模块中定义的全局变量一般不用加模块名。

12. 如果 A 和 B 都是整型变量,A 的值为 1,B 的值为 256,则变量 A 所占用的内存空间比变量 B 小。

13. 因为程序级和模块级范围不同,所以可以在同一个窗体模块中定义同名的程序级变量和模块级变量。

四、改错题

下面是窗体 Form1 的 Click 事件过程,要实现从第二次单击开始起,每次单击窗体时,窗体均向右移动 100 个单位。其中有几处错误,请改正。

```
1   Private Sub Form1_Click()
2      Dim intLeft As Integer
3      intLeft = intLeft + 100
4      form1.Left = intLeft
5   End Sub
```

请问,为什么第一次单击窗体时,窗体一般不向右移动,而是跳到屏幕左边附近?如果

希望无论窗体的初始位置在什么地方，每次单击（包括第一次）窗体都向右移100缇，程序应如何编写？

五、找出合法的直接常量

－0.0，.0，0.，2^3，1.2 * 10^3，log3，π，α，e，35.7°，""""，""""，""abc""，3＋5，12.3e，5e＋0，±&h007，3/2/99，&H123A

六、找出合法的变量名

3M，x_2，π，[，e，PI，OK，DIM，dim，＋a，we$，_name，a＋b