项目 5 制作 36 选 7 摇奖机

项目目标

本项目是制作一个 36 选 7 的摇奖机，包括摇出 6 个正选号码和 1 个特选号码，要具有没有重复号码；可以多次摇奖等功能。利用控件数组来设计摇奖机界面，产生随机数，通过循环控制语句和分支控制语句产生满足要求的摇奖号码，并把产生的摇奖号码显示在界面上。通过本项目，应掌握循环结构、随机数以及控件数组的使用方法和技巧。

项目主要知识点

1. Label 控件
   - 属性
   - 事件
   - 方法
2. 控件数组画法和使用
   - 复制－粘贴
   - 统一命名
3. 顺序程序结构
4. 随机函数的使用——Randomize
5. 产生随机数的方法
6. 框架控件的使用
7. 输出框
8. 循环结构初步知识
9. 循环结构初步
10. 程序调试，断点和监视的用法

项目实施步骤

1. 阅读相关基础知识
2. 界面设计——窗体设计
3. 程序设计
4. 程序调试，功能完善

5.1　相关基础知识

5.1.1　随机函数

随机数的产生在 VB 中用随机函数来实现。

1. Randomize 语句

初始化随机数生成器。

语法：

```
Randomize [number]
```

可选项 number 参数是 Variant 或任何有效的数值表达式。

说明：Randomize 用 number 将 Rnd 函数的随机数生成器初始化，该随机数生成器给 number 一个新的种子值。如果省略 number，则用系统计时器返回的值作为新的种子值。如果没有使用 Randomize，则（无参数的）Rnd 函数使用第一次调用 Rnd 函数的种子值。

注意：若想得到重复的随机数序列，在使用具有数值参数的 Randomize 之前直接调用具有负参数值的 Rnd。使用具有同样 number 值的 Randomize 是不会得到重复的随机数序列的。

2. Rnd 函数

返回一个包含随机数值的 Single。

语法：

```
Rnd[(number)]
```

可选项 number 参数是 Single 或任何有效的数值表达式。

返回值如表 5-1 所示。

表 5-1　参数与返回值关系

number 值	Rnd 生成
小于 0	每次都使用 number 作为随机数种子得到相同的结果
大于 0	序列中的下一个随机数
等于 0	最近生成的数
省略	序列中的下一个随机数

说明：Rnd 函数返回小于 1 但大于或等于 0 的值。

number 的值决定了 Rnd 生成随机数的方式。对最初给定的种子都会生成相同的数列，因为每一次调用 Rnd 函数都用数列中的前一个数作为下一个数的种子。在调用 Rnd 之前，先使用无参数的 Randomize 语句初始化随机数生成器，该生成器具有根据系统计时器得到的种子。

另：若要产生某个区间范围内的随机数，如[A,B]，则可利用下面的公式：

```
Int((B -A + 1) * Rnd +A)
```

这里，A 是随机数范围的上限；B 是随机数范围的下限。

5.1.2 VB 循环语句

与顺序结构、选择结构一样，循环结构是结构化程序中的 3 种基本程序结构之一。在程序中，凡是需要重复相同或相似的操作步骤，都可以用循环结构来实现。

循环结构由两部分组成：

（1）循环体，即要重复执行的语句序列。

（2）循环控制部分，即用于规定循环的重复条件或重复次数，同时确定循环范围的语句。

若想让计算机能够正常执行某循环，由循环控制部分所规定的循环次数必须是有限的，即循环体可以重复 0 次到若干次。

VB 支持的循环结构有：

- For…Next 循环。
- Do…Loop 循环。
- For Each…Next 循环。
- While …Wend 循环。

其中，For…Next 循环结构常用于设计已知循环次数的程序，而 Do…Loop 和 While…Wend 循环结构更适合于设计循环次数未知，而只知道循环结束条件的程序。

1. For …Next 循环结构

在已知循环要执行多少次时，最好使用 For…Next 循环。在 For…Next 循环中使用一个起计数器作用的循环变量，每重复一次循环之后，循环变量的值就会按一定的步长增加或者减少，直到超过某规定的终值时退出循环。

格式如下：

```
For <循环变量>=<初值> To <终值> [Step <步长>]
        <语句组 1>
[Exit For]
        <语句组 2>
Next  [<循环变量>]
```

说明：

- <循环变量>、<初值>、<终值>和<步长>都是数值型的，其中，<循环变量>、<初值>和<终值>是必需的。
- <步长>可正可负，也可以省略。若<步长>为正，则<初值>必须小于或等于<终值>，否则不能执行循环体内的语句；若<步长>为负，则<初值>必须大于或等于<终值>，否则不能执行循环体内的语句；若<步长>省略，则默认为 1。
- Exit For 语句可选，用于退出循环体，执行 Next 语句之后的语句。必要时，循环体中可以放置多条 Exit For 语句。该语句一般放在某条件结构中，用于表示当某种条件成立时，强行退出循环。
- Next 语句中的<循环变量>必须与 For 语句中的<循环变量>一致，也可以省略。

<步长>为正值时 For-Next 循环的逻辑流程如图 5-1 所示。

例 1：求 1+2+3+…+100 的值。

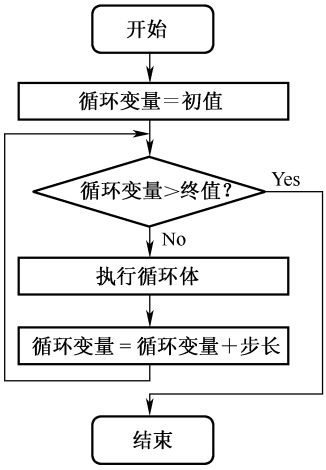

图 5-1　步长为正值时 For 循环执行逻辑流程

分析：在程序设计中，求取一批数据的“和”是一种典型的操作，通常称为“累加”。“累加”问题可以很方便地用循环来实现。设计时，一般引入一个存放“和”值的单元，如变量 Sum。首先设置该“和”值为 0，然后通过循环重复执行：和值=和值+累加项，即 Sum=Sum+i 来实现。

程序代码如下：

```
For i =1 to 100
  Sum=Sum+i
Next i
```

例 2：计算 100 以内的奇数的和。

程序代码如下：

```
For i =1 to 100 step 2
   Sum=Sum+i
Next i
```

步长为 2，则每次 i 值增加 2，即 i 值为 1，3，5，7，…，99。

2.　While 循环

格式如下：

```
While  <条件>
   <语句组>
Wend
```

功能：当条件为 True 时，反复执行循环，为 False 时退出循环。执行逻辑流程如图 5-2 所示。

例 3：求 1+2+3+…+100 的值。

程序代码如下：

```
Dim i As Integer, sum As Integer
   sum = 0
   i = 1
       Do While i <= 100
           sum = sum + i
           i = i + 1
```

```
    Loop
Print sum
```

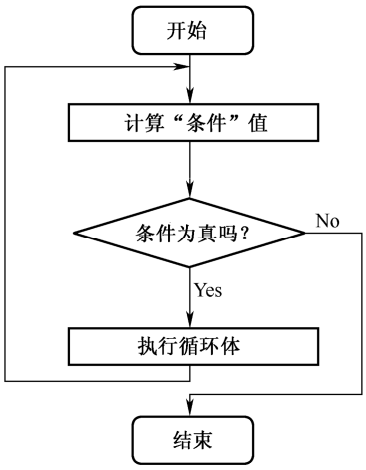

图 5-2　While…Wend 执行流程

3．Do…Loop 循环结构

Do 循环用于控制循环次数未知的循环结构，常见的形式如表 5-2 所示。

表 5-2　Do…Loop 循环结构常用形式

形式	形式 1	形式 2
格式	Do { While|Until }<条件> 　　语句块 　　[Exit Do 　　语句块] Loop	Do 　　语句块 　　[Exit Do 　　语句块] Loop { While|Until} <条件>
说明	先判断后执行，可能一次也不执行	先执行后判断，至少执行一次

说明：1）使用 While<条件>时，当指定的条件为 True 时，执行循环体中的语句块，而当条件为 False 时则退出循环，执行循环终止语句 Loop 之后的语句。

2）使用 Until<条件>时，当指定的条件为 False 时，执行循环体中的语句块，而当条件为 True 时则退出循环，执行循环终止语句 Loop 之后的语句。

例 4：求 1+2+3+…+100 的值。

用 Do...Loop 实现的程序核心代码为：

```
Dim i As Integer, sum As Integer
   sum = 0
   i = 1
Do
  sum = sum + i
   i = i + 1
Loop While i <= 100
```

用 Do Until…Loop 实现的核心代码如下：

```
Do Until i > 100          '条件不成立时执行循环体
    sum = sum + i
    i = i + 1
Loop
```

用 Do... Loop Until 实现的核心代码如下：

```
Do
sum = sum + i
i = i + 1
Loop Until i > 100        '条件不成立时执行循环体
```

4. 嵌套循环

通常，把循环体内不再包含其他循环的循环结构叫做单层循环。在处理某些问题时，常常要在循环体内再进行循环操作，而在内嵌的循环中还可以再包含循环，这种情况叫多重循环，又称为循环的嵌套。

VB 对循环的嵌套层数没有限制，当循环的层数太多时，程序的可读性会下降。习惯上，为了使循环结构更具可读性，总是用缩排的方式书写循环体部分。多层循环的执行过程是：外层循环每执行一次，内层循环就要从头开始执行一轮。

5. 双重 For 循环

语法格式：

```
For 循环变量 1=初值 1 To 终值 1 [Step 步长 1]
    For 循环变量 2=初值 2 To 终值 2 [Step 步长 2]
          循环体
    Next [循环变量 2]
Next [循环变量 1]
```

例 5：单击 Command1 按钮后输出以下乘法表，如图 5-3 所示。

1×1=1	2×1=2	3×1=3	4×1=4	5×1=5	6×1=6	7×1=7	8×1=8	9×1=9
1×2=2	2×2=4	3×2=6	4×2=8	5×2=10	6×2=12	7×2=14	8×2=16	9×2=18
1×3=3	2×3=6	3×3=9	4×3=12	5×3=15	6×3=18	7×3=21	8×3=24	9×3=27
1×4=4	2×4=8	3×4=12	4×4=16	5×4=20	6×4=24	7×4=28	8×4=32	9×4=36
1×5=5	2×5=10	3×5=15	4×5=20	5×5=25	6×5=30	7×5=35	8×5=40	9×5=45
1×6=6	2×6=12	3×6=18	4×6=24	5×6=30	6×6=36	7×6=42	8×6=48	9×6=54
1×7=7	2×7=14	3×7=21	4×7=28	5×7=35	6×7=42	7×7=49	8×7=56	9×7=63
1×8=8	2×8=16	3×8=24	4×8=32	5×8=40	6×8=48	7×8=56	8×8=64	9×8=72
1×9=9	2×9=18	3×9=27	4×9=36	5×9=45	6×9=54	7×9=63	8×9=72	9×9=81

图 5-3　乘法表图形

单击 Command1 按钮的核心代码如下：

```
Private Sub Command1_Click()
    Dim i As Integer, j As Integer
    For i = 1 To 9
        Print
        For j = 1 To9
            Print Tab(10 * j); j & "*" & i & "="; i * j;
        Next j
    Next i
End Sub
```

在以上的双重循环中，外层循环变量 i 取 1 时，内层循环就要执行 9 次（j 依次取 1、2、3、…、

9），接着，外层循环变量 i=2，内层循环同样要重新执行 9 次（j 再依次取 1、2、3、…、9）……，所以循环共执行 81 次。

例 6：实现如图 5-4 所示的九九乘法表。

```
1×1=1
1×2=2   2×2=4
1×3=3   2×3=6   3×3=9
1×4=4   2×4=8   3×4=12  4×4=16
1×5=5   2×5=10  3×5=15  4×5=20  5×5=25
1×6=6   2×6=12  3×6=18  4×6=24  5×6=30  6×6=36
1×7=7   2×7=14  3×7=21  4×7=28  5×7=35  6×7=42  7×7=49
1×8=8   2×8=16  3×8=24  4×8=32  5×8=40  6×8=48  7×8=56  8×8=64
1×9=9   2×9=18  3×9=27  4×9=36  5×9=45  6×9=54  7×9=63  8×9=72  9×9=81
```

图 5-4　九九乘法表图

单击 Command1 按钮的核心代码如下：

```
Private Sub Command1_Click()
    Dim i As Integer, j As Integer
    For i = 1 To 9
        Print
        For j = 1 To i
            Print Tab(10 * j); j & "*" & i & "="; i * j;
        Next j
    Next i
End Sub
```

说明：Tab[(n)]是列位置输出函数，可选的 n 参数是在显示或打印列表中的下一个表达式之前移动的列数。若省略此参数，则 Tab 将插入点移动到下一个打印区的起点。当 Print 方法与 Tab 函数一起使用时，打印的外观将会被分割为均匀、定宽的列。

在以上的双重循环中，外层循环变量 i=1 时，内层循环就要执行 1 次，接着，外层循环变量 i=2，内层循环同样要重新执行 2 次（j 再依次取 1、2）……，外层循环变量 i=9，内层循环同样要重新执行 9 次（j 依次取 1、2、3、…、9）。

同类循环可以嵌套，For…Next 循环和 Do…Loop 循环也可以互相嵌套。嵌套时，内层循环必须完全嵌套在外层循环之内。

5.1.3　数组

把一组具有相同属性、相同类型的数据放在一起，并用一个统一的名字作为标识，这就是数组。数组中的每一个数据称为一个数组元素，用数组名和该数据在数组中的序号来表示，序号称为下标。例如，一个班级有 30 名学生，可以用一个数组 score[29]来表示 30 个人的 VB 成绩：score（0）代表序号为 0 的同学的成绩，score（1）代表序号为 1 的同学的成绩，……，score（29）代表序号为 29 的同学的 VB 成绩。

在 VB 中如果没有特别说明，数组元素的下标是从 0 开始的，即第一个元素的下标是 0。

在 VB 中定义数组的一般格式为：

```
Dim 数组名（[下界 To] 上界）[As 数据类型]
```

例如：

```
Dim score(30) as Integer
```

定义了一个整数类型的一维数组 score，共有 31 个元素：score(0),score(1),…,score(29),score(30)。

例如：

```
Dim score(-1 to 28) as Integer
```

定义了类型为整型的一维数组 score，共有 30 个元素：score(-1),score(0),score(1)，…，score(28)。

1．数组的赋值

数组的赋值用两种方法：InputBox 函数输入和赋值语句赋值。

（1）数组元素一般通过 InputBox 函数输入，并用 For 循环语句反复输入多个数据。

```
Dim a(3) As Integer
For i = 0 To 3
a(i) = InputBox("Enter score:")
Print "a(" & i; ")=" & a(i)
Next i
```

通过这样的一段代码，可以实现将输入的 4 个数值连续赋给数组 a(0)、a(1)、a(2)、a(3)这 4 个元素，并显示在屏幕上。

（2）用赋值语句。可以用赋值语句为每个元素赋值。

```
a(0)=79:a(1)=80:a(2)=56:a(3)=90
```

2．数组的输出

输出数组元素可以使用标签控件、文本框控件等控件输出显示，或者使用 Print 方法直接打印在窗体上输出，如上代码所示。

5.1.4　程序的断点调试

当程序出现错误需要进一步排查时，通常用设置断点的方法来实现，可以获取一些变量的内容来检查是否符合要求，断点设置在排查区域内，便于发现问题。

“断点”是一个用于标识 Visual Basic 程序内代码行的标记，已标记断点的代码行以红色突出显示，且会在边界标识条中该代码行的相应位置处添加一个红点，如图 5-5 所示。

```
For i = 1 To 7
  Do
    x = Int(Rnd * 36) + 1    '产生1～36之间的随机数
    yes = False
    For j = 1 To i - 1
      If x = a(j) Then
         yes = True
         Exit For
      End If
    Next j
  Loop While yes = True
  Label1(i).Caption = x
  a(i) = x
  str = str & " " & x
Next i
MsgBox "本次中奖号码为：" & Chr(13) & Chr(13) & str
End Sub
```

图 5-5　断点

设置断点的方法如下：

- 在要添加断点的行旁边的边界标识条上单击。
- 光标停在要添加断点的行上，按下 F9 键。

断点使用方法：

在需要查看中间结果的代码前边界标识条上单击后，设置断点，单击图标运行程序，程序执行到断点时会从界面返回到代码窗口，黄色显示当前执行到的地方，按 F8 键逐语句执行，执行过的部分，光标放到附近能显示变量的当前值，如图 5-6 所示。

```
For i = 1 To 7
  Do
    x = Int(Rnd * 36) + 1    '产生1～36之间的随机数
  x = 27  = False
    For j = 1 To i - 1
      If x = a(j) Then
         yes = True
         Exit For
      End If
    Next j
  Loop While yes = True
  Label1(i).Caption = x
  a(i) = x
  str = str & " " & x
Next i
MsgBox "本次中奖号码为：" & Chr(13) & Chr(13) & str
End Sub
```

图 5-6　断点语句执行

为了进一步查看所关心的变量值及其变化过程，在程序执行时，通过添加监视的方法来实现对数值的观察。添加监视的方法为：

单击菜单“调试”→“添加监视”命令，即可出现如图 5-7 所示的对话框，在“表达式”文本框中填写需要关注的变量或者表达式。可以添加多个监视，本程序中添加两个监视，表达式分别为 a(i)和 str 变量，添加 str 监视的方法与添加 a(i)方法相同。

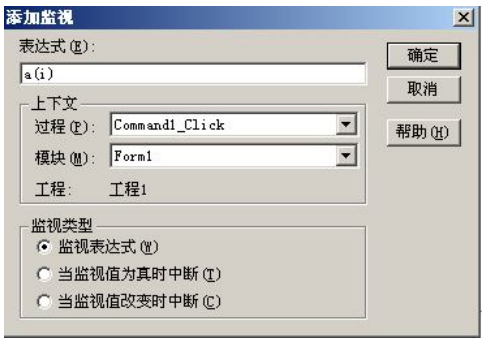

图 5-7　“添加监视”对话框

添加监视后，在程序的下方就会随着程序的执行显示所关注的表达式当前值，如图 5-8 所示。

图 5-8　监视表达式的当前值

当不需要断点时可清除断点，常用方法有两种：

（1）通过从“调试”菜单中选择“清除所有断点”命令或按下 Ctrl+Shift+F9 组合键，可以将

程序中设置的所有断点全部删除。

（2）退出 Visual Basic 环境时，会删除所有断点。

5.2　界面设计

（1）新建工程，并添加窗体 Form1，分别在窗体上添加以下控件：

1）两个框架 Frame1 和 Frame2。

2）在框架 1 中添加标签控件数组 Label1(0)～Label1(6)。

3）在框架 2 中添加 Label1(7)。

4）3 个命令按钮（Command1～Command3）。

（2）将下标为 0 的标签删掉

（3）将各控件按照图 5-9 所示进行排列。

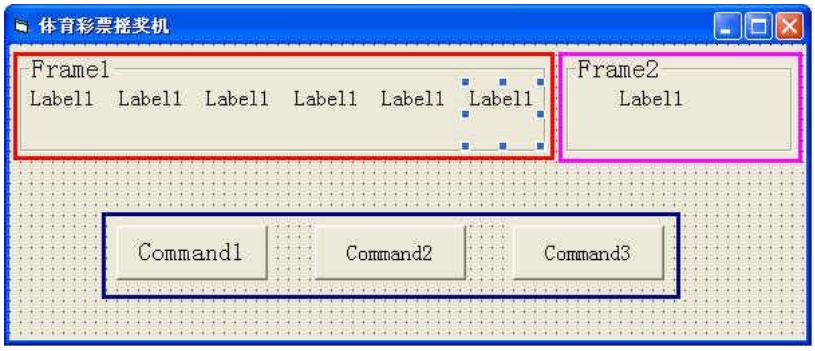

图 5-9　界面设计

（4）设置控件的属性如表 5-3 所示。

表 5-3　参数设置

控件名称	修改的属性
Form1	Caption：体育彩票摇奖机；MaxButton：False
Frame1	Caption：正选号码；ForeColor：红色
Frame2	Caption：特选号码；ForeColor：红色
Label1（1）～Label1（7）	Alignment：2-Center；ForeColor：红色
Command1	Caption：摇奖
Command2	Caption：清除
Command2	Caption：退出

参数设置完成后的界面如图 5-10 所示。

注：Alignment 是控件的对齐属性，通常有三个值；分别是：0-Left Justify（左对齐）、1-Right Justify（右对齐）、2-Center（居中对齐）。

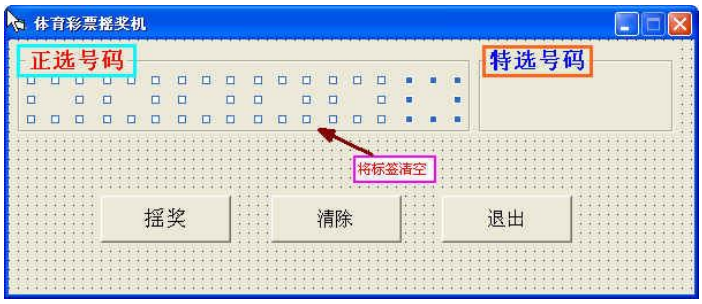

图 5-10　参数设置后界面

5.3　程序设计

编程思想：首先在“通用”部分定义全局数组，用来存放摇奖产生的数据

```
Dim a(1 To 7) As Integer
```

“摇奖”的过程就是产生 7 个随机数的过程，要求 7 个随机数不能重复。产生 1～36 之间的随机数的程序为：

```
Randomize
   x = Int(Rnd * 36) + 1
```

共两行代码。为了保证各次产生的随机数不重复，需要设定一个布尔变量来指示，布尔变量为 Yes，Yes 等于 False，说明随机数不与之前的随机数重复，可以使用。

在本程序中重复 7 次的工作是产生随机数，对于重复次数确定的操作，常用 For 循环来实现。For 循环的内部要实现 3 个内容的操作：

（1）将生成的符合条件的随机数赋值给标签，显示在界面上。

（2）保存生成的符合条件的随机数到数组中。

（3）后产生的随机数与先产生的随机数比较，用 for 循环来实现，循环的范围是从第一个到目前生成随机数的序号。例如，当前生成的随机数位置是 5，则循环将比较 a(1)、a(2)、a(3)、a(4)与新随机数是否相同。

（4）如果与先生成的随机数相等，则重新生成随机数，直到与前面的所有随机数都不同为止，随机数的产生用一个 While 循环来实现，循环执行的条件是有重复的数字。

分析得出双击 Command1 按钮“摇奖”的核心代码是：

```
Private Sub Command1_Click()    '“摇奖”按钮的 Click 事件
Dim yes As Boolean              '作为标志位，用于保证产生的随机数与前面的都不同
Dim str As String
Randomize
For i = 1 To 7
  Do
   x = Int(Rnd * 36) + 1        '产生一个 1～36 之间的随机数，存于 x 中
   yes = False
   For j = 1 To i - 1           '比较新随机数与前面产生的数字是否相同
     If x = a(j) Then
```

```
        yes = True                    '如果相同，跳出 for 循环
          Exit For
        End If
      Next j
    Loop While yes = True             '有重复的数字，则执行 while 循环再生成随机数
    Label1(i).Caption = x             '符合条件的数字通过 label 来显示
    a(i) = x
    '符合条件的数字保存在数组中，以备下一个数字生成时检查是否有重复
    str = str & " " & x               '字符串连接起来是为了一次在消息框中输出
  Next i
  MsgBox "本次中奖号码为：" & Chr(13) & Chr(13) & str
          'chr（13）回车换行
  End Sub
```

注：在测试、模拟和游戏程序中，经常要使用随机数。

编程效果如图 5-11 所示。

图 5-11　程序运行效果

1. “清除”命令按钮

双击“清除”按钮，Command2 的 Click 事件代码为：

```
Private Sub Command2_Click()      '“清除”命令按钮的事件
For i = 1 To 7
  Label1(i).Caption = ""          '清空每个标签
Next i
End Sub
```

2. “退出”命令按钮

双击“退出”按钮，Command3 的 Click 事件代码为：

```
Private Sub Command3_Click()
  End                             '退出当前窗体
End Sub
```

5.4　程序调试并完善功能

（1）36 选 7 的项目分别是 01 田径、02 游泳、03 跳水、04 水球、05 体操、06 举重、07 射击、……、33 蹼泳、34 围棋、35 象棋、36 桥牌，例如，“09062”期彩票的开奖结果是：“01 04 13 22 23 30 36”，对比开奖结果，需要对 1，2，3，…，9 这 9 个数字进行处理，让随机数“1”显示在界面上时为“01”，

随机数“2”显示在界面上时为“02”，对 10 以内的数字都进行处理。

（2）对摇奖界面进行美化，对图片和文字的格式进一步设置，让界面更美观。

项目总结

- 常用数据类型的使用，注意什么情况下使用何种数据类型。
- 控件数组的画法及使用方法。
- 通用过程的使用方法。
- 循环语句：For 循环和 While 循环的用法和适用场合。
- 随机数的产生方法：Randomize 语句和 Rnd 函数。
- MsgBox 函数和语句的简单使用。
- 程序断点调试方法。

操作练习

七星彩是指从 0000000～9999999 中选择任意 7 位自然数进行的投注，一组 7 位数的排列称为一注，如何设计七星彩体育彩票摇奖机？

复习思考

一、选择题

VB 支持的循环结构有（　　）。

A．For…Next　　B．Do…Loop

C．For Each…Next　　D．While …Wend

二、填空题

1．随机数的产生在 VB 中用_____来实现。

2．Rnd 函数返回_____的值。

3．若要产生某个区间范围内的随机数，如[A,B]，则可利用下面的公式：

_____。

4．循环结构由两部分组成：循环体和_____，_____用于规定循环的重复条件或重复次数，同时确定循环范围的语句。

5．数组中的每一个数据称为一个_____，用数组名和该数据在数组中的序号来标识，序号称为_____。