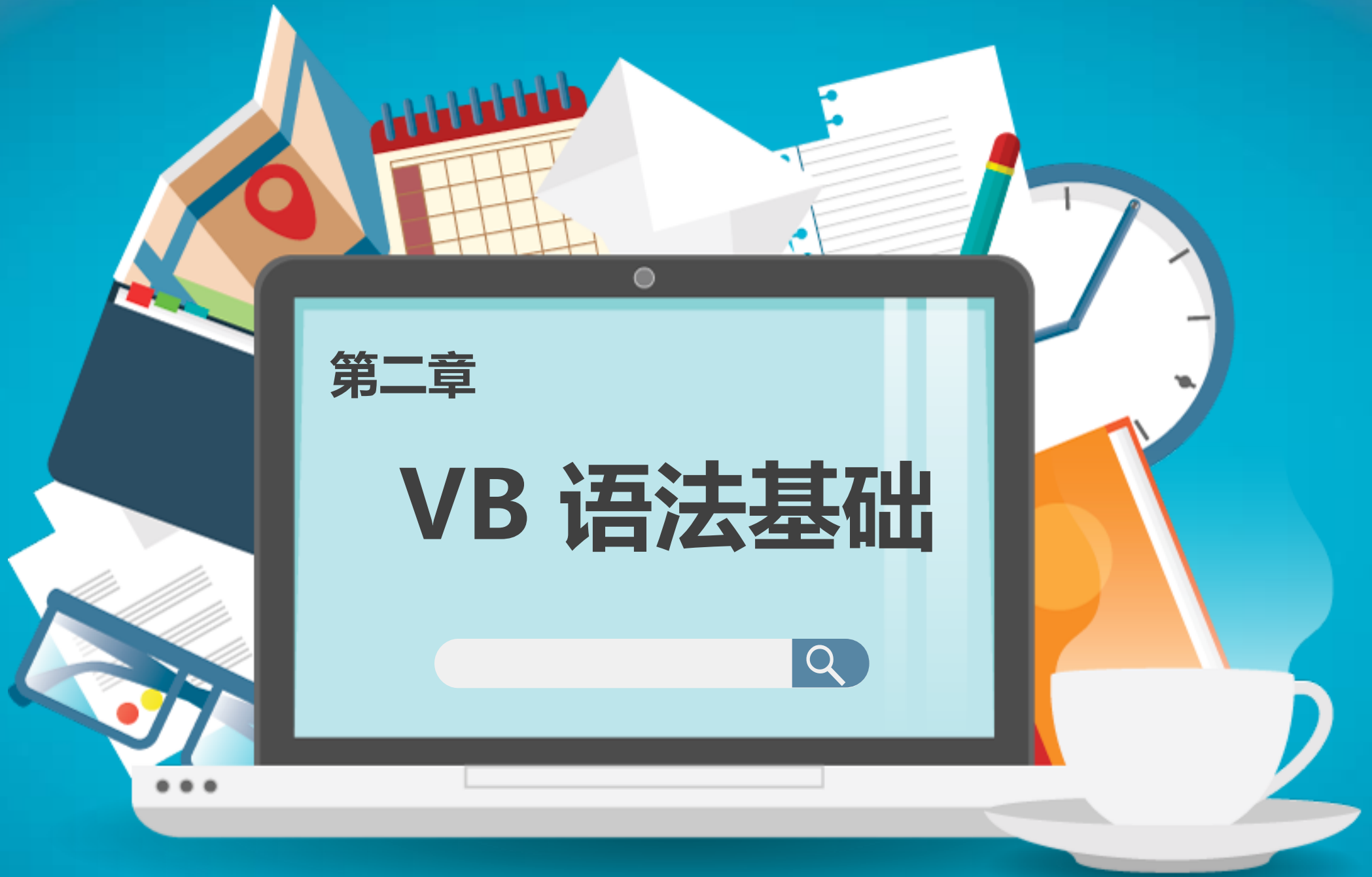
第二章
VB 语法基础

目录

目录

一、注释

变量注释，是代码中的一些“说明性文字”。注释本身不会参与程序的编译和运行，仅仅供程序员阅读

```
' 一 注释
' 打印并换行
Console.WriteLine("不积跬步，无以至千里")
```

二、变量

变量就是一种算法时可用来储值的容器

```
' 定义了一个变量n，变量的类型是Integer
Dim n As Integer
' 给n赋值为27
n = 27
' 改变变量n的值，将n+1
n = n + 1
' 打印n，{0}第0个参数为n
Console.WriteLine("n的值是{0}", n)
```

三、数据类型

定义一个变量时，需要指定存储在其中的数据类型。这就是数据类型，所有的程序语言都提供了大量不同的数据类型。

```
' 定义了一个变量n，变量的类型是Integer
 Dim n As Integer
```

3.1整型

VB处理数字时，要用到两种类型的数字： 整形和浮点型。

```
' 整形运算
Dim n2 As Integer
n2 = 16
n2 = n2 + 8
Console.WriteLine("add test n2=" & n2)
n2 = 24
n2 = n2 - 2
Console.WriteLine("sub test n2=" & n2)
n2 = 6
n2 = n2 * 6
Console.WriteLine("mul test n2=" & n2)
n2 = 12
n2 = n2 / 6
Console.WriteLine("div test n2={0}", n2)
```

3.1整型

VB处理数字时，要用到两种类型的数字： 整型和浮点型。

```
' 简写运算符
Dim n3 As Integer
n3 = 16
n3 += 8
Console.WriteLine("add test n3=" & n3)
n3 = 24
n3 -= 2
Console.WriteLine("sub test n3=" & n3)
n3 = 6
n3 *= 6
Console.WriteLine("mul test n3=" & n3)
n3 = 12
n3 /= 6
Console.WriteLine("div test n3={0}", n3)
```

3.1整型

整型运算的问题。

```
' 整型运算的问题
Dim n4 As Integer
n4 = 6
n4 = n4 * 10.23
Console.WriteLine("n4=" & n4)
n4 = 12
n4 /= 7
Console.WriteLine("n4=" & n4)
```

3.2浮点型

整形运算的问题。

```
' 浮点型运算
Dim n5 As Double
n5 = 45.34
n5 *= 4.333
Console.WriteLine("n5=" & n5)
n5 = 12
n5 /= 7
Console.WriteLine("n5=" & n5)
```

3.2浮点型

单精度浮点数

双精度浮点数能标识$-1.7X10^{308}$~$+1.7X10^{308}$

单精度只能表示$-3.4X10^{38}$~$+3.4X10^{38}$

用单精度浮点型来表示，精确度不像双精度浮点数那么高，但所需内存较少

```
' 单精度浮点型
Dim n5 As Single
```

3.3字符串

字符串就是字符的集合，使用双引号在首尾做标记

```
' 字符串
Dim n6 As String
n6 = "我是一个吊炸天的字符串！！"
Console.WriteLine("n6=" & n6)
' 字符串可以使用&符号进行连接
Console.WriteLine("n6的长度=" & n6.Length)
' 截取字符串
n6 = "当红" & n6.Substring(5, 1) & "子鸡"
Console.WriteLine(n6)
```

3.3字符串

```
' 格式化字符串
Dim n7 As Double
n7 = 12
n7 /= 7
Console.WriteLine("格式化之前的n7=" & n7)
Dim s As String
s = String.Format("{0:n3}", n7)
Console.WriteLine("格式化之后的n7=" & s)

' 替换字符串
Console.WriteLine(n6.Replace("子", "鸡"))
```

3.4日期

```
Dim d As Date
d = Date.Now()
Console.WriteLine("当前时间是=" & d)
Console.WriteLine("当前时间是=" & d.ToLongDateString)
Console.WriteLine("当前时间是=" & d.ToShortDateString)
Console.WriteLine("当前时间是=" & d.ToLongTimeString)
Console.WriteLine("当前时间是=" & d.ToShortTimeString)
Console.WriteLine("Year=" & d.Year)
Console.WriteLine("Month=" & d.Month)
Console.WriteLine("Day=" & d.Day)
Console.WriteLine("Hour=" & d.Hour)
Console.WriteLine("Minute=" & d.Minute)
Console.WriteLine("Second=" & d.Second)
Console.WriteLine("Day of week=" & d.DayOfWeek)
Console.WriteLine("Day of Year=" & d.DayOfYear)
Console.WriteLine(d.ToString("dddd"))
Console.WriteLine(d.ToString("MMMM"))
```

3.4日期处理

```
        Dim d2 As Date
        d2 = #5/5/1997 6:41:23 AM#
        Console.WriteLine(d2.ToLongDateString & " " &
d2.ToLongTimeString)
        ' 日期处理
        Dim d3 As Date
        d3 = #2/28/2040#
        Console.WriteLine(d3.AddDays(1))
        Console.WriteLine(d3.AddMonths(6))
        Console.WriteLine(d3.AddYears(-5))
```

3.5布尔型

布尔型（Boolean）变量的取值只能是True或False，这是计算机二进制本质的一种表示，因为计算机只能处理0和1，当程序做出决策时，布尔值非常重要。

四、数据类型转换

上一节我们学习了4种数据类型，也提到每种类型的变量只能存这种类型的数据。可是，有时候真的需要把不同类型的值放在一起运算，比如这种：3.5+8 这时候怎么办呢？有下面2种情况：

自动类型转换：2种不同类型的数据运算，低精度类型会自动转换为较高精度的类型。

以3.5+8为例，显然数字8的精度较低（Integer），而3.5的精度较高（Double），所以，8会自动转换为Double型，即转换为3.5+8.0进行运算，结果为11.5。

请看这个例子：
```
Dim d As Double
d=2
```
2的精度显然低于变量d的精度，所以2会自动转换为2.0然后赋值给d。

再看这个例子：
```
Dim i As Integer
i =3.0
```
变量i的精度低于3.0，但是由于i已经声明为int型的变量，变量的值可以变，但变量的**类型**可不能变来变去的，所以这条命令会出错的。

四、数据类型转换

强制类型转换：无法自动转换为我们需要的类型，可以用强制类型转换，比如上例可以这样完成：

Dim i As Integer = CInt(3.0) Cint表示强制转换为Integer，转换的目标类型为int，3.0会被强制转换为3。

需要注意， double **型强制转换为Integer型将失去小数部分**，比如Cint(2.8)，我们得到的将是2。

五、方法

方法是有关“做什么”的自包含代码块，方法是最基本的，其原因有二。第一它分解了程序，使之更容易理解。第二，它促进了代码的复用。

```
ComputeSum(1, 4)
Console.WriteLine(ComputeSub(3, 1))
 Private Sub ComputeSum(a As Integer, b As Integer)
   Console.WriteLine("a和b的和是: " & a + b)
 End Sub
 Private Function ComputeSub(a As Integer, b As Integer)
   Return a - b
 End Function
```

Private 表示这是一个私有函数
Sub表示这个方法是没有返回值的，Function表示这个方法有返回值，必须用Return返回
ComputeSum是这个方法的方法名（驼峰法命名，见名知意，Pascal casing首字母大写）
a和b是这个方法的参数，多个参数可用逗号隔开（camel casing首字母小写）
Integer是参数的类型
End Sub/End Function表示这个方法的结束位置

六、作用域

当方法开始时，在方法中定义的变量（换句话说，是Sub和Sub End或是Function和Function End之间的部分）是放在局部作用域内。作用域定义了程序的哪一部分可以查看变量，而局部是指“在方法内部“。

只有运行方法时，作用域中的变量才存在，方法结束时，变量就被删除了。

```
ComputeSum(1, 4)
Console.WriteLine(ComputeSub(3, 1))
Private Sub ComputeSum(a As Integer, b As Integer)
    Dim temp As String
    temp = "a和b的和是: "
    Console.WriteLine(temp & a + b)
End Sub
Private Function ComputeSub(a As Integer, b As Integer)
    Dim temp As String
    temp = "a和b的差是: "
    Return temp & a - b
End Function
```

七、练习

1.我上个月的收入是999.99元，怎样声明这个变量

A Dim salary As Integer
salary = 999.99

B Dim salary As String
salary = 999.99

C salary = 999.99

D Dim salary As Double
salary = 999.99

2.下面程序打印的是：

```
Sub Main()
Dim age As Integer
age = 18
age = 20
age = 20+1
Console.WriteLine(“我今年{0}岁”, age)
age = age - 2
End Sub
```

七、练习

3.算法交换

```
Dim a As String
a = “张飞” //第一个变量
Dim b As String
a = “关羽” //第二个变量
Dim temp As String//中间变量
 //第一步：将变量a赋值给中间变量
temp=a;//如同牛奶倒入空杯
//第二步：将变量b赋值给变量a
a=b;//如同咖啡倒入牛奶杯
//第三步：将中间变量赋值给变量b
b=temp;//如同空杯中的牛奶倒入咖啡杯
//此时交换完成，变量a存储了“文峰”，b存储了“振刚”
```

七、练习

3.算法交换练习
已知变量boy叫秀丽，变量girl叫伟强，请用你学会的交换算法交换他们的名字

七、练习

4.这些输出语句中（ ）会出错

A Dim d As Double = 2.3 B Dim d As Double = 3 C Dim i As Integer = 2.3 D Dim i As Integer = 3

5.()不会打印3

A Console.Write(Cint(3.6)) B Console.Write(CDbl(3.6)) C Console.Write(CInt(3)) D Console.Write(CDbl(3))