

21世纪高等学校计算机规划教材

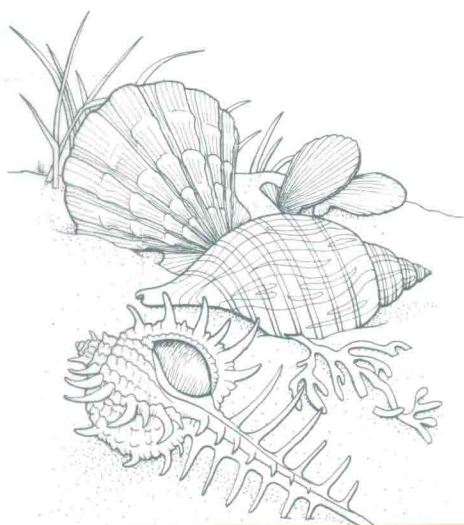
21st Century University Planned Textbooks of Computer Science

Visual Basic 程序设计实践教程

Experiment Instruction for Visual Basic
Programming

韦昌法 易钢 主编

刘青萍 李曼 周燃犀 王茜 副主编



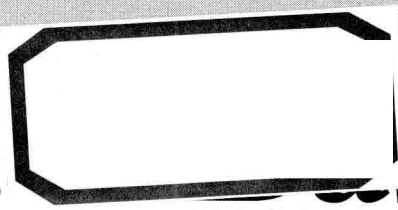
高校系列



人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

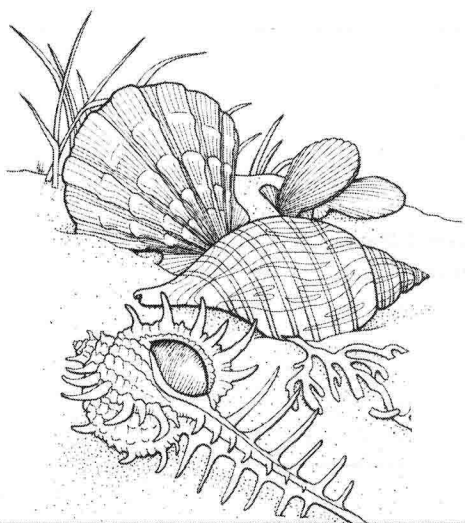
Visu  sic

程序设计实践教程

Experiment Instruction for Visual Basic
Programming

韦昌法 易钢 主编

刘青萍 李曼 周燃犀 王茜 副主编



高校系列

人民邮电出版社

北京

图书在版编目 (CIP) 数据

Visual Basic程序设计实践教程 / 韦昌法, 易钢主
编. — 北京: 人民邮电出版社, 2015.2
21世纪高等学校计算机规划教材
ISBN 978-7-115-37788-3

I. ①V… II. ①韦… ②易… III. ①BASIC语言—程
序设计—高等学校—教材 IV. ①TP312

中国版本图书馆CIP数据核字(2015)第001634号

内 容 提 要

本书是《Visual Basic 程序设计教程》一书的配套实验教材。全书分2章,第1章是 Visual Basic 程序编写、调试与发布,主要介绍了 Visual Basic 程序编写、程序调试与错误处理、发布应用程序以及 Visual Basic 帮助系统的使用;第2章是实验,安排了12组实验,包括 VB 开发环境及简单程序设计、VB 语言基础、VB 常用控件、顺序和选择结构、循环结构、数组程序设计、过程程序设计、应用程序界面设计、数据库程序设计、文件操作、图形操作以及综合实验。附录部分包括《Visual Basic 程序设计教程》一书的习题答案及全国计算机等级考试二级 Visual Basic 考试大纲。

本书可以作为非计算机专业本科或专科 Visual Basic 程序设计课程的实验教材,也可以作为计算机专业相关课程的辅助教材。

-
- ◆ 主 编 韦昌法 易 钢
副 主 编 刘青萍 李 曼 周燃犀 王 茜
责任编辑 邹文波
执行编辑 吴 婷
责任印制 沈 蓉 彭志环
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
北京艺辉印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
印张: 6
字数: 155千字
- 2015年2月第1版
2015年2月北京第1次印刷
-

定价: 20.00 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316
反盗版热线: (010)81055315

本书编委会

主 编 韦昌法 易 钢

副主编 刘青萍 李 曼 周燃犀 王 茜

编 委 丁长松 李 曼 李小智 刘青萍 涂 珊

王 茜 韦昌法 易 钢 周燃犀 周 知

Visual Basic 程序设计是大学的基础课程之一，也是全国计算机等级考试科目之一。为了适应现代信息社会发展的需要，进一步明确学习程序设计的意义、培养学生的计算思维能力、提高教学质量，编者根据多年的教学、教改经验和教材编写经验，编写了本套教材，即《Visual Basic 程序设计教程》（主教材）和《Visual Basic 程序设计实践教程》（实验教材）。

本套教材内容比较基础，但是覆盖面比较广，力求使初学者对 Visual Basic 有一个比较全面的了解。在介绍具体内容时深入浅出，理论联系实际。教材以医院的住院管理系统为主线，以丰富的程序设计实例为辅助，在阐述 Visual Basic 程序设计语言的同时，向学生演示了应用计算机解决实际问题的方法，加深学生对信息化工作的理解。

本套教材的编者都是中医药院校的一线教师，具有丰富的教学经验。编者们努力将知识点全面、具有医学特色的 Visual Basic 程序设计课程体系呈现出来。教材中精选了很多具有综合性和拓展性的典型教学案例以及实用性很强的住院管理系统，使学生在学习和实践过程中深入了解医学信息系统的重要性和实用性，培养学生熟练使用计算机程序处理医学信息的能力。本套教材的主要特点如下。

1. 以“住院管理系统”为主线，逐渐深入地阐述 Visual Basic 程序设计相关知识，培养学生编程解决实际问题的能力。在主教材《Visual Basic 程序设计教程》中，大部分章的最后一节根据该章知识点，围绕住院管理系统的某些功能模块进行实例讲解。实例讲解部分独立成节，有利于教师根据课时安排和授课对象水平进行选讲，也有利于学生循序渐进地进行自学。学生学完所有知识点之后可以自主地开发一些小系统，增强成就感。

2. 教材中配有丰富的教学例题。例题是帮助学生理解、掌握教学内容而设计的程序范例，学生通过阅读这些例题，可以做到举一反三，加深对所学内容的理解和掌握，逐步培养自己的程序阅读能力和编程能力。

3. 重视“面向对象”的可视化编程。主教材《Visual Basic 程序设计教程》除了在第 1 章和第 2 章介绍“面向对象”的可视化程序设计概念和方法之外，还根据读者对 Windows 软件图形界面相对比较熟悉的情况，在第 9 章介绍应用程序界面设计时，使用了记事本等 Windows 典型程序进行实例讲解。

4. 主教材《Visual Basic 程序设计教程》在每章后面都附有多种类型的习题和思考题，帮助学生复习、巩固所学知识。实验教材《Visual Basic 程序设计实践教程》安排了大量的由浅入深的实验，逐步培养学生的动手编程能力。

本套教材可以作为高等院校各学科非计算机专业或计算机应用专业的教材，也可以作为高职院校、专科学校的教材以及全国计算机等级考试（NCRE）或其他培训机构的培训教材。

主教材《Visual Basic 程序设计教程》共有 12 章。第 1、2 章由涂珊编写，第 3 章由王茜编写，第 4 章由周燃犀编写，第 5、8 章由韦昌法编写，第 6 章由周知编写，第 7 章由刘青萍编写，第 9 章由易钢编写，第 10 章由李小智编写，第 11 章由丁长松编写，第 12 章由李曼编写。

实验教材《Visual Basic 程序设计实践教程》共有 2 章。第 1 章由易钢、韦昌法编写；第 2 章

安排了 12 组实验，实验 1 由涂珊编写，实验 2 由王茜编写，实验 3 由周燃犀编写，实验 4、5、12 由韦昌法编写，实验 6 由周知编写，实验 7 由刘青萍编写，实验 8 由易钢编写，实验 9 由李小明编写，实验 10 由丁长松编写，实验 11 由李曼编写。

由于编者的水平和经验有限，教材中难免有疏漏和不妥之处，真诚地期望使用本套教材的师生们给予批评、指正。本套教材中所有的程序文件都可以在 Microsoft Visual Basic 6.0 环境下编译运行，使用本套教材的老师如果需要这些程序文件和相关的配套资料，可以直接与编者联系，邮箱地址为 nancy_tu@163.com。

编 者

2014 年 10 月

第 1 章 Visual Basic 程序编写

调试与发布1

1.1 Visual Basic 程序编写1

1.2 程序调试与错误处理2

1.3 发布应用程序14

1.4 Visual Basic 帮助系统的使用17

第 2 章 实验20

实验 1 VB 开发环境及简单程序设计20

实验 2 VB 语言基础23

实验 3 VB 常用控件28

实验 4 顺序和选择结构35

实验 5 循环结构40

实验 6 数组程序设计42

实验 7 过程程序设计49

实验 8 应用程序界面设计53

实验 9 数据库程序设计58

实验 10 文件操作63

实验 11 图形操作68

实验 12 综合实验71

附录 A 《Visual Basic 程序设计教程》习题答案83

附录 B 全国计算机等级考试二级 Visual Basic 考试大纲86

第 1 章

Visual Basic 程序编写调试与发布

1.1 Visual Basic 程序编写

一、Visual Basic 的工作模式

从设计到执行，一个 Visual Basic 应用程序处于不同的模式之中。Visual Basic 有 3 种模式：设计模式、运行模式和中断模式。这些模式是编写、调试和运行应用程序时，应用程序所处的不同状态。

1. 设计模式

启动 Visual Basic 后，打开一个工程窗口，即进入设计模式，在主窗口标题栏的最后面显示有“[设计]”字样，如图 1-1-1 所示。

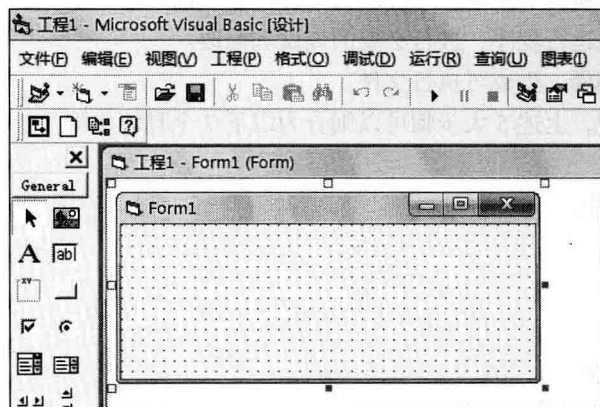


图 1-1-1 设计模式窗口

创建一个应用程序的所有工作都是在设计模式下完成。在设计时，可以设计窗体，绘制控件，编写程序代码，并使用“属性”窗口设置属性值或查看当前属性值等。当程序处于设计模式时，不能执行操作，也不能使用调试工具对之进行调试，但可设置断点和创建监视表达式。

2. 运行模式

选择“运行”菜单中的“启动”命令（或按【F5】键或单击工具栏上的“运行”按钮），应用程序即可进入运行模式。此时主窗口标题栏上原来显示的“[设计]”字样被“[运行]”字样取代。

在运行模式中, Visual Basic 将全部控制权交给应用程序, 用户可以对应用程序进行测试, 检验程序的运行结果, 并可查看程序代码, 但不能修改代码。

选择“运行”菜单中的“结束”命令(或单击工具栏上的“结束”按钮), 可返回设计模式。

选择“运行”菜单中的“中断”命令(或单击工具栏上的“中断”按钮或按【Ctrl+Break】组合键), 可进入中断模式。

3. 中断模式

当程序处于中断模式时, 暂停程序的执行。进入中断模式后, 主窗口标题栏上原来显示的“[设计]”或“[运行]”字样由“[break]”字样取代。在中断模式下, 因为变量和属性设置值被保留下来, 所以可以分析应用程序的当前状态并输入修改内容。这些修改将影响程序的运行。此时, 可以在应用程序中修改代码, 观察应用程序界面的情况, 确定哪个过程已被调用, 监视变量值、属性和语句, 并改变变量值和属性设置值, 查看或控制应用程序下一步运行的语句, 也可以立即运行 Visual Basic 语句, 手工控制应用程序的操作。

可以用以下 4 种方式进入中断模式。

- (1) 在运行模式下, 选择“运行”菜单中的“中断”命令。
- (2) 在程序中设置断点(Break Point), 程序执行到该断点时自动进入中断模式。
- (3) 执行程序时遇到了 Stop 语句。
- (4) 在程序执行过程中, 如果出现错误, 则将自动进入中断模式。

二、Visual Basic 程序编写步骤

用 Visual Basic 编写程序的过程可归纳为以下 5 大步骤。

- (1) 创建一个新的工程, 在工程中建立窗体, 在窗体上布置需要的各种控件对象。
- (2) 设置窗体和各个控件对象的属性(控件对象的属性也可以通过程序代码来设置)。
- (3) 根据各个控件所需要的响应事件编写事件过程代码。
- (4) 保存工程, 试运行程序, 进行必要的调试和修改。
- (5) 对工程进行编译, 生成可执行文件。

在实际编程过程中, 上述 5 大步骤可以细分为以下 7 个具体步骤。

- ① 新建工程。
- ② 向窗体添加控件。
- ③ 设置控件属性。
- ④ 编写代码。
- ⑤ 保存工程。
- ⑥ 运行、调试工程。
- ⑦ 编译工程、生成可执行文件。

1.2 程序调试与错误处理

一、错误类型

用户在编写程序时, 难免会出现一些语法和逻辑等错误, 这些错误称为 Bug。找出并纠正这些错误的过程称为 Debug(调试)。

Visual Basic 应用程序的错误一般有语法错误、编译错误、运行错误和逻辑错误 4 类。

1. 语法错误

语法错误是由于不正确地创建代码而导致的，即在语法不正确时出现这种错误。如错误地输入了关键字、丢失或写错了符号、遗漏了必需的语句成分、括号不匹配等。

Visual Basic 具有自动语法查错功能，在设计阶段输入程序代码时就能检查出语法错误。如以下代码。

```
Private Sub Command1_Click()  
    a=10;b=20  
    c=a+b  
    Print c  
End Sub
```

在输入代码时，若将第 3 行输为

```
c=a{b
```

按【Enter】键后就会显示出错提示框，并且刚输入的一行变为红色，出错的部分高亮显示，如图 1-2-1 所示。

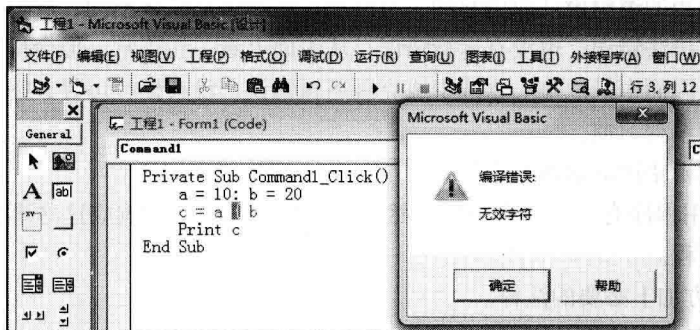


图 1-2-1 语法错误提示

只有改正该语法错误后，红色和高亮显示才会消失。

2. 编译错误

Visual Basic 在运行程序前先编译执行程序。如果用户未定义变量或遗漏关键字等，则 Visual Basic 将出现出错提示，并使有错误的程序行高亮显示，如图 1-2-2 所示，这种错误称为编译错误。出现这类错误后，Visual Basic 将停止编译，并返回有错误的程序代码窗口。

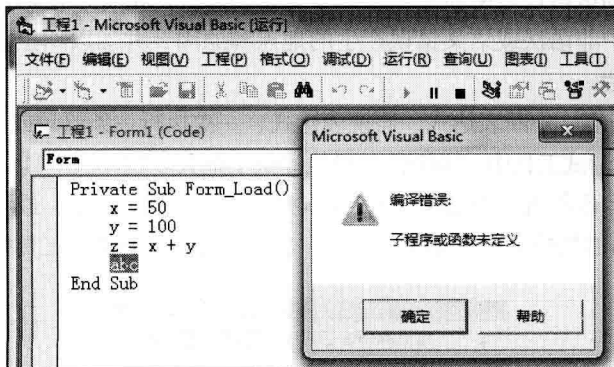


图 1-2-2 编译错误提示

3. 运行错误

若语法正确,但在程序运行期间一条语句试图执行一个不能执行的操作,就会产生运行错误。有运行错误的代码在一般情况下可以运行正常,但遇到非法数据或是系统条件禁止代码运行时就会产生错误。一个典型的运行错误是以 0 作为除数,运行时将打开提示框,如图 1-2-3 所示。

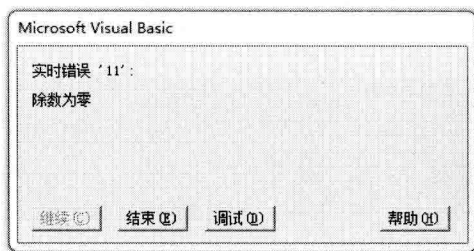


图 1-2-3 运行错误提示

4. 逻辑错误

程序逻辑错误是指应用程序未按设计进行运行,或生成无效甚至错误的结果。这种错误是由于程序代码中不恰当的逻辑设计而引起的,与语法错误和运行错误不同,逻辑错误一般不报告出错信息。它既没有语法错误,也没有运行错误,从表面上看一切正常,但会得到错误的结果。这类错误最难发现,因而也最危险。要减少或克服逻辑错误,没有捷径可寻,需要不断练习、积累经验并养成良好的编程习惯。

编程时减少逻辑错误的常用办法如下。

① 列出与应用程序有关的所有事件,然后考虑如何在程序中响应这些事件,认真定义每个事件过程及通用过程。

② 在关键地方加上必要的注释。

③ 注意变量名的一致性。

④ 设置断点,缩小错误的查找范围。

⑤ 中断程序的运行,检查变量的当前值。

二、程序调试

在程序设计过程中,错误是难免的,查找和修改错误的过程称为调试。Visual Basic 为程序调试提供了一组交互的、有效的调试工具。这些调试工具支持包括设置断点,逐语句、逐过程地控制程序的运行,利用调试窗口显示监视表达式、变量和属性的值等。使用这些调试工具,可以快捷、有效地检查程序中非语法或语义错误产生的地点和原因,并加以纠正。

1. “调试”工具栏

调试工具包括:断点、中断表达式、监视表达式、一次执行一条语句或一个过程、显示变量的值或属性值等。在各种调试工具中,“调试”工具栏提供了绝大多数调试工具。如果在屏幕中没有“调试”工具栏,则可选择菜单中的“视图”→“工具栏”→“调试”命令,或在 Visual Basic 6.0 工具栏中右键单击,在弹出的快捷菜单中选择“调试”命令,打开“调试”工具栏,如图 1-2-4 所示。



图 1-2-4 “调试”工具栏

工具栏中的按钮从左到右依次为“启动”、“中断”、“结束”、“切换断点”、“逐语句”、“逐过程”、“跳出”、“本地窗口”、“立即窗口”、“监视窗口”、“快速监视”和“调用堆栈”。所有调试工具的功能如表 1-2-1 所示。

表 1-2-1

调试工具的功能

工具	功能
启动	启动应用程序
中断	中断程序
结束	结束应用程序运行
切换断点	在光标所在行设置断点
逐语句	单步执行
逐过程	单步执行可执行语句，但不单步执行调用过程中的语句
跳出	执行该过程的剩余代码，在下一个过程第一行中断
本地窗口	显示本地变量的值
立即窗口	在程序中断方式下，可以执行代码或查询值
监视窗口	显示选中的表达式的值
快速监视	在程序中断方式下，列出表达式的当前值
调用堆栈	在中断方式下，显示所有被调用而未返回的过程



说明

应当注意，“逐过程”和“逐语句”调试工具有所不同，“逐语句”工具单步执行所有可执行代码，也包括在过程中调用的其他过程，而“逐过程”则不单步执行该过程中调用的其他过程。

2. 标识当前模式

在 Visual Basic 的标题栏中总是显示工程当前模式，可以很容易识别出当前所处理模式，如图 1-2-5～图 1-2-7 所示。

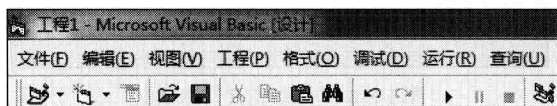


图 1-2-5 设计模式标题栏

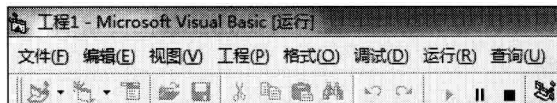


图 1-2-6 运行模式标题栏

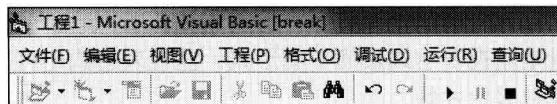


图 1-2-7 中断模式标题栏

可以通过选择“运行”菜单中的命令或单击工具栏中的命令按钮在 3 种模式间切换。例如，处于设计模式时，在“运行”菜单中选择“启动”命令，或单击“运行”按钮，可切换到运行模

式。选择“运行”菜单中的“结束”命令，或单击“结束”按钮，则返回“设计模式”。应用程序处于运行模式时，选择“运行”菜单中的“中断”命令，或单击“中断”按钮，或按【Ctrl + Break】组合键，可切换到中断模式。此外对于发生以下情况时，将自动切换到中断模式。

- ① 语句产生了非俘获的运行错误。
- ② 语句产生运行错误，并且“发生错误时中断”错误捕获选项被选中。
- ③ “添加监视”对话框中定义的中断表达式将发生变化或变成真，这取决于它的定义方式。
- ④ 执行到一个设有断点的行。
- ⑤ 执行到一个 Stop 语句。

3. 使用调试窗口

在中断模式下，用户可以通过“监视”窗口、“立即”窗口和“本地”窗口等调试窗口来观察相关变量的值。选择“视图”菜单，然后在下拉菜单中选择相应的命令可以打开上述调试窗口。

(1) “监视”窗口

“监视”窗口用于显示当前的监视表达式，用监视表达式可以查看或跟踪正在执行过程中的变量或表达式的值。

在设计阶段，选择菜单“调试”→“添加监视”命令或“调试”→“快捷监视”命令来添加监视表达式，设置监视类型，操作步骤如下。

- ① 选择菜单“调试”→“添加监视”命令，打开“添加监视”对话框，如图 1-2-8 所示。

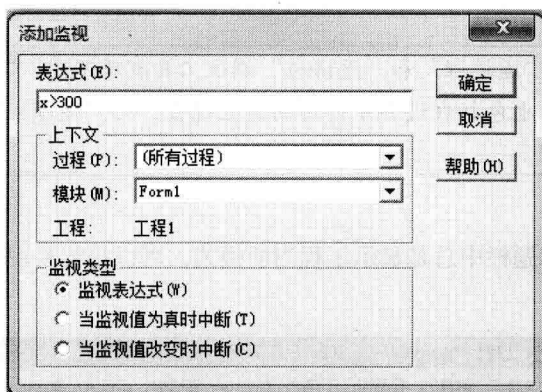


图 1-2-8 “添加监视”对话框

- ② 在“表达式”文本框中输入一个变量名或表达式。
- ③ 在“监视类型”选项区域中选中“监视表达式”单选按钮。
- ④ 单击“确定”按钮。

如果将某一表达式设置为监视表达式，则该表达式将出现在“监视”窗口中。按【F8】键运行该程序，单击窗体后，连续按【F8】键，此时是用“逐语句”方式执行程序，窗体上将显示程序的执行情况，同时在“监视”窗口中实时显示变量表达式的值。

值得提出的是，变量是监视表达式最简单的形式，监视表达式还可以是任何表达式，如算术表达式、关系表达式及逻辑表达式等。

另外，在“程序代码”窗口中也可以将变量或表达式选择为监视表达式。双击一个变量可将该变量选为监视表达式（变量名反相显示）；拖动并选取某个表达式，则该表达式也可被选为监视表达式（该表达式将反相显示）。

(2) “立即”窗口

程序进入中断模式后，将自动激活“立即”窗口，如图 1-2-9 所示。“立即”窗口用于显示过程代码中的调试语句生成的信息，或直接输入在窗口中的命令所生成的结果。在“立即”窗口中，可以输入并执行 Visual Basic 语句，每个语句一行，按【Enter】键执行，不影响窗口中的代码。“立即”窗口仅在程序处于中断模式时方可使用。

除了在中断模式下自动激活“立即”窗口外，用户还可以通过以下方法打开“立即”窗口。

- ① 选择菜单“视图”→“立即”命令。
- ② 单击“调试工具”工具栏中的“立即”按钮或按【Ctrl+G】组合键。
- ③ 还可以通过以下两种方式在“立即”窗口中输出信息。
- ④ 用 Debug.Print 语句。
- ⑤ 直接在“立即”窗口执行 Print 方法。

一般情况下，用 Print 方法可以把信息输出到窗体、打印机或控件中，其格式如下。

[对象.] Print [项目] [:]

其中，“对象”可以是打印机或控件，如省略“对象”则为当前窗体。在 Visual Basic 中 Debug 也是一个对象，如果用它作为 Print 方法的“对象”，即：

Debug.Print [项目] [:]

则可以把“项目”输出到“立即”窗口中。例如：

Debug.Print S

当程序执行到“Debug.Print...”语句时，Visual Basic 将自动激活“立即”窗口，并在该窗口中显示 S 变量的信息。

(3) “本地”窗口

在中断模式下，“本地”窗口显示了当前过程范围内的所有变量的值及其类型，而且当程序从一个过程切换到另一个过程时，“本地”窗口的内容仅仅反映当前过程中的变量。

如未显示“本地”窗口，则选择菜单“视图”→“本地窗口”命令或者单击“调试”工具栏中的“本地”按钮即可打开，如图 1-2-10 所示。

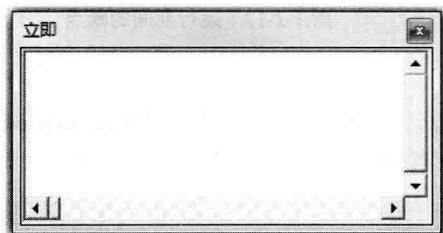


图 1-2-9 通过“立即”窗口监视

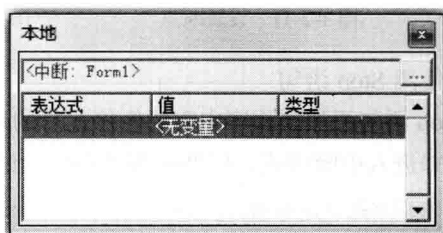


图 1-2-10 “本地”窗口

4. 使用中断

在前面已介绍了进入中断模式的方法，在中断模式下可以对程序进行修改，并可继续运行。在程序调试时，常用的中断方法有两种：设置断点和使用 Stop 语句。

(1) 设置断点

断点通常设置在程序中需要暂停执行的地方。利用断点可以分别测试程序段，或者通过断点使运行的程序在关键的地方停止，以方便测试一个变量的值，从而观察程序的实际情况。当在程序中遇到断点时，Visual Basic 就会进入中断模式。

断点在设计阶段或中断模式下设置，设置断点通常有以下 4 种方法。

① 在程序“代码”窗口中，把光标移到希望中断的语句上，然后选择“调试”→“切换断点”命令，或直接按【F9】键，即可把光标所在行设置为断点。同时，被设置为断点的语句行中的字符变为粗体并反相显示。

② 代码窗口的左边灰色区域称为边界标识条。当光标位于该区域时会变为右箭头，此时单击某一程序行，该行即被设置为断点，如图 1-2-11 所示。

③ 将鼠标在要设置为断点的程序行上右键单击，在弹出的快捷菜单中选择“切换”→“断点”命令，即把该行设置为断点。运行期间的断点如图 1-2-12 所示。

④ 在程序“代码”窗口中，把光标移到希望中断的语句上，再单击调试工具栏中的“切换断点”按钮，即把该行设置为断点。

以上 4 种方法用来切换断点，即无断点时设置为断点，有断点时则清除断点。如要清除全部断点，则可选择菜单“调试”→“清除所有断点”命令（或按【Ctrl+Shift+F9】组合键）。

设置好断点后运行程序，当程序执行到断点处，便暂停执行并进入中断模式，断点语句以黄色背景显示，并在边界标识条上显示一个箭头。此时就可以对断点之前的语句进行检查，把光标移到某个变量上，即可显示出该变量的当前值，如图 1-2-12 所示。

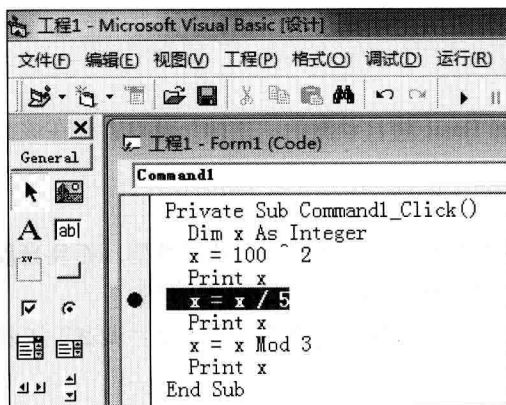


图 1-2-11 设置断点

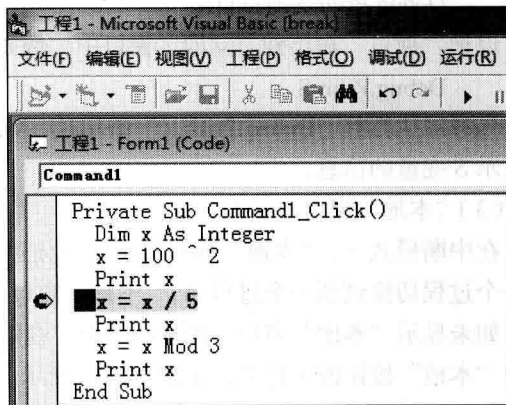


图 1-2-12 运行期间的断点

(2) 使用 Stop 语句

把 Stop 语句添加到程序中需要暂停运行的地方，当遇到 Stop 语句时，Visual Basic 将暂停程序执行，并进入中断模式，以便对程序进行调试。举例如下。

```
Private Sub Form_Click()
    x = 100
    y = 200
    z = 300
    a = x + y + z
    Stop
    Print a
    b = z - x + y
    Print b
End Sub
```

当执行到以上程序中的 Stop 语句时，程序暂停执行，进入中断模式。

以上介绍了使用中断的两种常用方法，在使用中断时，要注意以下几点。

① 从暂停运行进入中断模式，断点和 Stop 语句作用相同。使用断点无须修改代码，断点比 Stop 语句方便。但断点不如 Stop 语句灵活。在程序执行过程中每遇到一个断点，程序都无条件地中断执行，而 Stop 语句可使用程序在一定条件下暂停运行。举例如下。

```
Static BrkCount As integer
...
BrkCount = BrkCount + 1
If BrkCount 10 And A < 0 Then
    Stop
End If
...
```

在程序执行到 Stop 语句时不会立即暂停执行，必须在执行 10 次后当 A 的值小于 0 时才中断运行。这就是 Stop 语句的灵活之处。

② 要特别注意的是断点只在当前程序中存在，如果储存后再重新装入，断点将全部消失。而 Stop 语句则永远保留在程序中，除非将其删除。因此，在程序调试结束后，应删除程序中不需要的 Stop 语句。

③ 在断点或 Stop 语句处程序执行中断后，按【F5】键或选择菜单“运行”→“继续”命令可继续执行程序。

④ End 语句与 Stop 语句都可使程序停止执行。但 End 语句用来结束程序的执行而返回设计模式，而 Stop 语句是暂停程序执行，进入中断模式，可通过按【F5】键或选择菜单“运行”→“继续”命令继续执行程序。

⑤ 在“运行”菜单中有以下几个命令，需注意它们的用法。

- a. 启动（【F5】键）。
- b. 全编译执行（【Ctrl + F5】组合键）。
- c. 中断（【Ctrl + Break】组合键）。
- d. 结束。
- e. 重新启动（【Shift + F5】组合键）。

在不同的模式下，这几个命令是变化的。在设计阶段，只有前两个命令有效；在执行阶段，前两个命令无效，后 3 个命令有效；在中断模式下，第 2 个、第 3 个命令无效，第 1 个命令变为：继续（【F5】键）。

5. 运行程序指定部分

利用断点只能粗略地查出错误发生在程序的哪一部分，而程序出错的具体位置并不容易确定，这时需要在此范围内运行程序指定部分，一条条地执行语句，追踪程序的执行结果，这种运行方式称为“程序跟踪”。

Visual Basic 6.0 提供了在中断模式下的 4 种跟踪方式：逐语句、逐过程、运行到光标处、跳出，另外还提供了设置下一条执行语句的功能。

（1）“逐语句”跟踪

逐语句执行方式是一次执行一条语句，每执行一条语句后就暂停下来，用户可以使用“立即”窗口、“本地”窗口或“监视”窗口来判断执行语句是否正确。可以用以下 3 种方式来实现“逐语句”跟踪。

- ① 选择“调试”菜单中的“逐语句”命令。
- ② 按【F8】键。
- ③ 单击“调试”工具栏中的“逐语句”按钮。

设置了单步执行后，屏幕上将打开窗体，如图 1-2-13 所示，在执行图中“代码”窗口的程序时，要先单击窗体，才能开始单步执行。单步执行开始后，程序进入执行模式。执行完一条语句后，切换到中断模式，并把下一条语句作为“待执行语句”，待执行语句反相显示。

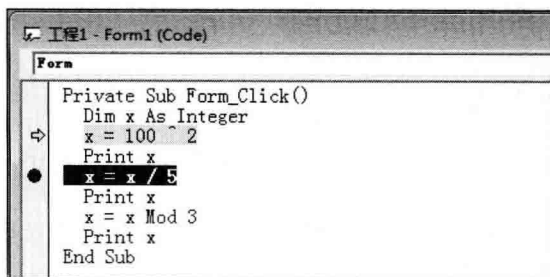


图 1-2-13 逐语句跟踪

(2) “逐过程”跟踪

“逐过程”跟踪的执行方式与单步跟踪基本相同，只是把被调用的过程当作一条语句一次执行完毕，然后停在下一条语句上，也称为“过程单步”。

如果确信某个过程不会有错误，则没有必要单步执行过程中的每个语句。在这种情况下，可使用“逐过程”跟踪。可以通过以下 3 种方式实现“逐过程”跟踪。

- ① 选择菜单“调试”→“逐过程”按钮。
- ② 按【Shift+F8】组合键。
- ③ 单击“调试”工具栏中的“逐过程”按钮。

(3) “运行到光标处”跟踪

“运行到光标处”即继续运行程序，运行到光标所在行暂停执行。可以通过以下 3 种方式实现“运行到光标处”跟踪。

- ① 选择菜单“调试”→“运行到光标处”按钮。
- ② 按【Ctrl+F8】组合键。
- ③ 单击“调试”工具栏中的“运行到光标处”按钮。

(4) “跳出”跟踪

“跳出”跟踪就是从当前过程中提前跳出，去执行过程调用者的下一条语句。一般用于当使用“逐语句”跟踪进入被调用过程后，如果从开始的几条语句就能断定出该过程没有问题，已没有必要再继续“逐语句”跟踪该过程的后继语句，那么就可以执行“跳出”跟踪方式。实现“跳出”跟踪有以下 3 种方式。

- ① 选择菜单“调试”→“跳出”命令。
- ② 按【Ctrl+Shift+F8】组合键。
- ③ 单击“调试”工具栏中的“跳出”按钮。

(5) 设置下一条语句

在前面的调试过程中，尽管可以随时中断程序的执行，但程序还是以正常的流程运行的。有时候，在更改了某变量或属性的值后，需要重新执行代码中的某一部分来观察更改后的运行结果，或暂时避开程序的某一部分，调试其他部分，这时可指定下一条要执行的语句。这种方式也称为“跳跃”执行。

设置下一条要执行的语句的方法如下。