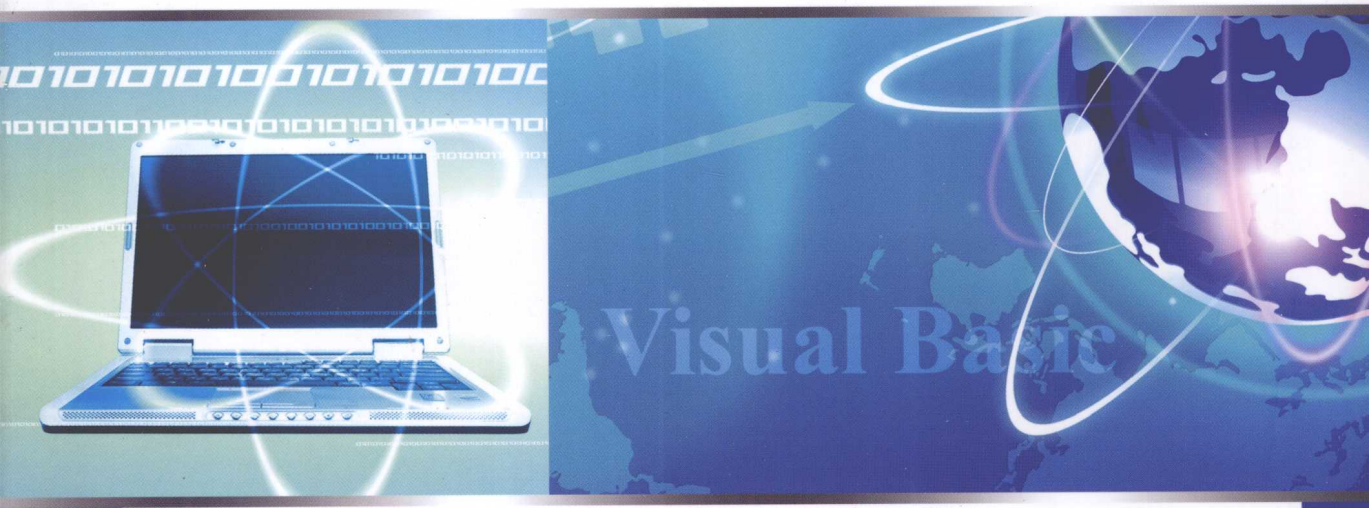
普通高等教育“十二五”规划教材

Visual Basic程序设计应用教程习题及实验指导



主　编　刘　利　汪华斌
副主编　黄　震　袁秀莲　李卫华
主　审　薛晓萍

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

Visual Basic 6.0 是一个功能强大的软件开发工具，使用 VB 6.0 可以快速地开发 Windows 环境下的应用程序。本书是与《Visual Basic 程序设计应用教程》配套的实验教材，共分四部分内容：程序调试与错误处理、习题及答案、实验指导和综合性设计性实验。实验指导部分根据主教材中的内容设计了十二个实验，涵盖了书中所有的重要知识点。

本书作为配套实验教材，结合课程教学和实验的特点合理安排教材内容。在程序调试与错误处理部分，详细介绍 VB 提供的调试技术和错误处理方法，并用实例讲解调试技术的应用；习题及答案部分按章归纳总结了大量的习题，其中大部分习题选自全国计算机等级考试二级 VB 考试试卷，全面涵盖主教材中的知识点；实验指导部分的实验内容紧跟主教材知识点，具备实用性、趣味性的特点，有助于进一步提高编程能力；综合性设计性实验部分提供了两个综合性的实验，由浅入深地帮助读者提高综合编程能力。

本书实验内容设计恰当、实验分析详细、习题丰富，既可作为普通高等院校 Visual Basic 程序设计课程的实验教材，也可作为广大计算机技术人员及全国计算机等级考试备考者的自学辅助用书。

图书在版编目（CIP）数据

Visual Basic程序设计应用教程习题及实验指导 / 刘利，汪华斌主编. -- 北京 : 中国水利水电出版社，2011.12

普通高等教育“十二五”规划教材

ISBN 978-7-5084-9266-7

Ⅰ. ①V… Ⅱ. ①刘… ②汪… Ⅲ. ①BASIC语言－程序设计－高等学校－教学参考资料 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2011)第261697号

策划编辑：陈宏华　　责任编辑：张玉玲　　封面设计：李　佳

书　名	普通高等教育“十二五”规划教材 Visual Basic 程序设计应用教程习题及实验指导
作　者	主　编　刘　利　汪华斌 副主编　黄　震　袁秀莲　李卫华 主　审　薛晓萍
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（发行部）、82562819（万水）
经　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	北京泽宇印刷有限公司
规　格	184mm×260mm　16 开本　13.5 印张　340 千字
版　次	2011 年 12 月第 1 版　2011 年 12 月第 1 次印刷
印　数	0001—3000 册
定　价	24.00 元

前　言

Visual Basic（简称 VB）是微软公司推出的 Windows 应用程序开发工具，是基于事件驱动、面向对象的可视化编程语言。VB 具有简单易学、功能强大、开发速度快等特点，深受广大程序开发人员的青睐，已成为世界上应用最广泛的高级程序设计语言之一。

本书从实践出发，辅助读者学习 VB，配合主教材进一步提高读者的编程能力。读者可以通过第一部分全面学习 VB 6.0 开发环境中所提供的调试技术，以及错误处理方法。为加深读者对知识点的理解，以及辅助读者参加各类 VB 考试，在第二部分中提供大量的习题，其中大部分习题选自近十年的全国计算机等级考试二级 VB 考试试卷，经过编者精心筛选汇编，并提供习题答案。实验指导部分，涵盖 VB 课程所有的重要知识点，按知识点设计了十二个实验，在每个实验的实验内容中，奇数题目给出详细的解答，偶数题目给出实验提示，既有助于读者自学，又能兼顾提高读者独立编程的能力。在综合性设计性实验部分，设计了两个综合实验，涉及书中大部分的知识点，通过完成此部分实验内容，可进一步加深读者对 VB 编程的理解，提高读者的综合编程能力。

本书主要内容如下：

第一部分程序调试与错误处理，主要介绍 VB 6.0 开发环境中所提供的调试技术，以及错误处理方法。

第二部分习题及答案，根据主教材的章节按章汇编了大量的选择题。

第三部分实验指导，根据《Visual Basic 程序设计应用教程》设计了如下十二个实验：

实验一　VB 环境及简单应用程序设计
实验二　数据类型、运算符、表达式和函数
实验三　顺序结构和选择结构程序设计
实验四　循环结构程序设计
实验五　常用控件
实验六　数组程序设计
实验七　过程程序设计
实验八　窗体界面设计
实验九　文件
实验十　图形设计
实验十一　数据库技术
实验十二　OLE 控件和 API 函数

第四部分综合性设计性实验，综合 VB 的主要知识点设计了如下两个综合实验并给出了详细的解答：

实验十三　小学生四则运算练习系统
实验十四　贪食蛇游戏

本书主要特点如下：

（1）实用性强。本书结合教学和实验的特点，根据教学内容设计丰富实验内容的同时，详细介绍 VB 程序调试技术和错误处理方法。所设计的实验具备实用性和趣味性等特点。

（2）丰富的习题。本书按章节汇编了大量的选择题，内容涵盖 VB 课程的主要知识点。其中大量题目选自近十年的全国计算机等级考试二级 VB 考试试卷，有助于读者加深对 VB 课程的理解和掌握，且可用于读者参加各类考试的练习准备。

（3）实验内容设计合理。实验内容的设计除考虑涵盖 VB 课程的主要知识点外，还对部分题目设计了思考题，有助于读者开拓思维，进一步提高编程能力。

本书由刘利、汪华斌任主编，黄震、袁秀莲、李卫华任副主编。具体分工为：第一部分、实验八、实验十和实验十二由汪华斌编写，实验一和实验三由李卫华编写，实验二和实验六由袁秀莲编写，实验四、实验五、实验七和实验十三由刘利编写，实验九、实验十一和实验十四由黄震编写，第二部分习题由大家共同整理。赵义霞、季军杰、兰远东、王健海、李慧、陈朝华、肖东、曾树洪、李旌燕对本书进行了校对，刘利和汪华斌对全书进行了统稿，薛晓萍教授对全书进行了审稿。

由于编者水平有限，书中难免还存在一些错误和不妥之处，恳请广大读者批评指正。

编　者

2011 年 10 月

目　　录

第一部分　程序调试与错误处理

无论用户如何仔细地编写计算机程序，都可能会发生错误。可能会有键入错误，程序可能不像预期的那样执行，或者可能根本无法运行。如果程序中发生错误，则需要找出该错误并修复它。查找并修复错误的过程称为“调试”。因此，用户在编写较大些的程序时，应该了解如何查找程序中的错误。本章介绍的就是如何处理应用程序中可能出现的各种错误，同时介绍如何使用 VB 中的调试工具查找逻辑错误，使程序可以适应各种复杂的情况。

1.1　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 有三种模式：设计模式、运行模式和中断模式，它们是编写、调试和运行应用程序时应用程序所处的不同状态。

1.1.1　设计模式

启动 Visual Basic 后，打开一个工程窗口，即进入设计模式。在主窗口标题栏上显示“[设计]”字样，如图 1 所示。

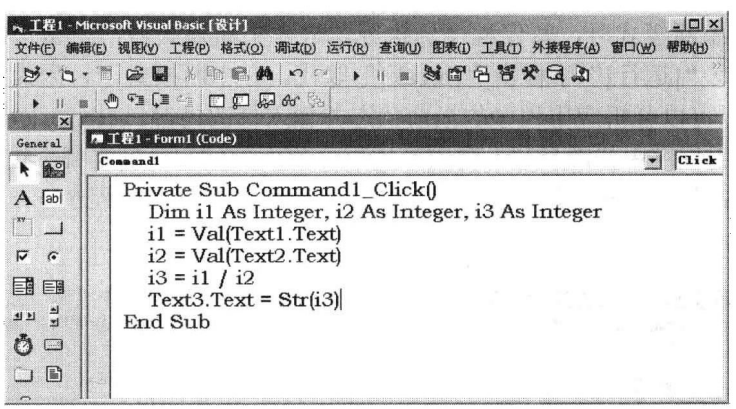


图1　程序设计模式

创建一个应用程序的所有工作都是在设计模式下完成的。在设计时，可以为主程序设计窗体，或在窗体上绘制控件，或编写程序代码及添加注释，并可使用“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其进行调试，但可以指定断点和创建监视表达式。

1.1.2　执行模式

选择“运行”→“启动”命令（或者按 F5 键或单击工具栏中的“运行”按钮 ），应用

程序即可进入执行模式，此时主窗口标题栏上原来显示的“[设计]”字样被“[运行]”取代，如图 2 所示。

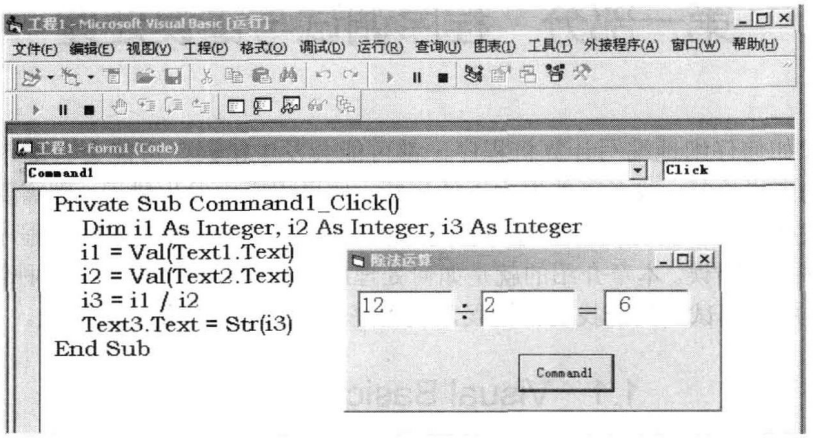


图2 程序运行模式

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，可以查看程序代码，检验程序运行的结果，但不能修改程序代码。

选择“运行”→“结束”命令（或者单击工具栏中的“结束”按钮）可返回设计模式。

选择“运行”→“中断”命令（或者单击工具栏中的“中断”按钮 或按 Ctrl+Break 组合键）可进入中断模式。

1.1.3 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，主窗口的标题栏中原来显示的“[设计]”或“[运行]”字样由“[break]”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，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，立即运行 Visual Basic 语句，手工控制应用程序的操作，如图 3 所示。

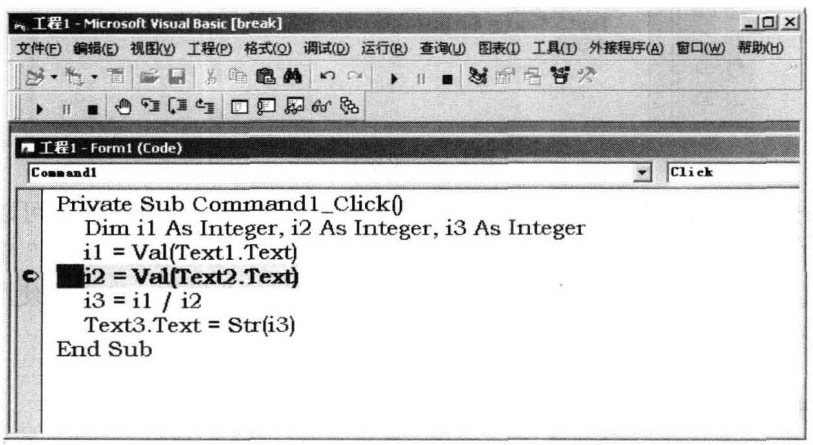


图3 程序中断模式

可以用以下 4 种方式来进入中断模式：

（1）在执行模式下，选择“运行”→“中断”命令。

（2）在程序中设置断点，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，则会自动进入中断模式。

1.2　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）。在学习有关调试过程的知识之前，了解需要找出并修复的 Bug 类型是很有帮助的。

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。

1.2.1　语法错误

语法错误是由于不正确地创建代码，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()
    Dim i1 As Integer, i2 As Integer, i3 As Integer
    i1 = Val(Text1.Text)
    i2 = Val(Text2.Text)
    i3 = i1/i2
    Text3.Text = Str(i3)
End Sub
```

在输入代码时，若将第 7 行代码输入为：

```
i3 = i1 {i2
```

程序运行后就会显示“语法错误”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 4 所示。

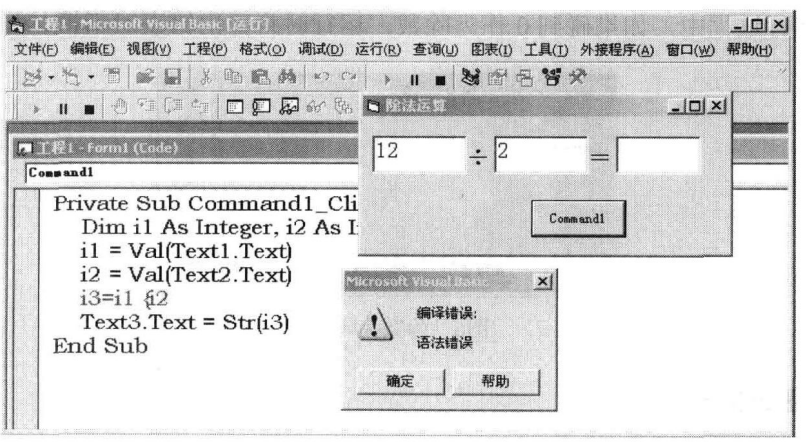


图4　语法错误

只有改正该语法错误后，红色和高亮显示才会消失。

1.2.2 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、遗漏关键字、函数未定义或找不到相应控件的方法与属性等，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 5 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。

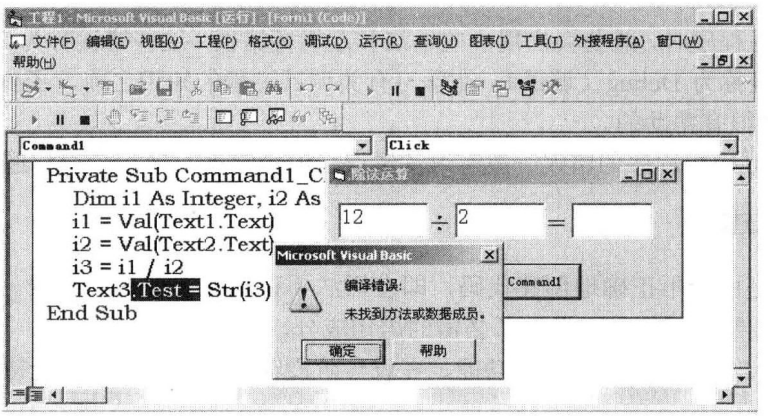


图5 编译错误

1.2.3 运行错误

若语法正确，并且程序运行期间，一个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。

常见的有：

（1）数据类型不正确。

（2）对象或控件不存在。

（3）0 作为除数。

在程序运行过程中，如果碰到 0 作为除数，运行时将弹出错误对话框，如图 6 所示。

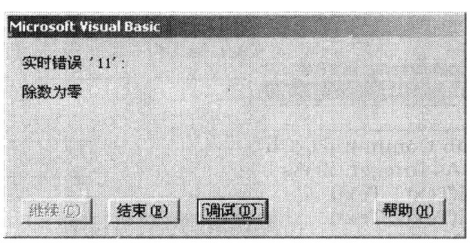


图6 除数为零错误

1.2.4 逻辑错误

程序逻辑错误是指应用程序未按设计运行或程序运行后得出的不是事先预定的正确结

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。逻辑错误产生的原因很多，运算符使用不正确、语句次序不对、循环的设置不对等，都可以产生逻辑错误。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。对于逻辑错误 VB 是检查不出来的，因此也没有提示信息产生，要减少或克服逻辑错误，没有捷径可循，只能靠我们自己仔细分析阅读程序，并充分使用调试工具，这样才能避免错误的发生。这类错误最难发现，因而也最危险。

1.3　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位置和原因，并加以纠正。

1.3.1　“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的值等。在各种调试工具中，“调试”工具栏提供了对绝大多数调试工具的访问。如果在屏幕中没有“调试”工具栏，则可选择“视图”→“工具栏”→“调试”命令，或者在 Visual Basic 6.0 工具栏中右击，在弹出的快捷菜单中选择“调试”命令，均会调出“调试”工具栏，如图 7 所示。

图7　“调试”工具条

1.3.2　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”窗口等这些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”窗口用来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者表达式的值。

在设计阶段，选择“调试”→“添加监视”或“快速监视”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“调试”→“添加监视”命令，弹出“添加监视”对话框，如图 8 所示。

（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。

2. “立即”窗口

程序进入中断模式后，将自动激活“立即”窗口。“立即”窗口用于显示过程代码中调试

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，如图 9 所示。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。

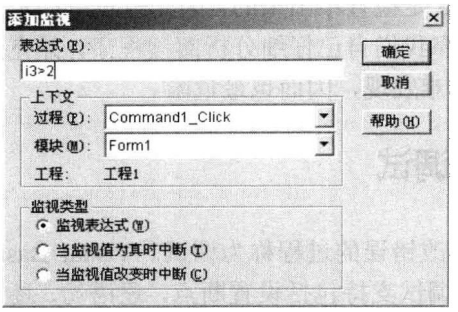


图8 “添加监视”对话框

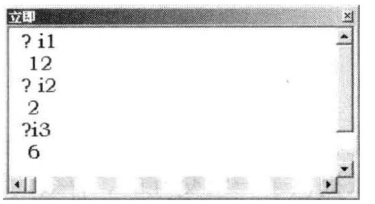


图9 “立即”窗口

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。

如果未显示“本地”窗口，则选择“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 10 所示。

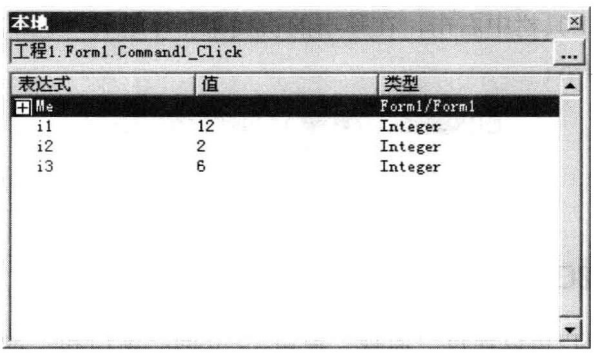


图10 “本地”窗口

1.3.3 使用中断

在设计状态下，可以改变应用程序的设计和代码，但不能立即看到这些变更对程序运行所产生的影响；在运行程序时，可以观察到程序的运行状态，但不能直接改变代码。通过设置运行断点，VB 系统可以中止程序的运行，使程序进入到中断模式。在中断模式下，系统保留着发生中断时的运行状态，包括各个变量和属性的设置值，供用户观察、分析，同时允许直接修改应用程序的代码，影响程序的运行。

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

设置运行断点通常有两种方法：

（1）在代码窗口中单击最左边的灰色区域，使之出现一个棕色标志●，对应的代码行被同时加亮，则此处便设置了一个断点。

（2）将光标移动到要设置断点的代码行，选择“调试”→“切换断点”命令亦可设置一个断点。

断点通常设置在程序中需要暂停执行的地方，如图 11 所示。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执行情况。当在程序运行中遇到断点时，Visual Basic 就会进入中断模式。

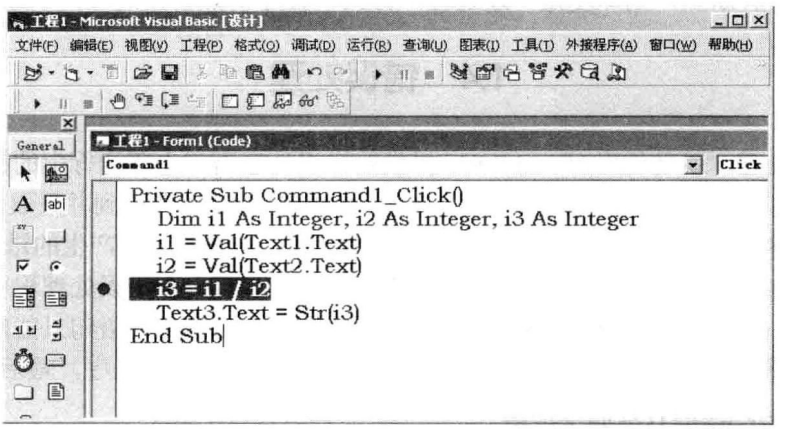


图11　断点设置

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暂停模式，以便对程序进行调试，如图 12 所示。

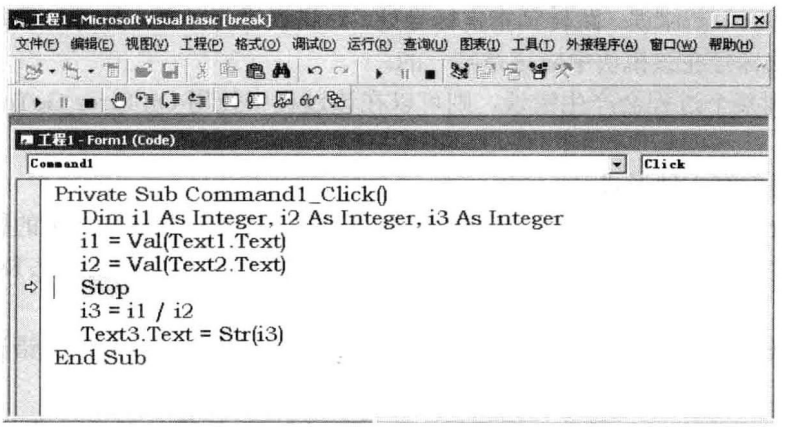


图12　Stop 语句

当执行以上程序中的 Stop 语句时，程序暂停执行，进入中断模式。

1.3.4　使用单步调试和跳跃调试

程序运行后遇到断点进入中断模式，此时断点所在的代码行还没有被执行。如要观察断点所在的行以及其后代码行的执行情况，可以使用“逐语句”功能进行单步调试。

使用 F8 快捷键或者“调试”→“逐语句”命令都可以进行单步运行。单步运行时，程序每次执行一行代码，然后继续等待，我们便可以观察到该行执行后的运行状态。反复使用“单

步运行”功能，就可以对程序执行的每一行进行全面跟踪。 如果当前代码行调用了一个过程，则利用“逐语句”功能还可以跟踪到过程的内部，了解该过程的执行情况。

使用 Shift+F8 快捷键或者“调试”→“逐过程”命令也可以进行单步运行。与“逐语句”功能的区别是，当被执行的代码行调用一个过程时，“逐过程”功能不向过程的内部跟踪，它把过程调用视为一个整体单元来执行，然后到达过程执行后的下一条语句。例外的情况是，被调用的过程中含有断点，则即便是“逐过程”方式，程序仍然会在过程内的断点处中断。

1.4 错误处理

所谓错误处理，就是使用 Visual Basic 提供的错误处理语句中断运行中的错误，并进行处理。对于运行期间程序可能会遇到的错误，在设计代码时就应该考虑到并采取预防措施，以避免程序的异常终止。预防的办法就是添加错误处理程序，捕获并处理产生的错误。由于 Visual Basic 不支持集中处理技术，所以每一个过程或事件都应该有一个错误处理程序来解决它自己的错误。Visual Basic 可以截获的错误称为可捕获的错误，错误处理语句只能对这类错误进行处理。错误处理过程一般是先设置陷阱捕获错误，再进入错误处理程序，最后退出错误处理。

1.4.1 错误的捕捉及处理子程序

1. 设置错误陷阱

Visual Basic 提供了 On Error 语句来设置错误陷阱，捕捉错误。该语句的使用语法如下：

```
On Error GoTo ErrorHandler
```

其中，ErrorHandler 是一个错误处理程序段的标号，当执行一条语句产生一个可捕获的错误时，该语句可以中断错误，跳转到指定标号处，对错误进行处理。标号可以是以字母开头的任意字符串，该标号在该模块中应该是唯一的。

如果预感到某个过程会产生错误，则可以在该过程中使用 On Error GoTo 语句设置错误陷阱。

On Error 语句有 3 种形式：

（1）On Error GoTo 语句标号：在发生运行错误时，转到语句标号指定的程序块执行错误处理程序。指定的程序块必须在同一过程中，错误处理程序的最后必须加上 Resume 语句，以告知返回位置。

（2）On Error Resume Next：在发生运行错误时，忽略错误，转到发生错误的下一条语句继续运行。

（3）On Error GoTo 0：停止错误捕捉，由 Visual Basic 直接处理运行错误。

2. 编写错误处理程序

编写错误处理程序一般要使用到 Err 对象，它是一个系统对象，在 Visual Basic 中，可以通过 Err 对象来获取错误的消息。当出现 Visual Basic 错误时，有关错误的信息将存储在 Err 对象中。Err 对象每次只维护一个错误信息，当出现新的错误时，Err 对象将更新为新的错误信息。

Err 对象的默认属性是 Number 属性，当运行错误发生时，将在 Err 对象的属性中添加明确识别错误的信息以及处理这个错误所使用的信息。

当 On Error 捕捉到错误后，Err 对象的 Number 属性指示对应的错误号，错误处理程序中

一般用 Select Case Err.Number 或 If Err.Number=语句确定可能发生的错误并对每种不同的错误提供错误处理方法。

3. 退出错误处理

在错误处理程序中，当遇到 Exit Sub、Exit Function、End Sub、End Function 等语句时，将退出错误捕获。

在错误处理程序结束后，要恢复运行，可用 Resume 语句。

Resume 语句应放置在出错处理程序的最后，以便错误处理完毕后指定程序下一步的操作。Resume 语句也有 3 种形式：

（1）Resume 语句标号：返回到标号指定的行继续执行，若标号为 0，则表示终止程序的执行。

（2）Resume Next：使程序返回到导致错误的语句之后的那条语句上开始执行，即跳到出错语句的下一条语句继续执行。

（3）Resume：使程序返回到导致错误的那条语句上重新执行。

1.4.2　Error 函数与 Error 语句

在 Visual Basic 中错误有 4 类：语法错误、编译错误、运行错误和逻辑错误。可以用 Error 语句模拟程序运行错误。

1. Error 函数

格式：字符串=Error$[(错误代码)]

在 Visual Basic 中，每个出错信息都有与之对应的错误代码，其取值范围为 0～65535。Error$函数返回与“错误代码”相对应的出错信息。如果省略“错误代码”，则返回最后执行错误语句的出错信息；如果程序中没有错误，则返回空字符串。如果所给的“错误代码”不是 Visual Basic 预定义的错误代码，则函数返回的信息是“应用程序定义或对象定义错误”。

例如：

```
Print Error$(11)     '将输出“除数为 0”
Print Error$(424)    '返回“要求对象”，说明代码中使用的对象未定义
```

2. Error 语句

格式：Error　错误代码

Error 语句用来模拟发生一个 Visual Basic 错误。错误代码的取值范围是 1～32767。如果所给出的“错误代码”属于 Visual Basic 预定义的错误代码，则模拟发生的错误，同时输出预定义的错误信息。

由于 Error 语句用来模拟错误的发生，因此应设置错误陷阱并定义错误处理子程序，否则也会发生错误而停止程序运行。

如果给出的错误代码不是 Visual Basic 预定义的，则可以定义；用户自己的错误代码，定义之后就可以在程序中使用。例如：

```
...
On Error GoTo msg_error
i1 = Val(Text1.Text)
i2 = Val(Text2.Text)
if i1 > 100 then Error 210
i3 = i1/i2
```

```
    Text3.Text = Str(i3)
msg_error:
    MsgBox "操作数不能大于 100"
    …
End Sub
```

上面的程序段预定义了一个错误代码 210，用这种方法定义的错误代码必须放在错误处理子程序中处理，即在错误处理子程序中定义与该码相对应的出错信息。

1.4.3 设计错误处理程序

设计错误处理程序的步骤如下：

（1）捕捉错误。当错误发生时，使用 On Error 语句捕获错误并把处理流程转向由标号指定的错误处理程序处。

（2）编写错误处理程序。对所有能预见的错误做出响应，并设计处理程序。

（3）退出错误处理程序。错误处理程序处理完错误后，应尽量恢复程序的正常运行。因此，在错误处理程序的最后使用一条不同形式的 Resume 语句来退出错误处理程序。

例如，从键盘上输入两个整数，计算并输出它们相除所得的商和余数。相应的程序如下：

```
Private Sub Command1_Click()
    Dim i1 As Integer, i2 As Integer, i3 As Integer
    i1 = Val(Text1.Text)
    i2 = Val(Text2.Text)
    i3 = i1/i2
    Text3.Text = Str(i3)
End Sub
```

上述程序运行后，利用窗体上的两个文本框输入两个数，程序代码计算并输出两个数相除所得的商。在该程序中，没有设置错误陷阱，将按正常方式处理所遇到的错误。当输入的除数 i2 为 0 时，程序停止运行并显示“除数为零”的信息，如图 13 所示。

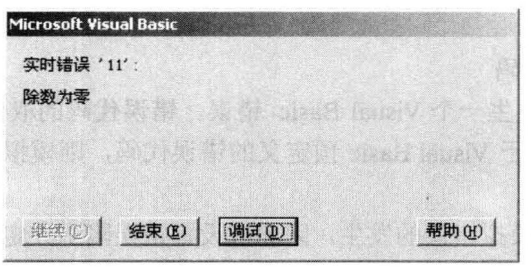


图13 除数为零错误

利用错误陷阱可以使上述程序在出现错误时不停止运行，并要求重新输入数据。修改后的程序如下：

```
Private Sub Command1_Click()
    On Error GoTo msg_error
    Static n As Integer
    Dim i1 As Integer, i2 As Integer, i3 As Integer
    Debug.Print
```

```
    Debug.Print "第" & n & "次执行"
    i1 = Val(Text1.Text)
    i2 = Val(Text2.Text)
    i3 = i1/i2
    Text3.Text = Str(i3)
    n = n + 1
    Exit Sub
msg_error:
    msgerr = "错误代码为："
    msgtest = "On Error 捕获错误实例"
    Select Case Err.Number
        Case 6          '溢出错误
            MsgBox Err.Description & vbCrLf & msgerr & Err.Number, , msgtest
        Case 11         '除数为 0 错误
            MsgBox Err.Description & vbCrLf & msgerr & Err.Number, , msgtest
            Text2.Text = ""
            Text2.SetFocus
        Case 13         '类型不匹配
            MsgBox Err.Description & vbCrLf & msgerr & Err.Number, , msgtest
        Case 424        '对象不存在
            MsgBox Err.Description & vbCrLf & msgerr & Err.Number, , msgtest
        End Select
End Sub
```

上述程序，当在 Text2 文本框中输入 0 时，提示错误如图 14 所示。

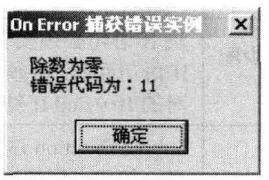


图14　除数为零错误

单击图 14 中的“确定”按钮后，Text2 中的内容被清空并获取到焦点，可重新输入，如图 15 所示。

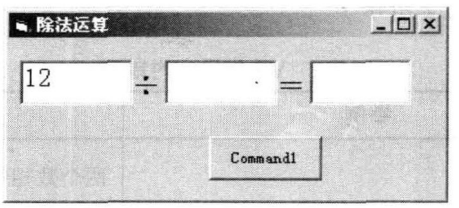


图15　重新输入除数

在该程序中用 On Error GoTo msg_error 来设置错误陷阱，当发生错误时，转移到 msg_error 开始的错误处理子程序。在程序中，用 MsgBox 语句使得语句输出明显易懂的出错信息，要求重新输入。单击“确定”按钮后，Resume Begin 语句使得程序在错误处理子程序之后从 Begin: 处继续运行。

同时上述程序还通过“Debug.Print "第" & n & "次执行"”在“立即”窗口中显示当前程序计算的次数。

1.5 常见程序调试技巧

1. 常见程序调试技巧

调试程序是一件复杂的工程，不但要求程序员对系统设计结果非常熟悉、思路清晰，更需要不断地在实践中积累经验。以下技巧可供参考：

（1）事先做好备份。

（2）分离受怀疑的程序。

（3）缩小搜索范围。

（4）使用 MsgBox 语句。

2. 常见编译错误（如表 1 所示）

表1 VB 常见编译错误

错误信息	举例	错误原因
缺少语句结束	Fori=1 To 10	For 关键字和 i 之间没有分隔符，VB 将 For 理解为一个变量，将 Fori=1 作为一个赋值语句来使用
	x=3y=24	x=3 作为一个完整的语句应该结束，两个赋值语句中间缺少冒号或分写在两行
缺少函数或变量	x=2+MySub(5,7)	过程调用不应该出现在表达式中，过程调用应作为独立的命令
If 块缺少 End If	使用块 If 语句时，缺少配对的 End If 语句	块 If 语句必须要有配对的 End If 语句。对于嵌入式 If…End If 语句而言，必须确保每个封闭的 If…End If 结构中应有配对的 If…End If 结构
子程序或函数未定义		Sub 或 Function 过程必须先定义，然后才能调用。如果已经定义，但仍出现该错误，可能是过程名称拼错。另外，在模块中声明为 Private 的过程不能被模块外部的过程调用
For 没有 Next 或 Next 没有 For		每一个 For 语句必须有配对的 Next，反之亦然

3. 常见实时错误（如表 2 所示）

表2 VB 常见实时错误

错误信息	举例	错误原因
溢出（错误 6）	Dim x As Iong x = 2000*365	两个数相乘的结果超过了整数的范围
类型不匹配（错误 13）	x% ="123a"	可将整个字符串视为整型
文件模式错误（错误 54）	将 Print 语句使用在非 OutPut 或 Append 访问方式所打开的文件上，或者将 Input#语句或 Line Input#语句使用在非 Input 访问模式所打开的文件上，就会导致此类错误	文件的处理方式，必须与打开的文件的模式一致