

· 高等学校计算机基础教材精选 ·

# 基于Visual C++的标准C 实用程序设计教程 ——上机指导与习题解

杨素英 孟军 冯毅 编著

TP312C



清华大学出版社

内容简介

· 高等学校计算机基础教育教材精选 ·

# 基于Visual C++的标准C 实用程序设计教程 ——上机指导与习题解

杨素英 孟军 冯毅 编著

清华大学出版社  
北京

## 内 容 简 介

本书是《基于 Visual C++ 的标准 C 实用程序设计教程》(杨素英等编著,清华大学出版社出版)的配套上机指导和习题解。本书章节独立、解题完整、深入浅出、循序渐进、综合性与实用性强。全书分两部分。第一部分为 Visual C++ 6.0 和 Turbo C 2.0 开发系统的使用方法,针对示例一步一步讲解不同系统调试程序的操作方法。第二部分为习题解题分析及程序设计。从第 4 章习题开始,针对单元内容,讲解上机调试的要点和程序测试运行的基本方法,使读者了解程序运行异常的原因。书中全部程序均通过调试和运行。

本书既可以作为程序设计课程教学的课后自学辅导书,也可以作为实践课程的上机指导用书。如果读者具备初级编程水平完全可以使用本书进行自学和编程训练指导。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

## 图书在版编目(CIP)数据

基于 Visual C++ 的标准 C 实用程序设计教程:上机指导与习题解 / 杨素英,孟军,冯毅编著. —北京:清华大学出版社,2010.4

(高等学校计算机基础教育教材精选)

ISBN 978-7-302-22143-2

I. ①基… II. ①杨… ②孟… ③冯… III. ①C 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(2010)第 024829 号

责任编辑:袁勤勇 李玮琪

责任校对:李建庄

责任印制:孟凡玉

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn>

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

质 量 反 馈:010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

印 装 者:北京国马印刷厂

经 销:全国新华书店

开 本:185×260

印 张:7.75

字 数:192 千字

版 次:2010 年 4 月第 1 版

印 次:2010 年 4 月第 1 次印刷

印 数:1~3000

定 价:16.00 元

---

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:010-62770177 转 3103 产品编号:036461-01

# 出版说明

高等学校计算机基础教材精选

在教育部关于高等学校计算机基础教育三层次方案的指导下,我国高等学校的计算机基础教育事业蓬勃发展。经过多年的教学改革与实践,全国很多学校在计算机基础教育这一领域中积累了大量宝贵的经验,取得了许多可喜的成果。

随着科教兴国战略的实施以及社会信息化进程的加快,目前我国的高等教育事业正面临着新的发展机遇,但同时也必须面对新的挑战。这些都对高等学校的计算机基础教育提出了更高的要求。为了适应教学改革的需要,进一步推动我国高等学校计算机基础教育事业的发展,我们在全国各高等学校精心挖掘和遴选了一批经过教学实践检验的优秀教学成果,编辑出版了这套教材。教材的选题范围涵盖了计算机基础教育的三个层次,包括面向各高校开设的计算机必修课、选修课以及与各类专业相结合的计算机课程。

为了保证出版质量,同时更好地适应教学需求,本套教材将采取开放的体系和滚动出版的方式(即成熟一本、出版一本,并保持不断更新),坚持宁缺毋滥的原则,力求反映我国高等学校计算机基础教育的最新成果,使本套丛书无论在技术质量上还是文字质量上均成为真正的“精选”。

清华大学出版社一直致力于计算机教育用书的出版工作,在计算机基础教育领域出版了许多优秀的教材。本套教材的出版将进一步丰富和扩大我社在这一领域的选题范围、层次和深度,以适应高校计算机基础教育课程层次化、多样化的趋势,从而更好地满足各学校由于条件、师资和生源水平、专业领域等的差异而产生的不同需求。我们热切期望全国广大教师能够积极参与到本套丛书的编写工作中来,把自己的教学成果与全国的同行们分享;同时也欢迎广大读者对本套教材提出宝贵意见,以便我们改进工作,为读者提供更好的服务。

我们的电子邮件地址是 [jiaoh@tup.tsinghua.edu.cn](mailto:jiaoh@tup.tsinghua.edu.cn)。联系人:焦虹。

清华大学出版社

# 前言

—— 基于 Visual C++ 的标准 C 实用程序设计教程 —— 上机指导与习题解 ——

为了达到使读者从实际训练中巩固基础、熟练编程方法的目的,本着培养大学生解决实际问题的能力及提高创新能力的目标,本书作者总结了几十年来累积的实践经验和教学经验,编写了用于 C 语言程序设计上机练习和实训设计相关实践环节的教学丛书。本书所有程序均由作者和参编老师认真编写和提炼,花费大量时间上机调试、运行,程序设计不仅涉及解题方法,同时渗透了程序设计中数据访问的安全性和代码优化问题,为读者掌握具有安全性程序设计思想起到了潜移默化的作用,这也是本书强调“实用”程序设计的一个重要教学目的。本书分为 Visual C++ 6.0 和 Turbo C 2.0 平台的上机指导和习题解两大部分。

本书的第一部分讲解开发工具的使用方法。分别使用当前广泛应用的 Visual C++ 6.0(VC++)和 Turbo C 2.0(TC 2.0)两个开发工具作为 C 程序的开发平台。VC++ 兼容标准 C 语言大部分文本方式下的程序运行环境,由于 VC++ 的运行环境是 Windows 操作系统,因此 VC++ 也具有微软 Word 办公软件的编辑功能,如文本的删除、复制、移动、粘贴等,其鼠标操作也比 DOS 系统方便得多,这也是当前很多人喜欢在这两个平台下开发 C 程序的原因。但是,C 语言就它的出身而言起源于 UNIX 系统,属于默认的文本模式,所以 VC++ 的可视化编程环境无法全部支持标准 C。C 语言强大之处并不是编写精美而庞大的软件,而是编写的程序代码具有高质量,同时又允许发挥它的高级语言程序设计之长处。

借助 VC++ 平台编写文本模式下的 C 程序,对于初学者来说入门快,尽早熟悉这个平台的调试工具,对进一步面向对象的程序设计实践是很有帮助的。但是因为 VC++ 环境下所提供的 C 语言编程资源有限,当需要发挥 C 的精、小、快等优势时,使用母本开发工具是更好的选择,如 DOS 运行环境的 Turbo C 2.0、Turbo C++ 3.0、Borland C 3.1 等都是专用于 C 程序的集成开发系统,其中 Turbo C++ 3.0 和 Borland C 3.1 本身也是面向对象的编程工具,提供鼠标操作。TC 2.0 也提供了完整的快捷键操作,这些按键操作可以通过在线帮助获得,易学易用。这些工具都完全兼容标准 C,开发资源可以得到充分利用,因此建议读者在 C 语言达到了一定程度时,可以转向使用 TC 开发系统,逐步编写应用范围更宽的 C 程序,例如彩色文本模式的图形编程、指针的底层接口驱动、中断系统多任务并行运行等相关程序设计,这些内容都涉及计算机硬件的程序设计。根据个人习惯充分利用两个平台的优势,发挥其不同的作用,使开发程序的空间更大,选择更灵活,也是本书编写的宗旨所在。

本书的第二部分是供读者参考的习题解答部分。为了使读者能够通过学习和编程练习,消化理解 C 语言的设计思想和设计方法,习题内容由浅入深、循序渐进。本书从第 4 章开始,在每章的习题解之前,讲解本章上机调试要点,阐述调试中的常用方法,另外,还讲解如何测试已经通过调试的程序运行结果的正确性,这些内容都是本书作者和参编同行老师从多年来的实践中积累的调试经验总结出来的,具有很强的实践指导意义,希望读者能从中受益。每章的习题程序设计解答部分都有题意分析,然后阐述解题的设计思想,部分题解给出了算法实现的步骤,部分题解给出了两种设计方法,便于对比,使读者掌握设计思路,有利于激发灵活的解题思维方式。虽然本书所有的程序都通过调试运行,但任何一个解题方法都不是唯一的,所以本书给出的程序解答仅供参考。

本书注重基础,并以应用为主,分析详细、解题完整。通过开发平台、调试与测试、解题思路、编写程序,使读者在学习 C 语言的过程中不但能够概念清晰,更重要的是找到解题入口点,熟练使用模块化设计方法解决各种实际问题,最终掌握实用程序设计从构思到运行的全过程。

本书第一部分由大连理工大学的杨素英、孟军编写,第二部分由杨素英、孟军、冯毅编写。全书由杨素英统稿,王金城审稿。张鑫、高苗苗、马颖等研究生与参编老师共同参加本书习题的修改和程序调试,在此作者表示诚挚的谢意,谢谢所有参编人员,正是他们的辛苦付出才使本书得以顺利完成。书中难免存在不妥与疏漏之处,欢迎读者及同行批评指正。

编者  
2010 年 3 月

# 目录

—— 基于 Visual C++ 的标准 C 实用程序设计教程 —— 上机指导与习题解 ——

|                                           |     |
|-------------------------------------------|-----|
| <b>第一部分 C 程序上机调试方法</b> .....              | 1   |
| 1.1 基于 Visual C++ 6.0 平台的 C 程序调试 .....    | 1   |
| 1.1.1 Visual C++ 6.0 环境下 C 程序调试基本方法 ..... | 1   |
| 1.1.2 C 程序调试方法 .....                      | 4   |
| 1.1.3 关闭工作区调试新程序 .....                    | 11  |
| 1.1.4 工程文件调试方法 .....                      | 13  |
| 1.2 DOS 操作系统基本操作 .....                    | 15  |
| 1.2.1 启动 DOS 操作系统 .....                   | 15  |
| 1.2.2 DOS 常用命令 .....                      | 16  |
| 1.3 Turbo C 2.0 集成开发系统 .....              | 17  |
| 1.3.1 集成开发系统启动与退出 .....                   | 17  |
| 1.3.2 选择工作目录 .....                        | 18  |
| 1.3.3 建立工作环境 .....                        | 19  |
| 1.3.4 编辑源文件 .....                         | 20  |
| 1.4 Turbo C 2.0 平台下程序调试 .....             | 22  |
| 1.5 C 程序调试要点 .....                        | 24  |
| 1.5.1 基本调试方法 .....                        | 24  |
| 1.5.2 工程文件的建立与调试 .....                    | 26  |
| 1.5.3 Turbo C 2.0 集成开发系统编辑快捷键 .....       | 32  |
| <b>第二部分 习题题解与程序设计</b> .....               | 34  |
| 第 1 章 C 语言概述习题 .....                      | 34  |
| 第 2 章 数据类型习题 .....                        | 35  |
| 第 3 章 运算符与表达式习题 .....                     | 41  |
| 第 4 章 数据输入输出与流程控制习题 .....                 | 44  |
| 第 5 章 模块化设计习题 .....                       | 62  |
| 第 6 章 指针基础习题 .....                        | 75  |
| 第 7 章 指针的应用习题 .....                       | 91  |
| 第 8 章 程序数据文件习题 .....                      | 104 |

C 语言开发系统较流行 DOS 运行环境下的 Turbo C、Borland C 等,虽然这些开发系统具有占用系统资源少、开发效率高等优点,但是 DOS 环境下的文本操作方式相对图形可视化的文本编辑存在操作不够灵活等不足。本教材的基础部分均采用 Visual C++ 6.0 开发系统,讲解 C 程序的编程和调试方法。同时兼顾标准 C 的支持性技术,在综合编程教学中采用 Turbo C 2.0 面向过程程序设计的开发环境,它更容易发挥 C 低级驱动和管理的功能,实现更大范围的程序开发。

## 1.1 基于 Visual C++ 6.0 平台的 C 程序调试

Visual C++ 6.0(以下简称 VC++) 作为面向对象的开发系统,提供了一套完整的程序设计方案,能够避免大量模式化工作。VC++ 不仅适用于 C++ 语言的编程,也兼容 C 语言的编程,借助 VC++ 平台学习 C 程序设计,从其提供的资源来看实为“大材小用”,但从教学目的来看,熟悉了这个平台,充分发挥其具有的文本编辑功能,编辑器能够支持 C 语言的大部分文本编程,而且符合标准 C 编程规范。从提高程序设计水平的角度,了解它的调试和资源调用方法,对进一步向面向对象软件设计水平的提高,可以起到巩固基础的作用。

### 1.1.1 Visual C++ 6.0 环境下 C 程序调试基本方法

VC++ 是一个庞大的语言集成工具,安装后占用几百兆的磁盘空间。本小节从启动调试平台、新建文件、编辑 C 文件、编译、链接及运行 6 个方面介绍在这个平台下调试 C 程序的方法。

#### 1. 启动 Visual C++ 6.0 及新建文件

编写调试程序,首先要进入开发系统,建立新文件,然后编辑程序代码,最后就可以进行调试运行了。如图 1.1(左)所示为启动 VC++ 后的开发界面。在 Windows 环境下,可按照如下步骤启动开发系统:

(1) 依次选择桌面左下角的“开始”→“程序”→Microsoft Visual Studio 6.0→Microsoft Visual C++ 6.0,即启动 Visual C++ 6.0 程序,进入 VC++ 软件开发平台的编

程环境,如图 1.1(右)所示为新建文件的对话框。

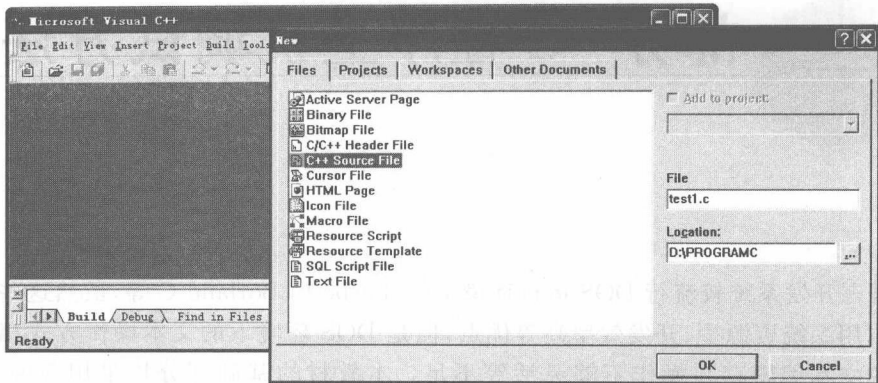


图 1.1 Visual C++ 6.0 开发平台启动后的界面(左)及 新建文件的对话框(右)

(2) 选择 File→New 命令,就会弹出一个对话框,切换至 Files 选项卡。在 File 文本框中输入要编辑的新文件名 test1.c,由于 VC++ 的源文件默认扩展名为 .cpp,为了编辑标准 C 文件,要指定扩展名为 .c,即需要输入完整的文件名 test1.c,否则系统将按照 C++ 的默认扩展名 .cpp 建立文件。

(3) 在 Location 文本框中选择预先建立的文件夹,如 D:\PROGRAMC,然后选择 C++ Source File 选项,单击“OK”按钮,在该文件夹中就建立了新文件 test1.c。

新建文件后,进入 VC++ 开发平台,如图 1.2 所示,可以看出开发平台界面中的上部为菜单和调试工具栏,中部为编辑窗口,下部为信息窗口。还可以使用直接打开文件的方法,建立新的 C 文件,单击窗口左上角的 New text file 按钮,然后选择 File 菜单中的 Save As 命令,手动输入带有扩展名 .c 的完整文件名,这样即将文件保存为标准 C 的文件格式,随后即可进行文件的编辑、编译及链接等操作。

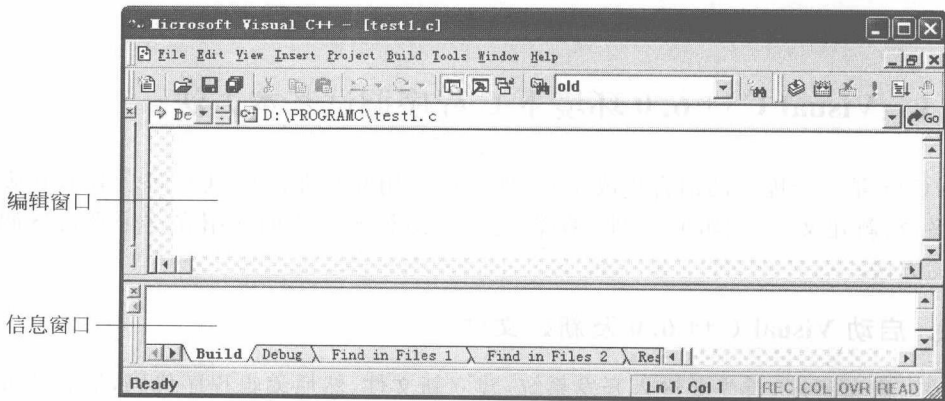


图 1.2 Visual C++ 界面

## 2. 编辑和保存 C 源文件

如图 1.3 所示为 File 菜单的命令集。在 Windows 环境下的 VC++ 编辑窗口中编辑 C 源程序,可通过鼠标和菜单输入和修改代码,十分方便。当程序编辑完成后要保存文件时,可选择 File→Save 命令将其保存为先前建立的源文件;通常该文件所保存的位置称为当前路径(文件夹),也可以按 Ctrl+S 组合键直接保存当前编辑好的 C 文件 test1.c;还可以选择菜单中的 Save As 命令,将当前编辑的 C 文件保存到其他文件夹中。

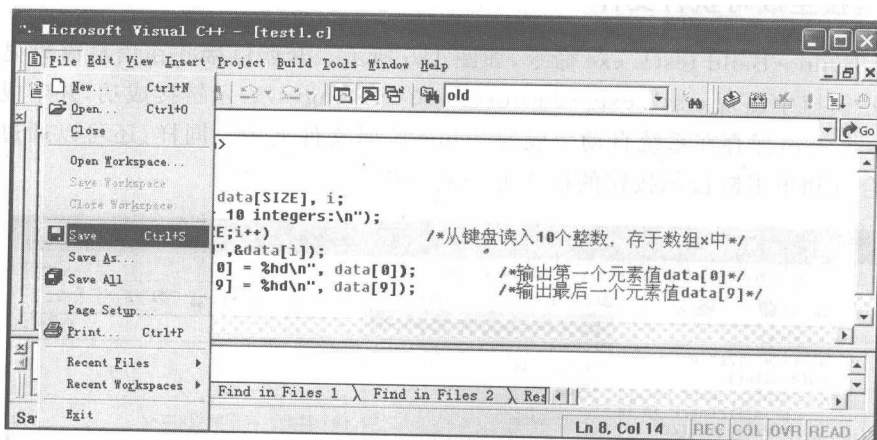


图 1.3 File 菜单

## 3. 编译 C 源文件生成目标文件

文件保存完成后,选择 Build→Compile test1.c 命令,编译此程序,如图 1.4 所示。第一次编译时会弹出一个提示框,单击 Yes 按钮,开始编译,并在信息窗口显示编译信息。

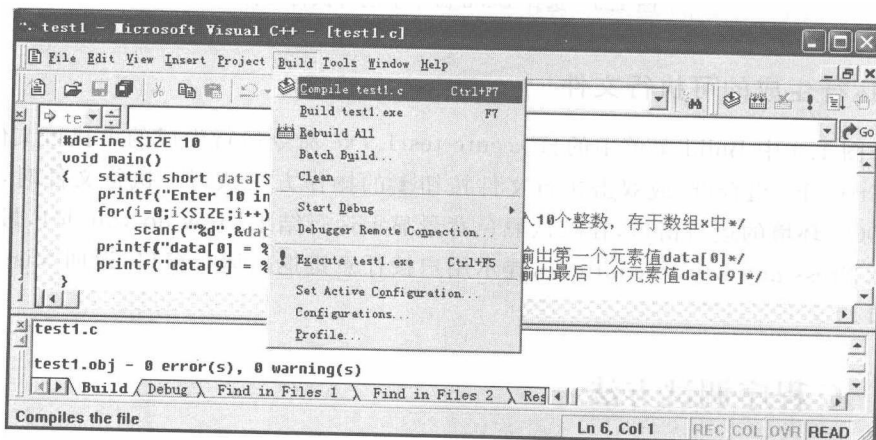


图 1.4 选择编译生成目标文件

信息窗口中出现 test1.obj - 0 error(s), 0 warning(s),表示编译正确,没有发现错

误和警告,并生成目标文件 test1. obj,此文件自动保存在系统建立的“debug”文件夹中。还可以通过按 Ctrl+F7 组合键或双击  按钮的操作方法编译文件。

如果在编译过程中,信息窗口出现编译错误信息,说明程序中存在语法或词法错误,要返回编辑窗口修改程序;如果出现警告信息,说明这些错误并未影响目标文件的生成,但最好查明原因,修改错误,因为有时忽略警告错误,也可以执行程序,但可能运行结果有错误。

#### 4. 链接生成可执行文件

选择 Build→Build test1. exe 命令,如图 1.5 所示。开始链接并在信息窗口显示链接信息,信息窗口中出现 test1. exe—0 error(s), 0 warning(s)表示链接成功,并生成了可执行文件 test. exe,保存在系统自动生成的 debug 临时文件夹中。同样,还可以通过按 Ctrl +F7 组合键和单击链接  按钮的操作方法链接调试。

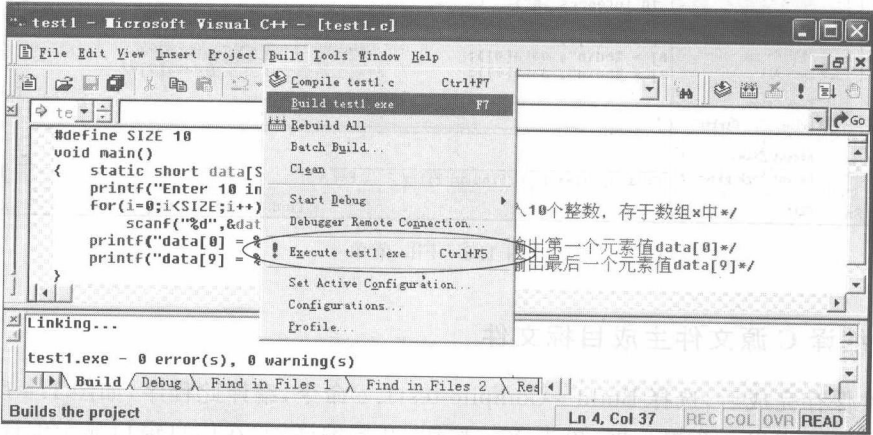


图 1.5 选择 Build 命令生成可执行文件

#### 5. 运行生成的可执行文件

选择图 1.4 中 Build 菜单中的!Execute test1. exe 命令运行生成的可执行文件,也可通过按 Ctrl+F5 组合键,或双击执行文件按钮  的操作方法执行。执行文件时;系统自动弹出 DOS 环境的运行窗口,在默认黑色背景显示运行结果 How do you do!,最后一行总是显示 Press any key to continue,提示用户按任意键退出运行窗口,返回 Visual C++ 编辑窗口。

### 1.1.2 C 程序调试方法

**例 1.1** 改正下列程序中的错误,求温度 100°F 对应的摄氏温度。计算公式如下,其中: c 表示摄氏温度, f 表示华氏温度,程序的源文件名为 error1. c。

$$c = \frac{5 \times (f - 32)}{9}$$

源程序(有错误的程序):

```
(1) /* .....error1.c..... */
(2) #include <stdoi.h>
(3) void main()
(4) {   float fahr;
(5)     float celsius;
(6)     fahr=100;
(7)     Celsius=5 * (fahr- 32)/9;
(8)     printf("fahr=f, celsius=%f\n", fahr ,celsius);
(9) }
```

程序运行结果(改改后程序的运行结果):

**fahr =100.000000, celsius =37.777778**

(1) 按照上一小节介绍的步骤,打开源程序 error1.c。

(2) 在上一小节中,使用菜单命令完成编译、链接和运行操作,现在介绍使用工具栏完成上述操作的方法。在工具栏或菜单栏上单击鼠标右键,弹出如图 1.6 所示的工具菜单,选择 Build MiniBar 命令,调出调试工具栏,如图 1.7 所示。调试工具栏中,按钮①表示编译(Compile),按钮②表示构建(Build),按钮③表示运行(Build Execute),按钮④表示断点跳转(Go);按钮⑤表示设置/取消断点(Insert/Remove Breakpoint)。

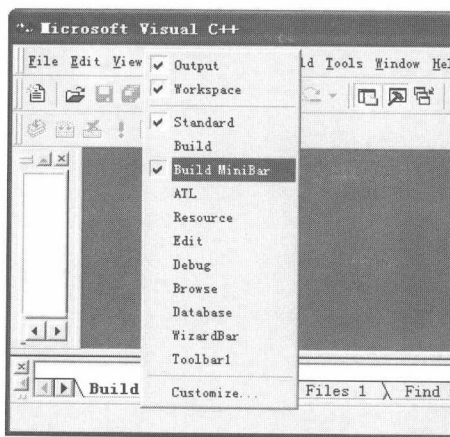


图 1.6 弹出的工具菜单

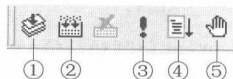


图 1.7 调试工具栏

(3) 单击 Compile 按钮①进行程序编译,编译时在信息窗口出现一条错误(Error)信息:

Cannot open include file: 'stdoi.h': No such file or directory

双击该错误信息,箭头指向源程序的第 2 行,错误信息指出不能打开库头文件 stdoi.h,

仔细观察可以发现,程序前面包含的头文件 `stdio.h` 误写为 `stdoi.h`。改正后重新编译,新产生的错误信息:

```
'Celsius' : undeclared identifier
```

双击信息窗口的错误信息,箭头指向源程序的第 7 行,错误信息指出 `Celsius` 变量没有被定义。返回程序检查,该变量已经定义了,再仔细观察后,发现变量引用处的 `celsius` 误写为 `Celsius`,即第一个字母大小写不一致,将第 7 行的第一个字母改为小写。重新编译,编译正确。

(4) 单击 Build 按钮 ② 进行程序链接。程序链接成功,并生成可执行文件 `error1.exe`。

(5) 单击 Build Execute 按钮 ③,执行程序。程序运行结果为

```
fahr =f, celsius =100.000000
```

运行结果不符合题目的要求,仔细检查源程序,发现函数 `printf()` 中 `fahr=f` 的 `f` 应为 `%f`。改正后,重新编译、链接和运行,运行结果与题目要求的结果一致。到此程序调试结束。

**例 1.2** 改正下列程序中的错误,输入  $x$ , 计算并输出下列分段函数  $f(x)$  的值(保留 2 位小数),源程序文件名称为 `error2.c`。

$$y = f(x) = \begin{cases} x & x \leq 1 \\ 1 & x > 1 \end{cases}$$

源程序(有错误的程序):

```
/* .....error2.c..... */
#include <stdio.h>
void main()
{ double x,y;
  printf("Enter x:");
  scanf("%lf",&x);
  if (x<=1)
    y=x
  else
    y=1;
  printf("f(%.2f)=%.2f\n", x,y);
}
```

改正后程序运行结果 1:

```
Enter x:0.5
f<0.50>=0.50
```

程序运行结果 2:

```
Enter x:3
f<3.00>=1.00
```

(1) 编译源文件 `error2.c`。

首先,打开 C 源程序文件 `error2.c`,单击“编译”按钮,查看信息窗口,可发现第一条编

译错误信息为 missing ';' before '}'。

双击该错误信息,箭头指向第一个“}”所在的行,错误信息指出在“}”前缺少分号。在“y=x”后面添加分号后,重新编译,新出现错误信息 local variable 'x' used without having been initialized。

双击该错误信息,箭头指向 scanf("%lf", x) 所在的行,可以看出错误原因是 x 前面缺少 &。在 x 前面添加 & 后,重新编译并连接,直到获得正确结果。

(2) 为了方便调试,从 Tools 菜单中调出 Debug 调试工具栏。

选择 Tools→Customize 命令,在如图 1.8 所示的 Customize 对话框的 Toolbars 选项卡中选中调试工具的 Debug 选项,打开如图 1.9 所示 Debug 调试工具栏。

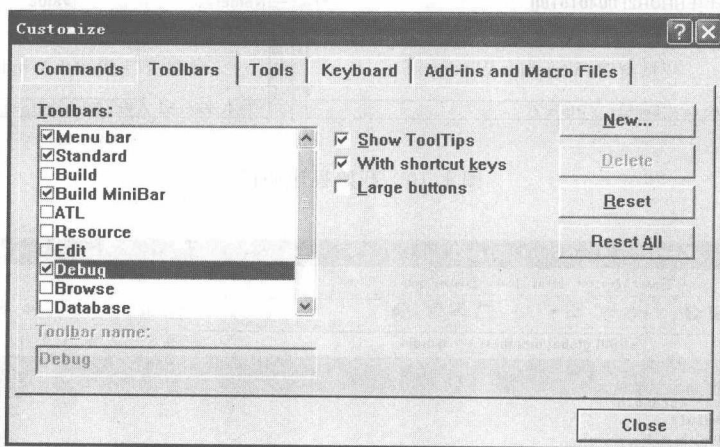


图 1.8 选择调试工具 Debug

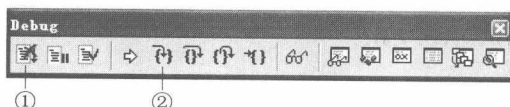


图 1.9 Debug 调试工具栏

在图 1.9 中,按钮①表示停止调试(Stop Debugging),按钮②表示单步调试(Step Over),单击该按钮则执行一条语句。调试程序时,有时为了跟踪程序的某个变量值或某个表达式计算结果,观察其值变化是否符合要求,即可使用单步调试。

(3) 程序编辑完成后,即可开始调试。单击“单步调试”(Step Over)按钮,每次执行一行语句,编辑窗口中的箭头即指向某一行,如图 1.10 所示;表示程序将要执行该行语句。图 1.9 中列出了变量窗口(Variables Windows)和观察窗口(Watch Windows),在观察窗口中可以改变变量的值。

(4) 单步执行观察变量 x。单击“单步调试”(Step Over)按钮两次,接着执行程序到输入函数 scanf("%lf",&x) 这一行,箭头位置如图 1.11 所示。在 DOS 界面中查看程序运行结果如图 1.12(左)所示,显示输入提示信息 Enter x:,此时将要执行输入函数 scanf()。继续单击“单步调试”(Step Over)按钮,在运行界面等待输入的光标处输入“0.5”,如图 1.12(右)所示。

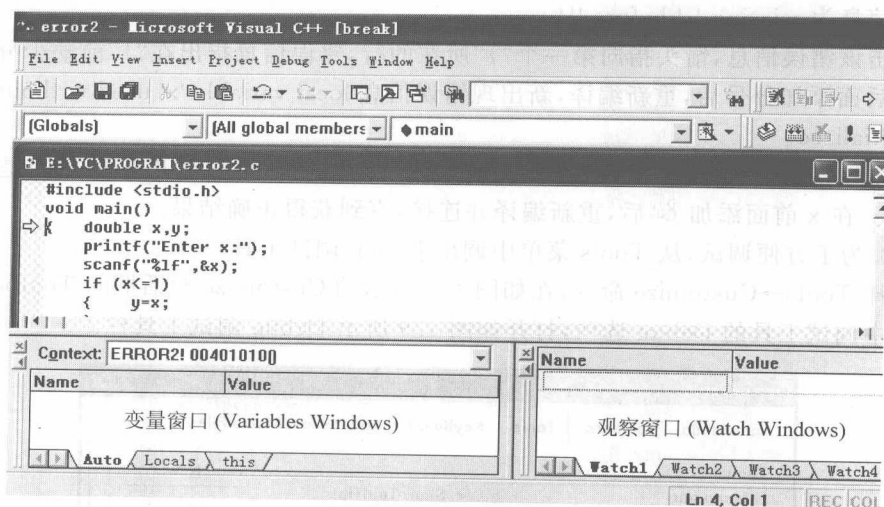


图 1.10 开始程序调试

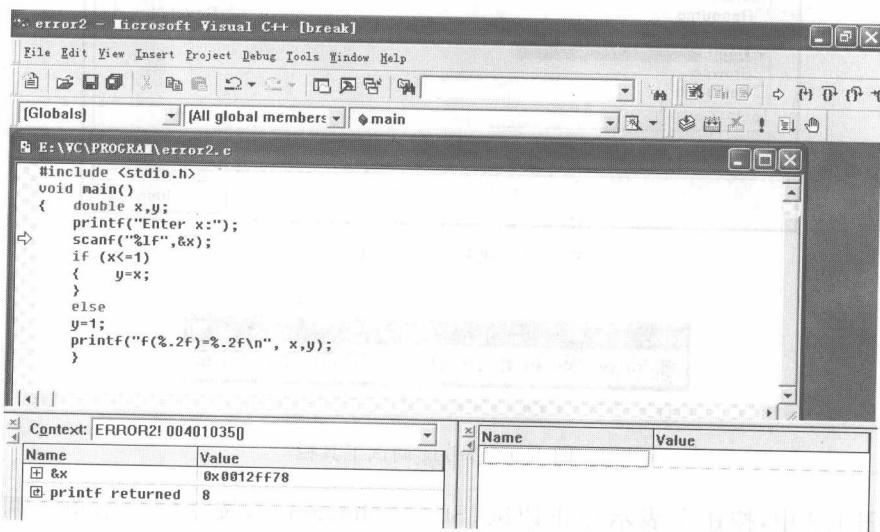


图 1.11 程序单步调试

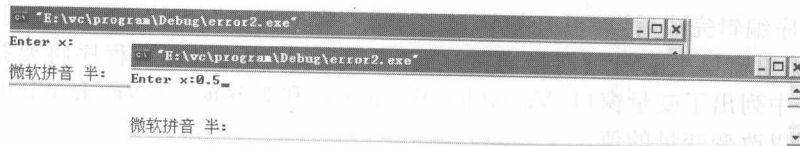


图 1.12 运行结果(左)及在运行窗口中输入变量 x 的值(右)

如果按 Enter 键,程序回到单步执行状态,箭头指向 if (x<=1)这一行,在如图 1.13 所示的变量窗口中看到此时变量 x 的值为 0.5000000000000000,说明当执行了输入函数 scanf()之后,输入的数据 0.5 保存到变量 x 中。因为使用 long 型控制字符"%lf",所以

显示浮点数,小数位格式超过了标准 C 规定的 6 位小数。

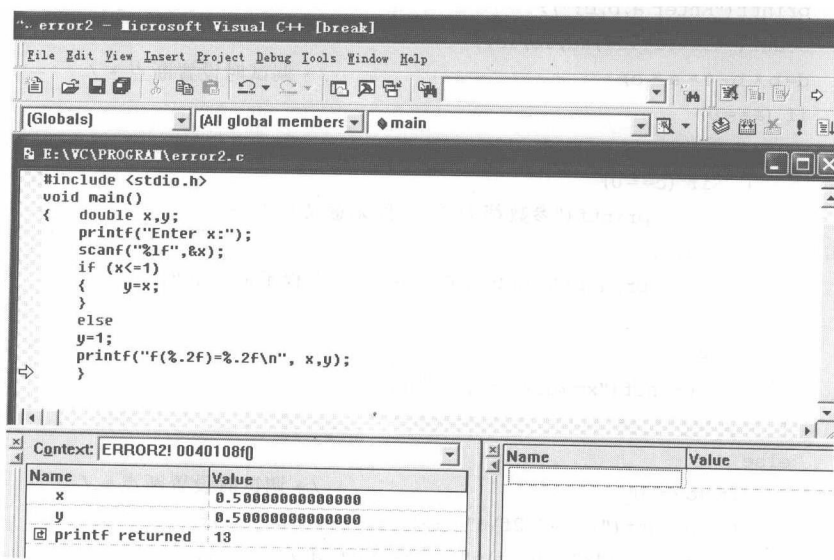


图 1.13 单步调试查看变量窗口

(5) 继续执行程序观察变量  $y$ 。单击“单步调试”(Step Over)按钮两次,箭头指向 else 语句这一行,执行后在图 1.13 的变量窗口中可观察到变量  $x$  和  $y$  的值均为 0.5000000000000000。

(6) 继续单击“单步调试”(Step Over)按钮两次,当执行最后的输出函数 `printf("f(%.2f)=%.2f\n", x,y);` 时,运行窗口显示的运行结果如图 1.14 所示,符合题目的要求。

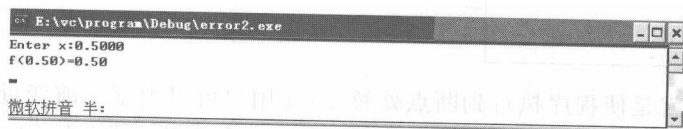


图 1.14 显示正确结果

当程序执行到某个位置时发现运行结果不正确,说明在此之前肯定有错误存在。如果确定某一小段程序可能有错,可按照上述步骤暂停在该段程序的第一行,再输入若干个查看变量,然后单步执行,即一次执行一行语句,逐行检查,查看到底是哪一行出现错误,从而能够确定错误语句并进行修改。

**例 1.3** 修改下列程序中的错误,输入参数  $a, b, c$ , 求一元二次方程  $ax^2 + bx + c = 0$  的根,源程序名称为 error3.c。

- (1) /\* ..... error3.c ..... \*/
- (2) #include <stdio.h>
- (3) #include <math.h>
- (4) void main()

```

(5) { double a,b,c,d;
(6)     printf("Enter a,b,c:");
(7)     scanf("%lf%lf%lf",&a,&b,&c);
(8)     d=b*b-4*a*c;                                /* 调试时设置断点 */
(9)     if (a==0)
(10)    { if (b==0)
(11)        { if (c==0)
(12)            printf("参数都为零,方程无意义!\n");
(13)        else
(14)            printf("a 和 b 为 0,c 不为 0,方程不成立\n");
(15)        }
(16)    else
(17)        printf("x=%0.2f\n",-c/b);
(18)    }
(19) else
(20)     if (d>=0)                                    /* 调试时设置断点 */
(21)     { printf("x1=%0.2f\n", (-b+sqrt(d)/(2*a)));
(22)       printf("x2=%0.2f\n", (-b-sqrt(d)/(2*a)));
(23)     }
(24) else
(25)     { printf("x1=%0.2f+%0.2fi\n", -b/(2*a),sqrt(d)/(2*a));
(26)       printf("x2=%0.2f-%0.2fi\n", -b/(2*a),sqrt(d)/(2*a));
(27)     }
(28) }                                                /* 调试时设置断点 */

```

程序运行结果(修改后程序的运行结果):

```

Enter a,b,c:2.2 8.8 3.3
x1=-0.42
x2=-3.58

```

断点的作用就是使程序执行到断点处暂停,使用户可以观察当前变量或表达式的值。设置断点时,先将光标定位到要设置断点的位置,然后单击调试工具栏上的 Insert/Remove Breakpoint 按钮,断点即设置完毕。

如果要取消已经设置的断点,只需将光标移动到要取消的断点处,单击 Insert/Remove Breakpoint 按钮,取消断点。如果想全部取消多个断点,可选择 Edit 菜单中的 Breakpoints 命令,打开 Breakpoints 窗口,窗口下方列出了所有断点,单击 Remove All 按钮将取消所有断点。断点通常用于调试较长的程序,可以避免使用 Run to Cursor 命令,经常要将光标定位到不同的位置。对于长度为上百行的程序,要寻找某位置并不太方便。

如果一个程序设置了多个断点,单击调试工具栏上的“断点跳转”(Go)按钮会暂停在第一个断点,再次单击“断点跳转”(Go)按钮会继续执行到第二个断点处暂停,这样依次执行下去。

打开源程序 error3.c,对程序进行编译,出现 3 个编译错误。第 20 行和第 21 行代码括号不匹配,各缺少一个右括号“)”;程序中花括号不匹配,可在第 25 行与第 26 行之间添