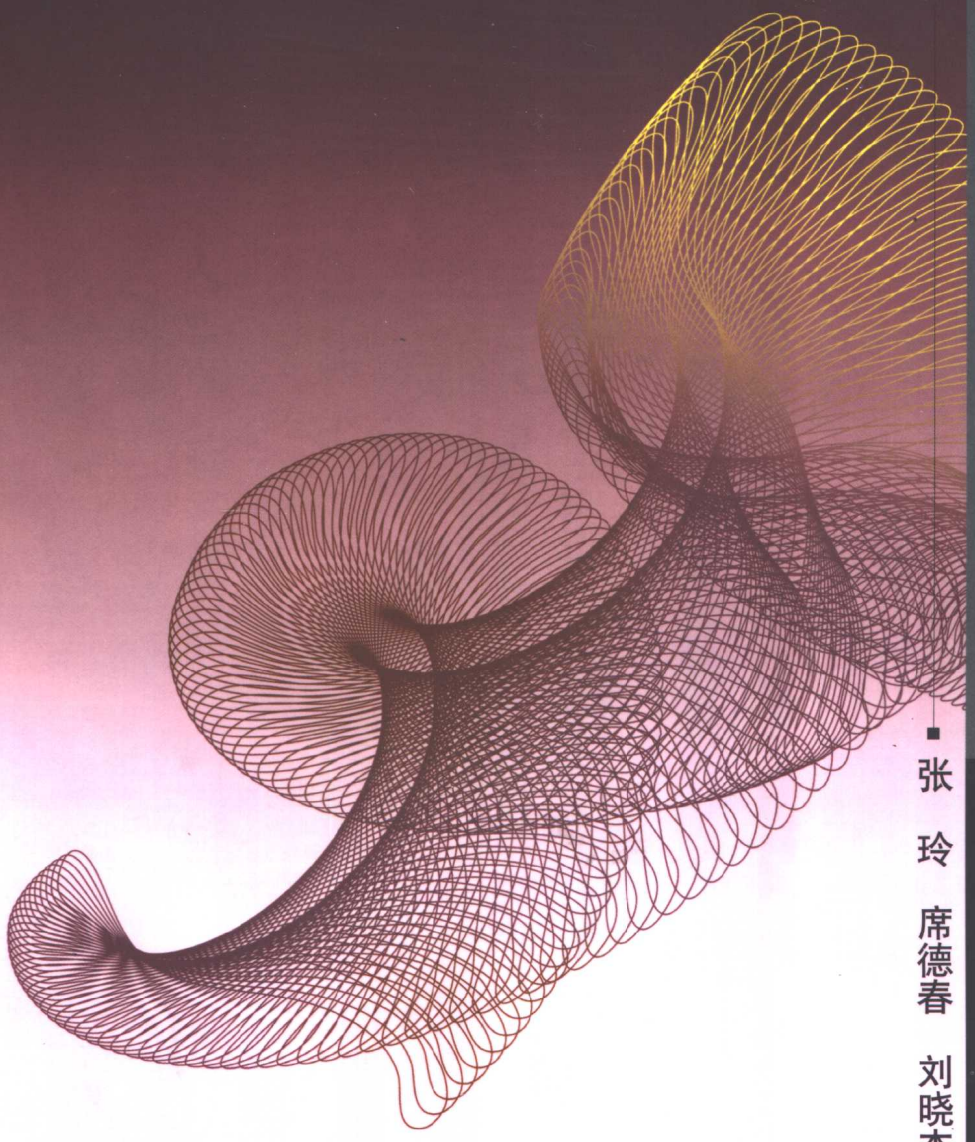
计算机实践指导系列教材

C++上机实践指导教程

张 玲 席德春 刘晓杰 等编著



机械工业出版社
CHINA MACHINE PRESS

计算机实践指导系列教材

C++上机实践指导教程

张　玲　席德春　刘晓杰　等编著

机械工业出版社

本书以目前高校普遍使用的C++教材为背景，通过一些经典的上机实例，使学生在掌握C++知识的同时提高上机操作能力。本书每章有多个上机实例，每个实例先给出其运行结果，然后介绍为实现该实例所涉及的基础知识和程序。最后，重点分析该实例的程序和在上机操作时容易出现的问题及注意事项。本书每章后都有复习题及解答、上机练习题，以加深对各章知识的理解与掌握。

本书可作为大专院校C++课程的上机实践指导书，也可作为有一定C++基础知识的读者的自学指导参考书。

图书在版编目（CIP）数据

C++上机实践指导教程/张玲等编著. —北京：机械工业出版社，2004.1
（计算机实践指导系列教材）
ISBN 7-111-13275-0

Ⅰ.C... Ⅱ.张... Ⅲ.C语言－程序设计－教材 Ⅳ.TP312

中国版本图书馆CIP数据核字（2003）第097059号

机械工业出版社（北京市百万庄大街22号 邮政编码 100037）
策　　划：胡毓坚
责任编辑：王　颖
责任印制：施　红

北京忠信诚胶印厂印刷·新华书店北京发行所发行

2004年1月第1版·第1次印刷
787mm×1092mm 1/16·14.75印张·360千字
0001—5000册
定价：21.00元

前　言

目前各理工类高校计算机公共课大多开设了 C++ 语言程序设计课程。该课程的教学和上机时间比例基本为 1:1。市场上有关 C++ 语言程序设计的教材很多，但如何指导学生上机操作的书籍却很少。学生往往在上机时不知干什么，或者在上机操作遇到问题时在教科书中找不到解决办法。如何指导学生上机实践操作，减轻教师工作量，是急需解决的问题。

根据目前的状况，我们编写了这本《C++ 上机实践指导教程》。本书以目前市场上流行的 C++ 语言程序设计教材为基础编写上机操作实例，每个实例的选择与教材的相关内容配合，目的是在上机操作的同时消化和理解教材的内容。在讲解每个实例时，首先让学生看到该实例上机后的结果，并介绍该实例的目的是让学生掌握哪些内容，然后介绍为实现该实例需要编写的程序。最主要的是，针对学生在上机操作时经常出现的问题以及操作错误，在每个实例中都给予说明。这样既可提高学生上机的效率，又可减轻教师答疑的负担。另外，在每一章中对有关的知识要点给予归纳总结。

另外，本书还安排了有助于对 C++ 知识进一步理解的复习题部分，并配有答案。

本书所使用的实例，经过反复挑选，既有利于学生掌握有关知识，又不失趣味性，在提高学生学习兴趣的同时，让学生学到 C++ 语言知识，相信本书一定会受到广大师生的欢迎。

本书主要由张玲、席德春、刘晓杰编写，另外参加编写工作的还有孟桃平、李晓梅、孙琪、张文忠、孟传、刘方义、高宁等。

由于作者水平有限，本书存在的不足之处，恳请读者提出宝贵意见和建议。

编　者

目　录

第1章 预备知识

C++语言源于C语言，它在C语言的基础上加入了面向对象的特性。C++不仅保持了C语言简洁、高效的特点，而且在模块化结构的基础上增加了面向对象程序设计的支持。它既可以做面向过程的模块化程序设计，又能支持面向对象程序设计的混合。

在这一章中，读者将练习在Visual C++环境下编辑、编译、构建和运行简单的C++程序。通过简单程序的编辑、运行，熟悉Visual C++的编辑环境。通过程序的调试，熟悉设置断点调试程序和逐行调试的操作，并且在程序运行过程中观察各变量值的变化。

实践 1　Visual C++6.0 窗口的基本操作

实践目的

本实践将介绍 Visual C++6.0 的操作环境，介绍如何在此环境下进行 C++ 源代码编辑、编译和运行等操作。

Visual C++6.0 介绍

程序语言的编写需要一个编辑环境，市面上有相当多的 C++ 编辑工具，本书中的程序都是在 Microsoft 公司出版的 Visual C++6.0 上编写、运行的。Visual C++6.0 是一套可视化的 C++ 程序编辑软件，它的编辑环境与一般的编辑软件相似，利用了简明易懂的可视化分割方式，将环境分成数个局部，包含一个简单的编辑区、一个功能齐全的工具栏，还有一个编辑功能菜单。

使用 Visual C++6.0 之前，必须将 Visual C++6.0 系统装到用户的计算机上。用户可以制定其安装目录，也可把 Visual C++6.0 创建成桌面快捷方式，方便使用。

进入 Visual C++6.0

启动 Visual C++ 的方法，可以选择单击屏幕左下角的【开始】按钮，出现菜单后，选择【程序】，接着选择【Microsoft Visual Studio6.0】，再选择【Microsoft Visual C++6.0】，就能启动 Visual C++。如果已经把 Visual C++ 创建成桌面快捷方式，则每次只要双击快捷方式的图标即可进入 Visual C++ 系统。

Visual C++6.0 的工作窗口

启动 Visual C++ 以后，会出现一个“Tip of the Day”的提示框，单击【Close】按钮后，就会显示如图 1-1 所示的 Visual C++ 窗口。

Visual C++ 窗口包括以下内容。

① 菜单栏：包括 File(文件)、Edit(编辑)、View(视图)、Insert(插入)、Project(项目)、Build(建构)、Tools(工具)、Window(窗口)和 Help(帮助)主菜单，每一个主菜单下还有相应的子菜单，通过菜单我们可以完成相应的操作。

② 标准工具栏：各项的功能与 Microsoft Word 的标准工具栏基本相同。

③ 向导栏：可列出程序中含有的类以及生成类向导。

④ 建立程序工具栏：从左至右依次包括 Compile(编译程序)、Build(建立可执行文件)、

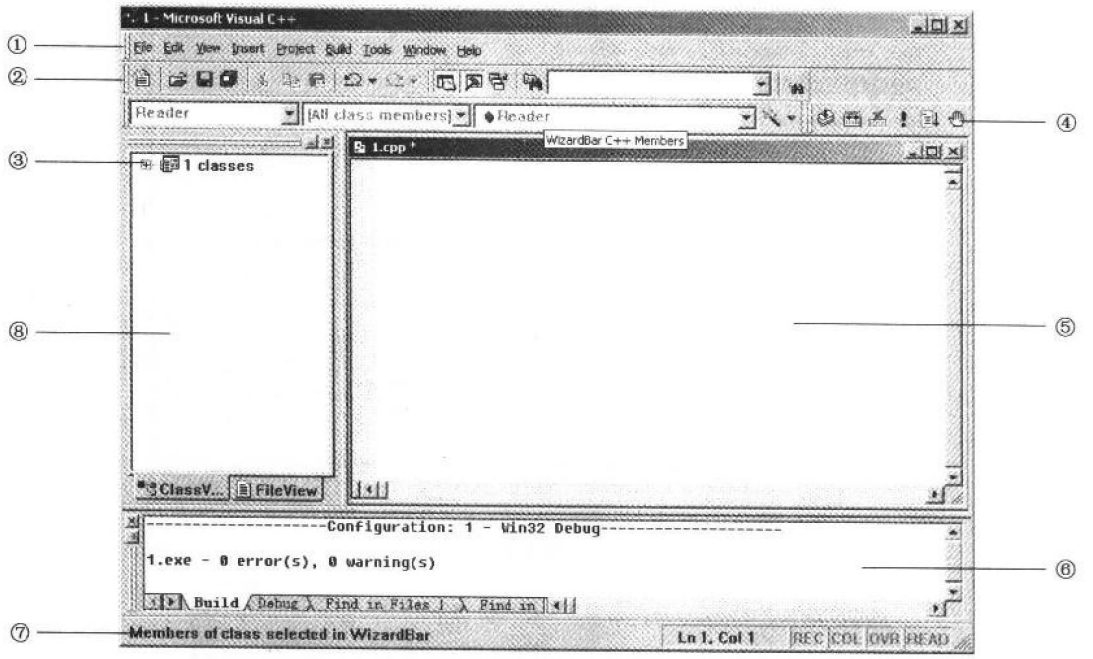


图 1-1

Stop Build(停止建立可执行文件)、Execute Program(执行程序)、Go(执行程序到下一个程序断点)、Insert/Remove Breakpoint(插入或删除程序的断点)等功能快捷键。

⑤ 程序编辑区:编写程序的区域。

⑥ 输出窗口:在此窗口显示程序编译期间所产生的输出信息,如 error(错误)、warning(警告)信息等。

⑦ 消息显示区:显示编辑程序或执行功能时的信息。

⑧ Workspace(工作空间)窗口:工作空间窗口底端显示两个标签,即 Class View 和 File View。

Visual C++ 6.0 的使用

1. 编辑一个新文件

第一步,用 AppWizard 创建控制台应用程序。

1) 执行【File】>【New】菜单命令,显示出"New"对话框。选择其中的【Projects】标签,并从列表框中选中【Win32 Console Application】项。在 Project Name 框中键入控制台应用程序的项目名称,此项目我们取名为 Game。之后,用户可以在 Location 下的编辑框中键入此项目所在的文件夹,或者单击【Browser】按钮选择已有的文件夹,如图 1-2 所示。

2) 单击【OK】键继续。接下来将会显示一个询问项目类型的应用向导,选中其中的 A simple application 项。如图 1-3 所示。

3) 单击【Finish】按钮,将出现"New Project Information"对话框,如图 1-4 所示,单击【OK】按钮,系统将自动创建此应用程序。

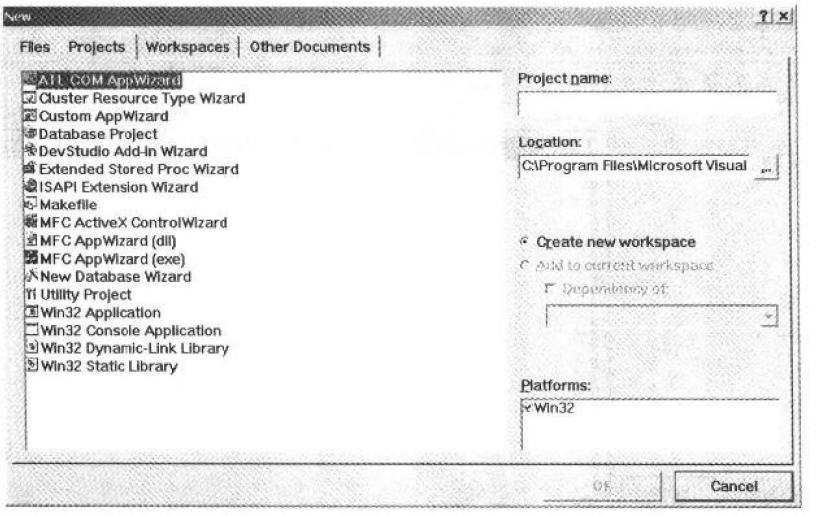


图 1-2

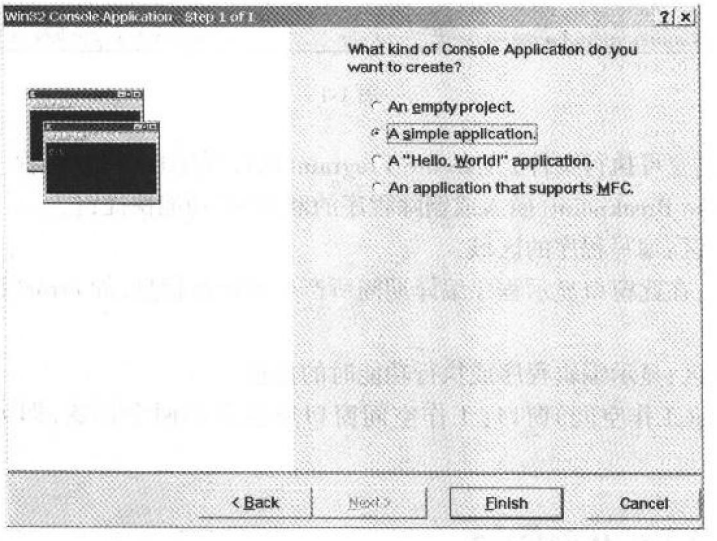


图 1-3

第二步，打开 Game.cpp 文件。

通过上述过程，创建好了一个控制台应用程序 Game 的程序框架。在项目工作区窗口中选择 FileView 标签项，看到 AppWizard 生成了 Game.cpp、StdAfx.cpp、StdAfx.h 以及 ReadMe.txt 四个文件。如图 1-5 所示。其中 Game.cpp 是 AppWizard 产生的程序源代码文件，用户几乎所有的代码都是添加在这个文件中的。

第三步，添加程序代码。

上述源代码只是一个框架，用户必须添加一些代码，以实现自己需要的功能，代码的添加过程如下所示。

1）将项目工作区窗口切换到 ClassView 页面，显示 Game 类的信息。单击各目录项前面的“+”，将所有的目录项打开。双击 main 函数名，在文档窗口将显示出 main 函数体所在的

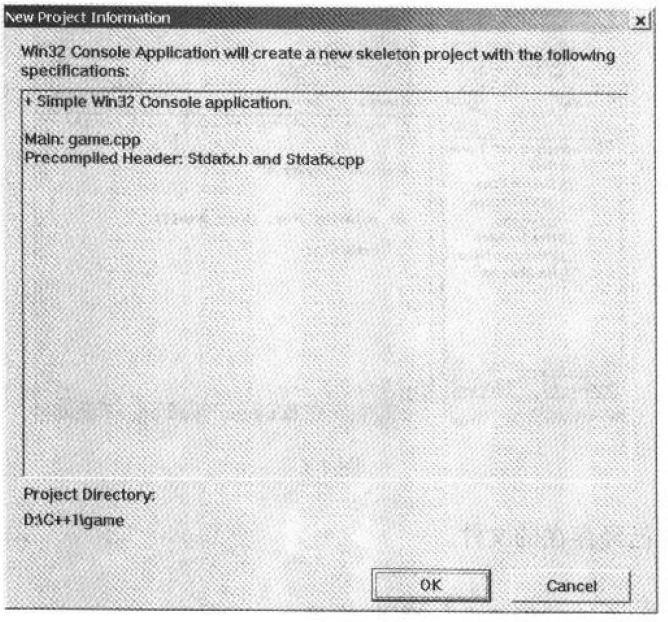


图 1-4

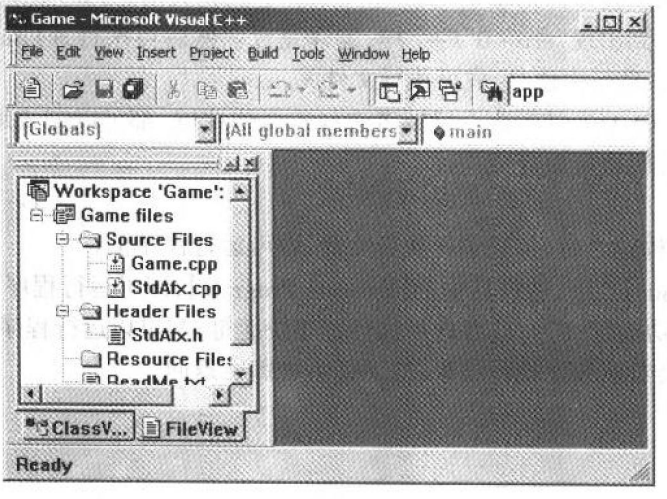


图 1-5

Game.cpp 源文件。

2）将光标移至此函数名前，键入代码如图 1-6 所示。

2. 保存文件

利用 File 菜单的【Save All】命令项或标准工具栏的【Save All】按钮将所有文件保存。

3. 打开一个已存在的文件

1）打开【File】菜单，单击选择其中的【Open】菜单，选择搜索路径，选择要打开的文件后，单击【Open】按钮即可打开所选中的可执行文件，可对此文件进行编译、运行。

2）此外，按下〈Ctrl〉+〈O〉组合键，或者单击打开文件的快捷键，重复选择搜索路径等操

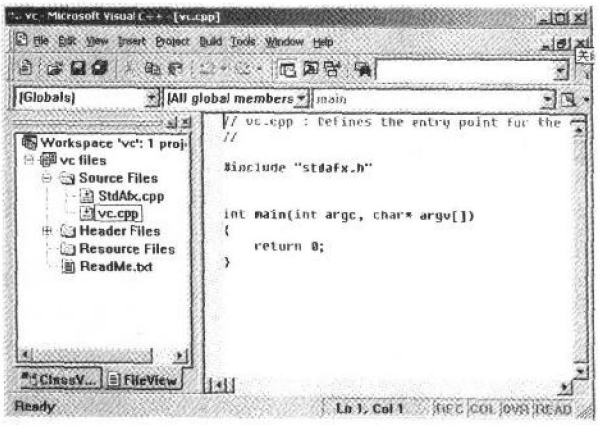


图 1-6

作,也可以打开一个已经存在的文件。

4. 编译和构建

编辑好源程序后,应该对程序进行编译、构建。

1) 打开【Build】菜单,选择其中的【Compile】编译器即可对程序进行编译,生成了 * .obj 文件。编译之后,在输出窗口即可输出 error 和 warning 的相关信息,如果出现错误提示信息,修改程序重新编译,直到出现"error 0"的字样,编译完成。然后再次打开【Build】菜单,选择其中的【Compile】,对程序进行构建,生成可执行文件 * .exe。

2) 此外,按下〈Ctrl〉+〈F7〉组合键或者选择编译快捷键,都可以完成编译。按下〈F7〉键,或者选择构建快捷键,也可以完成构建。

5. 运行

执行过编译和构建之后,生成了可执行文件,即可运行程序了。

1) 打开【Compile】菜单,选择其中的【Execute Program】,即可运行程序。

2) 此外,按下功能键〈F7〉,或者选择执行程序快捷键,都可以运行程序。

程序执行后,会看到程序执行结果,按下任意键即可返回。

实践2 第一个C++程序

——输出一个字符串“I like this game”

实践结果

本实例将编写一个简单的C++程序，在屏幕上输出一个字符串，如图1-7所示。

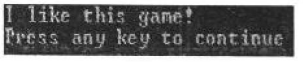


图1-7

实践目的

本实践将进一步了解开发环境的使用过程，向读者介绍C++程序的基本结构。

C++程序结构可以分为以下三个部分。

1）函数声明区。在程序中如果有独立的函数，必须在此声明函数的类型；如果使用函数库中的函数，必须使用#include指令将函数声明文件包含进来。

2）主程序区。使用main()开头，并用{}将主程序代码包含进来。因为程序是由许多函数组成的，包括主程序main()也是一个函数。

3）函数区。在主程序中，除了可以使用C++语言提供的函数库外，也可以使用在函数区自定义的函数。函数是指可以完成特定功能的独立运行程序，它是许多程序语句的集合。一个C++程序的内容主要是由一个或多个函数组成的。

程序源代码

```
#include"stdafx.h"
#include"iostream.h"                    //函数声明区

int main(int argc, char* argv[])        //主程序区
{
    cout<<"I like this game! \n";       //将此字符串输出到屏幕上

    return 0;                           //函数无返回值
}
```

程序分析与注意事项

1）在函数声明区的#include"iostream.h"是输入输出操作的头文件。此程序中没有独立

的函数,因此无需在此声明函数的类型。

2) 主程序中包含一条语句:cout << "I like this game! \ n";是将"I like this game!"显示在屏幕上,cout 是 C++ 特有的输出函数。

3) 在本例中,我们是通过 Visual Studio 内建的应用程序向导,自动产生一个应用程序。在主程序中加人自己的程序语句即可完成。

实践 3　程序动态调试方法

——VC 界面的 Debug 功能

实践目的

在本实践中，我们主要介绍调试器与断点的应用，介绍如何利用设置断点调试程序。当程序执行的结果未能达到预期的运算结果时，表示程序语句中有错误发生，调试器可以让用户检查程序的运行，并找出问题所在。如果设置了断点，调试器就可以允许在程序的执行过程中，检查程序中的数据数值的变化。

程序源代码

```
#include "stdafx.h"
#include "iostream.h"

int main(int argc, char* argv[])
{
    long value1 = 100;
    long value2 = 60;
    long temp = 0;
    long *valueptr = NULL;

    valueptr = &value1;
    *valueptr += 50;
    temp = *valueptr;

    cout << "&value1:" << valueptr << endl;
    cout << "value1 + 50 = " << temp << endl;
    cout << "****************************" << endl;

    valueptr = &value2;
    temp = *valueptr * 10;

    cout << "&value2:" << valueptr << endl;
    cout << "value2 * 10 = " << temp << endl;
    cout << endl;
```

```
    return 0;
}
```

1. 设置断点调试程序

断点是调试器在执行程序时会自动停止执行的位置。一般我们在认为程序会出错的地方设置断点，让程序暂停在第一个断点的位置，然后逐行执行到第二个断点，如此通过一个断点到下一个断点的跳跃检查方式来找出程序中的错误。

在本例中，我们将设置两个断点，以便检查 valueptr 指针以及 temp 值的变化。将光标位置移至要设置断点的位置，然后单击【Insert/Remove Breakpoint】按钮建立断点。如图 1-8 所示。除了利用菜单栏的按钮设置断点外，还有建立断点的方法：将光标移至要设置断点的程序位置，单击鼠标右键打开快捷菜单，执行【Insert/Remove Breakpoint】功能。

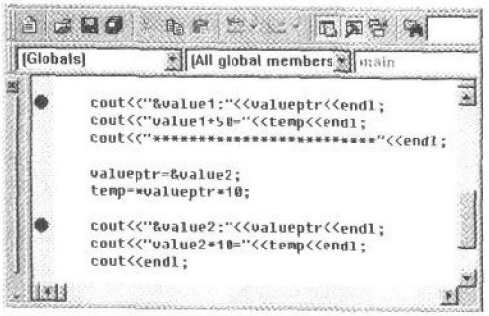


图 1-8

完成了断点的设置后，就可以进行调试。执行【Build-Start Debug】功能，开始【Debug】菜单，执行【Go】功能，直接执行程序，直到第一个断点的位置停止。利用这个功能可以让程序从一个断点执行至下一个断点。执行完【Go】功能后，系统就会显示三个默认窗口，第一个是程序源代码窗口；第二个是左下角的 Variable 窗口；第三个是在程序右下角的 Watch 窗口。如图 1-9 所示。

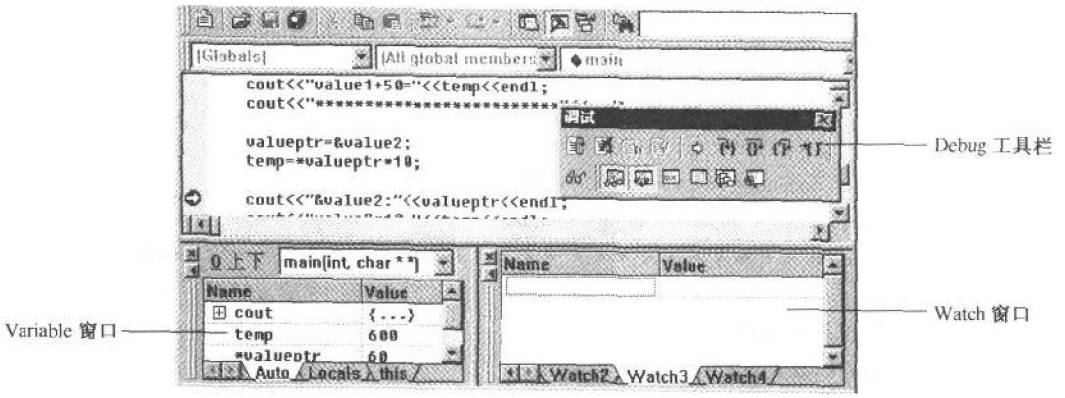


图 1-9

在 Variable 窗口会显示目前断点位置前的程序部分的变量值，也就是由目前断点的语句所派生的变量工程。如图 1-10 所示。