

高等院校计算机系列教材

C++程序设计教程

主 编 刘 宏
副主编 邱建雄 谢中科



WUHAN UNIVERSITY PRESS
武汉大学出版社

高等院校计算机系列教材

C++程序设计教程

主 编 刘 宏

副主编 邱建雄 谢中科

编 委 戴经国 张历卓 张小梅



WUHAN UNIVERSITY PRESS
武汉大学出版社

图书在版编目(CIP)数据

C++程序设计教程/刘宏主编;邱建雄,谢中科副主编. —武汉:武汉大学出版社,2005.8

(高等院校计算机系列教材)

ISBN 7-307-04583-4

I. C… II. ①刘… ②邱… ③谢… III. C语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2005)第 059717 号

责任编辑:杨华 黄金文 责任校对:刘欣 版式设计:支笛

出版发行:武汉大学出版社 (430072 武昌 珞珈山)

(电子邮件:wdp4@whu.edu.cn 网址:www.wdp.com.cn)

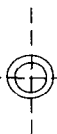
印刷:湖北省通山县九宫印务有限公司

开本:787×980 1/16 印张:16.125 字数:255千字

版次:2005年8月第1版 2005年8月第1次印刷

ISBN 7-307-04583-4/TP·163 定价:26.00元

版权所有,不得翻印;凡购买我社的图书,如有缺页、倒页、脱页等质量问题,请与当地图书销售部门联系调换。



前

言

针对将 C 与 C++ 截然分开的传统教学模式,基于目前新入学本科生的计算机基础普遍提高的现实,作者提出了将 C 与 C++ 结合起来进行系统讲述的新教学思路。为此,我们组织了多位长期从事程序设计、数据结构、面向对象技术教学的教师,依托丰富的教学经验,系统地讲述 C++(含 C)程序设计。这样有利于学生在低年级阶段尽快进入计算机专业,能够打下坚实的程序设计基础,更是提前培养了学生进行面向对象程序设计的思想与技能。这是一项有意义的教学改革尝试,为全省乃至全国的大专院校计算机专业的程序设计基础教学,率先探索出一条新路子。

C/C++ 代表了当前的计算机编程语言,无论是在通用计算机上的开发还是在嵌入式系统上的开发,都体现了它的明显优势,作为本科生学好 C/C++ 是有益的。尽管它相对其他编程语言来说有一定的难度,但通过努力学习,掌握 C/C++,那么学习其他编程语言时,就能融会贯通。

参加本教材编写工作的有:湖南农业大学张历卓(第一章、第二章),长沙大学邱建雄(第三章、第四章),长沙理工大学谢中科(第五章、第六章),湖南人文科技大学戴经国(第七章),湖南师范大学刘宏(第八章、第九章、第十章),黔东南民族高等师范专科学校张小梅(第十一章、第十二章)。本教材体现了作者多年的教学经验,经过多次讨论与修改才得以完成。

由于时间仓促,错误难免,恳请读者批评。

作者

2005.5

目 录

第一章 C++ 语言概述	1
1.1 C++ 语言简介	1
1.1.1 C++ 语言的发展	1
1.1.2 C++ 语言的特点	2
1.2 C++ 程序简介	2
1.3 C++ 程序的开发环境	4
1.3.1 Turbo C++ 介绍	4
1.3.2 Visual C++ 介绍	7
思考题一	10
实训一	10
第二章 C++ 语言编程基础	11
2.1 C++ 语言词法	11
2.1.1 注释	12
2.1.2 关键词	12
2.1.3 标识符	13
2.1.4 常量	13
2.1.5 变量	13
2.1.6 运算符	14
2.1.7 分隔符	14
2.2 基本数据类型	15
2.2.1 整型	16
2.2.2 浮点型	17
2.2.3 字符型	19
2.2.4 布尔型	20
2.2.5 类型转换	20
2.3 运算符与表达式	22
2.3.1 算术运算符及表达式	22
2.3.2 赋值运算符及表达式	23

2.3.3 关系运算符及表达式	24
2.3.4 逻辑运算符及表达式	24
2.3.5 位运算符	25
2.3.6 条件运算符	27
2.3.7 运算符的优先级	27
2.4 流程控制语句	28
2.4.1 C++ 语言语句	28
2.4.2 if 语句与条件选择控制	31
2.4.3 switch 语句与多项选择	36
2.4.4 while 语句	38
2.4.5 do...while 语句	39
2.4.6 for 语句	40
2.4.7 break 语句和 continue 语句	41
2.4.8 循环嵌套	43
2.4.9 程序设计综合举例	44
2.4.10 return 语句	46
思考题二	46
实训二	48
 第三章 函数与程序结构	 50
3.1 函数与程序结构概述	50
3.2 函数的定义与声明	52
3.2.1 函数的定义	52
3.2.2 函数声明	54
3.3 函数参数和函数调用	55
3.3.1 函数形式参数和实际参数	55
3.3.2 函数的返回值	56
3.3.3 函数调用	56
3.4 函数的嵌套与递归调用	57
3.4.1 函数的嵌套调用	57
3.4.2 递归调用	58
3.5 变量作用域和存储类型	59
3.5.1 局部变量与全局变量	59
3.5.2 动态存储变量和静态存储变量	60
3.6 内联函数	61
3.7 重载函数与默认参数函数	62

3.7.1 重载函数	62
3.7.2 默认参数函数	63
3.8 编译预处理	64
3.8.1 文件包含	64
3.8.2 宏定义	64
3.8.3 条件编译	65
小结	65
思考题三	65
 第四章 数组与字符串	66
4.1 数组的概念	66
4.2 数组的定义	67
4.2.1 一维数组	67
4.2.2 二维数组	74
4.3 数组作为函数的参数	79
4.3.1 用数组元素作函数参数	79
4.3.2 用数组名作函数参数	80
4.3.3 用多维数组名作函数参数	82
4.4 数组应用举例	84
4.5 字符串	93
4.5.1 字符串概念	93
4.5.2 字符串函数	96
4.5.3 字符串应用举例	99
小结	104
思考题四	104
 第五章 指针	107
5.1 指针的概念	107
5.2 指针变量	108
5.2.1 指针定义	108
5.2.2 指针运算符(& 和 *)	109
5.2.3 引用变量	110
5.2.4 多级指针与指针数组	112
5.2.5 指针与常量限定符	115
5.3 指针与数组	116
5.3.1 指针与一维数组	116



5.3.2 指针与二维数组	122
5.3.3 指针与字符数组	125
5.3.4 指针与函数	127
5.4 指针运算	131
5.5 动态存储分配	134
5.5.1 new 操作符	134
5.5.2 delete 操作符	135
小结	138
思考题五	139
第六章 结构体与共用体	141
6.1 结构体	141
6.1.1 结构体的声明	141
6.1.2 结构体变量的引用及初始化赋值	143
6.2 嵌套结构体	144
6.3 结构体数组	146
6.3.1 结构体数组的定义和初始化	146
6.3.2 结构体数组成员的引用	147
6.4 结构体指针	148
6.4.1 指向结构体变量的指针	148
6.4.2 指向结构体数组的指针	150
6.4.3 用结构体变量和指向结构体变量的指针作为函数参数	151
6.5 链表的基本操作	153
6.5.1 链表基本知识	153
6.5.2 内存动态管理函数	154
6.5.3 建立链表	155
6.5.4 输出链表	158
6.5.5 对链表的删除操作	159
6.5.6 对链表的插入操作	160
6.5.7 对链表的综合操作	162
6.6 共用体	164
6.6.1 共用体的概念	164
6.6.2 共用型变量的定义	165
6.6.3 共用型变量的引用	166
6.6.4 共用体类型数据的特点	167
6.6.5 共用体变量的应用	167

6.7 枚举类型	169
6.8 用 typedef 定义	172
思考题六	173
第七章 类与对象及封装性	175
7.1 类的抽象	175
7.2 类的定义与对象的生成	175
7.3 构造函数和析构函数	180
7.4 构造函数的重载	184
7.5 对象指针	185
思考题七	187
第八章 类的深入	189
8.1 友元函数	189
8.2 对象传入函数的讨论	193
8.3 函数返回对象的讨论	196
8.4 拷贝构造函数	198
8.5 this 关键字	202
思考题八	203
第九章 运算符重载	205
9.1 使用成员函数的运算符重载	205
9.2 友元运算符函数	210
9.3 重载关系运算符	213
9.4 进一步考查赋值运算符	214
9.5 重载 new 和 delete	216
9.6 重载[]	218
9.7 重载其他运算符	221
思考题九	223
第十章 继承性	224
10.1 继承性的理解	224
10.2 类的继承过程	225
10.3 基类访问控制	227
10.4 简单的多重继承	232



10.5	构造函数/析构函数的调用顺序	233
10.6	给基类构造函数传递参数	234
10.7	访问的许可	236
10.8	虚基类	237
	思考题十	239
 第十一章 多态性		241
11.1	指向派生类型的指针	241
11.2	虚函数	243
11.3	继承虚函数	245
11.4	多态性的优点	245
11.5	纯虚函数和抽象类	247
	思考题十一	248



第一章 C++ 语言概述

【学习目的和要求】

通过本章的学习,了解C++语言的发展、特点以及面向对象7程序设计的几个基本概念,掌握C++应用程序的一般结构。通过技能训练,掌握一种C++环境的基本使用方法。

1.1 C++ 语言简介

1.1.1 C++ 语言的发展

C语言是贝尔实验室于20世纪70年代初研制出来的,后来又被多次改进,并出现了多种版本。C语言既具有高级语言的特点,表达力丰富,可移植性好;又具有低级语言的一些特点,能够很方便地实现汇编级的操作,目标程序效率较高。刚开始形成的C语言,受到那些想建立更快、更有效代码的程序员们的欢迎。有一位名叫Bjarne Stroustrup的人却不满足于仅仅是生产快速代码,他想创建面向对象的C语言。他开始对C语言的内核进行必要的修改,使其能满足面向对象模型的要求。C++从此产生。

C++语言自诞生以来,经过开发和扩充已形成一种完全成熟的编程语言。现在C++已由ANSI、BSI、DIN、其他几个国家标准机构和ISO定为标准。ISO标准于1997年11月4日经投票正式通过。

C++标准演变了许多年。C++模板则是近几年来对此语言的一种扩展,模板是根据类型参数来产生函数和类的机制,有时也称模板为“参数化的类型”。使用模板,可以设计一个对许多类型的数据进行操作的类,而不需要为每个类型的数据建立一个单独的类。标准模板库(Standard Tempalte Library, STL)和微软的活动模板库(Active Tempalte Library, ATL)都基于这个C++语言扩展。

C++标准可分为两部分——C++语言本身和C++标准库。标准库提供了标准的输入/输出、字符串、容器(如矢量、列表和映射等)、非数值运算(如排序、搜索和合并等)和对数值计算的支持。应该说,C/C++包含了相对少的关键字,而且很多最有用的函数都来源于库,C++标准库实现容器和部分算法就是STL。

1.1.2 C++ 语言的特点

C++ 语言之所以被人们广泛认可,是因为它具有许多先进的技术特点。

1. 优越的性能

其性能有两个方面:算法速度和机器代码效率。一个算法可以定义为数据通过系统的概念化的路径,它描述一些点,在这些点上,数据能够被操作并可转换产生某个结果。例如,一个算法定义为获取一个字符串,计算字符串中的字符个数,并作为结果返回的过程。算法与语言是独立的,所以在编程之前必须设计算法,编写一个快速程序的第一个步骤是设计良好的算法,能以最少的操作步骤得出问题的答案。第二个步骤是选择语言,这也影响程序的速度。

从性能的角度考虑,用汇编语言编写程序是最佳的选择,它是计算机能理解的自然语言。但是,几乎没有人用汇编语言编写完整的程序,因为这样做极其乏味。另一个最佳的选择是 C 语言。然而,由 Visual C++ 提供的所有工具都产生 C++,而不是 C。使用 Visual C++ 的向导可以生成大量的实用代码,而不必人工地编写代码。从编写程序的难易程度和程序的性能综合考虑,C++ 是最佳的选择。

C++ 性能良好,因为它被编译为机器代码。对 VBScript 和 Java 等语言,代码在运行时由程序解释,而且每次运行程序时都要将代码转换为机器码,这样做效率比较低,不仅仅是已编译过的 C++ 程序运行得较快,而且微软 C++ 编译器已存在多年。这意味着微软的编译器程序员已经把许多优点集中到编译器上,以致它能产生非常高效的机器码。

2. 全面兼容 C

C++ 语言保持了 C 简洁、高效和接近汇编等特点,同时对 C 的类型系统进行了改革和扩充。C++ 对 C 的兼容性体现在许多 C 代码的程序不必修改就可被 C++ 所使用上。为保持这种兼容性,C++ 也支持面向过程的程序设计,因此 C++ 不是一个纯正的面向对象语言。

3. 支持面向对象的方法

C++ 是一种支持面向对象的程序设计语言 (Oriented Object Programming, OOP)。C++ 语言代码以类的形式组成,使得应用程序的开发变得十分容易。C++ 面向对象的特征主要有封装性、继承性和多态性,本书将在以后的内容中详细介绍。

1.2 C++ 程序简介

在开始学习 C++ 语言编程之前,应该了解一下 C++ 源程序的基本构成,以及如



何书写、编译和运行C++ 程序,以便建立一个总体的印象。

用C++ 语言编写应用程序,再到最后得到结果,需要经过3 个过程,即编写源程序、编译和运行。

1. 编写源程序

一个简单的C++ 应用程序如例 1-1 所示。

【例 1-1】 一个简单的C++ 应用程序。

```
// -----  
//    first.cpp  
// -----  
#include <iostream.h>  
void main ()  
{  
    // Output a string  
    cout << "This is my first C++ program." << endl;  
}
```

通过这个程序可以看到,C++ 应用程序的结构并不复杂。编写C++ 程序时必须遵循C++ 语言的编程原则。一个简单的C++ 应用程序的基本格式有以下几点规定:

(1) C++ 程序是无格式的纯文本文件,可以用任何文本编辑器(例如,记事本、写字板)来编写C++ 程序。

(2) C++ 程序(源代码)保存为文件时,建议使用默认扩展名.cpp。文件名最好有一定提示作用,能使人联想到程序内容或功能。

(3) 每个C++ 程序都由一个或多个函数组成,函数则是具有特定功能的程序模块。对一个应用程序来讲,还必须有一个 main () 函数,且只能有一个 main () 函数。该函数标志着执行应用程序时的起始点。例 1-1 中关键字 void 表示 main () 函数无返回值。

(4) 任何函数中可以有多条语句。例 1-1 的 main () 函数中只有一条语句,即:

```
cout << "This is my first C++ program." << endl;
```

该语句用来在屏幕上输出一个“This is my first C++ program.”字符串。cout 是 C++ 的一个对象,可通过它的操作符“<<”向显示设备输出信息。

(5) C++ 程序中的每条语句都要以分号“;”结束,包括以后程序中出现的类型说明等。

(6) 为了增加程序的可读性,程序中可以加入一些注释行,例如,用“//”开头的行。关于C++ 语言的注释定义符说明详见本书第二章。

(7) 在C++ 程序中,字母的大小写是有区分意义的,因此 main、Main、MAIN 都是

不同的名称。作为程序的入口只能是 `main()` 函数。

2. 编译

当 C++ 程序编写完成后,必须经过 C++ 编译器把 C++ 源程序编译成 .obj 的目标文件,然后使用连接工具将目标文件连接为 .exe 的应用程序。在 C++ 集成环境中,往往可以通过 Build 命令一次完成这两个步骤。

3. 运行

根据运行的不同目的,运行可分为应用运行、测试运行和调试运行。应用运行是指程序正式投入使用后的运行,目的是通过程序的运行完成预先设定的功能,从而获得相应的效益。测试运行是应用运行前的试运行,是为了验证整个应用系统的正确性,如果发现错误,应进一步判断错误的原因和产生错误的大致位置,以便加以纠正。调试运行则是专门为验证某段程序的正确性而进行的。运行时,通过输入一些特定的数据,观察程序是否产生预期的输出。如果没有,则通过程序跟踪方法,观察程序是否按预期的流程运行,程序中某些变量的值是否如预期的那样改变,从而判定出错的具体原因和位置,再加以纠正。

1.3 C++ 程序的开发环境

目前,比较流行的 C++ 程序集成开发环境有基于 Windows 平台的 Microsoft Visual C++ 和 Borland C++ Builder 以及基于 DOS 平台的 Turbo C++ 和 Borland C++。下面对 Visual C++ 和 Turbo C++ 开发环境的使用作简要介绍。

1.3.1 Turbo C++ 介绍

Turbo C++ 3.0 软件是 Borland 公司在 1992 年推出的强大的 C 语言程序设计与 C++ 面向对象程序设计的集成开发工具。它只需要修改一个设置选项,就能够在同一个集成开发环境 (IDE) 下设计和编译以标准 C 和 C++ 语法设计的程序文件。

1. 启动 Turbo C++

当 Turbo C++ 成功安装后,将自动在其安装目录下建立一个 BIN 子目录,该子目录下的 TC.exe 为 Turbo C++ 的启动程序,运行该程序可进入 Turbo C++ 集成开发环境,如图 1-1 所示。

2. Turbo C++ 集成开发环境

进入 Turbo C++ 集成开发环境后,可通过 File 菜单下的 New 选项新建 C++ 程序。图 1-2 演示了例 1-1 在 IDE 中编辑后并被保存为 first.cpp。

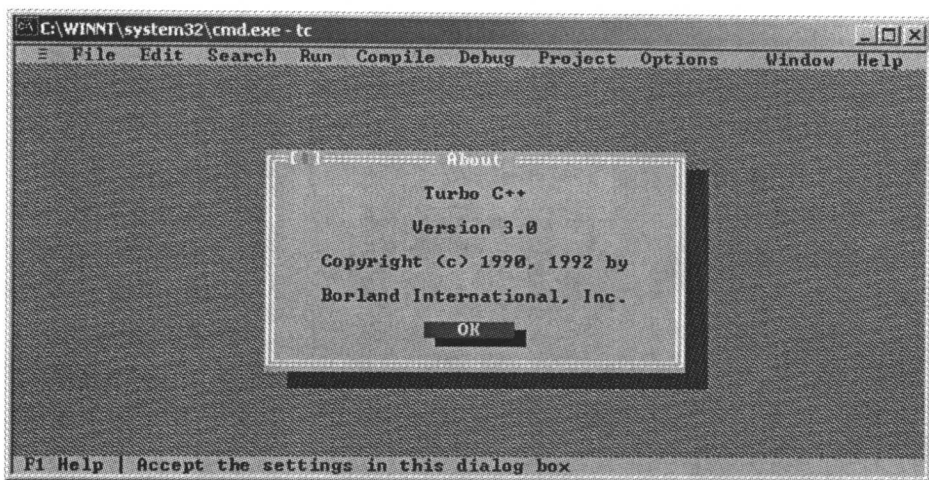


图 1-1 Turbo C++ 集成开发环境

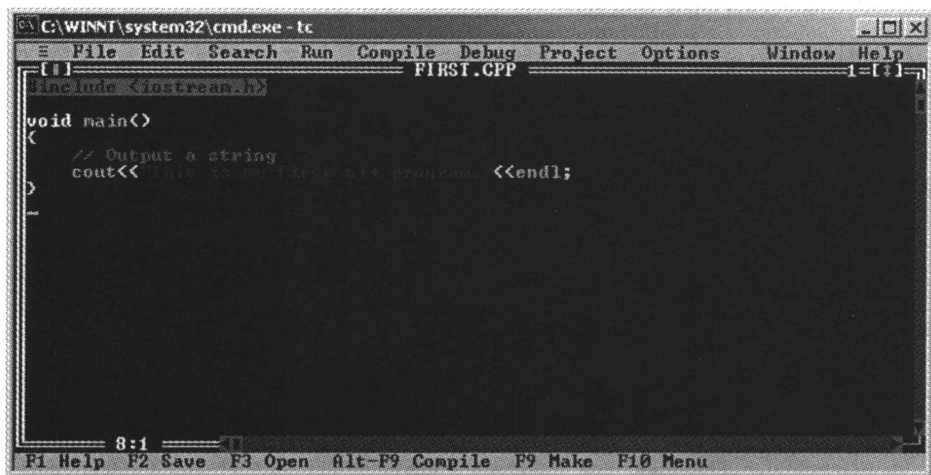


图 1-2 Turbo C++ 程序编辑器

3. C++ 程序编译和连接

在 Turbo C++ 集成环境中将源程序编辑完毕并且保存后,可以使用 Alt + C 组合键打开 Compile(编译)菜单,选择 Compile 选项(或快捷键 Alt + F9)对源程序进行编译,得到目标文件,然后选择 Link 选项将目标文件连接为可执行程序文件。出现错误则检查修改程序,然后重复上述步骤。C++ 程序的编译和连接如图 1-3 所示。

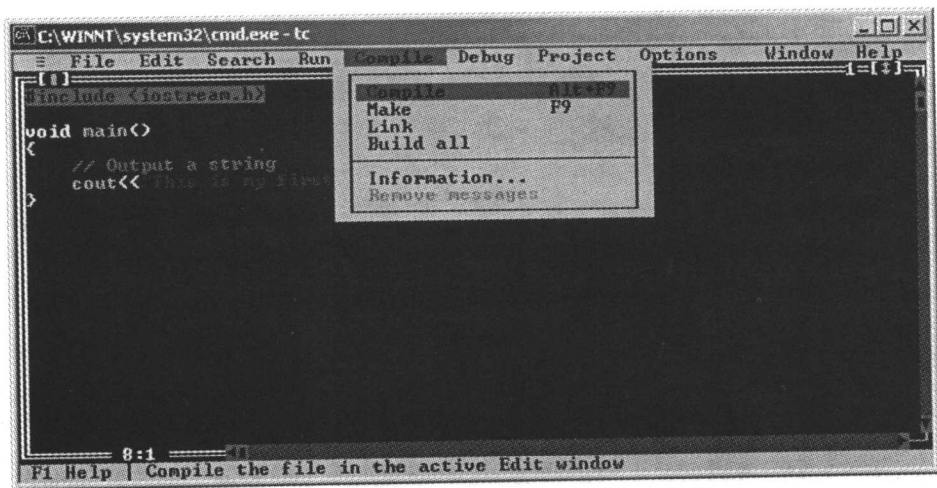


图 1-3 C++ 程序的编译和连接

4. C++ 程序执行和结果查看

C++ 程序在 Turbo C++ 集成环境中编译连接成功之后,可以使用 Alt + R 组合键打开 Run(运行菜单),选择 Run 选项(或 Ctrl + F9 快捷键)来运行程序。

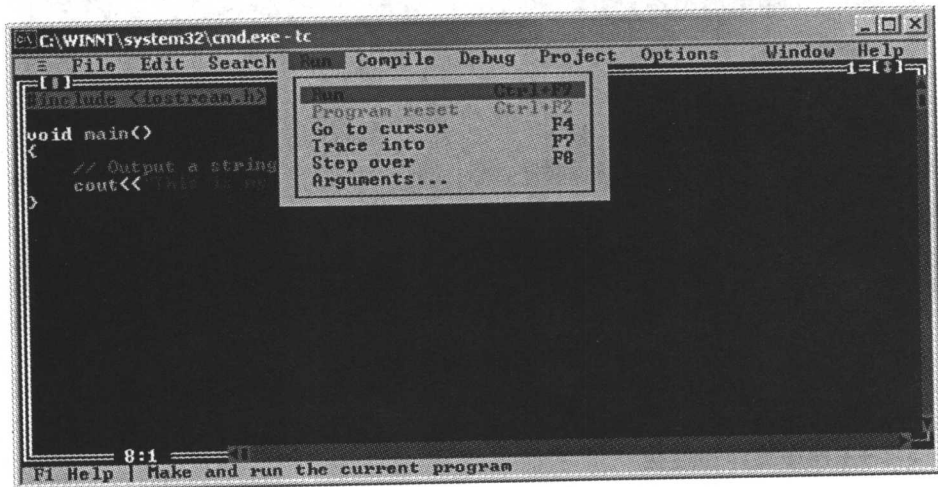


图 1-4 Turbo C++ 运行菜单

Turbo C++ 运行菜单如图 1-4 所示。

程序运行后,结果输出窗口需要使用 Windows 菜单下的 User screen 选项查看,



或者通过快捷键 Alt + F5 查看,然后按任意键返回集成环境界面。

1.3.2 Visual C++ 介绍

Visual C++ 是美国 Microsoft 公司最新推出的可视化 C++ 开发工具,是目前计算机开发者首选的 C++ 开发环境。它支持最新的 C++ 标准,它的可视化工具和开发向导使 C++ 应用开发变得非常方便快捷。

Visual C++ 已经从 Visual C++ 1.0 发展到最新的 Visual C++ 7.0 版本。不管使用何种版本,其基本操作大同小异。本节以 Visual C++ 6.0 为背景简单介绍 Visual C++ 的使用方法。

1. 启动 Visual C++

当 Visual C++ 成功安装后,通过选择 Windows 桌面的“开始”→“程序”→“Microsoft Visual Studio 6.0”→“Microsoft Visual C++ 6.0”就可以启动 Visual C++。Visual C++ 6.0 的集成开发环境如图 1-5 所示。

2. Visual C++ 集成开发环境

在 Visual C++ 环境中,开发应用程序的第 1 步是创建一个工程。Visual C++ 采用工程组织和维护应用程序。工程文件保存了与工程有关的信息。每个工程都保存在自己的目录中。每个工程目录包括一个工作区文件(.dsw)、一个工程文件(.dsp)、至少一个 C++ 程序文件(.cpp)以及 C++ 头文件(.h)。

Visual C++ 的工程向导简化了工程的创建,出现如图 1-6 所示的画面。当使用工程向导创建新工程时,向导会自动设置工程的目录框架,创建并保存工程属性;还可以为工程创建一个工程记录文件,该文件的信息会出现在工程源文件的注释中,并会出现在生成的文档中。

Visual C++ 6.0 提供了许多向导,可以极大地节省应用程序开发的时间。选择“File”菜单下的“New”命令项后,出现如图 1-6 所示的画面。

Visual C++ 提供的 Win32 Console Application 工程向导用来生成控制台应用程序,适合 C++ 程序设计的初学者编写简单应用程序。此界面中需要输入工程的名称和存放路径。确定工程名称和存放路径后,工程向导将询问应用程序的创建类型,出现如图 1-7 所示的画面。可以选择一个空的工程类型,单击 Finish(结束)按钮。至此,Visual C++ 工程向导已经创建了一个空的控制台工程。

3. 编辑 C++ 源程序

工程创建完成后,需要新建 C++ 源文件以进行源代码编写。此时选择“File”菜单下的“New”命令项后,出现如图 1-8 所示的画面。

在 Files 页面中选择 C++ Source File,同时输入加入工程的 C++ 程序文件的名称。