普通高等教育“十一五”计算机类规划教材

C++程序设计

张 桦 主 编
董 晨 副主编

免费电子课件

机械工业出版社
CHINA MACHINE PRESS

普通高等教育“十一五”计算机类规划教材

C++程序设计

主　编　张　桦
副主编　董　晨
参　编　夏承遗　李文杰

机械工业出版社

本书紧扣标准 C ++，涵盖C ++的主要语言特性，强调 C ++标准库的使用。全书共分8章：第1章简述 C ++语言的历史与演化、编译器与集成开发环境以及程序的编写、编译和运行；第2章介绍 C ++语言的基本数据类型，常量、变量以及运算符和表达式；第3章讲述数组和指针的概念、初始化和使用，介绍标准库 string 和 vector 类型；第4章介绍 C ++语言的程序流程控制以及编译预处理；第5章介绍 C ++语言中有关函数的特性和使用；第6章讲述类和对象，描述类的声明与定义，讨论类的构造和析构，类的静态成员、友元，类的运算符重载，对象指针、对象引用以及类成员指针；第7章重点讨论 C ++面向对象程序设计，介绍面向对象程序设计的设计思想、基本概念和基本方法；第8章介绍 RTTI 与异常处理。

本书适用于高等院校计算机专业/非计算机专业本科生，以及其他软件工程技术人员。为方便教师教学，本书配有教学课件，欢迎选用本书作为教材的老师索取，索取邮箱：llm7785@sina.com。

图书在版编目(CIP)数据

C ++程序设计/张桦主编. —北京：机械工业出版社，2008.6

普通高等教育“十一五”计算机类规划教材

ISBN 978-7-111-23846-1

Ⅰ. C… Ⅱ. 张… 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(2008)第045079号

机械工业出版社(北京市百万庄大街22号 邮政编码100037)

责任编辑：刘丽敏 责任校对：陈延翔

封面设计：张 静 责任印制：洪汉军

北京铭成印刷有限公司印刷

2008年6月第1版第1次印刷

184mm×260mm · 21.5印张 · 529千字

标准书号：ISBN 978-7-111-23846-1

定价：36.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

销售服务热线电话：(010)68326294

购书热线电话：(010)88379639、88379641、88379643

编辑热线电话：88379726

封面无防伪标均为盗版

前　言

本书作者长期从事本科生C++语言程序设计课程的教学工作，多次使用C++语言开发工程项目。在总结多年讲授C++语言的经验基础上，结合工程项目开发的实践，编写了本书。

本书体系完整，将C++作为一门独立的语言，而不是先讲授C语言，然后再讲C++扩展。全书紧扣C++标准，涵盖该语言的主要特性。在有限的篇幅内，由浅入深，通过大量的例题，在解决特定问题或程序设计任务的上下文环境中，讲授C++语言：从基本概念、数据类型和程序流程控制开始，介绍现代C++语言。讲述了string、vector、iostream，在讨论了函数程序设计之后，重点讲授C++支持面向对象程序设计的核心概念：封装、继承和多态，揭示隐藏在C++语法背后的语义，结合大量的代码阐述了C++面向对象程序设计的设计思想、基本概念和基本方法。

书中精心设计了程序范例，通过这些例题，贯穿前后章节的知识要点，培养学生综合应用能力。全书在介绍C++语言特性的同时强调编程风格。另外，每章都备有考核知识要点的练习题，内容全面，形式多样。读者通过对本书的学习，能够一窥C++语言的全貌，对C++语言有一个较深入和完整的认识，而不仅仅停留在介绍C++的语法和词法层面上，从而具备一定的软件设计与开发能力，能开发中小型的应用程序。

本书共分8章，第1章是开始学习C++，第2章是数据类型、运算符和表达式，第3章是string、vector、数组和指针，第4章是程序流程控制，第5章是函数，第6章介绍类和对象，第7章讲述面向对象程序设计，第8章介绍RTTI与异常处理，附录为C++库介绍。本书所有例题均在Visual C++6.0环境调试通过。

本书由张桦任主编，董晨任副主编。其中第2、3、7章由张桦编写，第1、5、8章和附录由董晨编写，第6章由夏承遗编写，第4章由李文杰编写。全书由董晨负责统稿。

在本书的编写过程中，得到了机械工业出版社高等教育分社刘丽敏的极大帮助，在此，对刘丽敏和出版社相关工作人员表示诚挚的感谢！本书也得到天津市《智能计算与软件新技术》重点实验室的大力支持，在此一并表示谢意！

由于作者水平有限，加之时间仓促，书中难免有一些疏漏和错误之处，敬请广大专家和读者批评指正。

编　者

目　录

第1章　开始学习C++

本章首先简要介绍C++语言，接着介绍如何编写一个C++“Hello World”程序，然后介绍程序在Windows和Linux平台下的编译与运行，最后对程序的各组成部分作简要介绍，其中包括：库函数、main函数、return语句和输入/输出流操作等，通过讲述“Hello World”程序，给读者一个C++语言程序设计的概略认识。

1.1　C++简介

C++，在程序员圈子中常读作“C加加”，英语中读作“C Plus Plus”，它是一种应用非常广泛的计算机编程语言。C++具有强大的灵活的语言机制，是由AT&T公司贝尔实验室的Bjarne Stroustrup博士在20世纪80年代发明并实现的。

最初，C++是作为C语言的增强版出现的。导致C++诞生的需求是1979年4月Bjarne博士等人分析UNIX内核，为了能有效地分析UNIX分布内核造成的网络流量，并将内核模块化，1979年10月，Bjarne博士完成了一个预处理程序，称为Cpre，为C语言增加了类机制，称作“C with Classes”，1983年8月，第一个C++实现并投入使用。

从给C语言增加类开始，虚函数(Virtual Function)、运算符重载(Operator Overload)、多重继承(Multiple Inheritance)、模板(Template)、异常(Exception)、名字空间(Namespace)等特性逐渐被加入，演变成今天的C++，C++语言发展大致可以分为3个阶段。

第一阶段从20世纪80年代到1995年，这一阶段，C++语言基本上是传统类型上的面向对象语言，并且凭借着接近C语言的效率，在工业界使用的开发语言中占据了相当大份额，根据《C++编程思想》(Thinking in C++)书中所评述的，C++与C的效率往往相差在-5%~+5%之间。

第二阶段从1995年到2000年，这一阶段由于标准模板库(STL)和后来的Boost等程序库的出现，泛型程序设计在C++中占据了越来越多的比重。但由于Java、C#等语言的出现和硬件价格的下降，C++受到一定的冲击。

第三阶段从2000年至今，由于以Loki、MPL等程序库为代表的产生式编程和模板元编程的出现，C++出现了发展历史上又一个新的高峰，这些新技术的出现以及和原有技术的融合，使C++已经成为当今主流程序设计语言中最复杂的成员之一。

1998年，国际标准组织(ISO)颁布了C++程序设计语言的国际标准ISO/IEC 14882—1998，称为C++98标准。2003年的ISO/IEC 14882—2003，称为C++03标准，是对C++98标准的补充和修正，目前正在制定中的C++0x的标准，预计于200x年通过并成为新标准(很可能x为9，即C++09标准)。对于C++0x的方向，Bjarne Stroustrup的总体思想是：对语言核心的改动应该是谨小慎微的，而对库的发展则应该是大胆而激进的[⊖]。

⊖ The Design of C++0x，Bjarne Stroustrup主页：http://www.research.att.com/~bs/rules.pdf

从某种意义上说，C ++仍然是一门很年轻的语言。从 1998 年 C ++的第一次标准化到现在，虽然语言的核心变化不大，但库的发展却未曾有一日停止过：STL、Boost、ACE、LOKI、MTL、Blitz ++…，其中最为瞩目的当属 STL。由于泛型编程(Generitic Programming，GP)机制的完美运用，STL 兼得了“鱼和熊掌”——效率和优雅。由于 GP 的强大表达力，在库的构建方面，像 Boost、ACE、LOKI、Blitz ++、MTL…等都不约而同地使用了 GP，并取得了巨大的成功，这些库所带来的思想和效益正逐渐深入工业界，并保持着 C ++长盛不衰的活力。本书将在附录中，介绍最著名的一些 C ++程序库。

C ++发展史上的主要事件见表 1-1。

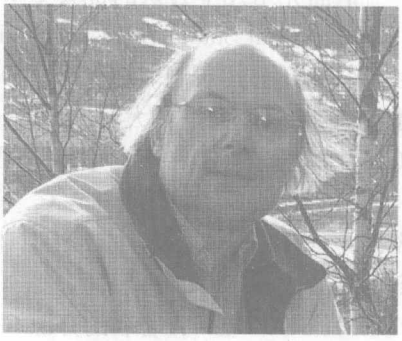
表 1-1　C ++发展史上的主要事件

1983 年 08 月	第一个 C ++实现并投入使用
1983 年 12 月	Rick Mascitti 建议命名为 C Plus Plus，即 C ++
1985 年 10 月	CFront 的第一个商业化版本发布，CFront Release 1.0
1985 年 10 月	The C ++ Programming Language 第一版出版
1986 年 11 月	第一个商用 CFront PC 移植(CFront 1.1，Glockenspiel)
1987 年 02 月	CFront Release 1.2 发布
1987 年 11 月	第一个 USENIX C ++会议在新墨西哥州举行
1990 年 7 月	增加模板机制
1990 年 11 月	增加异常机制
1991 年 6 月	The C ++ Programming Language 第二版出版
1991 年 6 月	第一次 ISO WG21 会议在瑞典召开
1991 年 10 月	CFront Release 3.0 发布
1993 年 3 月	增加运行时类型识别(RTTI)
1993 年 7 月	增加名字空间
1994 年 8 月	ANSI/ISO 委员会草案登记
1997 年 7 月	The C ++ Programming Language 第三版出版
1998 年 11 月	ISO 通过 C ++标准(ISO/IEC 14882—1998)
2003 年 10 月	ISO 发布 C ++标准的修正版(ISO/IEC 14882—2003)
2004 年	性能技术报告、库技术报告(散列(Hash)，正则表达式，智能指针)
2005 年	第一次关于 C ++0x 特征的投票，接纳 auto，static _ assert 和右值引用
2006 年	C ++0x 特征的第一次委员会投票(柏林，德国)

关于 C ++的历史与演化，Bjarne Stroustrup 博士写的“The Design and Evolution of C ++和 Evolving a language in and for the real world：C ++1991 ~2006”是非常好的参考文献。

C ++ 的创始人 Bjarne Stroustrup

Bjarne Stroustrup，C ++ 语言的设计者和第一位实现者，被称之为“C ++ 之父”，1950 年出生于丹麦奥尔胡斯市，在奥尔胡斯大学获得硕士学位，在英国剑桥大学获得博士学位。现任美国德州农工大学（Texas A&M University）计算机科学系教授，在此之前，他一直担任 AT&T 公司贝尔实验室的大规模程序设计研究部门的主管，也是 AT&T 公司贝尔实验室和 AT&T 公司的特别成员，ACM 的特别成员和 IEEE 的高级会员。他的研究方向包括分布式系统、操作系统、仿真、系统设计和程序设计。



Bjarne Stroustrup 一直积极推动 C ++ 语言的 ANSI/ISO 标准化，1990 年，他入选美国财富杂志评出的“全美 20 位青年科学家”；因其在程序设计语言方面的杰出成就于 1993 年获得 ACM 的 Grace Murray Hopper 奖；1995 年，入选 BYTE 杂志评出的“过去 20 年间计算机业界最具影响力的 20 人”。

Bjarne Stroustrup 是《The C ++ Programmig Language》（1985 年第 1 版，1991 年第 2 版，1997 年第 3 版，2000 年特别版）和《The Design and Evolution of C ++》的作者，前者自出版以来一直是 C ++ 语言领域最重要的著作之一，后者则以其对程序设计语言中思想和理想的碰撞、问题和现实约束的冲突的独到论述而广受关注。Bjarne 已经发表了 60 多篇学术论文，同时，还担任 Addison-Wesley 出版社的《C ++ In-Depth》系列书籍的顾问编辑。

1.2 编写、编译和运行 C ++ 程序

C ++ 程序设计与运行的基本步骤包括：

1）使用代码编辑器编写 C ++ 源文件。

2）调用 C ++ 编译器来生成可执行文件。

3）通过命令行运行可执行文件，或者从集成开发环境程序的菜单运行程序。

1.2.1 C ++ 编译器简介

程序源代码写好后需要经过编译，才能生成可执行程序。编译由 C ++ 编译器完成，具体包括 4 个步骤：预处理（Preprocessing）、编译（Compilation）、汇编（Assembly）和链接（Linking）。预处理对源代码中的#include、#define 等进行分析，编译、汇编生成以 .o 为后缀的目标文件，链接结合 .o 目标文件和程序库文件生成可执行程序。

AT&T 公司将 Bjarne Stroustrup 博士写的 C ++ 参考手册的所有权授给 ISO，ISO C ++ 标准是所有人可以免费使用的规范文档，C ++ 编译器生产与销售商不用支付版税给 ISO 和 AT&T。ISO C ++ 标准定义的是 C ++ 的功能和特性，而编译程序使用的编译器程序，是各个公司遵循 ISO C ++ 标准的实现产品，各公司的编译器都对 ISO C ++ 进行了改进，如 Microsoft Visual C ++ 6.0 使用 MFC，Borland C ++ Builder 使用 VCL，加入了自己的特点。

集成开发环境(Integrated Development Environment,IDE)使编辑器和编译器共同工作，在编辑器中创建源代码，使用编译(Compile)命令来启动编译器，当编译器发现错误时，会将编辑光标定位到出错语句处，以便于改正，IDE 为程序开发提供了一个整体环境。

下面介绍几个公认的优秀 C++编译器和 IDE。

1) Visual Studio 和 Visual Studio. NET 2002、2003、2005 以及 2008 中带的 C++编译器，由 Microsoft 公司研制。在 Visual Studio 6.0 中，编译器与 C++ 98 标准兼容相对较差，但是随着 C++编译器设计大师 Stanley Lippman 以及诸多 C++高手的加盟，在 Visual Studio. NET 2003 中，Visual C++编译器已经成为一个非常成熟可靠的 C++编译器了。

2) Borland C++ Builder X，其最大的特点是跨平台，跨编译器，多种 Framework 的集成，并且有一个 WxWindows 为基础的 GUI 设计器，对于 C++开发来说，从编译器，到库，到功能集成都是非常理想的。

3) GNU C++，著名的开源 C++编译器，是 GCC 的一部分。GCC 由 Richard. M. Stallman 在 1985 年开始编写，最初，GCC 指 GNU C Compiler，是 C 语言编译器，后来 GCC 迅速扩展，支持 C++、Fortran、Pascal、Objective-C、COBOL、Ada 和 Java 等其他语言，GCC 成了 GNU Compiler Collection，代表 GNU 编译器家族。GCC 支持非常广泛的软硬件平台，是支持 CPU 架构和操作系统最多的编译器，GCC 是 GNU/Linux 家族、BSD 家族、Mac OS X 和 NextStep 等操作系统的标准编译器。GCC 在符合 C++标准方面一直都非常好，GCC 有非常好的移植性，是编写跨平台、嵌入式程序的首选编译器。GCC 是 GPL 许可证的自由软件，是 GNU 计划的关键部分，GCC 目前在自由软件基金会的直接管理下，由数个程序小组维护。

4) Cygwin 是 Windows 下的 GNU/Linux 仿真环境，包括 Vi、Shell、tar、X Window 等 Linux 应用程序，包括 GCC、GDB 等 GNU 开发工具集，还有 MinGW(Minimalist GNU for Windows)包，可以与 Windows 的标准 C 运行库 msvcrt. dll 一起工作，Cygwin 遵守 GNU GPL 协议。

Cygwin/GCC 开发的程序可以无缝地用到 Linux，Cygwin 是在 Windows 下开发 Linux 程序的一个很好的选择。但是在 Cygwin/GCC 下编译出来的程序，在 Windows 下执行必须依赖 Cygwin1. dll，并且速度有些慢，如果不想依赖它，必须在 GCC 的编译选项中加入-mno-Cygwin，加入这个选项，GCC 编译器就会自动地选择链接 Cygwin/GCC 时安装的 MinGW 库，MinGW 是 GCC 的一个交叉编译。

说明：

交叉编译是嵌入式开发过程中的一项重要技术，它的主要特征是某机器中执行的程序代码不是在本机编译生成，而是由另一台机器编译生成，一般把前者称为目标机，后者称为主机。采用交叉编译的主要原因在于，多数嵌入式目标系统不能提供足够的资源供编译过程使用，因而只能将编译工作转移到高性能的主机中进行。

5) Dev C++是 Bloodshed 公司推出的基于 MingGW 的 C/C++IDE，是让 GCC 运行在 Windows 下的工具，Dev C++遵守 GNU GPL 协议。作为集成开发环境，提供了同专业 IDE 相媲美的语法，高亮代码提示，调试等功能，Dev C++的 IDE 是利用 Delphi 开发的。

6) Emacs + GCC，前面讲的是 Windows 环境下的 IDE，Linux 上开发者更倾向于使用

Emacs 来编辑 C++的源文件，用 Makefile 来控制 GCC 作编译，稍复杂，不适合初学者，本书不详细介绍。

7）Intel C++，著名 CPU 制造厂商 Intel 开发的编译器，Special Design for Intel x86，对于 Intel x86 结构的 CPU 经过特别的优化，特别是数值计算等高性能应用，采用 Intel 的编译器编译就能大幅度地提高性能。

GNU 的创始人 Richard. M. Stallman

Richard Matthew Stallman，简称 RMS，GNU 计划以及自由软件基金会(Free Software Foundation，FSF)的创立者，Stallman 1953 年出生于美国纽约，1971 年进入哈佛大学学习，期间 Stallman 开发了 Emacs 和 GNU C 编译器 GCC 软件。

Stallman 于 1983 年 9 月发起 GNU(GNU's Not Unix)计划，目标是创建一套完全自由，和 Unix 类似的操作系统。并发表著名的 GNU 宣言(GNU Manifesto)，解释为何发起该计划，其中一个理由就是要“重现当年软件界合作互助的团结精神”，从那时开始，许多程序员聚集起来共同开发一个自由的、高质量、易理解的软件。

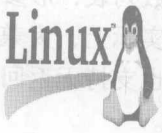
1985 年，Stallman 建立自由软件基金会，为 GNU 计划提供技术、法律以及财政支持。1989 年，Stallman 首创 GNU 通用公共协议证书(GNU General Public License，GNU GPL)，创造性地提出“Copyleft”概念，GNU GPL 是最广为采用的自由软件许可证：任何人在遵守 GPL 的条件下，可以免费使用和分发它，同时可以任意进行修改。

1992 年，Linux 遵守 GPL，Linux 核心与来自 GNU 计划的文字编辑器 Emacs、C/C++语言编译器 GCC、程序库和工具形成自由操作系统，称为 GNU/Linux 或简称 Linux，至此，完全自由的操作系统正式诞生，GNU 计划基本完成。

Emacs Logo

GNU Logo

GCC Logo



Linux Logo

建议初学者使用 IDE，可以避免学习复杂的编译参数，并方便程序调试，下面结合一个“Hello World”程序例子，简介 Visual C++ 6.0 的编译与运行。

1.2.2 程序的编译与执行

一个 C++程序可以由一个或多个函数组成，但是每个 C++程序都必须包含一个 main()函数，运行程序时，操作系统通过调用 main()函数开始执行程序，每个 C++程序都是从 main()函数开始执行，每个 C++程序有且仅有一个 main()函数，main()函数的通用形式如下：

```
int main(){
    …                    //完成函数功能的代码置于此处
}
```

函数体部分是完成函数功能的代码，函数体是函数定义中以左花括号“{”开始，并以右花括号“}”结束的语句块。下面是一个简单的 main()函数。

```
int main(){
    return 0;
}
```

例中唯一的语句是 return 0，它返回一个值 0 给操作系统，并终止函数。当 return 带上一个值(如 0)时，这个值称为函数的返回值，返回值的数据类型必须和函数的返回类型相同，或者可以转换成函数的返回类型。

ISO C++标准规定，main()函数必须有返回值，并且返回值必须是 int 类型(整型)。对于上例 main()函数，0 是 int 类型。main()函数的返回值是一个状态指示器，操作系统根据 main()函数的返回值确定程序是否成功执行完毕，返回值为 0 表示 main()函数成功执行完毕，返回值非 0 意味着有错误出现。按照 ISO C++标准，main()函数的形式只能是：

```
int main(){
    …
}
```

或者

```
int main(int argc,char* argv[]){
    …
}
```

main()函数的定义和其他函数一样，包含了函数定义必备的 4 个元素：返回类型、函数名、圆括号内的形参表和函数体。其中，函数通过函数名调用，形参表指明调用函数需要的参数列表，形参表可为空。函数的定义如图 1-1 所示。

实际上，为兼容标准以前的代码，Windows 平台的大部分编译器也支持无返回值的 main()函数，void 表示 main()函数没有返回值。

```
void main(){
    …
}
```

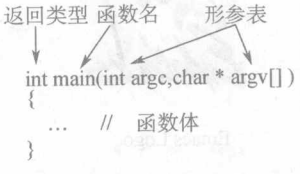


图 1-1 函数的定义

下面看一个程序，功能是将“Hello World!”输出到命令提示窗口，“Hello World”程序的源代码如下例所示：

【例 1.1】 输出并显示“Hello World!”。

```
//"Hello World!"program
#include <iostream>
using namespace std;
int main(){
```

```
    cout << "Hello World! " << endl;
    return 0;
}
```

如果不采用 C++集成开发环境(IDE)，编程的步骤是：首先，打开一个文本编辑器，如 Windows 的 notepad(记事本)程序，创建一个新文件，将上面源代码键入，并保存为 hello. cpp(源程序文件的扩展名为 . cpp)。然后，由 C++编译器将 hello. cpp 编译为 hello. obj(目标程序扩展名为 . obj)，hello. obj 链接到适当的 C++库生成 hello. exe(可执行文件扩展名为 . exe)程序。最后，通过在命令提示窗口中键入 hello 来执行 hello. exe 应用程序，程序执行后屏幕输出：

```
Hello World!
```

1.2.3 Microsoft Visual C++ 6.0 集成开发环境

1. 建立工程

启动 Microsoft Visual C++，单击菜单项：文件(F)→新建(N)...，在“工程”属性页对话框中，选择 Win32 Console Application，在位置[C]中选择项目文件的目录，例如 C:\DEMO\，在工程名称[N]中输入项目名称 helloWorld，默认生成可执行文件将是项目名称 . exe，即 helloWorld. exe，单击确定按钮，如图 1-2 所示。

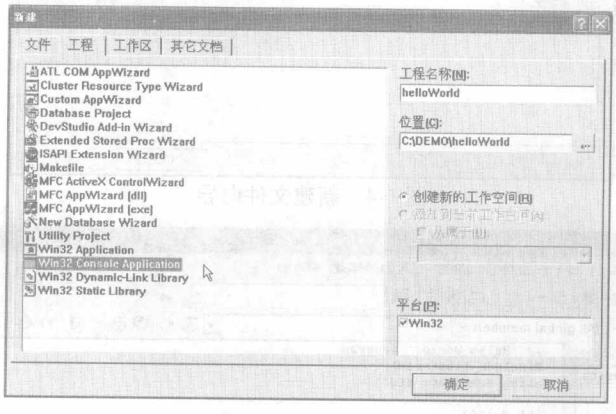


图 1-2 新建工程向导

2. 创建控制台程序的空工程

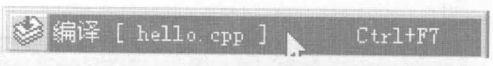
选择“一个空工程”，然后单击完成按钮，如图 1-3 所示。

3. 创建 C++源程序文件

(1) 文件(F)→新建(N)...，选择 C++Source File，输入文件名：hello，如图 1-4 所示。

(2) 输入并编辑源程序，如图 1-5 所示。

4. 编译源程序

选择组建[B]菜单中的编译[hello. cpp] 

，或直接按 Ctrl + F7 组合键，编译当前打开的源程序，编译结果如图 1-6 所示。

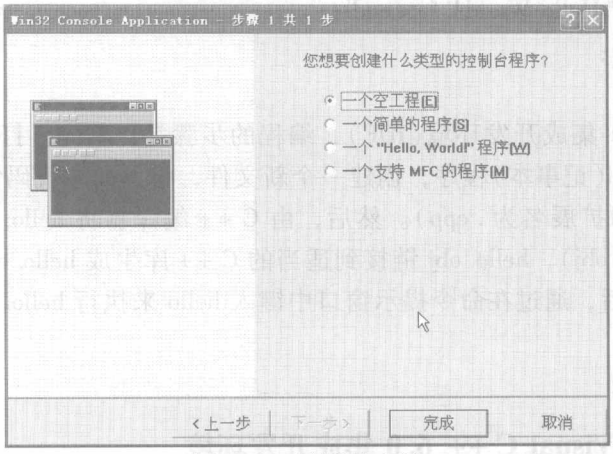


图 1-3　工程类型向导

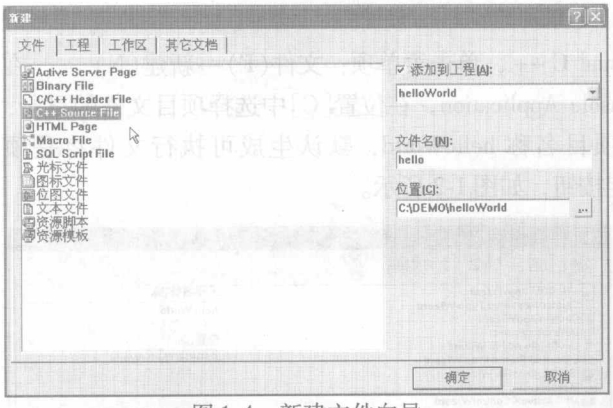


图 1-4　新建文件向导

图 1-5　输入并编辑源程序

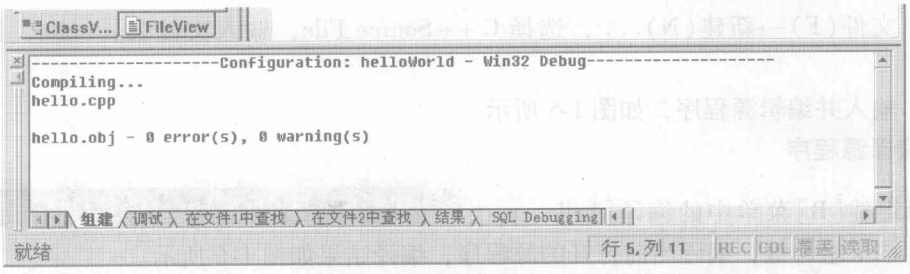
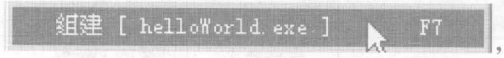


图 1-6　编译结果窗口

5. 链接程序

选择组建[B]菜单中的组建[helloWorld.exe] 

或直接按 F7 键，对当前工程进行链接，链接结果图 1-7 所示。

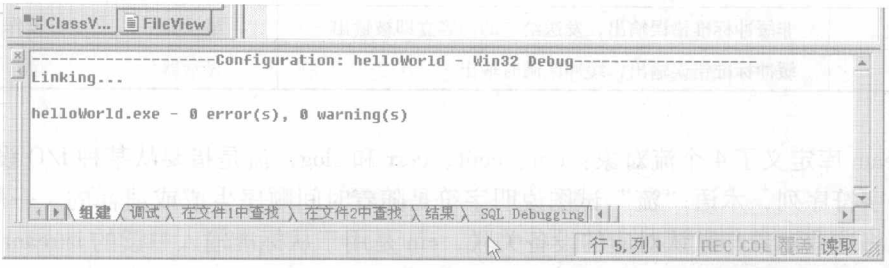
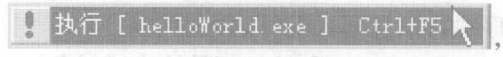


图 1-7 链接结果窗口

6. 运行程序

选择组建[B]菜单中的执行[helloWorld.exe] 

或直接按 Ctrl + F5 组合键运行程序，运行结果如图 1-8 所示。

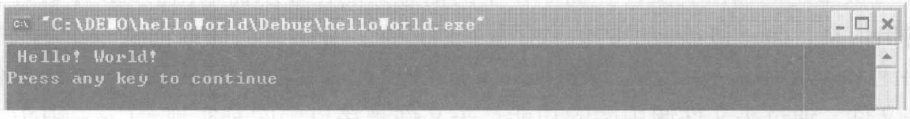


图 1-8 运行结果

1.2.4 “Hello World”程序详解

前面结合 IDE 介绍，编辑、编译并运行了“Hello World”程序：hello.cpp，下面详细解释 hello.cpp 中的每条语句。语句是程序的基本组成单元，程序由若干条语句组成，语句由单词组成，单词间用空格符分隔，语句以分号结束。在编程时应该注意 C++ 的书写格式，基本原则是：一行写一条语句，短语句可以一行写多个，长语句可以一条写多行，分行原则是不能将一个单词分开。语句如下：

```
1  //"Hello World!" program
2  #include <iostream>
3  using namespace std;
4  int main(){
5     cout <<"Hello World!" << endl;
6     return 0;
7  }
```

1. 标准输入与输出对象

C++ 语言没有内建输入或输出语句，I/O 操作是由标准库中的 iostream 库提供的，iostream 对象见表 1-2，这样做的目的是为了最大限度地保证语言与操作系统的平台无关性，因为输入输出操作是操作系统相关的，如果 C++ 为某种操作系统实现内部输入输出功能，那它也就被限制在这个操作系统上了。

表 1-2　iostream 中的 4 个流对象

iostream 对象	功　能	默认设备	发　音
cin	标准输入，从设备读入数据	键盘	读作 see-in
cout	标准输出，向设备输出或写数据，有缓冲	显示器	读作 see-out
cerr	非缓冲标准错误输出，发送给它的内容立即被输出	显示器	读作 see-err
clog	缓冲标准错误输出，缓冲区满时输出	显示器	读作 see-log

iostream 库定义了 4 个流对象：cin、cout、cerr 和 clog，流是指要从某种 I/O 设备上读入或写出的字符序列，术语“流”试图说明字符是随着时间顺序生成或消耗的。一般情况下，操作系统将这些对象与默认对应的设备关联，cin 是用于从标准输入中读的 istream 对象，cin 默认的输入设备是键盘。cout 用于写入到标准输出的 ostream 对象，用于程序的正常输出。cerr 绑定到标准错误输出的 ostream 对象(与 cout 相同的流)，输出 cerr 不缓冲，用于输出程序错误信息。clog 绑定到标准错误输出的 ostream 对象，写到 clog 时是带缓冲的，用于将程序执行信息写入到日志文件中。

cout、cerr 和 clog 都是输出流对象，cout、cerr 和 clog 的默认输出设备都是显示器，区别在于 cout 可被重定向输出到文件，而 cerr 中的信息只能在显示器输出。cerr 不缓冲，直接向显示器输出信息，cout 和 clog 中的信息存放在缓冲区，在缓冲区满或遇到 endl 时才输出。

C++程序要使用 C++标准库(包括 iostream 库)或者其他的程序库时，必须要包含该库相应的头文件，通过头文件调用库的功能，头文件是接口，库是实现，C++中头文件与库的划分体现了软件工程提倡的接口与实现相分离，因为在很多场合，源代码不便(或不准)向用户公布，只向用户提供头文件和二进制的库即可，用户按照头文件中的接口声明来调用库功能，不必关心接口是怎么实现的，而库的设计者，只要库接口不变，就可以在不影响用户已编写的代码前提下，实现库的发展与进化。

hello. cpp 使用了标准输出流 cout，所以必须将 iostream 库的头文件包含进来，具体代码是 hello. cpp 程序的第 2 行：

```
2   #include < iostream >
```

#include 告诉编译器要使用 iostream 库。尖括号里的名字是一个 iostream 库的头文件。#include指示必须单独写成一行——头文件名和#include 必须在同一行，头文件名和#include 之间由一个或者多个空格隔开。#include 指示应出现在任何函数的外部，通常，程序的所有 #include 指示都在程序的开头部分出现。

在编译之前，预处理器会用 iostream 里的全部内容来代替“#include < iostream >”这个预处理器指示符，链接时，编译器会从库中提取相应的代码。

2. 插入运算符

hello. cpp 的第 5 行语句使用插入运算符(<< 运算符)，在标准输出上输出“Hello World!”。

```
5   cout << " Hello World! " << endl;
```

插入运算符(<<，常发音为“put to”)接受两个操作数，把右操作数写到左操作数指定的输出流，如 cout << " hello "; 把 hello 写入到标准输出流。<< 运算符具有左结合性，<< 运算符可以链接在一起使用，第 5 行语句等价于：