21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

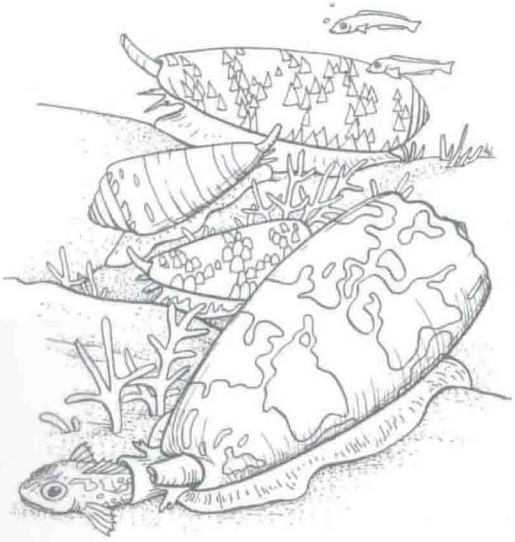
# C++语言程序设计

## C++ Programming Language

汤亚玲 胡增涛 主编

汪军 林芳 柯栋梁 李伟 姚红燕 副主编

- 内容由浅入深、通俗易懂
- 教学深入浅出、循序渐进
- 案例详细丰富、实践性强



高校系列

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等学校计算机规划教材

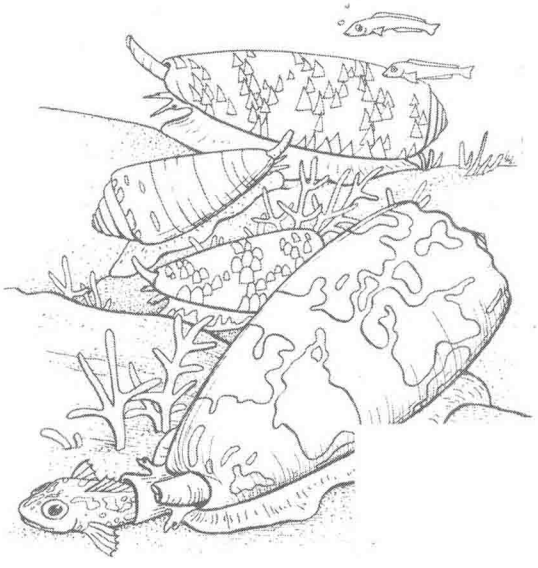
21st Century University Planned Textbooks of Computer Science

# C++语言程序设计

## C++ Programming Language

汤亚玲 胡增涛 主编

汪军 林芳 柯栋梁 李伟 姚红燕 副主编



高校系列

人民邮电出版社

北京

**图书在版编目（CIP）数据**

C++语言程序设计 / 汤亚玲，胡增涛主编. -- 北京 : 人民邮电出版社，2016.8
21世纪高等学校计算机规划教材
ISBN 978-7-115-42278-1

Ⅰ. ①C… Ⅱ. ①汤… ②胡… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2016)第177975号

## 内 容 提 要

本书系统地讲述 C++语言的基础知识、基本规则以及编程方法，详尽地讲述面向对象的基本特征—类和对象、继承性和派生类、多态性和虚函数等内容。每章配有丰富的例题和适量的练习题，便于自学。

本书文字简洁、精练，叙述清楚，通俗易懂。全书内容由浅入深，讲解突出重点，偏重对概念和语言机制的讲解。本书适合作为一般高等院校本科生的教材，也可作为教师和学生的参考书，以及广大电脑爱好者自学 C++语言的学习入门指导书。

◆ 主　　编　汤亚玲　胡增涛
副 主 编　汪　军　林　芳　柯栋梁　李　伟　姚红燕
责任编辑　吴　婷
责任印制　沈　蓉　彭志环

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京昌平百善印刷厂印刷

◆ 开本：787×1092　1/16
印张：19　　2016 年 8 月第 1 版
字数：501 千字　　2016 年 8 月北京第 1 次印刷

定价：45.00 元

读者服务热线：(010)81055256　印装质量热线：(010)81055316
反盗版热线：(010)81055315

# 前言

计算机程序设计是一门实践性很强的课程，也是计算机科学及其相关专业理论和实践相结合的一门必备课程。多年来，各大高校都把C/C++程序设计当作专业课程体系中的一门必修课。

C++自从诞生之日起，就以其简洁、高效、描述能力强被业界所重视。相比C语言，C++所新支持的面向对象编程模式，是一种分析问题、解决问题的新理念，这种模式更贴合实际情形，符合现实中人们解决问题的基本思路和方法。选择C++作为教学语言，其实用性和前瞻性不言而喻。C++通过类、对象、继承、多态、参数化程序设计以及异常等机制很好地支持了面向对象模式对实际问题的解决。因此，选择C++作为高等院校计算机及其相关专业学生的必修课程，有着非常重要的意义。

本书的编写组都是长年奋战在教学一线的老教师，有着较为深厚的理论功底和教学经验。在长期的实践教学中，他们深感一本言简意赅、叙述清楚、文字深入浅出，适合教学实情的教材的迫切性。这样的教材应该具有以下一些特征：一是能适应有良好C语言基础的学生学习的需要；二是能让没有能较好掌握C语言的学生有过渡和进行系统学习的机会；三是全书的知识体系要完整，章节、知识点的编排要合理，能适应一般工科院校的教学，让从教者能以清晰明了的教学思路传授C++的知识体系。

正是秉承着这样的编写指导思想，我们联合了省内外几所高校多位具有丰富教学经验的老教师，将自己平时在教学上积累的知识和当前主流的C++经典教材中的内容进行融合，并进行了适当的取舍，编写了本书。本书力争做到知识结构合理，各个章节相互联系又独立成篇，知识点过渡衔接自然，叙述清楚，简洁易懂，案例丰富同时避免枯燥乏味，让学习者能抓住知识的重点，并能比较轻松地构建自己学习C++语言的知识框架体系。

本书共分为12章，第1章介绍C++语言的一些基本概念和主要特征，让读者对于C++语言有一个总体的了解。第2章简要介绍了C++的一些基础知识，同时复习C/C++的一些基本语法结构。第3章介绍函数的相关知识，包括C++所支持的一些新的函数机制，如重载函数、内联函数等。第4章阐述类和对象的相关内容，本章是面向对象概念的主体部分，是全书的核心和重点之一。第5章讨论C++中对于数据共享的相关机制，以及一般C++程序的组织结构。第6章对C语言中已经学习过的数组、指针和字符串做进一步的讨论。第7章、第8章分别介绍继承和多态，继承和多态是面向对象中的高级技术，它们扩展了现有类的功能，并提供了更多的、以模拟现实的方式解决编程中问题的途径。第9章介绍流类库与输入输出，描述了在C++环境下，如何实现基本数据类型及其他类型数据输入输出的相关问题，以及文件使用的相关知识。第10章是异常处理，该章讨论面向对象编程中一种应对意外事件的解决方案——异常，它也是面向对象知识体系中的重点学习内容。第11章主要是针对时下主流的VC编译环境，介绍MFC的一些基础知识，让学生对VC系统自带的类库有个初步的了解。第12章是课程设计的内容，其目的在于学习前面章节之后，做一些综合性的训练。

本书可作为一般工科高等院校计算机类或者信息类相关专业“面向对象编程技术”课程的教材，建议理论课时为50~60学时，上机学时16学时左右，课程设计20学时左右。各院校可以根据本校的专业特点和具体学生情况，酌情增删学时。

本书由安徽工业大学汤亚玲、胡增涛老师任主编，负责全书的编撰和整理，以保证全书风格和内容的统一；由安徽工程大学汪军、姚红燕老师，福建工程学院林芳老师，安徽工业大学柯栋梁、李伟老师任副主编。其中第1章、第3章由汤亚玲编写，第7章、第8章、第12章由胡增涛编写，第11章由汪军编写，第2章、第5章由柯栋梁编写，第10章由李伟编写，第6章由姚红燕编写，第4章、第9章由林芳编写。全书由汤亚玲、胡增涛负责修改并统稿。

因编者水平有限，书中难免有不足甚至错误之处，敬请广大师生读者批评、指正。

如有任何建议或者意见，请联系 tangyl@ahut.edu.cn 或 huzengtao@ahut.edu.cn。

# 目 录

# 第1章 绪论

**教学提示：**本章是面向对象基本概念的导入篇，重点通过实例介绍面向对象的设计思想，以及C++对于面向对象的支持。教学中注意对比面向对象与面向过程的产生和各自的特点。

**教学目标：**了解C++的发展史和面向对象编程思想的渊源，理解面向对象在解决实际问题中的优势。

## 1.1 C++简介

在计算机诞生初期，人们要使用计算机就必须用机器语言或汇编语言编写程序。世界上第一种计算机高级语言是诞生于1954年的FORTRAN语言。之后又出现了多种计算机高级语言，其中使用最广泛、影响最大的当推BASIC语言和C语言。BASIC语言是1964年由Dartmouth学院的John G. Kemeny与Thomas E.Kurtz两位教授在FORTRAN语言的基础上简化而发明的，是适用于初学者设计的小型高级语言；C语言由美国贝尔实验室的D.M.Ritchie在1972年所开发，采用结构化编程方法，遵从自顶向下的原则。在操作系统以及需要对硬件进行操作的场合，用C语言明显优于其他高级语言，但在编写大型程序时，C语言仍面临着挑战。

在这种形势下，于20世纪80年代初，在美国贝尔实验室诞生了C++（plus plus）语言。它是由Bjarne Stroustrup博士主导，在C语言基础上设计并开发的一门新的语言。C++一经推出就获得了很大的成功，从很多方面大大提高了程序设计的效率，并转变着程序设计者的编程理念。

C++这个名字最初是由Rick Mascitti于1983年中所建议使用的，并于1983年12月首次使用。更早以前，尚在研究阶段的这门语言被称为“new C”，之后被称为“C with Class”。在计算机科学中，C++仍被称为C语言的上层结构。它最后得名于C语言中的“++”操作符（其对变量的值进行递增）。而且在共同的命名约定中，使用“+”以表示增强的程序。Bjarne Stroustrup说：“这个名字象征着源自于C语言变化的自然演进。”

C++相对于C语言一个很重要的改进，就是支持面向对象编程模式，并由此而产生了一系列新的特性。

## 1.2 面向对象与面向过程

简而言之，面向过程的核心思想是首先分析出解决问题所需要的步骤，然后用函数把这些步

骤一个个实现，最后依次调用。而面向对象是把构成问题的事务分解成各个对象，建立对象的目的不是为了完成一个步骤，而是为了描叙某个事物在解决整个问题的步骤中的行为。

例如五子棋，面向过程的设计思路就是首先分析出解决问题的步骤：

（1）开始游戏。
（2）黑子先走。
（3）绘制画面。
（4）判断输赢。
（5）轮到白子。
（6）绘制画面。
（7）判断输赢。
（8）返回步骤（2）。
（9）输出最后结果。

把上面每个步骤用函数分别实现，问题就解决了。

而面向对象的设计则是从另外的思路来解决问题。首先将整个五子棋系统分为以下三类对象。

（1）黑白双方：两方的行为相同。
（2）棋盘系统：负责绘制画面。
（3）规则系统：负责判定诸如犯规、输赢等。

第一类对象（玩家对象，即黑白双方）负责接受用户输入，并告知第二类对象（棋盘系统）棋子布局的变化，棋盘接收到了棋子的变化后负责在屏幕上面显示出这种变化，同时利用第三类对象（规则系统）来对棋局进行判定。

显然，面向对象是以概念来划分问题，而不是步骤。同样是绘制棋局，这样的行为在面向过程的设计中分散在了各个步骤中，很可能会出现不同的绘制版本，因为设计人员通常会考虑到实际情况而进行各种各样的简化。而在面向对象的设计中，绘图只可能在棋盘对象中出现，从而保证了绘图的统一。

功能上的统一保证了面向对象设计的可扩展性。例如需要加入悔棋的功能，按照面向过程的设计模式，那么从输入到判断到显示这一连串的步骤都要改动，甚至步骤之间的顺序都要进行大规模调整。如果以面向对象的设计模式，就只需要改动棋盘对象，棋盘系统保存了黑白双方的棋谱，进行回溯就可以；无须考虑显示和规则判断，同时整体对象功能的调用顺序没有变化，改动是局部。

另外，如要把五子棋游戏设计结果改为围棋游戏，在面向过程设计模式下，那么五子棋的规则就分布在了程序的每一个角落，改动的工作量和难度甚至超过重写。但是如果采用面向对象的设计模式，那么只需改动规则对象就可以了。五子棋和围棋的主要区别在于规则（棋盘大小一般有点差异，只需在棋盘对象中进行适当修改就可以了），而走棋的大致步骤，从面向对象的角度来看没有任何变化。

当然，要达到改动只是局部的目的，需要设计者有良好的设计和编程经验。使用对象不能保证你的程序就是面向对象，这需要充分学习面向对象的设计思想。初学者或者经验不够丰富的程序员很可能以面向对象之虚，行面向过程之实，这样设计出来的所谓面向对象的程序很难有良好的可移植性和可扩展性。

## 1.3 C++对面向对象的支持

C++既支持面向过程，又支持面向对象。面向过程的设计思想在处理一些简单问题时是可取的，比如一些单纯的、小规模的数值计算问题，C 语言或者 C++都可以胜任。但对于较为复杂的

大型软件的设计和开发，面向对象则体现出不可比拟的优势。

具体来说，C++主要通过支持以下一些面向对象的核心机制实现了分析、设计和求解的高效率。

**1. 抽象性**

抽象性是指将具有一致的数据结构和行为的对象抽象成类。一个类就是这样一种抽象，它反映了与应用有关的重要性质，而忽略其他一些无关内容。任何类的划分都是主观的，但一般与具体的应用有关。

**2. 继承性**

继承性是子类自动共享父类数据结构和方法的机制，这是类之间的一种关系。在定义和实现一个类的时候，可以在一个已经存在的类的基础之上来进行继承，把这个已经存在的类所定义的特性继承为自己的特性，并加入若干新的内容。

**3. 多态性**

多态性是指相同函数名的函数、过程可作用于多种类型的对象上，并获得不同的结果。或者说不同的对象收到同一消息可以产生不同的结果，这种现象称为多态性。

多态性允许每个对象以适合自身的方式去响应共同的消息。

## 1.4　C++的新特性

C++作为 C 语言之后的新一代语言，在很多方面提供了一些新的机制，使得语言整体上更为完善，效率也大大提高了。

C++语言的主要特点表现在两个方面：一是尽量兼容 C，二是支持面向对象的方法。它延续了 C 的简洁、高效地接近汇编语言等特点，对 C 的类型系统进行了改革和扩充，因此 C++比 C 更安全，C++的编译系统能检查出更多的类型错误。另外，由于 C 语言的广泛使用，从而极大促进了 C++的普及和推广。

C++对 C 的“增强”，主要表现在以下一些方面。

（1）C++的类型检查更为严格。

在 C++中，类型检查不仅延续了 C 中默认的低精度数据类型向高精度数据类型自动转换，支持多种形式的强制类型转换，以及多种不同类型指针的一致性检查；而且在通过赋值兼容规则、多态等多种形式支持家族类库中多个不同类型的兼容性，如果违背了这些规则，编译系统会做出警告或者错误的处理结果。

（2）C++增加了常类型。

常类型是 C++中对于共享数据的一种保护机制，是指使用类型修饰符 const 说明的类型。常类型的变量或对象的值是不能被更新的；因此，定义或说明常类型时必须进行初始化。

常类型的作用类似于 C 语言中的常量定义，这在 C 中是用宏定义形式：

#define　常量　数值

C++中，对于共享数据的保护的范围扩充到了多个范畴，包括基本数据类型、常对象、常引用、常类成员等，其定义的基本形式是：

const 类型符　标识符名；

C++中，常类型的定义形式要比 C 语言中更灵活、多变；特别是在类函数成员中，可以定义常成员函数，此函数可以和同名的其他他员函数构成函数重载。

（3）C++增加了泛型编程的机制。

泛型编程最初诞生于C++中，由 Alexander Stepanov 和 David Musser 创立。目的是为了实现C++的标准模板库（STL）。其语言支持机制就是模板（Templates）。模板的本质其实很简单，即参数化类型，其意思是，把一个原本特定于某个类型的算法或类当中的类型信息抽掉，抽出来做成模板参数 T。

泛型编程的代表作品 STL 是一种高效、泛型、可交互操作的软件组件。STL 以迭代器（Iterators）和容器（Containers）为基础，是一种泛型算法（Generic Algorithms）库，容器的存在使这些算法有东西可以操作。STL 包含各种泛型算法（Algorithms）、泛型迭代器（Iterators）、泛型容器（Containers）以及函数对象（Function Objects）。STL 并非只是一些有用组件的集合，它是描述软件组件抽象需求条件的一个正规而有条理的架构。

（4）C++增加了异常处理。

异常处理是编程语言或计算机硬件里的一种机制，用于处理软件或信息系统中出现的异常状况（即超出程序正常执行流程的某些特殊条件）。通过异常处理，我们可以对用户在程序中的非法输入或者程序执行时的意外情形进行控制和提示，以防程序崩溃。

C++异常处理机制是一个用来有效地处理运行错误的非常强大且灵活的工具，它提供了更多的弹性、安全性和稳固性，克服了传统方法所带来的问题。

异常的抛出和处理主要使用了以下三个关键字：try、throw 、catch。

抛出异常即检测是否产生异常，C++采用 throw 语句来实现，如果检测到产生异常，则抛出异常。该语句的格式为：

throw 表达式;

如果在 try 语句块的程序段（包括在其中调用的函数）中发现了异常，且抛弃了该异常，则这个异常就可以被 try 语句块后的某个 catch 语句所捕获并进行处理。捕获和处理的条件是被抛弃的异常的类型与 catch 语句的异常类型相匹配。由于 C++使用数据类型来区分不同的异常，因此在判断异常时，throw 语句中的表达式的值就没有实际意义，而表达式的类型就特别重要。

try-catch 语句形式如下。

```
try
  {
     包含可能抛出异常的语句
  }
 catch(类型名 [形参名]) // 捕获特定类型的异常
  {
     处理异常语句
  }
 catch(类型名 [形参名]) // 捕获特定类型的异常
  {
    处理异常语句
   }
  catch(...)    // 三个点（…）表示捕获所有类型的异常
   {
   }
```

（5）C++增加了运算符重载。

C++的另一个重大特性就是重载(overload)，通过重载可以把功能相似的几个函数合为一个，使得程序更加简洁、高效。在 C++中不止函数可以重载，运算符也可以重载。运算符重载最根本的出发点是扩展已有运算符的运算功能，能处理自定义数据类型。而且由于一般数据类型间的运算符没有重载的必要，所以运算符重载主要是面向对象之间的。

运算符重载的方法是定义一个重载运算符的函数，在需要执行被重载的运算符时，系统就自动调用该函数，以实现相应的运算。从形式上看，运算符重载是通过定义函数实现的，而实质上是函数的重载。

重载运算符的函数一般格式如下。

```
函数类型 operator 运算符名称（形参表列）
{
    // 对运算符的重载处理
}
```

（6） C++增加了标准模板库。

标准模板库（Standard Template Library，STL）是一个具有工业强度的、高效的 C++程序库。它被包含在 C++标准程序库（C++ Standard Library）中，是 ANSI/ISO C++标准中最新的，也是极具革命性的一部分。该库包含了许多在计算机科学领域里所常用的基本数据结构和基本算法，为广大 C++程序员们提供了一个可扩展的应用框架，高度体现了软件的可复用性。这种现象有点类似于 Microsoft Visual C++中的 MFC( Microsoft Foundation Class Library )，或者是 Borland C++ Builder 中的 VCL( Visual Component Library )。

STL 的一个重要特点是数据结构和算法的分离。这种分离使得 STL 变得非常通用。例如，STL 的 sort()函数是完全通用的，你可以用它来操作几乎任何数据集合，包括链表、容器和数组。

STL 的另一个重要特性是它不是面向对象的。但 STL 具有足够的通用性，它主要依赖于模板而不是封装、继承和虚函数（多态性）——OOP（Object-oriented Programming）的三个主要要素。在 STL 中找不到任何明显的类继承关系，这使得 STL 的组件具有广泛通用性的底层特征。另外，由于 STL 是基于模板，内联函数的使用使得生成的代码短小、高效。

从逻辑层次来看，在 STL 中体现了泛型化程序（Generic Programming）设计的思想，并引入了诸多新的名词，比如需求（Requirements）、概念（Concept）、模型（Model）、容器（Container）、算法（Algorithmn）、迭代子（Iterator）等。与 OOP 中的多态（Polymorphism）一样，泛型也是一种软件的复用技术。

从实现层次来看，整个 STL 是以一种类型参数化（Type Parameterized）的方式实现的，这种方式基于一个在早先 C++标准中没有出现的语言特性——模板（Templates）。考察任何一个版本的 STL 源代码，会发现模板是构成整个 STL 的基石。

## 1.5　小结

本章是对 C++、面向对象基础知识的导论，概要地介绍了 C/C++的发展史，通过一个五子棋程序实例的开发过程，展示了面向过程和面向对象在处理实际问题时的区别，以及面向对象分析模式所体现出来的优势和效率。

C++支持面向过程，也支持面向对象。抽象、继承、多态是 C++中面向对象理念的核心，也是本书后续章节内容的主体与重点。

相对于 C 语言，C++中一些新的机制，如类型检查、常类型、泛型编程、异常处理、运算法重载以及 C++所提供的标准模版库（STL），为编程程提供一个更加严格、完善的功能体系和强大的编程手段。

# 习 题 一

1. C++是何时产生的？它和C语言之间有何关系？
2. 面向过程和面向对象在处理实际编程问题时有何不同？
3. 面向对象的核心思想有哪些？
4. 相对于传统的C语言，C++引入了哪些新的特性？
5. 面向对象编程语言有哪些主要特点？
6. 什么是类？什么是对象？面向对象的方法的含义是什么？
7. 对比面向对象、面向过程以及结构化程序设计与非结构化程序设计的含义。

# 第2章 C++程序设计基础

**教学提示：**温故而知新，复习、巩固至提高。

**教学目标：**掌握 VC 的基本编辑环境，会使用标准 I/O 流对象进行简单的输入/输出，熟悉 C++基本语句结构。

本章主要介绍 C++程序设计的基础知识。“工欲善其事，必先利其器。”本章首先通过一个简单的 C++程序实例演示如何在 VC6.0 环境下编写 C++语言程序；接下来介绍构成 C++语言的基本元素——字符集、关键字、标识符等词法记号。程序是对数据集进行操控的代码的集合，本章与数据相关的基础知识主要包括基本数据类型、自定义数据类型以及数据的输入与输出等知识；与数据操控相关的基础知识主要包括程序的三种基本结构：顺序结构、选择结构、循环结构。

## 2.1 认识 C++程序

### 2.1.1 C++程序实例

【例 2.1】 一个简单的 C++程序。

```
1.  #include <iostream>
2.  using namespace std; //使用标准名空间
3.  void main(void )
4.  {
5.    cout<<"Hello World! "<<endl;
6.    cout<<"Welcome to C Plus Plus!"<<endl;
7.  }
```

运行结果：

```
Hello World !
Welcome to C Plus Plus!
```

上述程序中的第一行语句#include <iostream>使用 include 指令指示编译器对程序进行编译预处理时，将 iostream 头文件中的代码嵌入程序中该指令所在的位置。该文件中声明了 C++程序所需要的输入和输出操作的相关信息，使用 cout 对象的“<<”操作符来实现数据的输出。如果在编写的 C++程序中用 cin 与 cout 对象进行数据的输入与输出，需要在程序的第一行将 iostream 头文件使用 include 指令包含进来。第 2.3 节将简单介绍 C++的输入输出功能，其更详细的介绍参考第 11 章的内容。

程序的第二行语句 using namespace std 是命名空间的指令，表示本程序使用的命名空间是 std，这是一个标准的命名空间。命名空间相当于一个指定的作用域，系统会自动到命名空间对应的作

用域内去查找由 include 指令包含进来的文件，如本例中的 iostream 文件。有关命名空间的概念在后续的章节中会详细介绍。

从第三行代码开始是程序的主体，即一个名字为 main 的函数。一个 C++程序必须有一个全局命名空间中的 main 函数，表示程序执行的入口地址，该函数只能由系统调用，不能通过用户程序调用。如果 main 函数运行的结果没有返回值，则要在 main 函数前加上关键字 void。如果 main 函数运行时不需要参数，则需要在 main 函数后面的括号中加上关键字 void。如果 C++程序在运行时需要提供命令行参数，则一般形式如下。

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

其中 argc 表示在命令行中输入的参数个数，argv 是一个字符型的指针数组。数组的每个指针元素指向命令行中输入的参数在内存里的存储位置。

main 函数的函数体由一对 “{” 与 “}” 标识，函数体由多行 C++语句构成，每条语句以 “;” 结束。例 2.1 的 main 函数体中只有两条语句，分别使用 cout 对象的 “<<” 操作输出字符串 “Hello World!” 与 “Welcome to C Plus Plus!”，并换行；cout 对象可以利用 “<<” 操作连续进行输出操作，关键字 endl 表示输出换行。

通过文本编辑器将例 2.1 的程序录入并保存为后缀名为.CPP 文件之后（源程序文件），需要使用 C++编译程序将源程序文件编译成二进制目标文件，再经过链接生成可执行程序才可以运行。本书所有源程序文件都是在 VC6.0 环境下编译运行通过的。

### 2.1.2 字符集

字符集又称 “物理字符集”，是构成 C++程序的基本元素。一个 C++源程序在编译成目标程序时以一个个字符序列的形式被编译器程序读取，并映射为编译时的 “源字符集”。C++语言的字符集主要由下述几类字符构成。

（1）空格符：空格键，制表符，换行符，注释符（注释内容作为空格处理）

（2）英文字符：a……z，A……Z。

（3）数字字符：0……9。

（4）特殊字符：

```
_   {   }   [   ]   #   (   )   <   >   %   :   ;   .   ?
*   +   -   /   ^   &   |   ~   !   =   ,   \   "   '
```

### 2.1.3 词法记号

任何高级语言编写的源程序都是由一个个单词构成的文本文件，词法记号是构成 C++源程序的最小词法单元。词法分析程序与语法分析程序会查找对应的符号表判断源程序中的单词是否合法。C++中的词法记号主要包括关键字、标识符、常量、操作符、分隔符（标点符号）和空白等。

**1. 关键字**

关键字是系统已预定义的单词，它们在程序中表达特定的含义。C++中定义的主要关键字如下。

| | | | | | | |
|---|---|---|---|---|---|---|
| asm | do | if | return | typedef | auto | double |
| inline | short | typeid | bool | dynamic_cast | int | signed |
| typename | break | else | long | sizeof | union | case |
| enum | mutable | static | unsigned | catch | explicit | namespace |
| static_cast | using | char | export | new | struct | virtual |