普通高等教育“十一五”计算机类规划教材

C++程序设计与应用

C++ Chengxu Sheji Yu Yingyong

■ 周仲宁 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++程序设计与应用

主　编　周仲宁
副主编　曹晓军
参　编　杨海军　鹿　民　彭会萍
　　叶燕文　马　彦
主　审　沈　洁

机械工业出版社

本书以帮助读者掌握C++面向对象编程方法为目的，主要内容有语法基础、面向对象程序设计、流式输入输出、异常处理、多媒体编程、数据库编程、网络编程、多任务与多线程编程、容器和服务器、动态链接库、组件编程、活动模板库等。在参照98版C++标准的基础上，注重开发实例、开发经验、开发技巧是本书的特色。书中带有大量的代码实例，使读者不仅能够从理论上得以提高，而且能够轻松地在实践中应用。

本书是从入门到中高级程序设计人员的培训教材，可作为高等学校计算机相关专业学生的教材或参考书，也可供开发人员参考。本书配有免费电子课件，欢迎选用本书作教材的教师登录 www.cmpedu.com 下载或发邮件到 llm7785@sina.com 索取。

图书在版编目(CIP)数据

C++程序设计与应用/周仲宁主编. —北京：机械工业出版社，2009.2
普通高等教育“十一五”计算机类规划教材
ISBN 978-7-111-25686-1

Ⅰ.C… Ⅱ.周… 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187781号

机械工业出版社(北京市百万庄大街22号 邮政编码100037)
策划编辑：刘丽敏 责任编辑：刘丽敏 版式设计：霍永明
责任校对：闫玥红 封面设计：张 静 责任印制：李 妍
北京蓝海印刷有限公司印刷
2009年2月第1版第1次印刷
184mm×260mm ·28.5印张·707千字
标准书号:ISBN 978-7-111-25686-1
定价：45.00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：(010) 88379726
封面无防伪标均为盗版

前　言

C++是目前最流行的计算机程序设计语言之一，尽管早期的C++应用具有很强的系统程序设计色彩，但现在C++正被广大程序员应用到各个领域中，许多数值的、科学的以及工程的计算也是用C++完成的。自1998年C++语言国际标准推出后，C++开发环境逐步走向标准化，使程序的正确性不再受开发环境的影响。

C++之父Bjarne Stroustrup博士对C++语言的定义是："一种经过改进的更为优化的C语言，支持面向对象的程序设计，支持泛型程序设计"。C++语言是一门难学易用的语言，C++的难学，在于它提供了4种不同的程序设计思维模式：基于过程（Procedural - based）程序设计、基于对象（Object - based）程序设计、面向对象（Object - Oriented）程序设计和基于泛型的程序设计（Generic Paradigm）。本书试图从这4个方面阐述利用C++编程的基本方法和基本思想。

全书分两篇共21章，其中基础篇共12章，主要包括：C++语言概述、数据类型与表达式、C++程序的流程控制、函数、数组、引用和动态空间管理、类和对象的创建、类的继承、多态性、流类库和输入输出、异常处理、模板等。本部分以98版C++标准为蓝本，力争使读者领略结构化程序设计和面向对象程序设计的思想，掌握面向对象程序设计的基本方法，并对泛型的程序设计有所了解。

应用篇共9章，涵盖多个应用方向，目的是为后续课程的学习打下良好的编程基础，主要包括：多媒体编程、数据库编程、网络编程、多任务与多线程编程、容器和服务器、动态链接库、组件对象模型及ActiveX控件、活动模板库、开发案例等。本部分的特点是注重开发实例、开发经验、开发技巧和Windows高级应用。通过本书的学习，可以使读者掌握实际应用系统的开发方法过程，学会在Windows环境下开发出高水平的基于VC的应用程序。我们希望读者在阅读本书的过程中能够上机实践。读者每学完一个例子，可尝试着改变，或添加一点东西，并相应改变一些代码重新练习，这样将体验进步和成功的乐趣。

本书是从入门到中高级程序设计人员的培养教材，可作为高等学校计算机相关专业学生的教材或参考书，也可供应用开发人员学习参考。为方便读者，本书同时配有习题与实验指导书，供教师和学生自学选用，书中的代码及教学课件可登录出版社站点（www.cmpedu.com）下载。

本书由周仲宁任主编，曹晓军任副主编，杨海军、鹿民、彭会萍、叶燕文、马彦为参编，其中第1、11、15、17章由曹晓军执笔，第2、3、13、14章由彭会萍执笔，第4、7、21章由杨海军执笔，第5、6、8、12、20章由叶燕文执笔，第9、10、18、19章由马彦执笔，第16章由鹿民执笔，全书由周仲宁教授统稿。

本书由扬州大学沈洁教授任主审，他对全书进行了仔细的审阅，提出了许多极其宝贵的建议和意见，在此表示衷心的感谢！本书在编写中参考了有关的著作和文献，在此对这些文献的作者一并表示感谢！

限于作者水平，书中不当之处难免，请批评指正为盼！编者邮件地址是：zzn@lzcc.edu.cn。

编　者

目　录

基础篇

第 1 章　C++语言概述

1.1　概述

程序设计语言是人和计算机进行交流的工具，是计算机可以识别的语言，用于描述解决问题的算法，供计算机阅读和执行。

1.1.1　计算机语言的发展

第一代程序设计语言（1GL）是机器语言，直接使用 0 和 1 的序列作为机器指令来编程，程序能够直接被 CPU 执行，具有快速、高效的特点，但其缺点也十分明显，如难学、程序难写难读、编程效率低、代码的可移植性差等。随着计算机硬件结构越来越复杂，指令系统变得越来越庞大，为减轻程序设计人员的繁琐劳动，专家们开始用一些助记符来表示机器指令，产生了第二代程序设计语言（2GL），通常是指汇编语言。汇编语言代码容易读写，由于它是一个处理器的本机语言，有明显的速度优势，但运行程序前必须进行汇编语言到机器代码的映射，并且汇编语言程序依赖于特定的处理器家族和环境。汇编语言是机器语言的直接符号表示，所以也是低级语言，其语句功能不强，因此编写一个较大的程序仍然很繁琐，而且对机器的依赖性非常强。为了克服这些缺点，在 20 世纪 50 年代中期出现了与自然语言和数学语言更接近的、更容易被人们理解的第三代程序设计语言（3GL），即高级语言，如 1954 年由 John Backus 领导完成的主要面向专业人员的 FORTRAN 语言，1959 年 Grace Murray Hopper 博士领导的一个委员会设计的面向商业的通用语言 COBOL，1970 年 AT&T 的 Bell 实验室的 D. Ritchie 和 K. Thompson 为编写 UNIX 的系统程序共同发明的 C 语言等。其中 C 语言充分结合了汇编语言和高级语言的优点，高效而灵活，又容易移植，所以被广泛使用。第四代语言（4GL）是比较典型的高级语言，它更接近于人类语言，是一种非过程化的语言，加快了软件的发展过程。大部分 4GL 用于访问数据库，一个典型的 4GL 命令是：FIND ALL RECORDS WHERE COMPANY IS “JAVVIN”。第五代程序设计语言（5GL）使用赋予这个语言的约束、而不是使用由一个程序员写的算法、立足于解决问题的一种程序设计语言。大部分基于约束的逻辑程序设计语言和一些陈述性的语言是第五代语言。第五代语言主要应用在人工智能研究上，Prolog、OPS5 和 Mercury 是最知名的第五代语言。

1.1.2　C++语言

从本质上来说，C++语言是从 C 语言中继承而来的。C++在延续 C 语言的全部特征、属性和优点的基础上，添加了对面向对象编程（Object Oriented Programming，OOP）的完全支持。

1. C++的产生

由于 C 语言丰富的运算符和数据结构、程序执行效率高、语言简单灵活、可直接访问

计算机的物理地址、有大量的库代码和较多的开发环境、具有良好的可读性和可移植性，以及支持结构化程序设计等特点，产生之后成为使用最广泛的程序设计语言之一。但C语言也存在一些局限：如C语言相对较弱的类型检查机制使编译器无法检查到程序中的一些错误、语言结构几乎不支持代码复用、大规模程序开发中程序员很难控制程序的复杂性等。为此程序语言的设计专家们力图改变软件设计过程中的问题，提出程序设计语言应该具有数据类型的扩展能力。

1979年，贝尔实验室的Bjarne Stroustrup借鉴Simula中“Class”的概念，开始研究增强C语言使其支持面向对象的特性。为了使它能够尽快投入使用，B. Stroustrup写了一个转换程序“Cfront”把C++代码转换为普通的C代码，1983年正式取名为C++。到1986年，当B. Stroustrup出版《The C++Programming Language》时，C++已经开始受到关注。1988年第一个真正的C++编译系统诞生。1989年推出的C++2. 0版本对第1版作了重大的改进，引进了类的继承机制。1991年的C++3.0版本增加了模板，C++4.0版本则增加了异常处理、命名空间、运行时类型识别（RTTI）等功能。

C++的特点主要表现为4个方面：第一，C++完全包含C，在保持了C的简洁、高效和直接访问计算机的物理地址等优点的基础上，对C的类型系统进行了改革和扩充，比C更可靠，是更好的C，许多C代码不经修改就可为C++使用，用C编写的众多库函数和实用软件可以直接用在C++中。第二，用C++编写的程序可读性好，代码结构合理，可直接在程序中映射问题空间的结构。第三，C++生成的代码质量高，软件的可重用性、可扩充性、可维护性和可靠性等方面有更大提高，使大中型程序的开发变得更容易。第四，C++支持面向对象的机制，可方便地构造出模拟现实问题的实体和操作。

需要说明的是，由于C++与C保持兼容，使得C++不是一个纯正的面向对象的语言，被称为混合型程序设计语言。

2. C++的标准化

C++的诞生是软件史上一大进步，但在产生初期，由于没有标准可依，在开发C++编译器时，导致了无序化、大量重复劳动等问题。1989年，ANSI挂牌成立X3J16负责C++的标准化。由于当时还没有C++标准，1990年B. Stroustrup出版的《The Annotated C++Reference Manual》（简称ARM）成了事实上的标准。1991年，国际标准化组织（ISO）在其技术委员会下成立了负责C++语言标准化的工作组ISO/IECJTC1/SC22/WG21，开始进行C++语言标准化的工作。1993年，运行时类型识别（RTTI）和名字空间（Namespace）加入到C++中。1994年，C++标准草案出台，但之后标准模板库（STL）建议草案的提交又一次推迟了C++标准的出台。直到1998年，ANSI和ISO终于先后批准C++语言成为美国国家标准和国际标准。C++标准通过以后，ANSI和ISO开始酝酿新版本的C++标准。

标准化的目的是为语言和库制定一个规范，使其能够更好地服务于所有用户群体，而不至于偏向某个用户群、某个公司或某个国家。这是一个以保证质量和达成共识为目的的开放、公正的过程。ISO的C++标准（ISO/IEC 14882 Standard for the C++Programming Language）的通过，是C++发展史上的一个里程碑，开创了C++工具和技术稳定发展的新纪元。更重要的是，C++标准不仅仅是一份文档，它已经在各种C++实现产品中得到了体现，所有主要的C++实现产品现在都实现了标准化，只有几个例外。

3. C++的使用

早期的C++应用趋向于具有很强的系统程序设计色彩，例如，有几个主要的操作系统是在C++里写出的，许多系统用C++实现了其中的关键部分。现在C++被数以十万计的程序员应用到各个领域中，例如银行、贸易、保险业、远程通信以及军事。多年来，美国的长途电话系统的核心控制依赖于C++，所有的800业务（被叫方付款电话）都由一个C++程序控制路由。许多这样大规模的应用长期运行着。所以，稳定性、兼容性和可伸缩性都成为C++开发中被始终关注的问题。另外，还有许多数值的、科学的以及工程的计算也是在C++中完成的。从TIOBE公布的最流行开发语言排名中可以看到，2007年C++排名第三，有10.768%的程序员在使用C++作为开发工具。

4. 学习C++

学习编程的唯一秘诀是编程，在编程过程中不要放过任何一个看上去很简单的小问题，因为这些问题往往可能引申出很多知识点，一旦写程序写到一半发现自己用的方法很拙劣，请尽快将余下的部分粗略的完成而不是马上停手，以保证整个设计的完整性，然后再分析自己的错误并重新设计和编写，同时要注意保存好写过的所有程序，因为那是最好的积累。

C++语言和C++的集成开发环境要同时学习和掌握，但不要被VC、BC、MC、TC等词汇所迷惑，应该选择一个确定的集成开发环境使用而不是每天更换工具，这样有助于将重点放在对C++语言的学习上，一定阶段后，要学会控制集成开发环境，还要学会用命令行方式处理程序。

学习C++的第一个难点在于其编译器在幕后做了大量的工作。传统语言如C、Basic、Fortran等基本上都是通过函数调用完成程序功能的，你想做的工作，在程序代码中看得一清二楚，编译器仅仅为你的函数加上用以处理堆栈的一小段代码，而这段代码并不影响你对程序逻辑的思考。但在C++中，程序功能的实现，除了程序人员编写的代码外，与程序逻辑相关的代码很多工作都由编译器幕后完成，如对象诞生时构造方法的调用代码、对象消亡时析构方法的调用代码等，可以说，程序代码中看不到而却必须完成的所有与程序逻辑有关的动作，统统都是C++编译器完成的。当使用继承、多重继承、虚拟继承时，对对象成员的访问到程序运行时才能确定，所有这些问题都可归结为C++的对象模型。如果不知道这些底层机制，就只能把“将基类的析构函数设计为虚函数”、“不要使用多态性数组”等这类规则硬背下来，却不明白它的道理。

学习C++的第二个难点在于思维模式的转换。设计类，甚至使用类，都存在对思维模式和行为模式的转换问题。如在MFC中，使用了应用程序框架（一种大型的、凝聚性强的、有着对象向导公共基础设施的类库），框架提供的一大堆可改写的虚函数的意义与价值是什么？框架设计时，并不知道用户类的存在，为什么框架设计的属性以及方法可以施加于用户设计的类之上？这一系列问题的答案其实都可归结为C++的多态（Polymorphism）和泛型（Generalization），C++的这两项特性能够把新思维模式的威力发挥得淋漓尽致。

1.2 程序设计概述

计算机是通过执行指令完成特定任务的。通常所说的程序就是为完成某一特定任务的一系列指令的集合。高级语言通过语句来描述指令，故本书所称的“程序”，是指为完成某一

任务而组合在一起的一组语句的集合。程序设计（Programming）则是指设计、编制、调试程序的方法和过程。

早期的计算机存储器容量非常小，人们设计程序时首先考虑的问题是如何减少存储器开销，硬件的限制不容许人们考虑如何组织数据与逻辑，程序本身短小，逻辑简单，也无需人们考虑程序设计方法问题。与其说程序设计是一项工作，倒不如说它是程序员的个人技艺。随着大容量存储器的出现及计算机技术的广泛应用，程序的大小以算术基数递增，而程序的逻辑控制难度则以几何基数递增，人们不得不考虑程序设计方法。先后产生了结构化（面向过程）的及面向对象的程序设计方法。20 世纪 70 年代以前，程序设计方法主要采用流程图，结构化设计（Structure Programming，SP）思想也趋成熟，整个 20 世纪 80 年代 SP 是主要的程序设计方法。然而，随着信息系统的加速发展，应用程序日趋复杂化和大型化，传统的软件开发技术难以满足发展的新要求。20 世纪 80 年代后，面向对象的程序设计（Object Orient Programming，OOP）技术日趋成熟并逐渐地为计算机界所理解和接受。程序设计的方法正在从结构化程序设计走向面向对象的程序设计。经过多年的实践摸索，人们发现许多软件系统并不是都能完全按系统的功能来划分构件，许多重要的需求和设计决策，如安全、日志等，它们具有一种“贯穿特性”（Crosscutting Concerns），无论是采用面向对象语言还是过程型语言，都难以用清晰的、模块化的代码实现，于是面向方面程序设计、面向 Agent 程序设计等许多“后面向对象程序设计”方法正在逐步形成。

1.2.1 结构化程序设计

最早提出的方法是结构化程序设计方法，其核心是模块化。1968 年 Dijskstra 发表文章首次提到了“结构化程序设计”，至 1975 年起，在业界的共同努力下，结构化程序设计逐步形成既有理论指导又有切实可行方法的一门独立学科。SP 方法主张使用顺序、选择、循环 3 种基本结构来嵌套连接成具有复杂层次的“结构化程序”，严格控制 GOTO 语句的使用。用这样的方法编出的程序在结构上具有以下效果：

1）以控制结构为单位，具有单入口、单出口。

2）能够以控制结构为单位，从上到下顺序地阅读程序文本。

3）由于程序的静态描述与执行时的控制流程容易对应，所以能够方便正确地理解程序的动作。

结构化程序设计的要点是：“自顶而下，逐步求精”的设计思想，“独立功能，单入口、单出口”的模块仅用3 种基本控制结构的编码原则。自顶而下的出发点是从问题的总体目标开始，抽象低层的细节，先专心构造高层的结构，然后再一层一层地分解和细化。这使设计者能把握主题，高屋建瓴，避免一开始就陷入复杂的细节中，使复杂的设计过程变得简单明了，过程的结果也容易做到正确可靠。独立功能，单入口、单出口的模块结构减少了模块的相互联系，使模块可作为插件或积木使用，降低程序的复杂性，提高可靠性。程序编写时，所有模块的功能通过相应的子程序（函数或过程）的代码来实现。程序的主体是子程序层次库，它与功能模块的抽象层次相对应，编码原则使得程序流程简洁、清晰，增强可读性。

当然模块划分时不应随心所欲地把整个程序简单地分解成一个个程序段，而必须按照一定的方法进行。模块的根本特征是“相对独立，功能单一”。换言之，一个好的模块必须具有高度的独立性和相对较强的功能。模块的好坏，通常用“耦合度”和“内聚度”两个指

标从不同侧面来加以度量。所谓耦合度，是指模块之间相互依赖性大小的度量，耦合度越小，模块的相对独立性越大。所谓内聚度，是指模块内各成分之间相互依赖性大小的度量，内聚度越大，模块内各成分之间联系越紧密，其功能越强。因此模块划分应当做到“耦合度尽量小，内聚度尽量大”。

结构化程序相比于非结构化程序有较好的可靠性、易验证性和可修改性；结构化设计方法的设计思想清晰，符合人们处理问题的习惯，易学易用，模块层次分明，便于分工开发和调试，程序可读性强。其设计语言以 Ada 语言为代表，C 语言也能够很好地支持结构化程序设计。

1.2.2 面向对象的程序设计

20 世纪 90 年代，由于计算机硬件的飞速发展，对软件系统在规模和性能方面的要求也在不断的提高，传统的软件工具、软件技术和抽象层次越来越难以适应大规模复杂软件系统的开发特点，结构化程序设计方法也使得软件和硬件能力的差距迅速扩大。为了解决这些问题，20 世纪 70 年代中后期逐步产生了 OOP 的思想，并于 20 世纪 80 年代逐步代替了传统的 SP 方法，成为最重要的方法之一，至今 OOP 方法被广泛应用于各个领域。

面向对象的基本思想与结构化设计思想完全不同，面向对象的方法学认为世界由各种对象组成，任何事物都是对象，是某个对象类的实例，复杂的对象可由较简单的对象的某种方式组成。OOP 的基石是对象和类。对象是数据及对这些数据施加的操作结合在一起所构成的独立实体的总称；类是一组具有相同数据结构和相同操作的对象的描述。面向对象的基本机制是方法和消息，消息是要求某个对象执行类中某个操作的规格说明；方法是对象所能执行的操作，它是类中所定义的函数，描述对象执行某个操作的算法，每一个对象类都定义了一组方法。

OOP 有 3 个重要特性：封装性、继承性和多态性。封装性是指对象是数据和处理该数据的方法所构成的整体，外界只能看到其外部特性（消息模式、处理能力等），其内特性（私有数据、处理方法等）对外不可见。对象的封装性使得信息具有隐蔽性，它减少了程序成分间的相互依赖，降低了程序的复杂性，提高了程序的可靠性和数据的安全性。继承性（Inheritance）反映的是类与类之间的不同抽象级别，根据继承与被继承的关系，可分为基类和派生类，基类也称为父类，派生类也称为子类，子类从父类那里获得所有的属性和方法，并且可以对这些获得的属性和方法加以改造，使之具有自己的特点。继承性使得相似的对象可以共享程序代码和数据，继承性是实现程序可重用性的关键。多态性是指在形式上表现一样的方法，根据传递给它的参数的不同，可以调用不同的方法体，实现不同的操作。将多态性映射到现实世界中，则表现为同一个事物随着环境的不同，可以有不同的表现形态及不同的和其他事物通信的方式。多态性使程序员能在一个类层次中使用相同函数的多个版本，程序员可以集中精力开发可重用的类和方法而不必过分担心名字的冲突问题。

OOP 方法是以“对象”为中心进行分析和设计，紧抓“模型化世界”的对象，使这些对象形成了解决目标问题的基本构件，即解决从“用什么做”到“要做什么”。其解决过程从总体上说是采用自底向上方法，先将问题空间划分为一系列对象的集合，再将对象集合进行分类抽象，一些具有相同属性行为的对象被抽象为一个类，类还可抽象分为子类、超类（超类是子类的抽象）。其间采用继承来建立这些类之间的联系，形成结构层次。同时对于

每个具体类的内部结构，又可采用自顶向下逐步细化的方法由粗到细精化之。调试运行时通过向类对象发消息来完成，对象执行相应操作并返回结果，使对象集的初始状态变成了终态。故 OOP 总体来说主要是不断设计新的类和创建对象的过程。

OOP 方法具有许多优点，以“对象”为中心的设计方式能够再现人类认识事物和解决问题的方式；OOP 方法以对象为唯一的语义模型，整个软件任务是通过各对象（类）之间相互传递消息的手段协同完成，能尽量逼真地模拟客观世界及其事物；由对象和类实现了模块化，类继承实现了抽象对象，对象的内部状态和功能实现的细节对外都是不可见的，因此很好地实现了信息隐藏。建立在类及其继承性基础上的重用能力可应付复杂的大型软件开发。面向对象方法使得软件具有良好的体系结构、便于软件构件化、软件复用和软件的扩展和维护，抽象程度高，因而具有较高的生产效率。目前，面向对象程序设计语言以 Java、C++为代表，C++不仅能够很好地支持结构化程序设计，而且能够很好地支持面向对象的程序设计。

1.3 简单的 C++程序

1.3.1 程序开发过程

C++是一种编译语言，一般来讲，编译语言的源程序需要经过编译和连接两个步骤，在生成可执行文件后方可运行。使用 C++开发一个应用程序大致要经过以下步骤：

1）首先要根据实际问题确定编程思路，包括程序设计方法的选择以及解决问题的模型选择。

2）根据前述思路或解决问题的模型编写程序。除了非常简单的问题可以直接写出相应的 C++程序之外（在值得使用计算机解决的应用问题中，这种情况并不多），一般都应该根据选定的程序设计方法进行编程。

3）编辑源程序。首先应将源程序输入计算机，由于 C++的源程序是以纯文本的方式存储的，故这项工作可以通过任何一种文本编辑器（如 Visual C++集成环境中的文本编辑器）完成。输入的源程序一般以文件的形式存放在磁盘上（后缀为 CPP）。

4）编译和连接。在设计高级语言（包括 C++）时充分考虑了人（程序员）的需要，源程序很接近人类的自然语言。因此，需要将源程序转换为计算机可直接执行的指令。就 C++而言，这项工作又分编译和连接两个步骤，编译阶段将源程序转换成目标文件（后缀为 OBJ），连接阶段将目标文件连接成可执行文件（后缀为 EXE）。另外，这一步会涉及到编译器的选择问题。

5）上机反复调试程序，直到改正了所有的编译错误和运行错误。在调试过程中应该精心选择典型数据进行试算，避免因调试数据不能反映实际数据的特征而引起计算偏差和运行错误。

6）运行。如果是自用程序，在调试通过以后即可使用实际数据运行程序，得到计算结果；如果是商品软件或受委托开发的软件，则运行由用户实施。

1.3.2 一个简单的 C++程序

【例 1.1】 一个简单的 C++程序。

```
//*****************************************
//*  输出一行字符                          *
//*****************************************
#include <iostream>                    //用 cout 输出时需要用此头文件
using namespace std;                   //使用命名空间 std
int main()                             //程序入口
{                                      //函数体开始
  cout << "Hello,new world! \n";       //用 C++的方法输出一行
  return 0;                            //如果程序正常结束，向操作系统返回一个 0 值
}
```

程序运行结果：Hello, new world!

从程序中可以看到，C++的程序结构由注释、编译预处理和程序主体组成。具体说明如下：

1）注释是程序员为程序作的说明，是提高程序可读性的一种手段。一般分为两种：序言注释和注解性注释。前者用于程序开头，说明程序或文件的名称、用途、编写时间、编写人及输入输出说明等，后者用于在程序较难懂的地方。在 C++程序中，可以使用以“/* ……* /”形式的注释，它可以占用多行，也可以使用以“//”开头的注释，但它是单行注释。C++的注释不能嵌套。

2）每个以“#”开头的行称为编译预处理行。如“#include”称为文件包含预处理命令。“#include <iostream>”的作用是通知编译器在编译之前将文件“iostream. h”的内容装入到程序中。iostream 是“输入输出流”的意思，由于这类文件的使用都放在程序单元的开头，所以称为“头文件”（Head File），扩展名为“. h”。

3）程序第二行的“using namespace std;”的意思是“使用命名空间 std”。C++标准库中的类和函数几乎都是在命名空间 std 中声明的，因此程序中如果需要使用 C++标准库中的有关内容，就需要用“using namespace std;”语句做声明，表示要用到命名空间 std 中的内容。

4）main( )表示主函数，每一个 C++程序都必须有一个 main( )函数，main( )是程序的入口，一般在其前面加一个类型声明符 int，表示 main 函数的返回值为整型（标准 C++规定，main 函数必须声明为 int 性，即此函数将带回一个整型的函数值）。程序第 5 行的作用是返回 0。大部分时候，如果函数正常执行结束，其返回值为 0，否则返回一个非 0 值，如返回 -1。

5）在 C++程序中，一般用 cout 进行输出。cout 实际上是 C++系统定义的对象名，称为输出流对象。“<<”是插入运算符，与 cout 配合使用，其作用是将“<<”右边引号内的字符串“Hello, new world! \ n”插入到输出的队列 cout 中，C++系统将输出流 cout 的内容输出到标准输出设备（一般为显示器）上。为方便理解，把 cout 和“<<”实现输出的语句简称为 cout 语句。C++中，除了可以使用 cout 进行输出外，还可以使用 printf 函数输出。

6）C++语句必须以“;”结束。

【例 1.2】 输出两个数的和。

```
//定义函数，求两数之和
#include <iostream>
using namespace std;
int main( )
{
  int sum(int,int);                    //函数原型声明
  int a, b, s;                         //变量声明
  cin >> a >> b;                       //输入 a 和 b
  s = sum(a,b);                        //计算和
  cout << "a + b = " << s << endl;      //输出结果
  return 0;
}
int sum(int x,int y)
{
  return x + y;
}
```

本程序的作用是求两个整数 a 与 b 的和 s。具体说明如下：

1）程序第 6 行“int a, b, s;”是声明部分，指定变量 a、b、s 为整型（int）变量。

2）第 7 行“cin >> a >> b >> endl;”是输入语句，与 cout 类似，cin 是 C++系统定义的输入流对象。“>>”是“提取运算符”，与 cin 配合使用，其作用是从标准输入设备（如键盘）中提取数据到输入流 cin 中。用 cin 和“>>”实现输入的语句简称为 cin 语句。在执行 cin 语句时，从键盘输入的第 1 个数据赋给了整型变量 a，输入的第 2 个数据给 b。

3）第 8 行调用函数 sum 求 a 与 b 的和，并将返回值赋给整型变量 s。

4）第 9 行先输出字符串“a + b =”，然后输出变量 s 的值。后面的 endl 是 C++输出时的控制符，作用是换行（endl 是 end line 的缩写，表示本行结束，与“\n”相同）。

5）第 10 行开始定义了函数 sum。

【例 1.3】 一个包含类的简单的 C++程序。

```
//在程序中使用类
#include <iostream>
using namespace std;
class Goods                            //声明一个类，类名为 Goods
{
  private:                             //在做新的说明前，以下声明的都将是类的私有成员
        int num;                       //私有整型成员变量 num
        int amount;                    //私有整型成员变量 amount
        int price;                     //私有整型成员变量 price
  public:                              //在作新的说明前，以下声明的都将是类的公有成员
        void goodsdata( )              //定义公有成员函数 goodsdata，void 表示无返回值
  {
```

```
        cin >> num;
        cin >> amount;
        cin >> price;
    }
    void show( )                    //定义公有函数 show
    {
        cout << "num:" << num << endl;
        cout << "amount:" << amount << endl;
        cout << "price:" << price << endl;
    }
};                                  //类的声明结束
int main( )                         //主函数
{
    Goods goods1, goods2;           //定义 Goods 类的变量 goods1 和 goods2，称为对象
    goods1. goodsdata( );           //调用对象 goods1 的 gooodsdata 函数
    goods2. goodsdata( );           //调用对象 goods2 的 goodsdata 函数
    goods1. show( );                //调用对象 goods1 的 show 函数
    goods2. show( );                //调用对象 goods2 的 show 函数
    return 0;
}
```

本例程的目的是说明 C++程序设计中类的使用方法，细节性问题的讨论在本书的第 7 章，本处只做简单说明如下：

1）当使用面向对象的方法设计程序时，首先要声明（或派生）自己的类，使用 class 关键字完成类的声明，本例中取类名为 Goods。

2）C++的类由一系列数据成员和成员函数构成，本例中 num、amount 和 price 是数据成员，而 goodsdata 和 show 则是类 Goods 的成员函数。

3）为实现封装，C++类的成员声明时要指定访问控制属性，常用 private、public 和 protected 关键字指定私有、公有和保护。私有成员只限本类的成员函数访问，公有成员又称为接口，外界通过访问这些接口而实现其他成员的访问，保护成员用于类的继承时。

4）如同本例中看到的一样，在面向对象程序中，主函数所完成的工作是实例化对象，并通过实例化的对象访问接口来实现程序的功能，C++中，消息的传递也是通过访问接口实现的。

习　题　1

1. C++语言有哪些主要特点？C++程序的结构是怎样的？
2. 简述计算机程序设计语言的发展阶段。
3. 简述结构化和面向对象程序设计的基本思想和它们之间的联系与区别。
4. 简述程序开发的步骤和运行过程。