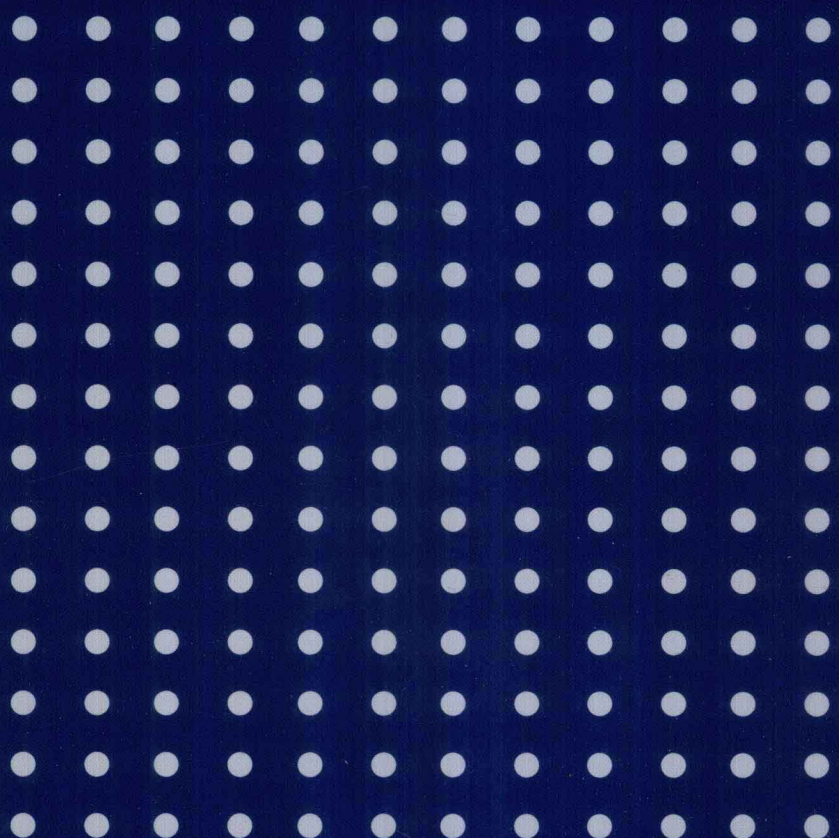
重点大学计算机专业系列教材

C++语言程序设计教程

蒋光远 田琳琳 赵小薇 于红 编著



清华大学出版社

内 容 简 介

C++是一种混合型的程序设计语言，支持面向过程与面向对象的程序设计方法。本书分别介绍面向过程的C++基础、面向对象的C++语言要素和应用C++开发的其他机制。面向过程部分总结C++面向过程的语法点，介绍数据类型、流程控制、函数、数组及指针，巩固基础知识的同时，对C++中引进的流、重载、引用、动态空间管理进行较为详尽的讲解。面向对象部分重点阐述面向对象思想，分析类、运算符重载、继承、多态和流等语法要素，通过浅显的例子解释知识点的意义与用法，对重点与难点语法采用大量的实例和图表来帮助理解，使读者能"知其然"，并能做到"知其所以然"。应用基础部分介绍应用C++编程的关键技术与高级机制，包括模板、STL、异常以及Windows编程，由于该部分涉及内容很多，采用向导式进行分析案例，使读者在简单应用中理解语法机制。本书注重案例设计的合理性，引导读者理解并应用面向对象程序设计的思想方法，从应用出发注重激发读者的学习兴趣。

图书在版编目(CIP)数据

C++语言程序设计教程/蒋光远编著. —北京：清华大学出版社，2012.1
(重点大学计算机专业系列教材)
ISBN 978-7-302-27500-8

Ⅰ. ①C… Ⅱ. ①蒋… 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(2011)第260584号

责任编辑：梁　颖　薛　阳
责任校对：白　蕾
责任印制：何　芊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京密云胶印厂
经　　销：全国新华书店
开　　本：185×260　**印　张**：21.25　**字　数**：518千字
版　　次：2012年1月第1版　　**印　　次**：2012年1月第1次印刷
印　　数：1～3000
定　　价：33.00元

产品编号：044810-01

INTRODUCTION

出版说明

随着国家信息化步伐的加快和高等教育规模的扩大，社会对计算机专业人才的需求不仅体现在数量的增加上，而且体现在质量要求的提高上，培养具有研究和实践能力的高层次的计算机专业人才已成为许多重点大学计算机专业教育的主要目标。目前，我国共有16个国家重点学科、20个博士点一级学科、28个博士点二级学科集中在教育部部属重点大学，这些高校在计算机教学和科研方面具有一定优势，并且大多以国际著名大学计算机教育为参照系，具有系统完善的教学课程体系、教学实验体系、教学质量保证体系和人才培养评估体系等综合体系，形成了培养一流人才的教学和科研环境。

重点大学计算机学科的教学与科研氛围是培养一流计算机人才的基础，其中专业教材的使用和建设则是这种氛围的重要组成部分，一批具有学科方向特色优势的计算机专业教材作为各重点大学的重点建设项目成果得到肯定。为了展示和发扬各重点大学在计算机专业教育上的优势，特别是专业教材建设上的优势，同时配合各重点大学的计算机学科建设和专业课程教学需要，在教育部相关教学指导委员会专家的建议和各重点大学的大力支持下，清华大学出版社规划并出版本系列教材。本系列教材的建设旨在“汇聚学科精英、引领学科建设、培育专业英才”，同时以教材示范各重点大学的优秀教学理念、教学方法、教学手段和教学内容等。

本系列教材在规划过程中体现了如下一些基本组织原则和特点。

(1) 面向学科发展的前沿，适应当前社会对计算机专业高级人才的培养需求。教材内容以基本理论为基础，反映基本理论和原理的综合应用，重视实践和应用环节。

(2) 反映教学需要，促进教学发展。教材要能适应多样化的教学需要，正确把握教学内容和课程体系的改革方向。在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养，为学生知识、能力、素质协调发展创造条件。

(3) 实施精品战略，突出重点，保证质量。规划教材建设的重点依然是专业基础课和专业主干课；特别注意选择并安排了一部分原来基础比较好的优秀教材或讲义修订再版，逐步形成精品教材；提倡并鼓励编写体现重点大学

计算机专业教学内容和课程体系改革成果的教材。

(4) 主张一纲多本,合理配套。专业基础课和专业主干课教材要配套,同一门课程可以有多本具有不同内容特点的教材。处理好教材统一性与多样化的关系;基本教材与辅助教材以及教学参考书的关系;文字教材与软件教材的关系,实现教材系列资源配套。

(5) 依靠专家,择优落实。在制订教材规划时要依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主编。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平的以老带新的教材编写队伍才能保证教材的编写质量,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

教材编委会

FOREWORD

前言

C++语言是一种典型的面向对象的程序设计语言，学习C++程序设计语言既要掌握其语法规则，更要理解面向对象(Object-Oriented，OO)的程序设计思想。只有在理解OO思想的基础上运用这些语法才能编写出真正的C++程序，才能够为后续专业课程(如“数据结构”、“编译原理”、“操作系统”和“软件工程”等)的学习提供支持，从而为软件开发工作奠定扎实的基础。

在笔者多年教学实践过程中发现，学生对C++面向对象程序设计的学习往往偏重于基本语法，忽略理解和掌握面向对象的程序设计思想。主要表现是：设计程序以完成基本功能为出发点，仍然采用结构化思想设计程序；尽管程序中定义了类，但没有体现封装、继承、多态的作用，构造的是基于对象而不是面向对象的程序。

C++是一个非常全面的程序设计语言，不仅具备面向对象的常规语言要素，如类、继承、多态、流、异常机制等，还包括诸多C++特有的语言要素，如多继承、复制构造、运算符重载、指针、引用、模板等。由于涉及的语法规则繁多且晦涩难懂，学生很难完全掌握，因此容易导致其畏难情绪。此外，C++教学往往与具体应用脱节，学习语法知识后学生不了解其应用方法，对应用程序开发无所适从。

基于以上问题，本书本着“理解与应用并重”的原则，强调案例设计的合理性，引导读者理解并应用面向对象程序设计的思想和方法，从应用出发培养学生的学习兴趣。在讲解基本语法规则之前，先通过浅显的例子帮助读者理解该知识点的本质，正所谓“知其然更需知其所以然”，进而使读者能够合理地规划程序结构并运用知识点。对重要的、难懂的知识点采用实用案例进行循序渐进的剖析，并引入大量简洁易懂的图表来帮助理解。将基础知识与标准模板库等相结合，使学生懂得利用已有的模板库和算法，能够提高程序的开发效率和可靠性，为实际研发打下基础。同时，为了培养学生学习兴趣，引入Windows编程部分，采用向导式介绍，让学生能够了解Windows程序设计的思路和应用，进一步增强对面向对象程序设计的理解。

本书由大连理工大学软件学院软件基础教研室组织编写，在总结各位教师多年教学经验的基础上，倾注了C++教学团队教师大量的心血。其中，由

蒋光远完成第1章、第2章、第3章的编写；田琳琳完成第4章、第9章以及附录的编写；赵小薇完成第5章、第6章、第8章的编写；于红完成第7章、第10章的编写。

这是一本主要面向研究型和教学型大学，针对计算机及相关专业的“C++程序设计语言”课程的教材，建议读者最好有一定的C语言程序设计基础。希望读者在学习C++语言语法的同时，能够真正理解和掌握面向对象程序设计的思想，并运用OO的分析与设计方法开发应用程序。

鉴于时间仓促，笔者水平有限，书中难免有纰漏，欢迎广大读者多提宝贵意见。

编　者

2011年12月于大连理工大学

CONTENTS

目录

CHAPTER 1

概　述　第1章

1.1　面向对象的由来

20世纪70年代流行的面向过程的程序设计方法，其目的主要是解决面向过程语言系统所存在的一些问题。它主要强调程序的模块化和自顶向下的功能分解。在涉及大量计算的算法类问题上，从算法的角度揭示事物的特点，面向过程对问题的分割是合理的。随着计算机硬件技术的飞速发展，计算机的容量、速度迅速提高，计算机取得了越来越广泛的应用，涉及社会生活的方方面面。面对变动的现实世界，面向过程的程序设计方法暴露出越来越多的不足，这就对软件开发提出了更高的要求。然而软件技术的发展却远远落后于硬件技术的进步，人们常常无法控制软件开发的周期和成本，软件的质量总是不尽如人意，经常是用之不灵、弃之可惜，有的软件甚至无法交付。

在面向过程的程序设计方法中存在以下问题。

(1) 基于模块的设计方式，导致软件修改困难。

(2) 功能与数据分离，不符合人们对现实世界的认识。要保持功能与数据的相容也十分困难。

(3) 自顶向下的设计方法，限制了软件的可重用性，降低了开发效率，也导致最后开发出来的系统难以维护。

为了解决结构化程序设计的这些问题，面向对象的技术应运而生。它是一种非常强有力的软件开发方法。它将数据和对数据的操作作为一个相互依赖、不可分割的整体，采用数据抽象和信息隐蔽技术，力图使对现实世界问题的求解简单化。它符合人们的思维习惯，同时有助于控制软件的复杂性，提高软件的生产效率，从而得到了广泛的应用，已成为目前最流行的一种软件开发方法。

面向对象技术产生的背景与结构化程序设计方法产生的背景类似，面向对象程序设计方法(OOP)是在结构化程序设计方法的基础上发展而来的。

20世纪60年代开发的Simula 67，是面向对象语言的鼻祖，它第一次提

出了对象的概念。20 世纪 60 年代中后期，Simula 语言在 ALGOL 基础上研制开发，它将 ALGOL 的块结构概念向前发展了一步，提出了对象的概念，并使用了类，也支持类继承。20 世纪 70 年代，Smalltalk 语言诞生，它吸取 Simula 的类为核心概念，它的很多内容借鉴于 Lisp 语言。由 Xerox 公司经过对 Smalltalk-72、Smalltalk-76 持续不断地研究和改进之后，于 1980 年推出商品化的版本，它在系统设计中强调对象概念，引入对象、对象类、方法、实例等概念和术语，采用动态联编和单继承机制。Smalltalk 语言是最有代表性、最有影响的面向对象语言，它丰富了面向对象的概念，实现了面向对象技术的机制。

从 20 世纪 80 年代起，基于以往已经提出的有关信息隐蔽和抽象数据类型等概念，以及由 Modula2、Ada 和 Smalltalk 等语言所奠定的基础，再加上客观需求的推动，人们进行了大量的理论研究和实践探索，不同类型的面向对象语言（如 Object-C、Eiffel、C++、Java、Object- Pascal 等）、面向对象程序设计方法广泛应用于程序设计，全新的面向对象的程序设计语言被开发出来。由此逐步地发展和建立起较完整的 OO 方法概念理论体系和实用的软件系统。

C++就是一个支持面向对象的程序设计语言，C++是 C 语言的超集，C++对 C 语言的最大改进是引进面向对象机制，同时 C++依然支持所有 C 语言特性，保留对 C 语言的兼容，这种兼容性使得 C++不是一种纯正的面向对象的程序设计语言。

1.2 面向对象的思想

结构化方法的本质是功能分解，从代表目标系统整体功能着手，自顶向下将一个大的问题按功能划分为一些小的功能模块，直到仅剩下若干个容易实现的子功能为止，然后用相应的工具来描述各个最低层的功能模块。在很长一段时间，结构化程序设计方法备受程序员的欢迎，尤其是在中小型问题的处理上。但随着问题复杂性的变化，用户需求的变化，基于过程设计的结构化程序设计方法就面临着灾难性的打击。其问题主要在于：软件重用性差，软件可维护性差，开发出的软件渐渐不能满足用户的需要。

面向对象的设计思想从现实世界中客观存在的事物（即对象）出发来构造软件系统，并在系统构造中尽可能运用人类的自然思维方式，强调直接以问题域（现实世界）中的事物为中心来思考问题，认识问题，并根据这些事物的本质特点，把它们抽象地表示为系统中的对象，作为系统的基本构成单位（而不是用一些与现实世界中的事物相去甚远，并且没有对应关系的其他概念来构造系统）。这可以使系统直接地映射问题域，保持问题域中事物及其相互关系的本来面貌。

从世界观的角度可以认为：面向对象的基本哲学是认为世界是由各种各样具有自己的运动规律和内部状态的对象所组成的；不同对象之间的相互作用和通信构成了完整的现实世界。因此，人们应当按照现实世界的本来面貌来理解世界，直接通过对象及其相互关系来反映世界。这样建立起来的系统才能符合现实世界的本来面目。

从方法学的角度可以认为：面向对象的方法是面向对象的世界观在开发方法中的直接运用。它强调系统的结构应该直接与现实世界的结构相对应，应该围绕现实世界中的对象来构造系统，而不是围绕功能来构造系统。

面向对象（Object Oriented，OO）是当前计算机界关注的重点，它从 20 世纪 90 年代开

始成为软件开发的主流方法。面向对象的概念和应用已超越了程序设计和软件开发，扩展到很宽的范围，如数据库系统、交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD 技术、人工智能等领域。

面向对象方法的定义有很多说法，常用的定义有以下两种。

定义一：面向对象方法是一种运用对象、类、封装、继承、多态和消息等概念来构造、测试、重构软件的方法。

定义二：面向对象方法是以认识论为基础，用对象来理解和分析问题空间，并设计和开发出由对象构成的软件系统(解空间)的方法。由于问题空间和解空间都是由对象组成的，这样可以消除由于问题空间和求解空间结构的不一致带来的问题。简言之，面向对象就是面向事情本身，面向对象的分析过程就是认识客观世界的过程。

1.3 面向对象的特征

基于面向对象的思想，面向对象方法具有以下几个特征。

1. 封装

封装是一种信息隐蔽技术，它体现于类的说明，是对象的重要特性。封装把数据和加工该数据的方法(函数)封装为一个整体，以实现独立性很强的模块，使得用户只能见到对象的外特性(对象能接受哪些消息，具有哪些处理能力)，而对象的内特性(保存内部状态的私有数据和实现加工能力的算法)对用户是隐蔽的。封装的目的在于把对象的设计和对象的使用分开，使用者不必知晓行为实现的细节，只需用设计者提供的消息来访问该对象。

封装有两层含义：一是把对象的全部属性和行为结合在一起，形成一个不可分割的独立单位，对象的属性值(除了公有的属性值)只能由这个对象的行为来读取和修改；二是尽可能隐蔽对象的内部属性和实现细节，对外形成一道屏障，与外部的联系只能通过公共接口实现。如某些类对外提供了调整时间、显示时间、库存查询、入库、出库等方法，而隐藏了时、分、秒及库存数据，同时隐藏了公共接口调整时间、显示时间、库存查询、入库、出库等方法的实现细节，也就是类的用户见不到其具体实现。

封装的信息隐蔽作用反映了事物的相对独立性，使用者可以只关心它对外所提供的接口，即能做什么，而不注意其内部细节，即怎么提供这些服务，如一台封装好的电视机，用户只需知道怎么使用换频道、调音量等操作，而不需要了解其怎么接收信号、处理信号等。

封装的结果使对象以外的部分不能随意存取对象的内部属性，从而有效地避免了外部错误对它的影响，大大减小了查错和排错的难度。另一方面，当对象内部进行修改时，由于它只通过少量的外部接口对外提供服务，因此同样减小了内部的修改对外部的影响。

封装机制将对象的使用者与设计者分开，使用者不必知道对象行为实现的细节，只需要用设计者提供的外部接口让对象去做即可。封装的结果实际上隐蔽了复杂性，并提供了代码重用性，从而降低了软件开发的难度。

在 C++中，最基本的封装单元是类，一个类定义着由一组对象所共享的行为(数据和代码)。一个类的每个对象均包含它所定义的结构与行为，这些对象就好像是一个模子铸造出来的，所以对象也叫做类的实例。

2. 继承

继承(Inheritance)是一种联结类与类的层次模型。有了继承,类与类之间不再是彼此孤立的,一些特殊的类可以自动地拥有一些一般的属性与行为,而这些属性与行为并不是重新定义的,而是通过继承的关系得来的。

在传统的程序设计中,人们往往要为每一项应用单独进行全新的程序开发。继承允许和鼓励类的重用,提供了一种明确表述共性的方法。一个特殊类既有自己新定义的属性和行为,又有继承下来的属性和行为,而这个类被它更下层的特殊类继承时,它继承来的和自己定义的属性和行为又被下一层的特殊类继承下去。因此,继承是传递的,体现了大自然中特殊与一般的关系。

继承性是子类自动共享父类数据和方法的机制,它由类的派生体现。一个子类直接继承父类的全部描述,同时可对其修改和扩充,继承是对父类的重用机制。

3. 多态

多态是指一个方法只有一个名字,但可以有许多形态,也就是程序中可以定义多个同名的方法,用"一个接口,多个方法"来描述。多态意味着同一类对象有着多重特征,可以在特定的情况下,表现不同的状态,从而对应着不同的方法。

多态的实现是在继承体系结构中,同一消息为不同的对象接受时可产生完全不同的行动,这种现象称为多态性。利用多态性用户可发送一个通用的信息,而将所有的实现细节都留给接受消息的对象自行决定,好像同一消息即可调用不同的方法。

基于以上特征,面向对象技术有以下几个基本概念。

对象:对象是要研究的任何事物。对象由数据(描述事物的属性)和作用于数据的操作(体现事物的行为)构成一个独立整体。从程序设计者来看,对象是一个程序模块,从用户来看,对象为他们提供所期望的行为。

类:类是对象的模板。即类是对一组有相同数据和相同操作的对象的定义,一个类所包含的方法和数据描述一组对象的共同属性和行为。类是在对象之上的抽象,对象则是类的具体化,是类的实例。类可有其子类,也可有其父类,形成类层次结构。

消息:消息是对象之间进行通信的一种规格说明,通过消息多个对象协作完成具体的功能。一般它由三部分组成:接收消息的对象、消息名及实际变元。

综上可知,在OO方法中,对象和传递消息分别表现事物及事物间相互联系的概念。类和继承是适应人们一般思维方式的描述范式。方法是允许作用于该类对象上的各种操作。这种对象、类、消息和方法的程序设计范式的基本点在于对象的封装性和类的继承性。通过封装能将类的定义和类的实现分开,通过继承能体现类与类之间的关系,以及由此带来的动态联编和实体的多态性,从而构成了面向对象的基本特征。

1.4 C++概述

C++是Bjarne Stroustrup在20世纪80年代早期开发的,是一种基于C的面向对象的语言。顾名思义,C++表示C的累加。由于C++基于C,所以这两种语言有许多共同的语法和功能,C中所有低级编程的功能都在C++中保留下来。但是,C++比其前身丰富得多,用

途也广泛得多。C++对内存管理功能进行了非常大的改进，C++还具有面向对象的功能，所以C在功能上只是C++的一个很小的子集。C++在适用范围、性能和功能上也增强很多。因此，目前大多数高性能的应用程序和系统都是使用C++编写的。

C++在几乎所有的计算环境中都非常普及，涉及个人电脑、UNIX工作站和大型计算机。如果考察一下编程语言的发展史，就可以看出C++的普及率是非常高的。除此以外，大多数专业程序员总是愿意使用他们已熟知的、使用起来得心应手的语言，而不是转而使用新的、不熟悉的语言，花大量的时间来研究其特性。当然，C++是建立在C的基础之上（在C++出现之前，许多环境都使用C语言），这对于C++的普及有很大的帮助。C++有许多优点：

（1）C++适用的应用程序范围极广。C++可以用于几乎所有的应用程序，从字处理应用程序到科学计算应用程序，从操作系统组件到计算机游戏等。

（2）C++可以用于硬件级别的编程，例如实现设备驱动程序。

（3）C++从C中继承了过程化编程的高效性，并集成了面向对象编程方式的功能。

（4）C++在其标准库中提供了大量的功能。

（5）有许多商业C++库，支持数量众多的操作系统环境和专门的应用程序。

（6）C++的可移植性非常强，因为几乎所有的计算机都可以使用C++编程，所以C++语言普及到几乎所有的计算机平台上。也就是说，把用C++编写的程序从一台机器迁移到另一台机器上不需要费什么力气。

C++的国际标准由ISO/IEC 14882文档定义，该文档由美国国家标准协会ANSI发表。读者可以获得该标准的副本，但要记住，该标准主要由编译器编写人员使用，而不是学习该语言的人使用。

标准化是把所编写的程序从一种类型的计算机迁移到另一种类型的计算机上的基础。标准的建立使语言在各种机器上的实现保持一致。在所有相容编程系统上都可用的一组标准功能意味着，用户总是能确定下一步会获得什么结果。C++的ANSI标准不仅定义了语言，还定义了标准库。使用ANSI标准后，C++使应用程序可以轻松地在不同的机器之间迁移，缓解了在多个环境上运行的应用程序的维护问题。

C++的ANSI标准还有另一个优点：它对用C++编程所需要学习的部分进行了标准化。这个标准将使后续的程序具有一致性，因为它只为C++编译器和库提供了一个定义参考。在编写编译器时，该标准的存在也使编写人员不再需要许可。读者在购买遵循ANSI标准的C++编译器时，就知道会得到什么语言和标准库功能。

C++程序都是由若干个源文件组成的，C++是一种编译语言。在执行C++程序之前，必须用另一个程序（即编译器）把它转换为机器语言。编译器会检查并分析C++程序，然后生成机器指令，以执行源代码指定的动作。

1.5 C++程序开发步骤

开发一个C++程序，需要按照以下4步顺序执行。

（1）编辑：程序员用任一编辑软件（编辑器）将编写好的C++程序输入计算机，并以文本文件的形式保存在计算机的磁盘上。编辑的结果是建立C++程序源文件，扩展名为.cpp（如