高等学校计算机专业“十一五”规划教材

C++面向对象程序设计

主　编　李　兰
副主编　任凤华

西安电子科技大学出版社
http://www.xduph.com

高等学校计算机专业“十一五”规划教材

C++面向对象程序设计

主　编　李　兰
副主编　任凤华

西安电子科技大学出版社

内 容 简 介

面向对象程序设计是目前流行的软件开发方法。本书根据“面向对象程序设计”课程的基本教学要求，针对面向对象的本质和特性，系统地讲解了面向对象程序设计的基本理论和基本方法，阐述了用C++语言实现面向对象基本特性的关键技术。本书的内容主要包括：面向对象程序设计概述、C++语言基础、封装性、继承性、运算符重载、多态性、模板和STL、输入/输出流、异常处理等。

本书可作为高等院校计算机及相关专业“C++面向对象程序设计”课程的教材，也可作为从事计算机开发和应用的工程技术人员的参考书。同时，也适合初学程序设计或有一定编程实践基础、希望突破编程难点的读者作为自学教材。

图书在版编目(CIP)数据

C++面向对象程序设计 / 李兰主编.
—西安：西安电子科技大学出版社，2010.9
高等学校计算机专业“十一五”规划教材
ISBN 978-7-5606-2444-0
Ⅰ. ① C… Ⅱ. ① 李… ② 任… Ⅲ. ① C语言—程序设计—高等学校—教材
Ⅳ. ① TP312

中国版本图书馆CIP数据核字(2010)第151557号

策　　划　陈　婷
责任编辑　许青青　陈　婷
出版发行　西安电子科技大学出版社(西安市太白南路2号)
电　　话　(029)88242885　88201467
网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com
经　　销　新华书店
印刷单位　陕西华沐印刷科技有限责任公司
版　　次　2010年9月第1版　2010年9月第1次印刷
开　　本　787毫米×1092毫米　1/16　印张22.75
字　　数　535千字
印　　数　1～3000册
定　　价　33.00元
ISBN 978－7－5606－2444－0 / TP・1218
XDUP 2736001-1

高等学校计算机专业“十一五”规划教材

编审专家委员会

前　言

C++语言为面向对象技术提供了全面支持，是目前应用较广的一种优秀的高级程序设计语言，也是一个可编写高质量的用户自定义类型库的工具。C++保留了对传统的结构化程序设计方法的支持，同时又增加了对面向对象程序设计方法的完全支持，但后者是其主要特色和应用。C++ 语言是一种具有代表性的面向对象的程序设计语言，它具有丰富的数据类型和各种运算功能，带有庞大的函数库和类库，既支持面向过程的程序设计，又支持面向对象的程序设计。

目前，国内外许多高等院校计算机专业和相关专业都开设了“面向对象程序设计”课程，其目的是让学生掌握面向对象程序设计的概念和方法，深刻理解面向对象程序设计的本质，并用面向对象技术来开发软件。

本书力求在内容、编排顺序和教学方法上有所创新和突破，以帮助学生快速理解与程序设计相关的基本概念，掌握程序设计语言的基本知识，建立程序设计的基本思想，并通过实际动手领会和掌握 C++的精髓，最终获得面向对象 C++程序设计的真实本领，为今后的进一步学习和开发应用打下坚实的基础。

基于这些理念，本书对 C++语言的基本概念、原理和方法的叙述由浅入深，条理分明、循序渐进；以“概念—语法—举例”的形式进行讲解，并指出了学生常犯的错误和容易混淆的概念；各章均配有大量的习题，以帮助学生深入理解面向对象语言和 C++的性质。

本书的主要特色体现在以下几个方面：

(1) 内容精练，语言严谨。编者综合了教学及软件设计经验，使得本书既具有较强的理论性，又具有较强的实用性。书中采用最新的 C++标准，对庞杂的知识进行了认真的取舍，对概念进行了清楚准确的解释，并结合实例及作者的教学经验进行了说明讲解。

(2) 知识介绍深入浅出、简明易懂。对 C++语言的基本概念、原理和方法的叙述由浅入深，条理分明，循序渐进地介绍了 C++面向对象程序设计的相关知识。各个章节层层展开、环环相扣，目的是希望读者通过对本书的学习，能够编写出规范、稳定的程序。

(3) 强调理论与实践紧密结合。为了使读者能快速掌握 C++相关知识的应用方法，不仅说明了知识点，更重要的是表明了其应用方法，以激发读者对程序设计的兴趣，使读者可以在有趣、高效的应用中学习枯燥的语法。

(4) 强调“练中学”。初学者考试时往往会感到茫然而不知所措，为了巩固其中灵活、难解的语法知识，每章都有配套习题，习题包括选择题、填空题、判断题、阅读程序和编程题。

(5) 本书各章提供了常见编程错误，这部分内容对初学者及长期编程的人都很有用。

(6) 本书配有全部的程序源文件和电子教案。

总之，本书信息量大，综合面广，实用性强，可读性好，具有鲜明的特色。

本书的编者长期从事面向对象程序设计的教学，具有丰富的教学、实践经验和独到的见解，这些经验和见解都已融入到了本书的内容之中。全书共 9 章。第 1 章面向对象程序

设计概述，阐述了C++的主要特点；第2章C++语言基础，讲述了各种常见的数据类型，各种表达式及其求值，表达式的优先级、结合性，顺序结构、选择结构和循环结构三种基本结构及其相应的控制语句，变量的存储和作用域，程序的文件结构及编译预处理命令；第3章函数，主要介绍了C++中函数的定义与声明、函数调用、函数重载和内联函数等内容；第4章类与对象，讲解了C++中类的定义、对象的创建、对象成员的访问，以及友元与静态成员等基本内容；第5章继承，着重介绍了C++的基类和派生类，单继承、多继承等继承方法，二义性和虚基类等；第6章多态与虚函数，介绍了多态性，重点介绍了运算符重载、虚函数和抽象类；第7章模板，介绍了模板编程方法，并对模板容易出现的编程问题进行了详细的讨论；第8章输入/输出(I/O)流，重点讨论了标准输入/输出流类、文件操作与文件流类；第9章异常处理，主要介绍了C++中异常的概念、基本原理以及异常处理方法和多路捕获。本书前3章属于基础部分，后6章属于面向对象的程序设计部分。为满足不同层次的教学需求，教师可采取多种方式，根据学生的背景知识以及课程的学时数对内容进行取舍。

本书由李兰主编并统稿。李兰编写第1、2、3、5、6、7章，任凤华编写第4章及第8、9章的部分内容，王金龙参与编写第8、9章的初稿，熊晓芸编写部分章节的常见错误。本书在编写过程中得到了各方面的大力支持。在此，编者要感谢一起工作的同事们，他们对本书的编写给予了极大的关注和支持：感谢本书所列参考文献的作者；感谢为本书出版付出辛勤劳动的西安电子科技大学出版社的工作人员，他们为本书的出版倾注了大量热情；感谢使用本书的读者。希望各位读者能够提出宝贵的意见或建议，并将对本教材的建议或意见寄给编者，您的意见将是我们再版修订时的重要参考。

尽管作者力求严谨，并尽了最大努力，但由于水平有限，时间仓促，疏漏与不妥之处在所难免，敬请各位读者不吝赐教。

在使用本书时如遇到问题需要与编者交流，或想索取本书例题的源代码与电子讲稿，请与编者联系。编者的电子信箱为：lqdlanli@163.com。

编　者

2010年6月

目　录

第 1 章　面向对象程序设计概述

(Introduction to Object-Oriented Programming)

****

【学习目标】

- 了解计算机语言的发展。
- 理解结构化程序设计和面向对象程序设计的特点。
- 了解面向对象程序设计语言中的基本概念。
- 掌握程序开发的过程。
- 理解名称空间的概念，学会运用名称空间。
- 掌握 C++程序的基本结构。

****

1.1　计算机程序设计语言的发展

(Computer Programming Languages Development)

1.1.1　程序设计语言概述(Introduction to Programming Languages)

计算机的每一步发展几乎都会在软件设计和程序设计语言中得到充分体现。随着软件开发规模的扩大和开发方式的变化，人们开始将程序设计语言作为一门科学来对待，程序设计方法和技术在各个时期的发展直接导致了一大批风格各异的程序设计语言的诞生。

众所周知，在用计算机解决某一个问题之前，必须先把求解问题的步骤描述出来，这种描述称为算法。算法不能直接输入到计算机中，因为用自然语言表达的算法，计算机并不理解。正如人与人之间通过语言进行沟通一样，要计算机做某事，就要用计算机能够理解的语言将其表述出来，这种语言称为计算机语言。将算法用某种特定的计算机语言表达出来，输入到计算机，这便是计算机编程。

计算机的工作体现为顺序执行程序。程序是控制计算机完成特定功能的一组有序指令的集合。编写程序所使用的语言称为程序设计语言，它是人与计算机之间进行信息交流的工具。计算机程序设计语言是计算机可以识别的语言，用以描述解决问题的方法，供计算机阅读和执行，与一般语言一样，它具有一套语法、词法规则系统。

从1946年世界上第一台计算机诞生以来，短短的60多年计算机科学得到了迅猛发展，程序设计语言的发展也从低级到高级，经历了机器语言、汇编语言、高级语言、面向对象语言等多个阶段。

1.1.2 机器语言与汇编语言(Machine Language and Assemble Language)

1. 机器语言

最基本的计算机语言是机器语言。机器语言(也称为第一代语言)使用二进制位来表示程序指令。计算机的硬件系统可以直接识别和执行的二进制指令(机器指令)的集合称为这种计算机的机器语言。每种计算机都有自己的机器语言，并能直接执行用机器语言所编写的程序。虽然绝大多数计算机完成的功能基本相近，但不同计算机的设计者都可能会采用不同的二进制代码集来表示程序指令。所以，不同种类的计算机使用的计算机语言并不一定相同，但现代计算机都是以二进制代码的形式来存储和处理数据的。

在早期的计算机应用中，程序都是用机器语言编写的，程序员需要记住各种操作的机器语言指令。同时，为了存取数据，程序员还必须记住所有数据在内存中的存储地址。例如，字母A表示为1010，数字9表示为1001。由于机器语言的指令码可能有多种形式，还要考虑运算中的进位、借位和符号溢出等各种情况，因此更增加了程序员的记忆负担。这种需要记住大量的编码指令来编写程序的方法不仅难以实现，而且非常容易出错。直接使用机器语言来编写程序是一种相当复杂的手工劳动，它要求使用者熟悉计算机的有关细节，一般的工程技术人员是难以掌握的。

2. 汇编语言

汇编语言(也称为第二代语言)的出现简化了编程人员的工作。汇编语言将机器指令映射为一些可以被人读懂的助记符，如ADD、MOV等，它实际上是与机器语言相对应的语言。汇编语言的主要特征是可以用助记符来表示每一条机器指令。汇编语言同机器语言相比，并没有本质的区别，只不过是把机器指令用助记符号代替。由于汇编语言比机器语言容易记忆，因此其编程效率比机器语言前进了一大步，而且改进了程序的可读性和可维护性。直到今天，仍然有人用汇编语言进行编程。但汇编语言程序的大部分语句还是和机器指令一一对应的，与机器的相关性仍然很强。

由于计算机只能执行机器指令，因此汇编语言需要在编译后才能被识别，这一过程称为汇编。也就是说，用汇编语言编好的程序还需要由相应的翻译程序(称为汇编程序)将其翻译成计算机可执行的机器语言程序。用程序设计语言写成的程序称为源程序。源程序可以在具有这种语言编译系统的不同计算机上使用。源程序必须翻译成机器语言才能执行。逐条翻译并执行的翻译程序称为解释程序，而将源程序一次翻译成目标程序然后再执行的翻译程序称为编译程序。早期的计算机由于速度慢、内存小，因此衡量程序质量高低最重要的指标是机器执行的效率。但是，随着计算机技术的发展，机器硬件的性能大幅度提高，程序的复杂度也在增加，人们越来越希望把简单、重复性的工作交给机器去做，而人更多地从事创造性的工作，因此，程序的可读性和可维护性渐渐成为衡量程序质量高低的最重要的指标。

虽然汇编语言较机器语言已有很大的改进，但仍是低级语言，它有以下两个主要缺点：

(1) 涉及细节太多；

(2) 与具体的计算机相关。

所以，汇编语言也被称为面向机器的语言。为了进一步提高编程效率，改进程序的可读性、可维护性，又出现了许多高级语言(也称为第三代语言)。

1.1.3 高级语言(Advanced Language)

20 世纪 60 年代，出现了高级语言。既然机器语言和汇编语言都是计算机可以理解的语言，使用它们可以完全控制计算机的行为，那么为什么人们还要创造并使用高级程序设计语言呢？因为机器语言和汇编语言都是低级语言，是面向机器的，与具体的计算机相关，学习起来困难，编程效率也低，可读性、可维护性也差，而高级语言编写的程序借助于编译器就可以在特定的机器上运行。高级语言编写的程序由一系列语句(或函数)组成，其中每一条语句都对应着几条、几十条甚至上百条机器指令的序列，这样一条语句的功能显然增强了，所以用它开发程序比用低级语言开发效率高得多。同时，由于高级语言的编写方式更接近人们的思维习惯，因此用高级语言编写的程序易读、易懂、易于维护。

高级语言的另一个优点是：编写的程序具有一定的通用性。低级语言涉及到计算机硬件细节，所以不具有通用性。要使用高级语言编写的程序在某一计算机上运行，只要该计算机提供该语言的翻译系统即可。

既然高级语言有着低级语言无法比拟的优势，那么是不是可以完全放弃低级语言呢？回答是否定的。首先，机器语言是最终操作计算机硬件的语言，任何高级语言程序要在计算机上执行，必须翻译成机器指令；其次，虽然高级语言具有众多优点，但是执行速度比不上同样功能的低级语言，并且在对硬件的操作上，也不如低级语言灵活，所以在对程序速度要求高的场合(如过程控制的实时系统中或者编写某种新硬件的驱动程序时)，仍然可以看到低级语言(主要是汇编语言)的影子。

使用高级语言编程，一般不必了解计算机的指令系统和硬件结构，只需掌握解题方法和高级语言的语法规则，就可以编写程序。高级语言在程序设计时着眼于问题域中的过程，是一种面向过程的语言。这使得在书写程序时可以联系到程序所描述的具体事物，使计算机的编程语言前进了一大步。

1.1.4 面向对象语言(Object-Oriented Programming Language)

20 世纪 80 年代，出现了面向对象的编程语言。面向对象的编程语言与以往各种编程语言的根本不同点在于：它的设计出发点就是为了能更直接地描述客观世界中存在的事物(即对象)以及它们之间的关系。面向对象语言是比面向过程语言更高级的一种高级语言。面向对象语言的出现改变了编程者的思维方式，使程序设计的出发点由问题域中的过程转向问题域中的对象及其相互关系，这种转变更加符合人们对客观事物的认识。因此，面向对象语言更接近于自然语言，是人们对于客观事物更高层次的抽象。

面向对象的语言分为几大类别：一类是纯面向对象的语言，如 SmallTalk 和 Eiffel；另一类是混合型的面向对象语言，如 C++ 和 Objective C；还有一类是与人工智能语言相结合形成的语言，如 LOOPS、Flavors、CLOS 以及适合网络应用的 Java 语言等。

C++ 语言是由 AT&T 公司贝尔实验室的 Bjarne Stroustrup 博士开发的，是在 C 语言的基础上扩展而来的一门高效、实用的混合型程序设计语言。它包括两部分内容：一部分是支持面向过程部分，该部分以 C 语言为核心；另一部分是支持面向对象部分，是 C++ 对 C 的扩充部分。这样一来，它既支持面向对象程序设计方法，又支持面向过程的结构化程序设计方法，具有广泛的应用基础和丰富的开发环境的支持。因此 C++ 语言在短短的几年内迅速流行，成为当今面向对象程序设计的主流语言，同时也大大促进了面向对象编程的应用和发展。

1.2 程序设计方法

(Programming Methodology)

计算机程序设计方法的发展经历了面向过程的方法和面向对象的方法两个阶段。

1.2.1 结构化程序设计方法(Structured Programming)

结构化程序设计(Structured Programming, SP)方法是由 E.W.Dijkstra 等人于 1972 年首先提出的，它建立在 Bohm、Jacopini 证明的结构定理的基础之上。该结构定理指出：任何程序逻辑都可以用顺序、选择和循环三种基本结构来表示。结构化程序设计方法是从程序结构和风格上来研究程序设计。用结构化程序设计方法编写出来的程序不仅结构良好、易读易写，而且其正确性易于证明。

结构化程序设计方法是为了解决早期计算机程序难于阅读、理解和调试，难于维护和扩充，以及开发周期长、不易控制程序的质量等问题而提出的，它的产生和发展奠定了软件工程的基础。

结构化程序设计方法强调程序结构的规范性，即强调程序设计的自顶向下、逐步求精的演化过程。在这种方法中，待解问题和程序设计语言中的过程紧密相联。

结构化程序设计方法的优点如下：

(1) 自顶向下，逐步细化。结构化程序设计方法的主要思想是功能分解并逐步求精。当一些任务复杂以致无法描述时，可以将它拆分为一系列较小的功能部件，直到这些完备的子任务小到易于理解的程度，这种方法叫“自顶向下，逐步细化”。

(2) 模块化设计。在程序设计中常采用模块化设计的方法，尤其是当程序比较复杂时，更有必要。模块化是指在拿到一个程序模块(实际上是程序模块的任务书)以后，根据程序模块的功能将它划分为若干个子模块，如果这些子模块的规模仍较大，则可以划分为更小的模块。这个过程采用自顶向下的方法来实现。结构化程序设计方法可以解决人脑思维能力的局限性和所处理问题的复杂性之间的矛盾。

(3) 结构化编码。在设计好一个结构化的算法之后，还要善于进行结构化编码，即用高级语言语句正确地实现顺序、选择、循环三种基本结构。

此外，各模块可以分别编程，使程序易于阅读、理解、调试和修改；新功能模块的扩充较为方便；功能独立的模块可以组成子程序库，有利于实现软件复用等。因此，结构化程序设计方法出现以后，很快被人们接受并得到了广泛应用。

结构化程序设计方法以解决问题的过程作为出发点，其方法是面向过程的。它把程序

定义为“数据结构+算法”，程序中数据与处理这些数据的算法(过程)是分离的。这样对不同的数据结构作相同的处理，或对相同的数据结构作不同的处理，都要使用不同的模块，从而降低了程序的可维护性和可复用性。同时，这种分离导致了数据可能被多个模块使用和修改，难于保证数据的安全性和一致性。因此，对于小型程序和中等复杂程度的程序来说，结构化程序设计方法是一种较为有效的技术，但对于复杂的、大规模的软件的开发来说，它就不尽如人意了。

通过上面的分析可以看出，结构化程序设计方法的核心思想是功能的分解，其特点是将数据结构与过程分离，着重点在过程。随着程序规模与复杂性的增长，面向过程的结构化程序设计方法存在明显的不足之处：

(1) 数据安全性问题。

(2) 可维护性及可重用性差。

(3) 图形用户界面的应用程序很难用过程来描述和实现，开发和维护也都很困难。

1.2.2　面向对象程序设计方法(Objected-Oriented Programming)

面向对象程序设计方法建立在结构化程序设计方法的基础之上，避免了结构化程序设计方法中所存在的问题。

面向对象是一种认识世界的方法，也是一种程序设计方法。面向对象的观点认为，客观世界是由各种各样的实体，也就是对象组成的。每种对象都有自己的内部状态和运动规律，不同对象间的相互联系和相互作用就构成了各种不同的系统，并进而构成了整个客观世界。按照这样的思想设计程序，就是面向对象的程序设计。“面向对象”不仅仅作为一种技术，更作为一种方法贯穿于软件设计的各个阶段。

面向对象的方法是把软件系统分解成为相互协作而又彼此独立的对象的集合。对象是数据结构和算法的封装体，每个对象在功能上相互之间保持相对独立，即每个对象有自己的数据、操作和功能，对象被看做由数据及可以施加在这些数据上的操作所构成的统一体。当对象的一个成员函数被调用时，对象执行其内部的代码来响应这个调用，这使对象呈现出一定的行为，行为及其结果就是该对象的功能。根据这个定义，在对象中，不但存有数据，而且存有代码，使得每个对象在功能上相互之间保持相对独立。当然，对象之间存在各种联系，但它们之间只能通过“消息”进行通信。程序可表示为：

程序 = 对象 + 类 + 继承 + 消息通信

通过这个等式可以知道面向对象程序的基本结构。

面向对象程序设计着重于类的设计。类正是面向对象语言的基本程序模块，通过类的设计可完成实体的建模任务。一般情况下，面向对象程序都由三部分构成：类的声明、类的成员的实现和主函数。

面向对象程序设计是在面向过程程序设计的基础上的质的飞跃。面向对象方法的产生是计算机科学发展的要求。面向对象的技术在系统程序设计、数据库及多媒体应用等领域都得到了广泛应用。

【例 1-1】　用 C++ 语言描述，用结构化程序设计方法计算三角形的面积(已知一个三角形的 3 个顶点的坐标)。