面向21世纪课程教材
Textbook Series for 21st Century

面向对象程序设计基础

（第二版）

李师贤　李文军　周晓聪
李宏新　林　瑛　编著

高等教育出版社

面向21世纪课程教材

Textbook Series for 21st Century

面向对象程序设计基础

（第二版）

李师贤　李文军　周晓聪　李宏新　林　瑛　编著

高等教育出版社

内 容 提 要

本书第一版被列入“面向21世纪课程教材”，自出版以来，深受读者欢迎。作为给程序设计初学者提供的一本入门教材，本书以循序渐进、深入浅出的方式，引导众多学子走进了面向对象程序设计的大门。新版教材在归纳多年教学体会的基础上，以继续保持原书的特色为前提，对前版教材进行了修改和补充，使相关概念阐述得更加通俗易懂，并适当增加了相关的例子，以求使读者在学习时能更好地理解和领会。新版教材内容包括了程序设计基础、程序设计语言、算法与复杂性和软件工程等内容，可作为高校计算机专业本科生程序设计课程的入门教材，也可供相关专业高年级学生作为面向对象程序设计课程教材使用。

与本书配套的电子教案可从高等教育出版社高等理工教学资源网下载，网址：http://www.hep-st.com.cn/。

图书在版编目(CIP)数据

面向对象程序设计基础/李师贤等编著. —2版.
—北京：高等教育出版社，2005.3(2006重印)
ISBN 7-04-016650-X

Ⅰ.面… Ⅱ.李… Ⅲ.面向对象语言-程序设计-高等学校-教材 Ⅳ.TP312

中国版本图书馆CIP数据核字(2005)第019233号

策划编辑 倪文慧　　**责任编辑** 倪文慧
封面设计 于文燕　　**责任印制** 尤　静

出版发行	高等教育出版社	购书热线	010-58581118
社　　址	北京市西城区德外大街4号	免费咨询	800-810-0598
邮政编码	100011	网　　址	http://www.hep.edu.cn
总　　机	010-58581000		http://www.hep.com.cn
		网上订购	http://www.landraco.com
			http://www.landraco.com.cn
经　　销	蓝色畅想图书发行有限公司	畅想教育	http://www.widedu.com
印　　刷	北京铭成印刷有限公司		
		版　　次	1998年8月第1版
			2005年3月第2版
开　　本	787×1092　1/16	印　　次	2006年6月第4次印刷
印　　张	31.5	定　　价	32.00元
字　　数	670 000		

物料号　16650-00

第二版前言

本书第一版被列入“面向21世纪课程教材”，自出版以来，深受读者欢迎。作为给程序设计初学者提供的一本入门教材，它以循序渐进、深入浅出的方式，引导众多学子走进了面向对象程序设计的大门；不少高校采用它作为大学生首门程序设计课程的教材。

在归纳多年教学体会的基础上，我们以继续保持原书的特色为前提，对本书进行了修改和补充，以便相关的概念阐述得更加通俗易懂；并适当地增加了相关的例子，以求使读者在学习时能更好地理解和领会。例如，在第四章增加了常用编译预处理命令的说明；在第四、五、六、七、八、九等章加入了针对性的应用实例等。

程序设计课程教学的宗旨是培养学生掌握程序设计的基本概念、基本思想和方法。我们采用C++语言作为学习面向对象程序设计的工具；但是，不能认为C++就是面向对象程序设计，C++语言仅仅是学习所用的工具而已，它并非学习的最终目标。所以，本书也不打算编成C++大全，涉及C++语言的众多细节、C++语言各种具体版本的制约等内容都未列入本书讨论的范畴。透过现象抓本质，这也正是我们重要的体会。

本书的内容包含了《中国计算机科学与技术学科教程2002》（简称CCC2002）里所列举的知识体系一览中的以下内容：

“PF程序设计基础”中的PF1至PF5；

“PL程序设计语言”中的PL1至PL6；

同时，还涉及到“AL算法与复杂性”和“SE软件工程”当中的一些基本概念和相关知识。

对于已经有一定程序设计基础的读者，不妨跳过第一章到第四章的内容，直接从第五章开始学习；对于面向对象程序设计而言，第五章至第十章无疑是极其重要的。

程序设计是实践性很强的课程。除了精心组织好课堂教学外，认真组织好上机实习是学好程序设计必不可少的重要环节。随着课堂教学进程的进展，应该配备有相应规模、相应难度和相应强度的练习和上机实习实践训练；并且，每一次实习都应该做到有计划、有组织、有落实、有指导、有考核、有总结、有提高，才能收到良好的效果。课堂教学和上机实习的时间安排，建议不低于1:1的关系。

中山大学计算机科学系2000级至2003级的同学在使用本书（第一版）进行课程学习的过程中，对本书的补充和修改提供了不少宝贵的建议；特别是2001级的陈凯生、陈强超、宋岩、陈子聪、肖茵茵、许金萍以及2002级的彭丽珊、赵少东、蓝天恩等同学在本书修改、补充过程中做了不少工作，在此谨向他们致以衷心的感谢。

计算机科学与技术发展迅速，而程序设计的内容和技术同样也在迅速发展，由于编者水

平所限和时间关系，本书可能存在不少谬误，恳请广大读者不吝指正。

编　者

2004 年 12 月 30 日

引　言

我们编写本书的目的是为了给程序设计初学者提供一本清晰的入门教材，该教材以**面向对象程序设计**(Object - Oriented Programming，OOP)为核心，并选用C++语言作为工具。

程序设计是计算机专业非常重要的基础课程，学好程序设计的入门课程对于软件工程、编译原理、形式语言与自动机、形式语义学等后续课程有很大帮助。程序设计课程包括两个相辅相成的方面：程序设计方法与程序设计语言。自从第一台计算机诞生以来，程序设计方法与程序设计语言一直在不断发展，20 世纪 70 年代是结构化程序设计的时代，而 80 年代以来面向对象程序设计逐渐占据了程序设计的主流。“面向对象”不再是软件开发中的一个时髦名词，而是对软件开发人员一种很基本的要求。

在过去的十多年中，程序设计入门教材一般选用 Pascal 或 C 作为程序设计语言，并贯穿结构化程序设计思想。随着面向对象技术逐步成为程序设计的主流，从一开始就向学生传授面向对象程序设计思想是十分必要和有益的。面向对象程序设计与结构化程序设计是解决问题的两种不同的思维方式，前者注重事物的结构，而后者注重事物表现的行为。学习面向对象程序设计并不要求学生先完全掌握结构化程序设计，在程序设计技术迅速发展的今天，面向对象程序设计方法与面向对象程序设计语言可以并应该作为初学者的入门选择。

本书正是为满足初学者的这种需要而编写的，书中向第一次学习程序设计的读者介绍了面向对象程序设计方法与C++程序设计语言。本书的内容大致可分为两部分，一部分从数据结构和控制结构两个角度介绍程序设计语言的一般概念，另一部分围绕类与对象介绍面向对象程序构造的基本思想。本书选用C++作为程序设计语言是因为C++是一种广泛使用的面向对象程序设计语言，它为面向对象程序设计思想提供了较好的支持，同时也保留了 C 语言的灵活性特点。在介绍某些语言特征时，本书针对C++语言在面向对象程序设计理论上的不完善之处，参考了其他语言的相关概念进行了解释(例如类属、程序断言与异常处理等机制参考了 Eiffel 和 Ada 语言的有关概念)。

与其他面向对象程序设计教材相比，本书有以下特色：

一、以循序渐进、深入浅出的方式引导读者步入面向对象程序设计的大门

传统的面向对象程序设计教材都假设读者已经掌握了结构化程序设计方法以及 C 语言或 Pascal 语言，因而难以作为程序设计入门教材。本书则从程序设计最基本的概念开始，详细介绍基本数据类型与控制结构，逐步过渡到函数、类、继承、多态性、类属等复杂机制，最后还包括提高程序可靠性的断言机制与异常处理机制、提高程序可重用性的规范方法等高级

内容。例如，对于初学者而言，变量是一个比较难理解而又是十分重要的概念，本书在开始时对变量的概念做直观描述，以求给读者一个感性认识，在引入指针概念之前，又对变量的名字、属性、关联和值等要素进行了深入讨论。

二、OOP != C++

"!="是C++语言的运算符，表示"不等于"。我们想用这个式子表达两重含义：一方面，学习面向对象程序设计不仅仅是学习使用C++语言，在C++语言之外还有许多其他的优秀语言支持面向对象程序设计，如Smalltalk、Eiffel、Objective C以及最近在Internet上广泛应用的Java语言；另一方面，用C++语言编写的程序未必是面向对象的，正如彩色电视机上播出的可能是黑白节目一样，用一种优秀的程序设计语言也可能写出非常糟糕的程序。学习面向对象程序设计并不只是学习面向对象程序设计语言。本书不仅介绍面向对象程序设计语言，而且引导读者采用面向对象的程序设计方法去解决实际问题。在介绍C++语言的各种机制时，我们力求使得读者不仅会使用它们，而且可以理解这些机制。只有这样，读者才可能容易地转向其他程序设计语言。

三、尽早引入类的概念，以开拓面向对象程序设计思维方式

类是构造面向对象程序的基本单元。类中的操作是以函数来表达的，在本书中函数更多地是以类中某种操作的形式出现的，而不是作为构造程序的基本单位。提早引入类的概念有助于初学者采用面向对象而不是传统结构化的思维方式去解决实际问题，有助于构造良好的程序结构，为日后处理大型程序打好基础。因而本书在介绍了所必需的基本数据类型与基本控制结构后，即引入函数的概念，紧接着就引入类的概念。

四、本书不打算成为一本C++语言手册大全

C++语言是一种优秀的面向对象程序设计语言，然而C++语言提供的程序设计机制十分复杂，本书不打算作为一本C++语言手册大全。我们的目标是从C++语言十分复杂的概念中整理出最核心的概念介绍给初学者，放弃那些明显为了效率的原因而引入的机制（如联合、内联成员等）或与面向对象程序设计原则有抵触的机制（如goto语句等）。

五、本书各章均精选典型习题

本书每一章均附有精选的典型习题，以帮助读者掌握该章节内容。习题中包括一些讨论题和实习题，供有兴趣的读者进一步学习。

采用本书学习面向对象程序设计有两条有效途径。途径之一是亲自动手多编写程序并上机调试。程序设计是一门实践性很强的学科，本书中的例题、习题中的程序均可作为实习内容。此外，在学习过程中如果遇到关于C++语言机制的疑难问题，也可尝试自己动手编写一些小程序上机解决。途径之二是阅读别人编写的程序，并从程序的可靠性、可理解性、可重用性、可维护性等方面对程序进行评价。这样既可从好的程序中吸取精华，又可从差的程序中吸取教训。

此外，多思考与多提问也是一种好的学习方法。学习一种新的语言机制时不仅要学会

使用它们，还应思考与提出一些问题：

为什么要引入这些机制？

这些机制在内部是如何实现的？

是否有其他的解决方式？

不同解决方式相比较有何异同？

通过这些问题，我们不仅学会了这些语言机制，也真正理解了这些语言机制。

本书既可供计算机专业本科学生作为程序设计课程的入门教材，也可供非计算机专业高年级学生作为面向对象程序设计课程的教材。任课教师可根据自己的教学目标、学生基础、实践经验以及学校特点对书中的部分内容进行调整。例如，抽象数据类型、多重继承与重复继承、运算符重载、程序断言、异常处理等内容均可根据需要进行扩展或裁剪，其中有些知识将在后续课程中做进一步深入探讨。

安排并指导学生上机实习对于学好本课程具有重要意义。任课教师既可在每一章布置不同的上机实习题，也可安排两三道典型上机实习题贯穿全书内容。例如，不断改进上机实习题 6－1 中的 MEGA_INTEGER 类可覆盖前 6 章及函数与运算符重载、输入/输出流、异常处理等知识单元，而比较 8.6 节给出的多态数据结构与 9.2 节给出的包容数据结构，则覆盖了继承机制、运行时多态性、类属机制、文件流等知识单元。

对于计算机专业本科学生，建议本课程课内学时数为 72 学时（其中含习题课 10 学时，除第一章与第十二章外每两章安排一次习题课），学生上机学时数共 40 学时。对于非计算机专业学生，课内学时数至少 54 学时，上机学时数不少于 36 学时。

如果安排 72 个课内学时、40 个上机学时，教学计划可参考以下学时数安排（各章后的括号中列出的分别是课内学时数与上机学时数，其中课内学时数不包括习题课）：第一章（2/2）、第二章（4/2）、第三章（6/4）、第四章（6/4）、第五章（6/4）、第六章（8/6）、第七章（8/4）、第八章（6/4）、第九章（4/4）、第十章（6/4）、第十一章（4/2）、第十二章（2/0）。

南京大学计算机科学系许满武教授认真审阅了书稿，并提出了宝贵的意见。中山大学计算机科学系张治国、乔琳、吴向军诸位老师多次参加了书稿的讨论，编者从中获益良多。在此谨向他们致以衷心的感谢。

由于编者水平所限，书中谬误之处在所难免，恳请广大读者不吝批评指正。

编　者

目　录

第一章　程序设计与C++语言初步

1.1　计算机程序

1.1.1　算法

计算机出现前，人类已经积累了许多解决问题的经验。解决问题时并不一定使用计算机，如果使用计算机，只不过在解决问题的时间、空间、精度等方面提供更大的方便而已。

给定两个正整数 p 和 q，如何求出 p 和 q 的最大公约数 g？对于这样一个问题，数学家欧几里德(Euclid)给出了一个解决方案，这个方案由三个步骤完成，如例 1.1.1 所示。

例 1.1.1　求解最大公约数的欧几里德算法。

步骤 1：如果 $p<q$，则交换 p 和 q。

步骤 2：令 r 是 p/q 的余数。

步骤 3：如果 $r=0$，则令 $g=q$ 并终止；

否则令 $p=q$，$q=r$ 并转向步骤 2。

按照上述步骤，可以逐步地计算出任意两个正整数的最大公约数。计算工具可以是纸和笔，也可以是先进的计算机。这种用来解决问题的、由有限多个步骤组成的具体过程称为**算法**(algorithm)。

从以上的简单算法，可以看出一个算法具有的基本特征。算法是由一些能够机械执行的操作组成的，欧几里德算法中包含了除法、比较、转向等操作；算法可以具有多个输入和输出，欧几里德算法的输入是正整数 p 和 q，输出是 p 和 q 的最大公约数 g。算法对于任何输入都应该是可终止的，欧几里德算法交替执行步骤 2 和步骤 3，执行步骤 3 时可能重新转向执行步骤 2，也可能终止算法，可以证明最多执行 $1+2q$ 步之后算法一定会终止。算法的主要操作对象是数据，欧几里德算法中除了输入/输出数据外，还包括保存中间计算结果的数据 r。

1.1.2　实体

除数学问题外，现实生活中的许多**行为**(behaviour)也可以用算法来表示。例如，在银行账户中存款或从银行账户中取款这两种行为可以表示为例 1.1.2 和例 1.1.3。

例 1.1.2　在银行账户中存款。

输入：存款金额 m 和当前余额 b。

输出：新余额 b'。

步骤：令 $b'=b+m$ 并终止。

例 1.1.3　从银行账户中取款。

输入：取款金额 m、当前余额 b 和透支限额 v。

输出：已取金额 m' 和新余额 b'。

步骤 1：如果 $m>b+v$，则令 $m'=0$，$b'=b$ 并转向步骤 2；

否则令 $m'=m$，$b'=b-m$ 并终止。

步骤 2：提示超额透支并终止。

这种表示方式忽略了很重要的一点，即存款行为与取款行为所作用的数据均包含账户当前的余额，存款与取款行为是密切相关的。

按照我们日常的思维习惯，例如银行账户这一类的个别事物通常被看做是一个**实体**(entity)，其中包括了账号、户名、地址、密码以及当前余额等信息，都可完成存款、取款等行为；而不同的账户实体所具有的信息可能是不相同的，例如每一个账户都有惟一的账号、自己的姓名、与别人不同的密码、当前的存款余额等。通常一个实体应具有一个名字、一组表示该实体特征的数据以及若干作用在这些数据上的行为。实体具有的数据表示了它的状态，而这些状态可由实体的行为来改变。

银行账户可描述为实体，如例 1.1.4 所示，存款与取款只是表现在这一实体上的行为。

例 1.1.4　银行账户实体。

实体：银行账户。

属性：账号、户名、地址、密码、当前余额 b、透支限额 v。

行为：1）存款

输入：存款金额 m。

输出：无。

步骤：令 $b=b+m$ 并终止。

2）取款

输入：取款金额 m。

输出：已取金额 m'。

步骤 1：如果 $m>b+v$，则令 $m'=0$ 并转向步骤 2；

否则令 $m'=m$，$b=b-m$ 并终止。

步骤 2：提示超额透支并终止。

从实体出发来把握事物或从行为出发来把握事物，其实是理解事物的两种不同思维方式：前者是从事物内在结构的角度出发，而后者则是从事物外部表现出来的功能出发。通过观察与分析事物表现出来的行为来探讨事物的内在结构，掌握了事物的内在结构后又可用

来解释或预测事物的行为。在处理复杂问题时,从实体出发比从行为出发更容易把握问题的复杂性。

1.1.3 程序

有了实体后,如何利用计算机来解决问题呢?**程序**即是实体在计算机中的体现。然而实体中的属性与行为是如何转换为程序而在计算机中工作的呢?必须首先了解计算机的工作原理。计算机内部只能表示0或1这样的二进制数,因而所有由计算机处理的数据和处理这些数据的程序都必须表示为二进制数。

1. 数据在计算机内部的表示

计算机中的数据可以分为数值型和字符型两大类。数值型数据包括整数、浮点数、有符号数、无符号数等,字符型数据包括字母、数字、标点等符号。无论是数值型还是字符型数据,在计算机内部都是通过二进制编码来表示的。由于二进制数书写困难且易产生错误,所以经常用八进制或十六进制数来表示,因为三者之间有比较直接的转换方法,如表1.1.1所示。

表 1.1.1 不同数制的对应表示

十进制(decimal)	二进制(binary)	八进制(octal)	十六进制(hexadecimal)
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

在计算机内部表示整数的方法有原码、补码和反码等表示方法。如果数值有符号,通常用最高有效位表示(如0表示+,1表示-)。实数的表示则有定点和浮点两种形式。在计算机内部表示数值型数据要特别注意容量问题,即给定字长和表示方法,可以表示的最大数