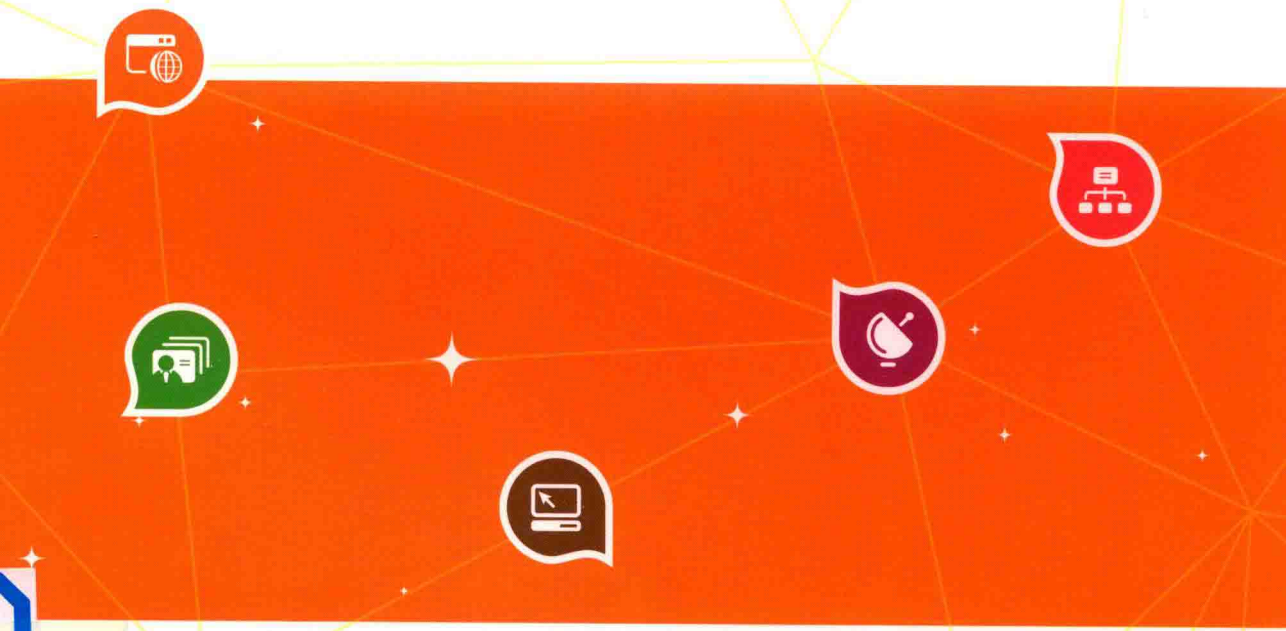
普通高等院校计算机基础教育“十三五”规划教材

C++面向对象程序设计

C++ MIANXIANG DUIXIANG CHENGXU SHEJI

李　文　黄丽韶　吕兰兰　主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等院校计算机基础教育“十三五”规划教材

C++面向对象程序设计

李　文　黄丽韶　吕兰兰　主　编

郭力勇　何　琛　扈乐华　周鹏　邵金侠　副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

C++继承了C语言效率高的优点，实现了面向对象技术的抽象、封装、继承和多态等核心特性，使得C++成为开发大型复杂软件的首选编程语言。

本教材以面向对象程序设计思想为主线，介绍使用C++语言进行程序设计的基本知识和方法。主要内容包括：C++语言基础，类与对象初步，数据的共享与保护，继承与派生，多态性，流类库与输入输出，异常处理，个人银行账户管理系统。

本书注重程序实例的合理性，注重引导读者理解并学会应用面向对象程序设计的思想和方法，力求从应用出发培养学生的学习兴趣，适合作为普通高等院校计算机及其相关专业本科生的教材。

图书在版编目（CIP）数据

C++面向对象程序设计/李文，黄丽韶，吕兰兰主编. —北京：中国铁道出版社，2018.2（2019.1重印）

普通高等院校计算机基础教育“十三五”规划教材

ISBN 978-7-113-24219-0

Ⅰ.①C… Ⅱ.①李… ②黄… ③吕… Ⅲ.①C++语言－程序设计－高等学校－教材Ⅳ.①TP312.8

中国版本图书馆CIP数据核字(2018)第023773号

书　　名：C++面向对象程序设计
作　　者：李　文　黄丽韶　吕兰兰　主编

策　　划：韩从付　　**读者热线**：（010）63550836
责任编辑：刘丽丽　李学敏
封面设计：刘　颖
责任校对：张玉华
责任印制：郭向伟

出版发行：中国铁道出版社（100054，北京市西城区右安门西街8号）
网　　址：http://www.tdpress.com/51eds/
印　　刷：三河市航远印刷有限公司
版　　次：2018年2月第1版　2019年1月第2次印刷
开　　本：787 mm×1 092 mm　1/16　**印张**：13　**字数**：305千
书　　号：ISBN 978-7-113-24219-0
定　　价：40.00元

前 言

C++作为一门优秀的面向对象编程语言，已经成为近十年来最流行、应用范围最广的编程语言之一，被广泛应用于众多工程技术领域。C++在继承 C 语言效率高的优点的同时，还实现了面向对象技术的抽象、封装、继承和多态等核心特性。由于这些特性，使得面向对象程序相比传统的结构化程序而言，具有更高的可复用性、可扩展性和可维护性。这使得 C++成为开发大型复杂软件的首选编程语言。同时，C++面向对象程序设计也成为计算机科学与技术、软件工程等相关专业的基础课程之一。

C++既支持面向过程的程序设计，也支持面向对象的程序设计。因此，大多数 C++书籍中通常既包含面向过程的内容，也包含面向对象的内容，书中有大量篇幅介绍 C 语言中的知识。考虑到计算机科学与技术、软件工程等相关专业的学生，在学习 C++面向对象程序设计课程之前，大多已学过 C 语言程序设计课程。因此，本书在内容组织安排上选择了以面向对象为主的方式，书中包含的 C 语言面向过程设计部分的内容较少。

本书编者长期从事计算机科学与技术专业、软件工程专业的 C++程序设计教学工作，在教学中遇到了一些问题，例如：C++语法规则繁多，学生很难完全理解，容易导致畏难情绪；学生学习语法知识后不了解其应用方法，在程序开发时无法灵活应用所学知识等。同时，本书编者也积累了一些教学经验，因此萌生了编写一本 C++教材的想法。

基于作者在 C++程序设计教学实践中遇到的问题，本书注重程序实例的合理性，注重引导读者理解并学会应用面向对象程序设计的思想和方法，力求从应用出发培养学生的学习兴趣。例如，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相应知识点，在此基础上再力求使读者达到运用相应知识点的目标；对教学中的重点和难点内容，精心设计实例进行细致分析。

本书的主要特色如下：

1. 注重基础

本书较全面地介绍了 C++的基本语法和相关知识，并将面向对象设计的思想融合于问题的解决中。本书面向大学计算机相关专业的低年级学生，建议读者最好具有一定的 C 语言程序设计基础。

2. 深入浅出

本书力求用简洁浅显的语言讲述复杂的概念，力求做到通俗易懂、深入浅出。本书的宗旨是让读者不但要知其然，还要知其所以然，因此对于 C++的一些语法特性，不但介绍如何使用，还会介绍 C++为什么会有这个语法特性。

3. 方便裁剪

书中每个章节安排适当，符合计算机相关专业的教学需求，不同学校可以针对自身的教学特点，选择不同的章节组合进行教学，教师也可以根据授课对象的实际情况进行灵活裁剪。

本书内容上主要分 9 章。

绪论：介绍了程序设计语言的历史和特点，面向对象方法的基本概念，面向对象的软件开发过程，以及程序开发的基本概念。

C++语言基础：介绍了 C++程序设计的基础知识，简述了 C++语言与 C 语言的区别，并介绍 C++语言对 C 语言进行的扩展，包括基本数据类型和自定义数据类型、数据的输入与输出、三种基本控制结构（顺序、选择和循环结构）、指针和引用、函数重载等。

类与对象初步：介绍了面向对象程序设计的一大特性——封装。从数据封装的角度，介绍了面向对象程序设计中的核心概念——类，包括类的声明和实现，类成员的初始化（类的构造函数和析构函数），以及如何使用类解决具体问题（类的组合）。此外，还对比了类与结构体和联合体的区别，简单介绍了如何用 UML 描述类的特性。

数据的共享与保护：从数据共享的角度，介绍了标识符的作用域与可见性、对象的生存期、类的静态成员、类的友元；从保护共享数据的角度，介绍了常对象和常引用。最后介绍了多文件结构和编译预处理命令。

继承与派生：介绍了面向对象程序设计的另一大特性——继承。介绍了继承与派生的基本概念、派生类的声明和实现方式、派生类成员的初始化（派生类的构造函数和析构函数），讨论了三种不同派生方式下派生类对基类成员的访问控制方式，以及公有派生下派生类和基类的兼容规则，分析了多继承存在的二义性问题并提出了虚基类的解决方案。最后，分析了在解决实际问题时，如何合理运用类与类之间的继承、组合与使用关系。

多态性：介绍了面向对象程序设计的另一大特性——多态。简单介绍了多态的概念，详述 C++支持的两种多态类型：静态多态（函数重载）和动态多态（虚函数），分析了虚析构函数的声明方式和必要性，以及纯虚函数和抽象类的使用场合。

流类库与输入/输出：介绍了输入/输出流的概念，以及 C++输入/输出流类库的结构和使用方法。

异常处理：讲述了异常处理概念和基本思想，以及 C++异常处理机制的实现，还简单介绍了异常处理中析构函数的处理和标准程序库异常处理。

个人银行账户管理系统：本章主要的目的是培养学生综合运用面向对象程序设计基本知识和方法进行项目设计的能力；初步培养学生运用软件工程思想进行项目设计的能力。

本书由湖南科技学院电子与信息工程学院组织编写，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。本书由李文、黄丽韶、吕兰兰任主编，郭力勇、何琛、扈乐华、周鹏、邵金侠任副主编，其中第 1、2 章由李文编写，第 3、4 章由黄丽韶编

写，第5、6章由吕兰兰编写，第7章由郭力勇编写，第8章由何琛、周鹏编写，第9章由扈乐华、邵金侠编写。全书由李文、黄丽韶、吕兰兰统稿。在本书的写作过程中，编者参考了国内外许多优秀的同类教材以及网络资源，在此向其作者表示衷心感谢。感谢所有支持和帮助过本书编写的人们。

感谢以下项目的资助：

【1】湖南省普通高等学校“十三五”专业综合改革试点项目（湘教通〔2016〕276号）；

【2】湖南省普通高校校企合作创新创业教育基地（湘教通〔2016〕436号）；

【3】湖南科技学院计算机应用技术重点学科建设项目；

【4】教育部高教司产教合作项目（201601021003，201701034028，201702065165）；

【5】湖南省教育厅教改项目（湘教通〔2016〕400号）；

【6】湖南省教育科学“十三五”规划课题（XJK17QGD008）。

虽然编者在高校从事了多年的C++教学，但对这门与时俱进的语言仍然有不能掌控的地方。由于编者水平所限，加之时间仓促，书中难免存在不当和疏漏之处，恳请广大读者及同仁们批评指正，以便于我们在今后的版本中进行改进。

编　者

2017年12月

目 录

第1章 绪论

本章首先从发展的角度概要介绍面向对象程序设计语言的产生和特点、面向对象方法的由来及其基本概念，以及什么是面向对象程序设计的软件工程，最后介绍程序的开发过程。

1.1 计算机程序设计语言的发展

语言是一套具有语法、词法规则的系统。语言是思维的工具，思维通过语言来表述。计算机程序设计语言是计算机可以识别的语言，用于描述解决问题的方法，供计算机阅读和执行。

1.1.1 机器语言与汇编语言

从1946年2月世界上第一台数字电子计算机ENIAC诞生以来，在这短暂的70多年间，计算机科学得到了迅猛发展，计算机及其应用已渗透到社会的各个领域，有力地推动了整个信息化社会的发展，计算机已成为信息化社会中必不可少的工具。

计算机系统包括硬件和软件。计算机之所以有如此强大的功能，不仅因为它具有强大的硬件系统，而且依赖于软件系统。软件包括了使计算机运行所需的各种程序及其有关的文档资料。计算机的工作是用程序来控制的，离开了程序，计算机将一事无成。程序是指令的集合。软件工程师将解决问题的方法、步骤编写为由一条条指令组成的程序，输入到计算机的存储设备中。计算机执行这一指令序列，便可完成预定的任务。

所谓指令，就是计算机可以识别的命令。虽然在人类社会中有丰富的语言用来表达思想、交流感情、记录信息，但计算机却不能识别它们。计算机所能识别的指令形式，只能是简单的0和1的组合。一台计算机硬件系统能够识别的所有指令的集合称为它的指令系统。

由计算机硬件系统可以识别的二进制指令组成的语言称为机器语言。毫无疑问，虽然机器语言便于计算机识别，但对于人类来说却是晦涩难懂，更难以记忆。可是在计算机发展的初期，软件工程师们只能用机器语言来编写程序。这一阶段，在人类的自然语言和计算机编程语言之间存在着巨大的鸿沟，软件开发的难度大、周期长，开发出的软件功能却很简单，界面也不友好。

不久，出现了汇编语言，它将机器指令映射为一些可以被人读懂的助记符，如ADD、SUB等。此时编程语言与人类自然语言间的鸿沟略有缩小，但仍与人类的思维相差甚大。因为它的抽象层次太低，程序员需要考虑大量的机器细节。

尽管如此，从机器语言到汇编语言，仍是一大进步。这意味着人与计算机的硬件系统不必非得使用同一种语言。程序员可以使用较适合人类思维习惯的语言，而计算机硬件系统仍只识别机器指令。那么两种语言间的沟通如何实现呢？这就需要一种翻译工具（软件）。汇编语言

的翻译软件称为汇编程序，它可以将程序员写的助记符直接转换为机器指令，然后再由计算机去识别和执行。

1.1.2 高级语言

高级语言的出现是计算机编程语言的一大进步。它屏蔽了机器的细节，提高了语言抽象层次，程序中可以采用具有一定含义的数据命名和容易理解的执行语句。这使得在书写程序时可以联系到程序所描述的具体事物。

20 世纪 60 年代末开始出现的结构化编程语言进一步提高了语言的层次。结构化数据、结构化语句、数据抽象、过程抽象等概念使程序更便于体现客观事物的结构和逻辑含义，使得编程语言与人类的自然语言更接近。但是二者之间仍有不少差距，主要问题是程序中的数据和操作分离，不能够有效地组成与自然界中的具体事物紧密对应的程序成分。

目前应用比较广泛的几种高级语言有 FORTRAN，BASIC，Pascal，C 等。当然本书介绍的 C++语言也是高级语言，但它与其他面向过程的高级语言有着根本的不同。

1.1.3 面向对象的语言

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，它设计的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。

开发一个软件是为了解决某些问题，这些问题所涉及的业务范围称为该软件的问题域。面向对象的编程语言将客观事物看作具有属性和行为（或称服务）的对象，通过抽象找出同一类对象的共同属性（静态特征）和行为（动态特征），形成类。通过类的继承与多态可以很方便地实现代码重用，大大缩短了软件开发周期，并使得软件风格统一。因此，面向对象的编程语言使程序能够比较直接地反映问题域的本来面目，软件开发人员能够利用人类认识事物所采用的一般思维方法来进行软件开发。

面向对象的程序设计语言经历了一个很长的发展阶段。例如，LISP 家族的面向对象语言、Simula 67 语言、Smalltalk 语言以及 CLU，Ada，Modula-2 等语言，或多或少地都引入了面向对象的概念，其中 Smalltalk 是第一个真正的面向对象的程序语言。

然而，应用最广的面向对象程序语言是在 C 语言基础上扩充出来的 C++语言。由于 C++对 C 兼容，而 C 语言又早已被广大程序员所熟知，所以，C++语言也就理所当然地成为应用最广的面向对象程序语言。

1.2 面向对象的方法

程序设计语言是编写程序的工具，因此程序设计语言的发展恰好反映了程序设计方法的演变过程。下面首先初步介绍面向对象方法的基本概念和基本思想，学习完本书之后，相信你会对面向对象的方法有一个深入、完整的认识。

1.2.1 面向对象方法的由来

在面向对象的方法出现以前，我们都是采用面向过程的程序设计方法。早期的计算机是用于数学计算的工具，例如，用于计算炮弹的飞行轨迹。为了完成计算，就必须设计出一个计算

方法或解决问题的过程。因此，软件设计的主要工作就是设计求解问题的过程。

随着计算机硬件系统的高速发展，计算机的性能越来越强，用途也更加广泛，不再仅限于数学计算。由于所处理的问题日益复杂，程序也就越来越复杂和庞大。20 世纪 60 年代产生的结构化程序设计思想，为使用面向过程的方法解决复杂问题提供了有力的手段。因而，在 20 世纪 70 年代到 80 年代，结构化程序设计方法成为所有软件开发设计领域及每个程序员都采用的方法。结构化程序设计的思路是：自顶向下、逐步求精；其程序结构是按功能划分为若干个基本模块，这些模块形成一个树状结构；各模块之间的关系尽可能简单，在功能上相对独立；每一模块内部均是由顺序、分支和循环 3 种基本结构组成；其模块化实现的具体方法是使用子程序。结构化程序设计由于采用了模块分解与功能抽象以及自顶向下、分而治之的方法，从而有效地将一个较复杂的程序系统设计任务分解成许多易于控制和处理的子任务，便于开发和维护。

虽然结构化程序设计方法具有很多优点，但它仍是一种面向过程的程序设计方法。它把数据和处理数据的过程分离为相互独立的实体，当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于老问题的新方法都要带来额外的开销，程序的可重用性差。另外，由于图形用户界面的应用，使得软件使用起来越来越方便，但开发起来却越来越困难。一个好的软件，应该随时响应用户的任何操作，而不是请用户按照既定的步骤循规蹈矩地使用。例如，我们都熟悉文字处理程序的使用，一个好的文字处理程序使用起来非常方便，几乎可以随心所欲，软件说明书中绝不会规定任何固定的操作顺序，因此对这种软件的功能很难用过程来描述和实现，如果仍使用面向过程的方法，开发和维护都将很困难。

那么，什么是面向对象的方法呢？首先，它将数据及对数据的操作方法放在一起，作为一个相互依存、不可分离的整体——对象。对同类型对象抽象出其共性，形成类。类中的大多数数据，只能用本类的方法进行处理。类通过一个简单的外部接口与外界发生关系，对象与对象之间通过消息进行通信。这样，程序模块间的关系更为简单，程序模块的独立性、数据的安全性就有了良好的保障。另外，通过后续章节中介绍的继承与多态性，还可以大大提高程序的可重用性，使得软件的开发和维护都更为方便。

面向对象的方法有如此多的优点，然而对于初学程序设计的人来说，是否容易理解、容易掌握呢？回答是肯定的。面向对象方法的出现，实际上是程序设计方法发展的一个返璞归真过程。软件开发从本质上讲，就是对软件所要处理的问题域进行正确的认识，并把这种认识正确地描述出来。面向对象方法所强调的基本原则，就是直接面对客观存在的事物来进行软件开发，将人们在日常生活中习惯的思维方式和表达方式应用在软件开发中，使软件开发从过分专业化的方法、规则和技巧中回到客观世界，回到人们通常的思维方式。

1.2.2　面向对象的基本概念

下面简单介绍面向对象方法中的几个基本概念。当然我们不能期望通过几句话的简单介绍就完全理解这些概念，在本书的后续章节中，会不断帮助读者加深对这些概念的理解，以达到熟练运用的目的。

1. 对象

从一般意义上讲，对象是现实世界中一个实际存在的事物，它可以是有形的（一辆汽车），也可以是无形的（一项计划）。对象是构成世界的一个独立单位，它具有自己的静态特征（可

以用某种数据来描述）和动态特征（对象所表现的行为或具有的功能）。

面向对象方法中的对象，是系统中用来描述客观事物的一个实体，它是用来构成系统的一个基本单位。对象由一组属性和一组行为构成。属性是用来描述对象静态特征的数据项，行为是用来描述对象动态特征的操作序列。

2. 类

把众多的事物归纳、划分成一些类，是人类在认识客观世界时经常采用的思维方法。分类所依据的原则是抽象，即忽略事物的非本质特征，只注意那些与当前目标有关的本质特征，从而找出事物的共性，把具有共同性质的事物划分为一类，得出一个抽象的概念。例如，石头、树木、汽车、房屋等都是人们在长期的生产和生活实践中抽象出的概念。

面向对象方法中的“类”，是具有相同属性和服务的一组对象的集合。它为属于该类的全部对象提供了抽象的描述，其内部包括属性和行为两个主要部分。类与对象的关系犹如模具与铸件之间的关系，一个属于某类的对象称为该类的一个实例。

3. 封装

封装是面向对象方法的一个重要原则，就是把对象的属性和服务结合成一个独立的系统单位，并尽可能隐蔽对象的内部细节。这里有两个含义：第一个含义是把对象的全部属性和全部服务结合在一起，形成一个不可分割的独立单位。第二个含义也称作“信息隐蔽”，即尽可能隐蔽对象的内部细节，对外形成一个边界（或者说一道屏障），只保留有限的对外接口使之与外部发生联系。

4. 继承

继承是面向对象技术能够提高软件开发效率的重要原因之一，其定义是，特殊类的对象拥有一般类的全部属性与服务，称作特殊类对一般类的继承。

继承具有重要的实际意义，它简化了人们对事物的认识和描述。比如我们认识了轮船的特征之后，再考虑客轮时，因为知道客轮也是轮船，于是可以认为它理所当然地具有轮船的全部一般特征，从而只需要把精力用于发现和描述客轮独有的那些特征。继承对于软件复用有着重要意义，使特殊类继承一般类，本身就是软件复用。不仅于此，如果将开发好的类作为构件放到构件库中，在开发新系统时便可以直接使用或继承使用。

5. 多态性

多态性是指在一般类中定义的属性或行为，被特殊类继承之后，可以具有不同的数据类型或表现出不同的行为。这使得同一个属性或行为在一般类及其各个特殊类中具有不同的语义。例如，可以定义一个一般类“几何图形”，它具有“绘图”行为，但这个行为并不具有具体含义，也就是说并不确定执行时到底画一个什么样的图（因此不知道“几何图形”到底是一个什么图形，“绘图”行为当然也就无从实现）。然后再定义一些特殊类，如“椭圆”和“多边形”，它们都继承一般类“几何图形”，因此也就自动具有了“绘图”行为。接下来，可以在特殊类中根据具体需要重新定义“绘图”，使之分别实现画椭圆和多边形的功能。进而，还可以定义

“矩形”类继承“多边形”类，在其中使“绘图”实现绘制矩形的功能。这就是面向对象方法中的多态性。

1.3　面向对象的软件开发

在整个软件开发过程中，编写程序只是相对较小的一个部分。软件开发的真正决定性因素来自前期概念问题的提出，而非后期的实现问题。只有识别、理解和正确表达了应用问题的内在实质，才能做出好的设计，然后才是具体的编程实现。

早期的软件开发所面临的问题比较简单，从认清要解决的问题到编程实现并不是太难的事。随着计算机应用领域的扩展，计算机所处理的问题日益复杂，软件系统的规模和复杂度增加，以至于软件的复杂性和其中包含的错误已达到软件人员无法控制的程度，这就是 20 世纪 60 年代初期的“软件危机”。软件危机的出现，促进了软件工程学的形成与发展。

我们学习面向对象的程序设计，首先应该对软件开发和维护的全过程有一个初步了解。因此，在这里先简要介绍一下什么是面向对象的软件工程。面向对象的软件工程是面向对象方法在软件工程领域的全面应用。它包括面向对象的分析（OOA）、面向对象的设计（OOD）、面向对象的编程（OOP）、面向对象的测试（OOT）和面向对象的软件维护（OOSM）等主要内容。

1.3.1　分析

在分析阶段，要从问题的陈述着手，建立一个说明系统重要特性的真实情况模型。为理解问题，系统分析员需要与客户一起工作。系统分析阶段应该扼要精确地抽象出系统必须做什么，而不是关心如何去实现。

面向对象的系统分析，直接用问题域中客观存在的事物建立模型中的对象，无论是对单个事物还是对事物之间的关系，都保留它们的原貌，不做转换，也不打破原有界限而重新组合，因此能够很好地映射客观事物。

1.3.2　设计

在设计阶段，是针对系统的一个具体实现运用面向对象的方法。其中包括两方面的工作，一是把 OOA 模型直接搬到 OOD，作为 OOD 的一部分；二是针对具体实现中的人机界面、数据存储、任务管理等因素补充一些与实现有关的部分。

1.3.3　编程

编程是面向对象的软件开发最终落实的重要阶段。在 OOA 和 OOD 理论出现之前，程序员要写一个好的面向对象的程序，首先要学会运用面向对象的方法来认识问题域，所以 OOP 被看作一门比较高深的技术。现在，OOP 的工作比较简单了，认识问题域与设计系统成分的工作已经在 OOA 和 OOD 阶段完成。OOP 工作就是用一种面向对象的编程语言把 OOD 模型中的每个成分书写出来。

尽管如此，我们学习面向对象的程序设计仍然要注重学习基本的思考过程，而不能仅仅学习程序的实现技巧。因此，虽然本书面向的是初学编程的读者，介绍的主要是 C++语言和面向对象的程序设计方法，但仍然用了一定的篇幅，通过例题介绍设计思路。

1.3.4 测试

测试的任务是发现软件中的错误，任何一个软件产品在交付使用之前都要经过严格的测试。在面向对象的软件测试中继续运用面向对象的概念与原则来组织测试，以对象的类作为基本测试单位，可以更准确地发现程序错误，提高测试效率。

1.3.5 维护

无论经过怎样的严格测试，软件中通常还是会存在错误。因此软件在使用的过程中，需要不断地维护。

使用面向对象的方法开发的软件，其程序与问题域是一致的，软件工程各个阶段的表示是一致的，从而减少了维护人员理解软件的难度。无论是发现了程序中的错误而追溯到问题域，还是因需求发生变化而追踪到程序，道路都是比较平坦的；而且对象的封装性使一个对象的修改对其他对象的影响很少。因此，运用面向对象的方法可以大大提高软件维护的效率。

读者在初学程序设计的时候，教科书中的例题都比较简单，从这些简单的例题中读者很难体会到软件工程的作用。而且题目本身往往已经对需要解决的问题做了清楚准确的描述。尽管如此，我们也不应直接开始编程，而应该首先进行对象设计。当然，本书主要的目的是介绍编程方法，建议读者在熟练掌握了 C++语言编程技术后，另外专门学习面向对象的软件工程。

1.4 程序开发的基本概念

在学习编程之前，首先来简单了解一下程序的开发过程及基本术语。在后续章节的学习和编程实践中，读者将对它们有不断深入的理解。

1.4.1 基本术语

源程序：用源语言编写的、有待翻译的程序，称为“源程序”。源语言可以是汇编语言，也可以是高级程序设计语言（比如 C++语言），用它们写出的程序都是源程序。

目标程序：是源程序通过翻译加工以后所生成的程序。目标程序可以用机器语言表示（因此也称之为“目标代码”），也可以用汇编语言或其他中间语言表示。

翻译程序：是指用来把源程序翻译为目标程序的程序。对翻译程序来说，源程序是它的输入，而目标程序则是其输出。翻译程序有 3 种不同类型：汇编程序、编译程序、解释程序。

汇编程序：其任务是把用汇编语言写成的源程序翻译成机器语言形式的目标程序。所以，用汇编语言编写的源程序先要经过汇编程序的加工，变为等价的目标代码。

编译程序：若源程序是用高级程序设计语言所写，经翻译程序加工生成目标程序，那么，该翻译程序就称为“编译程序”。所以，高级语言编写的源程序要上机执行，通常首先要经编译程序加工成为机器语言表示的目标程序。若目标程序是用汇编语言表示，则还要经过一次汇编程序的加工。

解释程序：这也是一种翻译程序，同样是将高级语言源程序翻译成机器指令。它与编译程序的不同点就在于：它是边翻译边执行的，即输入一句，翻译一句，执行一句，直至将整个源