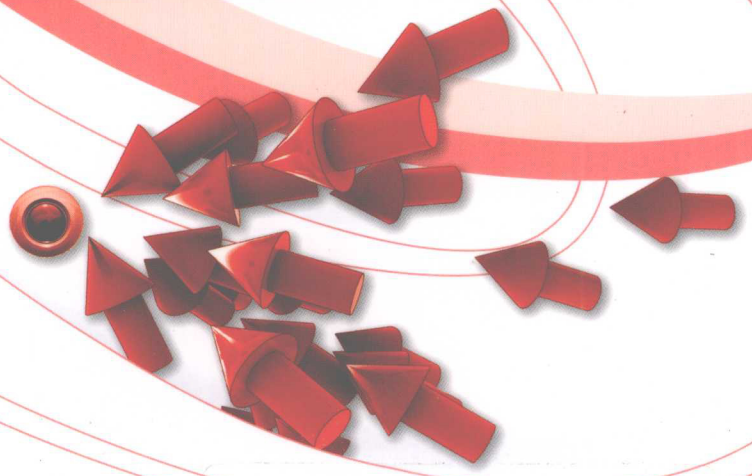

高等学校计算机规划教材

C++面向对象程序设计

姚全珠 主编　　李 薇 王晓帆 副主编

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

高等学校计算机规划教材

C++面向对象程序设计

姚全珠　主编

李　薇　王晓帆　副主编

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书针对已有 C 程序设计基础、学习 C++面向对象程序设计的读者。全书分为 12 章，第 1 章首先介绍面向对象程序设计的思想和基本概念，以及 C 与 C++的关系；在第 2、3、6 章介绍 C++类、对象、重载的概念及使用方法；第 4、5 章详细介绍常量及变量的生存期与作用域；第 7、8 章详细讲解类的继承、组合、多态及模板问题；第 9、10、11 章详细介绍 I/O 流、异常处理及 Visual C++ 2008 环境；最后给出了一个综合实例。全书内容由浅入深，采用案例教学的方法，力求将复杂的概念用简洁浅显的语言表达，并用实例对方法进行说明。书中还配有大量的习题。本书配有电子课件、习题解答等教学资源。

本书可作为高等学校 C++面向对象程序设计课程的教材，也可作为工程技术人员的参考书。

图书在版编目(CIP)数据

C++面向对象程序设计/姚全珠主编. —北京：电子工业出版社，2010.8
高等学校计算机规划教材
ISBN 978-7-121-11427-4

I. ①C… II. ①姚… 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(2010)第 140872 号

策划编辑：史鹏举
责任编辑：史鹏举
印　　刷：北京市李史山胶印厂
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1092　1/16　印张：19　字数：548 千字
印　　次：2010 年 8 月第 1 次印刷
定　　价：29.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：(010) 88258888。

前　言

面向对象的程序设计(OOP)已成为目前大型程序设计的主流方法，由于其具有的封装、继承、多态等特点，使设计者可以方便地将现实世界的对象抽象封装在一起，对象中既描述对象的属性(数据)，也有处理这些数据的方法，形成一个封装体黑盒，其他对象要和它交互时可方便地通过它所提供的接口来实现，这就保证了对象的稳定与安全特性；为了最大限度地实现代码复用，在面向对象程序设计方法中又提供了继承方法，它允许子类继承父类的所有属性与方法，并可以灵活地在子类中对从父类继承来的属性和方法进行扩充和修改，以实现子类的特例化；为了实现处理方法的同名意不同(函数名相同，具体处理的参数数据类型及个数，以及处理过程可能不相同)，在面向对象程序设计中又提供了多态性处理方法，允许对函数和运算符重载，增强了程序处理的灵活性。

C++是从 C 语言发展而来的，全面兼容了 C 语言，是一种面向对象的编程语言。对于具有 C 语言基础的人来说，学习 C++会非常容易。本书是 Visual C++.NET 入门教材，在 C 语言的基础之上，紧密结合 C++标准，从 C 语言顺利过渡到 C++语言，涵盖了该语言的主要特征，使初学者能很快学习掌握 C++。本书在内容组织上采用案例教学的思想，由浅入深，对每个 C++的理论方法从需求到应用做了详细的描述。从基本的数据单元“类”开始点滴扩展，逐步深入，讲述 C++设计中的重要理念：面向对象的封装、继承、多态等方法，揭示 C++设计的初衷。并在各章章后配备了相应的习题与思考题，有助于 C++学习人员灵活掌握各知识点。

本书作者长期从事 C++项目开发及本科生 C++程序设计课程的教学工作，具有丰富的教学与程序开发经验。作者也主持和参与了多项教改项目，以及相应的省级精品课程建设工作。

全书共分 12 章，第 1 章介绍面向对象的设计方法和几种常用的面向对象语言；第 2 章为类与对象；第 3 章为函数重载与内联函数；第 4 章为常量与引用；第 5 章为静态与命名控制；第 6 章为运算符重载；第 7 章为组合、继承与多态性；第 8 章为模板；第 9 章为输入/输出流；第 10 章为异常处理；第 11 章为 Visual C++ 2008 开发环境；第 12 章为综合实例。本书所有例题均在 Visual C++ 2008 下调试通过。全书由姚全珠教授负责，并组织课堂教学一线的教师共同完成。其中，第 1、2、3、5、6、8 章由李薇编写，第 4、7、9、10、11、12 章由王晓帆编写，由姚全珠统稿。

本书配有电子课件、习题答案等教学资源，需要者可从华信教育资源网 http://www.hxedu.com.cn 免费注册下载。

在本书的编写过程中，得到电子工业出版社史鹏举编辑的大力帮助，对史鹏举编辑及电子工业出版社的有关工作人员表示衷心的感谢!

由于作者水平有限，时间仓促，难免有疏漏和错误之处，敬请各位专家和读者批评指正。

咨询、意见和建议，可反馈至本书责任编辑邮箱：shipj@phei.com.cn。

编　者

目　录

第1章　面向对象的方法学

学习目标

(1) 了解面向对象技术的发展历程。
(2) 了解面向对象软件开发的过程。
(3) 掌握面向对象程序设计的特点。
(4) 掌握面向对象程序设计的相关术语。
(5) 了解面向对象的编程风格。
(6) 了解目前常用的面向对象程序设计语言。

传统的软件开发方法曾经给软件产业带来了巨大的进步，尤其是在开发中小规模软件项目时获得了成功。但是随着硬件性能的提高和图形用户界面的推广，软件的应用更加普及与深入，当开发大型软件产品时，由于面对的问题越来越复杂，在使用传统软件开发方法时，成功率较低。

随着面向对象编程语言 Simula 67 中首次引入了类和对象的概念，人们逐渐开始注重面向对象分析和面向对象设计的研究，因此产生了面向对象方法学。到了 20 世纪 90 年代，面向对象方法学已经成为人们在开发软件时的主流软件设计方法。

1.1　面向对象的发展

OO 方法(Object-Oriented Method，面向对象方法)是一种把面向对象思想应用于软件开发过程中，指导开发活动的系统方法，简称 OO 方法，它是建立在“对象”概念基础上的方法学。OO 方法起源于面向对象的编程语言(简称为 OOPL)。20 世纪 50 年代后期，在用 FORTRAN 语言编写大型程序时，常出现变量名在程序不同部分发生冲突的问题。因此，ALGOL 语言的设计者在 ALGOL 60 中采用了以“Begin……End”为标识的程序块，使程序块内的变量名变成局部的，以避免它们与程序中块外的同名变量相冲突。这是编程语言中首次提供封装(保护)的尝试。此后程序块结构广泛用于高级语言，如 Pascal、Ada、C 之中。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、76 持续不断的研究和改进之后，于 1980 年推出商品化的 Smalltalk 80，它在系统设计中强调对象概念的统一，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，使人们注意到 OO 方法所具有的模块化、信息封装与隐蔽、继承性、多样性等独特之处，这些优异特性为研制大型软件，提高软件可靠性、可重用性、可扩充性和可维护性，提供了有效的手段和途径。

C++是美国贝尔实验室的 Bjarne Stroustrup 博士在 C 语言的基础上，弥补了 C 语言存在的一些缺陷，增加面向对象的特征，于 1980 年开发出的一种过程性与对象性相结合的程序设计语言。最初他把这种新的语言叫做“含类的 C”，到 1983 年才取名为 C++。为了使 C++具有良好的可移植性，1990 年，美国国家标准学会(American National Standards Institute, ANSI)设立了委员会专门负责制定 C++标准。接着，国际化标准组织(International Organization for Standardization, ISO)也成立了自己的委员

会。同年，ANSI 和 ISO 将这两个委员会合并，统称为 ANSI/ISO，共同合作进行标准化工作。1998 年发布了 C++国际标准。20 世纪 90 年代中期由 Booch、Rumbaugh 和 Jacoson 共同提出了统一建模语言 UML(Unified Modeling Language)，把众多面向对象分析和设计方法综合成一种标准，使面向对象的方法成为主流的软件开发方法。目前面向对象方法已被广泛应用于程序设计语言、形式定义、设计方法学、操作系统、分布式系统、人工智能、实时系统、数据库、人机接口、计算机体系结构、并发工程、综合集成工程等，在许多领域的应用都得到了很大的发展。

1.2 面向对象方法学概述

传统的软件开发方法采用结构化技术(结构化分析、结构化设计和结构化实现)来完成软件开发的各项任务，该方法强调的是将一个较为复杂的任务分解成许多易于控制和处理的子任务，自顶向下顺序地完成软件开发各阶段的任务。然而，人类认识客观世界、解决现实问题的过程实际上是一个渐进的过程。人类的认识是在继承已有的有关知识的基础上，经过多次反复才能逐步深化。面向对象方法学就是尽量模拟人类习惯的思维方式，使软件开发的方法与过程尽可能接近人类认识世界、解决问题的方法与过程，从而使描述问题的问题空间(即问题域)与实现解法的解空间(即求解域)在结构上尽可能一致。

软件设计的基本目标是为了解决日常生活中存在的各种实际问题。面向过程是将要处理的问题转变为数据和过程两个相互独立的实体来对待，强调的是过程。当存储数据的数据结构需要变更的时候，必须修改与之有关的所有模块。例如，学生信息管理系统，该系统所处理的学生类型是研究生，允许用户进行输入学生信息、输出学生信息、插入(学生)、删除(学生)、查找(学生)等操作。这时如果需要再增加一种学生类型——在职研究生，则原来的程序就不能处理了，因为学生类型不同，不同类型的学生对应不同的处理，因此就需要重新编写程序代码。因此，面向过程的程序可重用性差，维护代价高。面向对象是将客观事物看做具有属性和行为的对象，通过对客观事物的抽象找出同一类对象的共同属性(静态属性)和行为(动态特征)，形成类。每个对象有自己的数据、操作、功能和目的。通过对类的继承与派生、多态等技术提高软件代码的可重用性。例如，在上例中，采用面向对象的思想，可以先定义一个学生类，包括学生的基本信息如姓名、年龄、班级等和学生所对应的相应的操作；当需要再增加一种学生类型时，可以采用继承和派生的方式，在学生类的基础上派生出一个新类，这样该新类不仅可以继承学生类的所有特性，而且可以根据需要增加必要的程序代码，从而避免了公用代码的重复开发，实现了代码重用。此外在解决问题时，面向对象的思想与人类处理问题的过程是一致的。例如，挪开凳子，人类处理问题的过程是拿起凳子，移到一边。面向过程的思想是将凳子和挪开作为两个实体来对待(描述凳子的数据和移动凳子的动作)；面向对象的思想是选择一个对象——凳子，然后向这个对象施加一个动作——挪开。由此可见，面向对象程序设计思想更接近人类的思维活动。

了解面向对象软件工程的基本概念有助于掌握面向对象软件的开发与设计。面向对象软件工程是面向对象软件方法在软件工程领域的全面应用，包括面向对象分析(OOA)、面向对象设计(OOD)、面向对象实现等重要内容。

1.2.1 面向对象分析

分析是问题抽象(即做什么)。结构化方法采用面向过程的方法对问题进行分解，由于对过程的理解不同，面向过程的功能细分所分割出的功能模块有时会因人而异。面向对象分析是指在深入、全面理解问题本质需求的基础上，准确地抽象出系统必须做什么，提炼出面向对象软件开发中所需的各种要素，即确定类与对象、属性，建立继承关系，确定服务等。这样通过对对象细分，从同一问题领

域的对象出发，不同人得出相同结论的概率较高。在分析过程中，系统分析员还应与用户反复进行讨论、协商，以便能正确提炼出用户的需求；此外，系统分析员还应深入理解用户需求，在此基础上抽象出目标系统本质属性，并用模型准确地表示出来。

1.2.2　面向对象设计

分析是提取和整理用户需求，建立问题精确模型的过程，即做什么。设计是问题求解（即怎么做），是对分析阶段所建立的模型进行精雕细凿，并逐渐扩充的一个过程。也就是用面向对象观点建立求解域模型的过程。

优秀的设计是权衡了各种因素，从而使得系统在其整个生命周期中的总开销最小的设计。对大多数软件系统而言，60%以上的软件费用都用于软件维护。因此，优秀软件设计的一个主要特点就是容易维护。在面向对象设计过程中，应遵循软件工程中关于软件设计的基本准则。

(1) 模块化。面向对象设计中，对象就是模块，它是把数据结构和操作这些数据的方法紧密地结合在一起所构成的模块。

(2) 抽象。抽象是指将现实世界中的事物、状态或过程所存在的相似方面集中和概括起来，暂时忽略它们之间的差异。面向对象方法不仅支持过程抽象，而且支持数据抽象。类实际上是一种抽象数据类型，它对外开放的公共接口构成了类的规格说明（即协议），这种接口规定了外界可以使用的合法操作符，利用这些操作符可以对类实例中包含的数据进行操作。此外，某些面向对象的程序设计语言还支持参数化抽象。所谓参数化抽象，是指当描述类的规格说明时并不具体指定所要操作的数据类型，而是把数据类型作为参数。这使得类的抽象程度更高，应用范围更广，可重用性更高。例如，C++程序设计语言提供的“模板”机制就是一种参数化抽象机制。

(3) 信息隐藏。信息隐藏是程序把函数过程或对象看成“黑箱”的能力，使用它实现指定的操作，而不需要知道内部的运转。在面向对象方法中，信息隐藏通过对象的封装性实现。通常，对象属性的表示方法和操作的实现算法是隐藏的。

(4) 高内聚与低耦合。内聚是衡量一个模块内各个元素彼此结合的紧密程度。耦合是指一个软件结构内不同模块之间关联的紧密程度。在面向对象方法中，对象是最基本的模块，因此，耦合主要指不同对象之间相互关联的紧密程度。在设计时应该尽量做到高内聚、低耦合。

(5) 可重用性。软件重用是提高软件开发质量的重要途径。重用包括尽量使用已有的类，如果确实需要创建新类，则在设计这些新类的协议时，应该考虑将来的可重复使用性。

1.2.3　面向对象实现

实现是问题的解(即结果)。在面向对象分析和面向对象设计的基础上，使用面向对象程序设计语言编写程序代码，最终实现一个软件系统的过程就是面向对象方法的实现。由于软件在开发中很有可能存在各种隐含错误，因此程序实现之后还应该进行软件测试，以便及时发现程序中的潜在错误。目前软件代码的规模越来越庞大，即使经过多次测试，软件中仍然不可避免存在各种各样的隐含错误，因此软件在使用过程中，需要开发人员或专业软件维护人员进行必要和合理的维护。

1.3　面向对象程序设计的特点

面向对象的程序设计方法强调在软件开发过程中面向待求解问题域中的事物，运用人类认识客观世界的普遍思维方法，直观、准确、自然地描述客观世界中的相关事物。面向对象程序设计方法的

基本特征主要有抽象性、封装性、继承性和多态性。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，以便于熟练掌握和运用。

1.3.1 抽象性

抽象就是从众多的事物中抽取出共同的、本质性的特征，舍弃其非本质的特征。例如，苹果、香蕉、酥梨、葡萄、桃子等，它们共同的特征就是水果。得出水果概念的过程，就是一个抽象的过程。共同特征是指那些能把一类事物与其他类事物区分开来的特征，这些具有区分作用的特征又称本质特征。而共同特征是相对的，是指从某一个片面来看是共同的。例如，对于汽车和大米，从买卖的角度看都是商品，都有价格，这是它们的共同特征，而从其他方面比较时，它们则是不同的。所以在抽象时，哪些是共同特征，决定于从什么角度进行抽象。抽象的角度取决于分析问题的目的。抽象的目的主要是为了降低复杂度，以得到问题域中较简单的概念，好让人们能够控制其过程或以宏观的角度了解许多特定的事态。

抽象包含两个方面：一方面是过程抽象；另一方面是数据抽象。过程抽象就是针对对象的行为特征，如鸟会飞、会跳等，这些方面可以抽象为方法，即过程，写成类时都是鸟的方法。数据抽象就是针对对象的属性，如建立一个鸟这样的类，鸟会有以下特征：两个翅膀，两只脚，有羽毛等，写成类时都应是鸟的属性。

例如，用面向对象程序设计方法设计学生信息管理系统。由于管理的对象是学生，分析的重点应该是学生，通过分析学生信息管理系统的各种功能、操作和学生的主要属性(学号、姓名、班级、年龄、各科成绩等)，找出其共性，将学生作为一个整体对待，并抽象成一个类(Student)，将学生群体抽象为一个类的过程如图 1-1 所示。在该抽象过程中，首先有高低、胖瘦、俊丑、学习好坏等各不相同的学生 1、学生 2……但他们都属于学生，都具有学号、姓名、班级、年龄、性别、成绩等属性(数据)；还有输入学号、修改班级、打印各科成绩等行为(方法)。因此可以把这些属性和方法封装起来而形成类。有了类后，就可以建立类的实例，即类所对应的对象。在此基础上还可以派生出其他类，从而实现代码的重用。

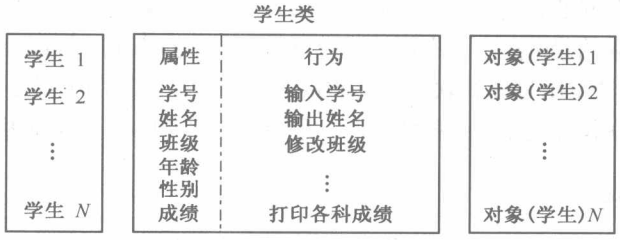

图 1-1 抽象过程示意图

1.3.2 封装性

封装是面向对象方法的一个重要特点，即将对象的属性和行为封装在对象的内部，形成一个独立的单位，并尽可能隐蔽对象的内部细节。对数据的访问只允许通过已经定义好的接口，也就是通过预先定义的关联到某一对象的服务和数据的接口，而无须知道这些服务是如何实现的。例如，一台洗衣机，使用者无须关心它的内部结构，也无法(当然也没必要)操作洗衣机的内部电路，因为它们被封装在洗衣机里面，这对于用户来说是隐蔽的、不可见的。使用者只需要掌握如何使用机器上的按键，如启动 / 暂停、选择等。这些按键安装在洗衣机的表面，人们通过它们与洗衣机交流，告诉洗衣机应

该做什么。面向对象就是基于这个概念，将现实世界描述为一系列完全自治、封装的对象，这些对象通过固定受保护的接口访问其他对象。在上例的学生对象中，其他对象可通过直接调用方法“打印各科成绩”来实现学生成绩的打印，而不必关心打印的具体实现细节。

1.3.3　继承性

继承性是子类自动共享父类数据结构和方法的机制，这是类之间的一种关系。在定义和实现一个类的时候，可以在一个已经存在的类的基础之上进行，把这个已经存在的类所定义的内容作为自己的内容，并加入若干新的内容。对象的一个新类可以从现有的类中派生，这叫做类的继承。新类继承了原始类的特性，新类称为原始类的派生类或子类，原始类称为新类的基类或父类。子类不仅可以从父类那里继承父类的数据成员和方法，而且可以增加新的数据成员和方法，或者修改已有的方法使之满足需求。图 1-2 说明了人、学生、大学生之间的继承关系，箭头的方向指向其父类。在此例中“学生”也是“人”，具有身高、体重、性别、年龄等人类的共同属性，除此之外，学生还有自己所特有的属性，如班级、学习成绩等特性。同样，大学生除了继承学生的全部属性外，还有所学专业、所修学分等特有属性。

继承性是面向对象程序设计语言不同于其他语言的最重要的特点，是其他语言所没有的。在类层次中，子类只继承一个父类的数据结构和方法，则称为单重继承或单继承，如图 1-3 所示。子类继承了多个父类的数据结构和方法，则称为多重继承或多继承，如图 1-4 所示。通过类的继承关系，可以使公共的特性能够共享，提高了软件的重用性。

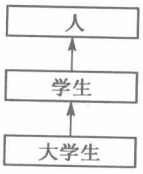

图 1-2　继承关系

图 1-3　单继承

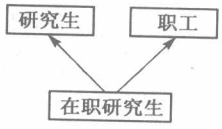

图 1-4　多继承

1.3.4　多态性

多态性是面向对象方法的重要特征。不同的对象，收到同一消息可以产生不同的结果，这种现象称为多态性。多态性允许每个对象以适合自身的方式去响应共同的消息。例如，一个学生拿着象棋，对另一个学生说：“咱们玩棋吧。”另一个学生听到请求后，就明白是下象棋。一个小朋友拿着跳棋对另一个小朋友说：“咱们玩棋吧。”另一个小朋友听到请求后，就明白是玩跳棋。在这两件事情中，学生和小朋友都是在“玩棋”。但他们听到请求以后的行为是不同的，这就是多态性。多态性使得同一个属性或行为在父类及其各派生类中具有不同的语义。如图1-5所示，父类是“学生”类，它具有“输入学生信息”、“输出学生信息”的行为。派生类“大学生”、“研究生”、“在职研究生”等都继承了父类“学生”的输入学生信息、输出学生信息的功能，但具体输入、输出的信息却各不相同。当发出“输入学生信息”或“输出学生信息”的消息后，“大学生”、“研究生”、“在职研究生”等类的对象接收到这个消息后，将执行不同的功能，这就是面向对象方法中的多态性。多态性丰富了对象的内容，扩大了对象的适应性，改变了对象单一继承的关系。多态性增强了软件的灵活性和重用性。

C++支持两种多态性，即编译时的多态性和运行时的多态性。编译时的多态性是通过重载来实现的，运行时的多态性是通过虚函数来实现的。本书将在后续章节对此进行介绍。

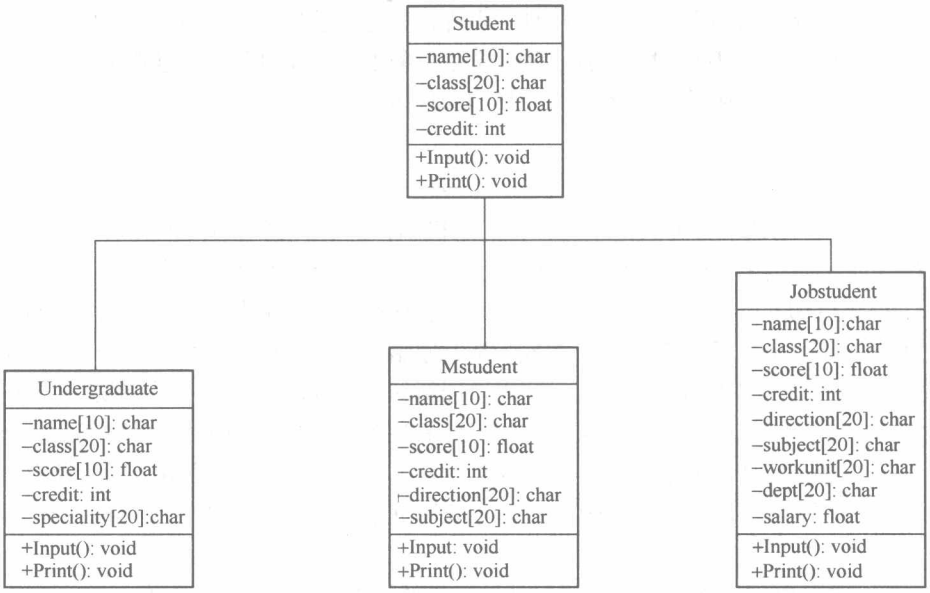

图 1-5 多态性

1.4 面向对象的术语

1. 类

“类”是对一组具有共同属性特征和行为特征的对象的抽象。例如，学生张三、学生王明，虽然是不同的学生，但是他们的基本特征是相似的，都有姓名、年龄、班级、学习成绩等，因此将他们统称为“学生”类。

2. 对象

对象是封装了数据结构及可以施加在这些数据结构上的操作的封装体。对象中的数据表示对象的状态，对象中的操作可以改变对象的状态。在现实世界中，对象是我们认识世界的基本单元，可以是人，也可以是物，还可以是一件事。对象既可以是一个有形的具体存在的事物，如一个球、一个学生、一辆车；也可以是无形的、抽象的事件，如一节课、一场球赛等。对象既可以是简单对象，也可以是由多个对象构成的复杂对象。术语“对象”既可以是指一个具体的对象，也可以泛指一般的对象。

3. 实例

实例是一个类所描述的一个具体的对象。例如，通过“大学生”类定义一个具体的对象学生王明就是大学生类的一个实例，就是一个对象。姓名：王明；年龄：20；班级：网络 021；专业：网络工程专业；高等数学成绩：80；大学物理成绩：90，这些就是对象中的数据。输入学生信息、输出学生信息等操作就是对象中的操作。

类和对象之间的关系是抽象和具体的关系。类是对多个对象进行综合抽象的结果，对象是类的个体实物，一个具体的对象是类的一个实例。例如，手工制作糕点时，先制作模子，然后将面塞进模子里，再进行烘烤，这样就可以制作出外形一模一样的糕点了。这个模子就类似于“类”，制作

出的一个个糕点就好比是类的“实例”。再比如，在 C 语言中 int(整型)是一个数据类型(类)，int a 则说明 a 为整型变量，a 就是一个整型(类)对象；当执行语句 a=5 时，就是令整型对象 a 取得一个实例值 5。

4．属性

属性，就是在类中所定义的数据。它是对客观世界实体所具有的性质的抽象。例如，Student 类中所定义的表示学生的姓名、年龄和成绩的数据成员就是 Student 类的属性。类的每个实例都有自己特有的属性值。如上例中学生王明的姓名：王明；年龄：20；班级：网络 021；专业：网络工程专业；高等数学成绩：80；大学物理成绩：90，就是实例王明自己特有的属性值。

5．消息

在面向对象程序设计中，消息就是要求某个对象执行定义该对象的类中的某个操作的规格说明。消息具有三个性质：

(1) 同一个对象可以接收不同形式的多个消息，做出不同的响应；

(2) 相同形式的消息可以传递给不同的对象，所做出的响应可以是不同的；

(3) 接收对象对消息的响应并不是必须的，对象可以响应消息，也可以不响应。

在面向对象程序设计中，消息分为两类：公有消息和私有消息。公有消息是由其他对象直接向它发送的。私有消息是它向自己发送的。

例如，MyStudent 是一个 Student 类的对象，当要求它在第 2 个位置上插入一个学生信息时，在 C++中应该向它发下列消息：

MyStudent.Insert_Student(2, x)；

其中 MyStudent 是接收消息的对象的名字，Insert_Student 是消息选择符(即消息名)，括号内 2 和 x 是消息的变元。当 MyStudent 接收到这个消息后，将执行在 Student 类中所定义的 Insert_Student 操作。

6．方法

方法是对象所执行的操作，也是类中所定义的服务。方法描述了对象执行操作的算法、响应消息的方法。在 C++中把方法称为成员函数。

例如，为了让 Student 类中的对象能够响应插入运算，在 Student 类中必须给出成员函数 int Insert_Student(int i, datatype x)的定义，也就是要给出这个成员函数的实现代码。

7．重载

在解决问题时，经常会遇到一些函数，它们的功能相同，但参数类型不同或参数个数不相等。例如，求一个数的立方或求最大值问题，参数类型可能是整型，也可能是实型；可能是求两个参数的最大值，也可能是求三个参数的最大值。但很多程序设计语言要求函数名必须唯一，因此就需要定义不同的函数名，使得程序员需要记忆很多不同的名字，增加了程序员的负担。针对这类问题，C++提供了重载机制，即允许具有相同或相似功能的函数使用同一函数名，从而减少了程序员记忆多个函数名字的负担。C++提供的重载包括函数重载和运算符重载。函数重载是指在同一作用域内的若干个参数特征不同的函数可以使用相同的函数名字；运算符重载是指同一个运算符可以施加于不同类型的操作数上。也就是说，相同名字的函数或运算符在不同的场合可以表现出不同的行为。

1.5 其他面向对象程序设计语言

1.5.1 Java 语言

1. Java 简介

Java 语言是一种通用、并发、基于类的面向对象程序设计语言。Java 语言的名字来源于印度尼西亚的一个岛名“爪哇”(印度尼西亚盛产咖啡的一个岛屿)。

Java 是由 Sun Microsystems 公司于 1995 年 5 月推出的 Java 程序设计语言和 Java 平台的总称。这是一种纯粹的面向对象程序设计语言。传统的程序语言编写的软件往往与具体的实现环境有关，而 Java 编写的软件具有在执行代码上的兼容特性。只要计算机系统提供了 Java 解释器，用 Java 编写的软件就可以在各种系统上运行。

Java 平台由 Java 虚拟机(Java Virtual Machine)和 Java 应用编程接口(Application Programming Interface, API)构成。Java 应用编程接口为 Java 应用提供了一个独立于操作系统的标准接口，可分为基本部分和扩展部分。在硬件或操作系统平台上安装一个 Java 平台之后，Java 应用程序就可运行。现在 Java 平台已经嵌入到了几乎所有的操作系统。这样，Java 程序可以只编译一次，就可以在各种系统中运行。Java 应用编程接口已经从 1.1x 版发展到 1.2 版。目前常用的 Java 平台基于 Java 1.4，最近版本为 Java 1.7。

Java 分为三个体系 JavaSE(Java 2 Platform Standard Edition, Java 平台标准版)，JavaEE(Java 2 Platform Enterprise Edition, Java 平台企业版)，JavaME(Java 2 Platform Micro Edition, Java 平台微型版)。

2. Java 的主要特性

(1) 简捷性

Java 语言的语法与 C 语言、C++语言很接近，使得大多数程序员很容易学习和使用 Java。另外，Java 摒弃了 C++中很少使用的、很难理解的、令人迷惑的那些特性，如操作符重载、多继承、自动强制类型转换。特别地，Java 语言不使用指针，并提供了自动废料收集，使得程序员不必为内存管理而担忧。

(2) 面向对象

Java 语言提供类、接口和继承。为了简单起见，只支持类之间的单继承，但支持接口之间的多继承，并支持类与接口之间的实现机制(关键字为 implements)。Java 语言全面支持动态绑定，而 C++语言只对虚函数使用动态绑定。总之，Java 语言是一个纯面向对象程序设计语言。

(3) 分布式

Java 语言支持 Internet 应用的开发，在基本的 Java 应用编程接口中有一个网络应用编程接口(java.net)，它提供了用于网络应用编程的类库，包括 URL、URLConnection、Socket、ServerSocket 等。Java 的 RMI(远程方法激活)机制也是开发分布式应用的重要手段。

(4) 健壮性

Java 的强类型机制、异常处理、废料自动收集等是 Java 程序健壮性的重要保证。对指针的摒弃是 Java 的明智选择。Java 的安全检查机制使得 Java 更具健壮性。

(5) 安全性

Java 通常被用在网络环境中，为此，Java 提供了一个安全机制以防恶意代码的攻击。除了 Java