21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C++面向对象程序设计

C++ Object-Oriented Programming

陈维兴 陈昕 编著

- 不追求深奥的理论，强调C++的实践技能
- 不追求抽象的概念，注重C++的实际应用
- 不追求完整的语法，突出C++的基本内容

精品系列

人民邮电出版社
POSTS & TELECOM PRESS

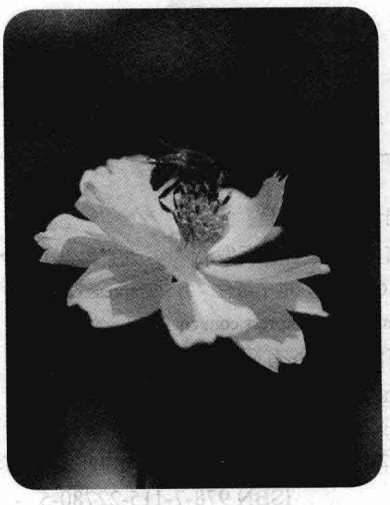
21世纪高等学校计算机规划教材

21st Century University Planned Textbooks of Computer Science

C++面向对象程序设计

C++ Object-Oriented Programming

陈维兴 陈昕 编著



精品系列

人民邮电出版社

北京

图书在版编目（CIP）数据

C++面向对象程序设计 / 陈维兴，陈昕编著. — 北京 : 人民邮电出版社，2010.10
21世纪高等学校计算机规划教材
ISBN 978-7-115-22780-5

Ⅰ. ①C… Ⅱ. ①陈… ②陈… 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117253号

内 容 提 要

本书介绍了C++面向对象程序设计的基本知识和编程方法，以及C++面向对象的基本特征。针对初学者的特点，本书力求通过大量实例、习题和上机实验题，以通俗易懂的语言讲解复杂的概念和方法，使读者能深刻理解和领会面向对象程序设计的特点和风格，掌握其方法和要领，以期帮助读者尽快地迈入面向对象程序设计的大门。

本书以应用为目的，大力加强实践环节，注重培养应用能力，适合作为高等院校各专业学生学习C++程序设计课程的教材，也可作为C++语言自学者的参考用书。

21世纪高等学校计算机规划教材

C++面向对象程序设计

◆ 编　　著　陈维兴　陈　昕
责任编辑　武恩玉
◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街14号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷
◆ 开本：787×1092　1/16
印张：19.75　　2010年10月第1版
字数：521千字　　2010年10月河北第1次印刷

ISBN 978-7-115-22780-5

定价：33.00元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

面向对象程序设计是不同于传统程序设计的一种新的程序设计范型。它对降低软件的复杂性，改善其重用性和维护性、提高软件的生产效率，有着十分重要的意义。因此面向对象的程序设计被普遍认为是程序设计方法学的一场实质性的革命。

C++是在 C 语言基础上扩充了面向对象机制而形成的一种面向对象程序设计语言，它除了继承 C 语言的全部优点和功能外，还支持面向对象程序设计。C++现在已成为介绍面向对象程序设计的首选语言。学习 C++不仅可以深刻理解和领会面向对象程序设计的特点和风格，掌握其方法和要领，而且可以使读者掌握一种十分流行和实用的程序设计语言。

目前市场上的 C++教材很多，但大多数教材都是为没有 C 语言基础的学生编写的。据作者了解，当前无论在大学里或社会上，有相当一批人已经学过 C 语言。很多高校的培养计划，仍是先开设 C，随后再开设 C++。本书就是为那些已经学过 C 语言，具有一定程序设计基础的大学本科生编写的。因此，本书是符合高校教学需要的。在取材方面，舍去了 C 语言已经学过的内容，只讲 C++面向对象程序设计部分的内容。这样既节省了教学时间，也减轻了学生的学习负担。根据多年师生反馈的信息，本书的取材是合适的，深度也是适宜的。

本书体现了“以学生为中心”的理念，内容叙述力求通俗易懂，由浅入深，符合认知规律，力求做到多讲实例，循序渐进地引出概念，将复杂的概念用简洁浅显的语言来讲述。力求教学内容富有启发性，便于学生学习。本书以应用为目的，大力加强实践环节，注重培养应用能力。在本书中配有大量的例题、上机实验题和习题，以利于学生举一反三，从中学习方法和技巧，注意培养学生的实践能力和创新能力。

本书共分 11 章。第 1 章介绍面向对象程序设计的基本概念；第 2 章介绍 C++对 C 语言在非面向对象方面的扩充；第 3 章至第 10 章详述了 C++支持面向对象程序设计的基本方法，包括类、对象、派生类、继承、多态性、模板、流类库、异常处理和命名空间等；第 11 章介绍了面向对象程序设计的一般方法和技巧，并安排了一个应用实例，供读者借鉴。在附录中介绍了 Visual C++ 6.0 的开发环境，对如何建立和运行 C++单文件程序和多文件程序进行了较详细的介绍。每一章后面给出了上机实验题和习题，供读者练习。

本书的上机实验部分由陈昕编写，其他章节由陈维兴编写。全书由陈维兴组织编写并统稿。本书中所有程序都经作者在 Visual C++ 6.0 上调试通过。

在本书的编写和出版过程中得到了周涛、林小茶、李春强、孙若莹的帮助和支持，在此表示诚挚的感谢。

最后，借用本书出版的机会，向关心本书的各位专家、老师和广大读者表示衷心的感谢，欢迎您对本书的内容和编写方法提出批评和建议。

编　者

2010 年 6 月

目 录

第 1 章 面向对象方法概述

面向对象程序设计（Object-Oriented Programming，OOP）的思想已经被越来越多的软件设计人员所接受。之所以这样，不仅因为它是一种最先进的、新颖的计算机程序设计思想，更主要的是这种新的思想更接近人的思维活动，人们利用这种思想进行程序设计时，可以很大程度地提高编程能力，减少软件维护的开销。面向对象程序设计方法是通过增加软件的可扩充性和可重用性来提高程序员的编程能力。这种思想与我们以前使用的方法有很大的不同，并且在理解上有一些难点，希望本章的内容能对读者有所帮助。

1.1 什么是面向过程程序设计方法

面向对象程序设计方法和面向过程程序设计方法比较起来，有许多优越性。那么，什么是面向对象程序设计方法呢？在介绍它之前，首先讨论面向过程程序设计方法。

1.1.1 面向过程程序设计方法概述

面向过程程序设计方法是流行很广泛的程序设计方法。在面向过程程序设计中，程序设计者不仅要考虑程序要“做什么”，还要解决“怎么做”的问题。首先要明确程序的功能，程序设计的重点是如何设计算法和实现算法。在面向过程程序设计中，一种普遍采用的优化方法是使用结构化程序设计方法，即所有的程序都可以由顺序、分支和循环 3 种基本结构组成。这样在问题求解过程中，我们可以先进行整体规划，将一个复杂的任务按功能分解成一个个易于控制和处理的子任务。然后对每个子任务按功能再进行细化，依此进行，直到不需要细分为止。具体实现程序时，每个子任务对应一个子模块，模块间尽量相对独立，通过模块间的调用关系或全局变量而有机地联系起来。

在 C 语言中，可以将每一个子模块对应设计成一个函数，各个函数及函数间的调用关系组成了程序。下面，用面向过程程序设计方法编写一个简单的计算面积的 C 语言程序。

例 1.1　利用面向过程程序设计方法计算圆和三角形的面积。

设圆的半径为 r，圆周率取 3.14，则圆面积 cs 的计算公式为 cs=3.14*r*r，设三角形的高为 h，底为 w，则三角形面积 ts 的计算公式为 ts=0.5*h*w。

根据结构化程序设计方法，该任务有一个主模块（对应主函数 main），两个子模块，即计算圆面积的子模块（对应函数 circle）和计算三角形面积的子模块（对应函数 triangle）。其程序的架构可以用图 1-1 来表示。

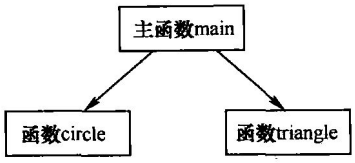


图 1-1　例 1.1 的程序框架示意图

下面，用面向过程程序设计方法编写一个简单的计算面积的 C 语言程序。

```
#include<stdio.h>
double circle(double r)                          //定义函数 circle
{ return 3.14*r*r;
}
double triangle(double h, double w)              //定义函数 triangle
{ return 0.5*h*w;
}
int main()                                       //定义主函数 main
{ double r, h, w;
  double cs, ts;
  printf("Input r, h, w: ");
  scanf("%lf%lf%lf", &r, &h, &w);                //输入圆的半径和三角形的高和底的值
  cs= circle(r);                                 //调用函数 circle
  ts= triangle(h, w);                            //调用函数 triangle
  printf("The area of circle is:%f\n", cs);      //输出圆的面积
printf("The area of triangle is: %f\n", ts);     //输出三角形的面积
return 0;
}
```

程序的一次运行结果如下：

```
Input r, h, w: 10 20 10
The area of circle is:314.000000
The area of triangle is: 100.000000
```

这个简单的 C 语言程序设计表示了面向过程程序设计的主要特征：程序由过程定义和过程调用组成（所谓过程，简单地说，就是程序执行某项操作的一段代码，函数是最常用的过程。），从这个意义出发，基于面向过程的程序可以用以下的公式来表述：

程序 = 过程 + 调用

推而广之，如果把函数看成是一种过程，那么面向过程程序方法所设计的程序的架构可以用图 1-2 来概括。

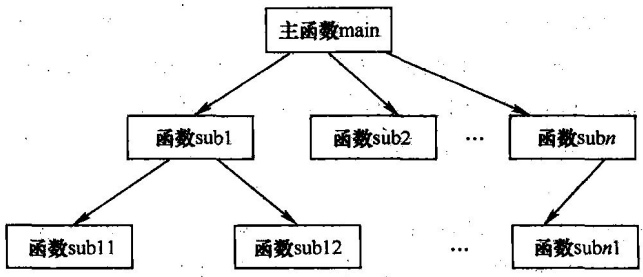


图 1-2　面向过程方法的程序设计框架示意图

结合图 1-2 所示，可以看出：

（1）面向过程程序设计方法一般采用自上而下、逐步求精的结构化程序设计方法。当编写一个较大的程序时，可以把它按照功能逐级划分成许多相对独立的小模块。每个小模块的功能由一个函数实现，再通过适当的方法将这些函数组织在一起协同工作，就能够完成整个程序所规定的任务。

（2）使用面向过程程序设计方法编写出的 C 语言程序包括一个主函数 main 和若干用户定义函数。主函数由操作系统调用，它是整个程序的入口。主函数的主要任务就是完成对其他函数的有机调用。其他函数之间也可以相互调用，并且同一个函数可以被一个或多个函数调用任意多次。

1.1.2　面向过程程序设计方法的局限性

众所周知，计算机的发明和使用，对人类社会的进步与发展发挥了巨大的作用。随着计算机大规模地推广、普及与应用，人们对软件的功能要求越来越多，对软件的性能要求越来越高。传统的程序设计已不能满足这些日益增长的需要。面向过程程序设计中普遍采用的优化方法是使用结构化程序设计方法，其局限性体现在以下几个方面。

1. 面向过程程序设计方法开发软件的生产效率低下

按照结构化程序设计的要求，当需要解决一个复杂的任务时，首先应将它按功能划分为若干个小任务，每个小任务又可以按功能划分为若干个更小的任务，依次类推……直到最低一层的任务容易用程序实现为止，然后将所有的小任务全部解决并把它们组合起来。

这种传统程序设计的生产方式仍是采用较原始的方式进行，程序设计基本上还是从语句一级开始。软件的生产中缺乏大粒度、可重用的构件，软件的重用问题没有得到很好地解决，从而导致软件生产的工程化和自动化屡屡受阻。

面向过程程序设计的特点是数据与其操作分离，而且对同一数据的操作往往分散在程序的不同地方。这样，如果一个或多个数据的结构发生了变化，那么这种变化将波及程序的很多部分甚至遍及整个程序，致使许多函数必须重写，严重时会导致整个软件结构的崩溃。数据和操作相分离的结构，使得维护数据和处理数据的操作过程要花费大量的精力和时间，严重地影响了软件的生产效率。

2. 面向过程程序设计方法难以应付日益庞大的信息量和多样的信息类型

随着计算机科学与技术的飞速发展和计算机应用的普及，当代计算机的应用领域已从数值计算扩展到了人类社会的各个方面，所处理的数据已从简单数字和字符发展为具有多种格式的多媒体数据，如文本、图形、图像、影像、声音等，描述的问题已从单纯的计算问题发展到仿真复杂的自然现象和社会现象。于是，计算机处理的信息量与信息类型迅速增加，程序的规模日益庞大，复杂度不断增加。这些都要求程序设计语言有更大的信息处理能力。然而，面对这些庞大的信息量和多样的信息格式，面向过程程序设计方法是无法应付的。

3. 面向过程程序设计方法难以适应各种新环境

当前，并行处理、分布式、网络和多机系统等，已经或将是程序运行的主流方式和主流环境。这些环境的一个共同特点是都具有一些有独立处理能力的节点，节点之间有通信机制，即以消息传递进行联络。显然传统的程序设计技术很难适应这些新环境。

综上所述，面向过程程序设计方法不能够满足计算机技术迅猛发展的需要，软件开发迫切需要一种新的程序设计方法的支持。那么，面向对象程序设计是否能够担当此任呢？下面

我们再分析一下面向对象程序设计的方法。

1.2 什么是面向对象程序设计方法

1.2.1 面向对象程序设计方法的基本概念

为了理解面向对象程序设计方法，我们从最基本的概念入手。本节介绍的内容是面向对象程序设计的理论基础，这些内容不依赖于具体的程序设计语言，也就是说，无论使用哪种面向对象语言进行面向对象程序设计，本节内容都有理论意义。

1. 对象

面向对象程序设计方法是一种自下而上的程序设计方法，它不像面向过程程序设计方法那样，需要在一开始就要用主函数 main 概括出整个程序，它往往是从问题的一部分着手，一点一点地构建出整个程序。面向对象程序的基本元素是对象。

对象就是可以控制和操作的实体。在现实世界中，任何事物都是对象。它可以是一个有形的、具体存在的事物，例如一张桌子、一个学生、一辆汽车甚至一个地球。它也可以是一个无形的、抽象的事件，例如一次演出、一场球赛、一次出差等。对象既可以很简单，也可以很复杂，复杂的对象可以由若干简单的对象构成，整个世界都可以认为是一个非常复杂的对象。

现实世界中的对象既具有静态的属性（或称状态），又具有动态的行为（或称功能）。例如，一个人就是一个对象，每个人都有姓名、性别、年龄、身高、体重等属性，都有吃饭、走路、睡觉、学习等行为。所以在现实世界中，任何一个对象都具有这两个要素，即属性和行为，它能根据外界给出的信息进行相应的操作。一个对象往往是由一组属性和一组行为构成的。

现实世界中的对象，具有以下特性：

（1）有一个名字：每一个对象必须有一个名字，称为对象名，以区别于其他对象；

（2）有一组属性：用属性来描述它的某些特征，一般可以用数据来表示，所有的属性都有值；

（3）有一组行为：对象的行为或功能也称为方法，一般用一组操作来描述；

（4）有一组接口：除施加于对象内部的操作外，对象还提供了一组公有操作用做与外界的接口，从而可以与其他对象建立关系。

面向对象的程序设计采用了以上人们所熟悉的这种思路。在面向对象程序设计中，对象是描述其属性的数据以及对这些数据施加的一组操作封装在一起构成的统一体。在 C++语言中，每个对象都是由数据（属性）和操作代码（行为，通常用函数来实现）两部分组成的，如图 1.3 所示。在面向对象程序设计中，用数据来体现上面提到的“属性”，如一个圆对象，它的半径就是它的属性。函数是用来对数据进行操作的，以实现某些功能。例如，可以通过半径计算出圆的面积，并且输出圆的半径和面积。计算圆的面积和输出有关数据就是前面提到的行为。

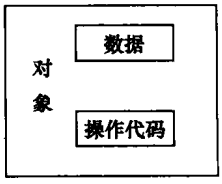


图 1-3 对象结构示意图

2. 类

在实现世界中，“类”是一组具有相同属性和行为的对象的抽象。例如，虽然张三、李四、王五……，每个人的性格、爱好、职业、特长等各有不同，但是他们的基本特征是相似的，都具有相同的生理构造，都能吃饭、说话、走路等，于是把他们统称为“人”类，而具体的每一个人是人类的一个实例，也就是一个对象。

类在现实世界中并不真正存在。例如，在地球上并没有抽象的“学生”，只有一个个具体的学生，如张三、李四、王五……。同样，世界上也没有抽象的“教师”，只有一个个具体的教师。

在面向对象程序设计中，“类”就是具有相同的数据（属性）和相同的操作代码（函数）的一组对象的集合。对象是具体存在的，如一个圆可以作为一个对象，10 个不同半径的圆就是 10 个对象。如果这 10 个圆对象有相同的数据（属性）和操作代码（函数），就可以将它们抽象为一种类型，称为圆类型。在 C++语言中，可以将这种类型定义为类（class）。需要说明的是，这里所说的数据相对于现实世界中属性，具体指圆的半径，各个圆对象的半径值可以不同。在 C++语言中把类中的数据称为数据成员，类中的操作是用函数来实现的，这些函数称为成员函数。

类和对象之间的关系是抽象和具体的关系。类是多个对象进行综合抽象的结果，一个对象是类的一个实例。例如“学生”是一个类，它是由许多具体的学生抽象而来的一般概念。同理，桌子、教师、计算机等都是类。

在面向对象程序设计中，总是先声明类，再由类生成其对象。类是建立对象的“模板”，按照这个模板所建立的一个个具体的实例，通常称为对象。打个比方，手工制作月饼时，先雕刻一个有凹下图案的木模，然后将事先揉好的面塞进木模里，用力挤压后，将木模反扣在桌上，一个漂亮的图案就会出现在月饼上了。这样一个接着一个就可以制造出外形一模一样的月饼，但每个月饼中的面和馅的量可以是不同的。这个木模就好比是“类”，制造出来的糕点好比是“对象”。

3. 消息

现实世界中的对象不是孤立存在的实体，它们之间存在着各种各样的联系，正是它们之间的相互作用、联系和连接，才构成了世间各种不同的系统。

在面向对象程序设计中，对象之间也需要联系，我们称为对象的交互。面向对象程序设计技术必须提供一种机制允许一个对象与另一个对象的交互，这种机制称为消息传递。一个对象向另一个对象发出的请求称为“消息”。当对象接收到发向它的消息时，就调用有关的方法，执行相应的操作。例如，有一个教师对象张三和一个学生对象李四，对象李四可以发出消息，请求对象张三演示一个实验，当对象张三接收到这个消息后，确定应完成的操作并执行之。

一般情况下，我们称发送消息的对象为发送者或请求者，接收消息的对象为接收者或目标对象。发送者发送消息，接收者通过调用相应的方法对消息作出响应。这个过程不断重复，系统不停地运转，最终得到相应的结果。

一个对象可以同时向多个对象发送消息，也可以接收多个对象发来的消息。相同形式的消息可以发送给不同的对象，同一个对象也可以接收不同形式的消息。消息的发送者可以不考虑具体接收者，只是反映了发送者的请求。由于消息的识别、解释取决于接收者，对象可以响应消息也可以不响应消息。

4. 方法

在面向对象程序设计中的消息传递实际是对现实世界中的信息传递的直接模拟。调用对象中的函数就是向该对象传送一个消息，要求该对象实现某一行为（功能）。对象所能实现的行为（功能），在程序设计方法中称为方法，它们是通过调用相应的函数来实现的。在 C++语言中，方法是通过成员函数来实现的。

方法包括界面和方法体两部分。方法的界面给出了方法名和调用协议（相对于 C++语言中成员函数的函数名和参数表）；方法体则是实现某种操作的一系列计算步骤，也就是一段程序（相对于 C++语言中成员函数的函数体）。消息和方法的关系是：对象根据接收到的消息，调用相应的方法；反过来，有了方法，对象才能响应相应的消息。

面向对象程序设计是一种新的程序设计范型。这种范型的主要特征如下：

程序 = 对象 + 消息

对于面向对象的程序设计，程序员注重的是类的设计和编写，即问题域中涉及几个类，各个类之间的关系如何，每个类包含哪些数据和函数（操作代码），再由类生成其对象。程序中的一切操作都是通过向对象发送消息来实现的，对象接收到消息后，启动有关方法（通过成员函数）完成相应的操作。

下面我们将例 1.1 用 C++面向对象程序设计语言进行重新编写。在此，读者不需要完全理解，只要有一个初浅的了解即可。

例 1.2 利用面向对象思想求解计算圆和三角形的面积。

首先利用面向对象的思想来分析这个问题，本题可声明两个类：圆类 Circle 和三角形类 Triangle。

圆类 Circle 中含有数据成员（属性）r 和 cs，r 和 cs 分别表示圆的半径和圆的面积；含有成员函数（操作代码）radius_input 和 c_area_out，成员函数 radius_input 用于输入圆半径 r 的值，成员函数 c_area_out 用于计算与输出圆面积 cs 的值。由圆类 Circle 定义的各个大小不同的圆都是圆类的对象。

三角形类 Triangle 中含有数据成员（属性）h、w 和 ts，h 表示三角形的高，w 表示三角形的底边，ts 表示三角形的面积；含有成员函数（操作代码）h_w_input 和 t_area_out，成员函数 h_w_input 用于输入高 h 与底边 w 的值，成员函数 t_area_out 用于计算与输出三角形面积 ts 的值。由三角形类 Triangle 定义的各个大小和形状不同的三角形都是三角形类的对象。

```
#include<stdio.h>
class Circle {                    //声明圆类 Circle
    double r;                     //定义数据成员 r(表示圆的半径)
    double cs;                    //定义数据成员 cs(表示圆的面积)
    public:
    void radius_input()           //定义成员函数 radius_input (用于输入圆半径 r 的值)
    { printf("Input r : ");
      scanf("%lf", &r);
    }
    void c_area_out()             //定义成员函数 c_area_out (用于计算与输出圆面积)
    { cs=3.14*r*r;
      printf("The area of circle is:%f\n", cs);
    }
};                                //圆类 Circle 到此结束
```

```
class Triangle{                    //声明三角形类 Triangle
    double h, w;                   //定义数据成员 h 和 w(表示三角形的高和底边的值)
    double ts;                     //定义数据成员 ts(表示三角形的面积)
  public:
  void h_w_input()                 //定义成员函数 h_w_input(用于输入三角形的高 h 与底边 w 的值)
  { printf("Input h, w: ");
    scanf("%lf%lf", &h, &w);
  }
  void t_area_out()                //定义成员函数 t_area_out (用于计算与输出三角形面积)
  { ts=0.5*h*w;
    printf("The area of triangle is: %f\n", ts);
   }
};                                 //三角形类 Triangle 到此结束
int main()                         //定义主函数 main
{ Circle  c1;                      //定义圆类 Circle 的对象 c1
  c1.radius_input();               //向对象 c1 传送一个消息，即调用函数 radius_input，输入圆半径 r 的值
  c1.c_area_out();                 //向对象 c1 传送一个消息，即调用函数 c_area_out
                                   //计算与输出圆面积
  Triangle t1;                     //定义三角形类 Triangle 的对象 t1
  t1.h_w_input();                  //向对象 t1 传送一个消息，即调用函数 h_w_input，输入三角形的高 h 与
                                     底边 w 的值
  t1.t_area_out();                 //向对象 t1 传送一个消息，即调用函数 t_area_out，计算与输出三角形面积
  return 0;
}                                  //主函数 main 到此结束
```

程序的一次运行结果如下：

```
Input r : 10
The area of circle is:314.000000
Input h, w: 10 20
The area of triangle is: 100.000000
```

为了使程序简单明了，以上程序只定义了一个圆类对象 c1 和一个三角形对象 t1。实际使用时，可根据需要定义多个对象。读者可以通过例 1.1 和例 1.2 的对比分析，初步了解面向过程程序设计方法和面向对象程序设计方法的区别。

需要说明的是，基于面向过程程序设计方法的语言称为面向过程性语言，如 C 语言等就是典型的面向过程性语言。但是，某一种程序设计语言不一定与一种程序设计方法相对应。实际上存在具有两种或多种程序设计方法的程序设计语言，即混合型语言。例如，C++语言就是具有面向过程程序设计方法和面向对象程序设计方法的混合型程序设计语言。

1.2.2　面向对象程序设计方法的基本特征

面向对象程序设计方法模拟人类习惯的解题方法，代表了计算机程序设计的新颖的思维方法。这种方法的提出是对软件开发方法的一场革命，是目前解决软件开发面临困难的最有希望、最有前途的方法之一。本节介绍面向对象程序设计的 4 个基本特征。

1. 抽象

抽象是人类认识问题的最基本的手段之一。抽象是将有关事物的共性归纳、集中的过程。

在现实生活中，人们能看到的都是一些具体的事物，例如男人、女人、大人、小孩等，这些都是具体的人，把这些具体的人归纳为一类，称为“人”，这就是一种抽象。再假如，把所有具有大学学籍的人归为一类，称为“大学生”，这也是一个抽象。抽象是对复杂世界的简单表示，抽象并不打算了解全部问题，而只强调感兴趣的信息，忽略了与主题无关的信息。例如，在设计一个学生成绩管理系统的过程中，只关心学生的姓名、学号、成绩等，而对学生的身高、体重等信息就可以忽略。而在学生健康信息管理系统中，身高、体重等信息必须抽象出来，而成绩则可以忽略。

在面向对象程序设计中，抽象是通过特定的实例（对象）抽取共同特性后形成概念的过程。C 和 C++语言中的基本数据类型就是对一批具体的数的抽象。例如，“整型数据”是对所有整数的抽象。

2. 封装

在现实世界中，封装就是把某个事物包围起来，使外界不知道该事物的具体内容。在面向对象程序设计中，封装是指把数据和实现操作的代码集中起来放在对象内部，并尽可能隐蔽对象的内部细节。对象好像是一个不透明的黑盒子，表示对象属性的数据和实现各个操作的代码都被封装在黑盒子里，从外面是看不见的，更不能从外面直接访问或修改这些数据及代码。使用一个对象的时候，只需知道它向外界提供的接口而无须知道它的数据结构细节和实现操作的算法。这很像电视机、录音机、洗衣机等，从其外形来看，都是各种电子或机械部件被封装在盒子内部。使用这些电器的人并不需要知道电器内部有哪些部件，它们是如何组装的，它们的工作原理又如何。使用者只需要会使用电器提供的几个外部按钮（对应于对象的外部接口），就可以实现自己所需要的功能。将电器部件封装在盒子内部，既可以避免各种人为的损坏，也便于维护和管理。

C++程序中，对象的函数名就是对象的对外接口，外界可以通过函数名来调用这些函数来实现某些行为（功能）。这些将在以后详细介绍。

封装的好处是可以将对象的使用者与设计者分开，大大降低了人们操作对象的复杂程度。使用者不必知道对象行为实现的细节，只需要使用设计者提供的接口的功能，即可自如地操作对象。封装的结果实际上隐藏了复杂性，并提供了代码重用性，从而减轻了开发一个软件系统的难度。

封装是面向对象程序设计方法的一个重要特性。封装具有两方面的含义：一是将有关的数据和操作代码封装在一个对象中，各个对象相对独立、互不干扰；二是将对象中某些数据与操作代码对外隐蔽，即隐蔽其内部细节，只留下少量接口，以便与外界联系，接收外界的消息。这种对外界隐蔽的做法称为信息隐蔽。信息隐蔽有利于数据安全，防止无关人员访问和修改数据。

3. 继承

继承在现实生活中是一个很容易理解的概念。例如，我们每一个人都从我们的父母身上继承了一些特性，如种族、血型、眼睛的颜色等，我们身上的特性来自我们的父母，也可以说，父母是我们所具有的属性和行为的基础。

利用继承可以简化人们对事物的认识和叙述，简化工作程序。例如“柯利狗”继承了“狗”的特性，现在要叙述“柯利狗”的特征，显然不需要从头介绍什么是狗，而只要说明“柯利狗是尖鼻子、红白相间的颜色、适合放牧的狗”即可。利用继承可以简化程序设计的步骤。例如，在软件开发中已经建立了一个类 A，又需要建立另一个与 A 基本相同，但增加了一些

属性和行为的类 B，这时没有必要从头设计一个新类，只需在类 A 的基础上增加一些新的内容即可。这就是面向对象程序设计中的继承机制。

以面向对象程序设计的观点，继承所表达的是类之间相关的关系。这种关系使得某一类可以继承另外一个类的特征和能力。

若类之间具有继承关系，则它们之间具有下列几个特性：

（1）类间具有共享特征（包括数据和操作代码的共享）；

（2）类间具有差别或新增部分（包括非共享的数据和操作代码）；

（3）类间具有层次结构。

假设有两个类 A 和 B，若类 B 继承类 A，则类 B 包含了类 A 的特征（包括数据和操作），同时也可以加入自己所特有的新特性。这时，我们称被继承类 A 为基类或父类；而称继承类 B 为 A 的派生类或子类。同时，我们还可以说，类 B 是从类 A 中派生出来的。

如果类 B 是类 A 的派生类，那么，在构造类 B 的时候，我们不必描述 B 的所有特征，我们只需让它继承类 A 的特征，然后描述与基类 A 不同的那些特性。也就是说，类 B 的特征由继承来的和新添加的两部分特征构成。

具体地说，继承机制允许派生类继承基类的数据和操作（即数据成员和成员函数），也就是说，允许派生类使用基类的数据和操作。同时，派生类还可以增加新的操作和数据。

采用继承的方法可以很方便地利用一个已有的类建立一个新的类，这就可以重用已有软件中的一部分甚至大部分，在派生类中只需描述其基类中没有的数据和操作。这样，就避免了公用代码的重复开发，增加了程序的可重用性，减少了代码和数据冗余，大大节省了编程的工作量，这就是常说的“软件重用”思想。同时，在描述派生类时，程序员还可以覆盖基类的一些操作，或修改和重定义基类中的操作。具体的实现方法将在以后详细介绍。

从继承源来分，继承分为单继承和多继承。

单继承是指每个派生类只直接继承了一个基类的特征。例如，图 1-4 表示了一种单继承关系，它表示 Windows 操作系统窗口之间的继承关系。

单继承并不能解决继承中的所有问题，例如，小孩喜欢的玩具车既继承了车的一些特性，也继承了玩具的一些特征，这就是多继承的一个例子，如图 1-5 所示。

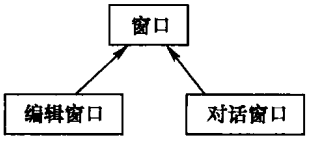


图 1-4　单继承示意图

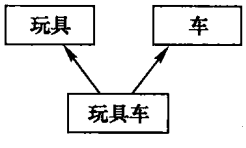


图 1-5　多继承示意图

多继承是指多个基类派生出一个派生类的继承关系。多继承的派生类直接继承了不止一个基类的特征。

4. 多态

多态性也是面向对象系统的重要特性。在讨论面向对象程序设计的多态性之前，我们还是来看看现实世界的多态性。现实世界的多态性在自然语言中经常出现。假设一辆汽车停在了属于别人的车位，司机可能会听到这样的要求：“请把你的车挪开”。司机在听到请求后，所做的工作应该是把车开走。在家里，一把凳子挡住了孩子的去路，他可能会请求妈妈“请

把凳子挪开”，妈妈过去搬起凳子，放在一边。在这两件事情中，司机和妈妈的工作都是“挪开”一样东西，但是他们在听到请求以后的行为是截然不同的，这就是现实世界中的多态性。对于“挪开”这个请求，还可以有更多的行为与之对应。“挪开”从字面上看是相同的，但由于作用的对象不同，操作的方法也就不同。

面向对象程序设计借鉴了现实世界的多态性。面向对象系统的多态性是指不同的对象收到相同的消息时执行不同的操作。例如，有一个窗口（Window）类对象，还有一个棋子（Piece）类对象，当我们对它们发出“移动”的消息时，“移动”操作在 Window 类对象和 Piece 类对象上可以有不同的行为。

C++语言支持两种多态性，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函数重载（包括运算符重载）来实现的，运行时的多态性是通过虚函数来实现的。本书将分别在第 2 章、第 6 章和第 7 章对函数重载、虚函数和运算符重载进行详细介绍。

多态性增强了软件的灵活性和重用性，为软件的开发与维护提供了极大的便利。尤其是采用了虚函数和动态联编机制后，允许用户以更为明确、易懂的方式去建立通用的软件。

1.2.3 面向对象程序设计方法的主要优点

面向对象程序设计方法是软件开发史上的一个重要里程碑。这种方法从根本上改变了人们以往设计软件的思维方式，从而使程序设计者摆脱了具体的数据格式和过程的束缚，将精力集中于要处理对象的设计和研究上，极大地减少了软件开发的复杂性，提高了软件开发的效率。面向对象程序设计主要具有以下优点。

1. 可提高程序的重用性

面向对象程序设计方法能比较好地解决软件重用的问题。对象所固有的封装性和信息隐藏等机理，使得对象内部的实现与外界隔离，具有较强的独立性，它可以作为一个大粒度的程序构件，供同类程序直接使用。

有两种方法可以重复使用一个对象类：一种方法是建立在各种环境下都能使用的类库，供相关程序直接使用；另一种方法是从它派生出一个满足当前需要的新类。继承性机制使得子类不仅可以重用其父类的数据和程序代码，而且可以在父类代码的基础上方便地修改和扩充，这种修改并不影响对原有类的使用。

2. 可控制程序的复杂性

面向对象程序设计方法采用了封装和信息隐藏技术，把数据及对数据的操作放在一个个类中，作为相互依存、不可分割的整体来处理。这样，在程序中任何要访问这些数据的地方都只需简单地通过传递信息和调用方法来进行，这就有效地控制了程序的复杂性。

3. 可改善程序的可维护性

在面向对象程序设计方法中，对对象的操作只能通过消息传递来实现，所以只要消息模式即对应的方法界面不变，方法体的任何修改不会导致发送消息的程序修改，这显然对程序的维护带来了方便。另外，类的封装和信息隐藏机制使得外界对其中的数据和程序代码的非法操作成为不可能，这也就大大地减少了程序的错误率。

4. 能够更好地支持大型程序设计

类是一种抽象的数据类型，所以类作为一个程序模块，要比通常的子程序的独立性强得多，面向对象技术在数据抽象上又引入了动态连接和继承性等机制，进一步发展了基于数据抽象的模块化设计，使其更好地支持大型程序设计。