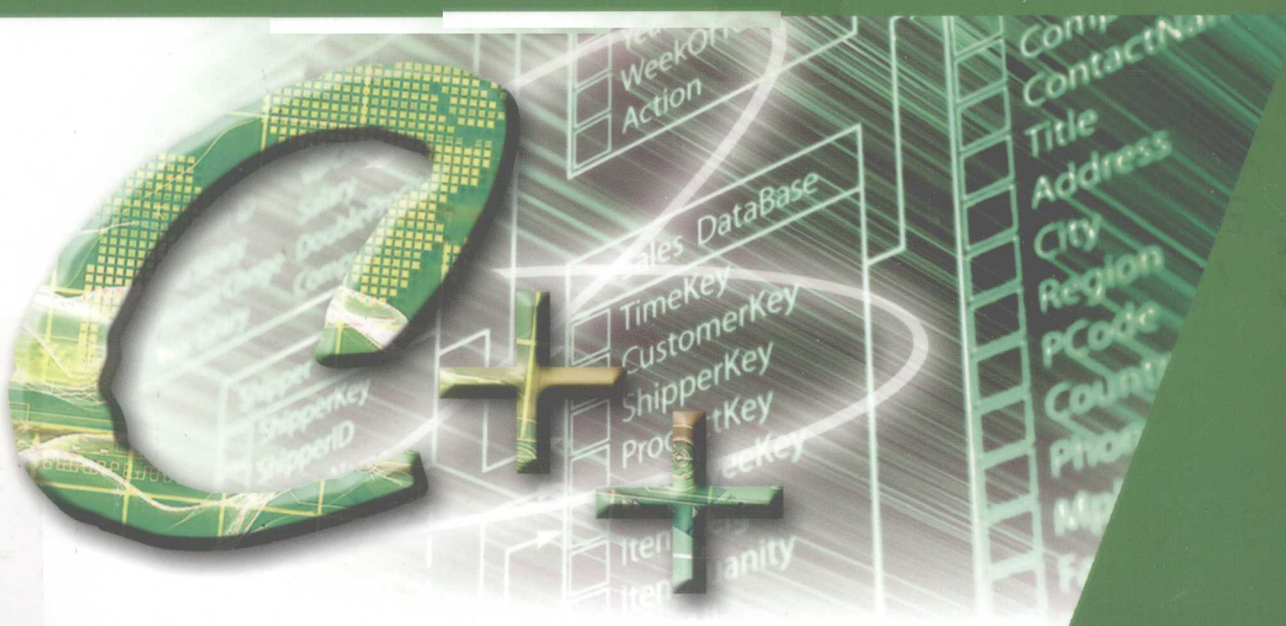
21世纪

高等学校精品规划教材

程序设计课程系列

C++面向对象程序设计

主编 栗青生 王爱民



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书从实际应用出发，系统地介绍C++面向对象程序设计的原理、方法和技巧。重点突出，叙述清楚，深入浅出，论述详尽，使读者既能深刻领会面向对象程序设计的思想，了解面向对象程序设计的特征，又能掌握C++语言的编程与应用。全书共8章，主要内容包括：面向对象程序设计语言概述、C++语言基础知识、类和对象、对象成员和友元、继承和派生、多态性和运算符重载、模板、C++的输入/输出流。在每一章的知识点后面，都给出了相应的程序设计实例，这些实例不仅有助于读者巩固知识点的内容，而且有助于读者的创新能力的培养。

本书适合作为普通高等院校计算机及其相关专业C++程序设计教材，也可供从事计算机软件开发的科研人员使用。

本书所配电子教案、程序源代码等均可从中国水利水电出版社网站以及万水书苑下载，网址为：http://www.waterpub.com.cn/softdown/和 http://www.wsbookshow.com，也可以与编者（aylqs@163.com）联系，或登录该课程教学网站（www.aylqs.cn），获取更多教学资源。

图书在版编目（CIP）数据

C++面向对象程序设计 / 栗青生，王爱民主编. --
北京 : 中国水利水电出版社，2010.2
21世纪高等学校精品规划教材
ISBN 978-7-5084-7197-6

Ⅰ. ①C… Ⅱ. ①栗… ②王… Ⅲ. ①
C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023003号

策划编辑：雷顺加　责任编辑：宋俊娥　加工编辑：陈　欣　封面设计：李　佳

书　名	21世纪高等学校精品规划教材 C++面向对象程序设计
作　者	主　编　栗青生　王爱民 副主编　刘明亮　杨玉星　刘国英　吴琴霞
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路1号D座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（营销中心）、82562819（万水）
经　售	全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	北京市天竺颖华印刷厂
规　格	184mm×260mm　16开本　14.25印张　350千字
版　次	2010年3月第1版　2010年3月第1次印刷
印　数	0001—4000册
定　价	25.00元

前　言

作为学习可视化面向对象技术的入门基础，C++语言已开始代替传统的 C 语言成为计算机教学语言。C++以类、对象、继承、封装、消息等概念提供了对面向对象特征的全面支持，又向下兼容了传统的 C 语言的结构化程序设计特征。因此，全面系统地学习 C++面向对象的程序设计语言，是可视化面向对象语言编程的基础。

任何一种计算机语言都离不开实践，本教材更注重理论和实践的统一，通过在每一章后面提供的程序实例、思考练习题和课本最后的实验，向读者提供丰富的操作、实验和实践题目，以期读者在实践中掌握面向对象程序设计语言的精髓。

本书共分 8 章，其中第 1 章主要讲述面向对象的基本思想；第 2 章讲述 C++语言基础；第 3 章到第 8 章讲述面向对象 C++的类、对象、派生、多态、重载等技术的理论、实例和应用，这是本教材的重点。

本书具有如下特点：

1．采用“理论＋实例＋实践”三结合的教学体系，更加重视学生实践能力的培养。

2．结合作者多年讲授“C++面向对象程序设计”的经验，灵活地安排课程的结构和内容，重点突出、难点易懂，即使没有 C 语言基础的读者也能系统地掌握。

3．考虑到不同学校实验平台的差异，精心设计的例题和实例在 Microsoft Visual C++ 6.0 系统和 Microsoft Visual Studio 2005/2008 系统上都能调试通过。

4．本教材的配套教学资源十分丰富，不仅有针对教师和学生的学习课件、配套的教学网站，而且还有教学视频，更方便学生自学。

本书由栗青生、王爱民任主编，刘明亮、杨玉星、刘国英、吴琴霞任副主编，参加编写的还有张长青、吴兴丽、杜科、郑小明、王胜金等，王娟、高春玲、刘超群、张智会、沙飞翔、潘中豪等同学参与了程序的调试，在此，对他们的帮助表示衷心的感谢。

本书配有电子教案、教学课件、课程辅导网站，需要者可以直接与编者联系。E-mail：aylqs@163.com。

教学网站：www.aylqs.cn。

由于编写时间仓促，教材中难免存在不足之处，敬请读者指正。

编　者

2010 年 1 月

目　　录

第 1 章　面向对象程序设计语言概述

学习过 C 语言的都知道，C 是面向过程的程序设计语言，也就是说 C 程序的设计首要考虑的是如何通过一个过程，对输入（或环境条件）进行运算处理得到输出（或实现过程（事务）控制）。这个过程的重点在于算法和数据结构。对于 C++，程序设计首先要考虑的是如何构造一个“模型”，让这个模型能够契合与之对应的问题域，这样就可以通过获取“模型”的状态信息得到输出（或实现过程（事务）控制）。这里的“模型”也称之为“对象模型”，因此 C++是面向对象的程序设计语言。

许多人都把 C++称为“带类的 C”。的确，对象模型的状态信息是由一些抽象的信息实例化得到的。这些抽象信息也就是我们说的“类”，所以说 C++是在 C 的基础上引入面向对象的“类”的机制而形成的一门程序设计语言。C++几乎继承了 C 的所有特点，同时添加了面向对象的特征。C++既支持面向过程的程序设计，又支持面向对象的程序设计。面向过程的程序设计语言是基于功能分析的，以算法为中心的程序设计方法；而面向对象的程序设计语言是基于结构分析的，以数据为中心的程序设计方法。

面向对象的程序设计方法具有三大特征：封装性、继承性和多态性，其基本思想是尽可能模拟人类的自然思维方式来构造软件系统，不仅可以提高对用户需求的适应性，而且支持软件复用。

- 了解面向对象程序设计的思想
- 掌握面向对象中出现的基本概念
- 掌握面向对象中出现的基本特征
- 掌握面向对象程序设计的优点

1.1　面向对象程序设计概述

软件工程学家 Coad/Yourdon 认为：面向对象=对象+类+继承+消息。如果一个计算机软件系统采用这些概念来建立模型并予以实现，我们就说它就是面向对象的。下面将阐述面向对象的程序设计和面向对象的设计方法。

1.1.1　面向对象程序设计

面向对象程序设计（Object Oriented Programming，缩写 OOP），指一种程序设计范型，同

时也是一种程序开发的方法论。它将对象作为程序的基本单元，将程序和数据封装其中，以提高软件的重用性、灵活性和扩展性。

当我们提到面向对象的时候，它不仅指一种程序设计方法，它更多意义上是一种程序开发范式。在这一方面，我们必须了解更多关于面向对象系统分析和面向对象设计（Object Oriented Design，简称 OOD）方面的知识。

面向对象程序设计的雏形早在 1960 年的 Simula 语言中即可发现，当时的程序设计领域正面临着一种危机：在软硬件环境逐渐复杂的情况下，软件如何得到良好的维护？面向对象程序设计在某种程度上通过强调可重复性解决了这一问题。20 世纪 70 年代的 SmallTalk 语言在面向对象方面堪称经典，以至于 40 年后的今天依然将这一语言视为面向对象语言的基础。

面向对象程序设计可以被视作一种在程序中包含各种独立而又互相调用的单位和对象的思想，这与传统的思想刚好相反：传统的面向过程程序设计主张将程序看作一系列函数的集合，或者直接就是一系列对电脑下达的指令。面向对象程序设计中的每一个对象都应该能够接受数据、处理数据并将数据传达给其它对象，因此它们都可以被看作一个小型的“机器”，或者说是负有责任的角色。

目前已经被证实的是，面向对象程序设计推广了程序的灵活性和可维护性，并且在大型项目设计中广为应用。

1.1.2 面向对象的软件工程

传统的软件工程方法曾经给软件产业带来了巨大进步，部分缓解了软件危机，但随着人们对软件产品需求的日益增加，其缺点越来越突出。为了克服传统工程开发的缺点，20 世纪 70 年代提出了面向对象方法，现在它已经有很广泛的应用。面向对象软件工程是面向对象方法在软件工程领域运用的结果。

1.1.3 面向对象的主要概念

1. 对象

对象是一个实体，可以是现实世界中具体的物理实体或概念化的抽象实体。

在现实生活中，任何事物都是对象，具体存在的事物，比如一个学校是对象，桌、椅是对象，抽象事物比如规章制度也是对象。对象既可以很简单，也可以很复杂。一般一个对象都有一个区别于其他事物的名字，也有描述它特征的属性，以及一组操作，这组操作既可以是自身所承受的，也可以是施加给其他对象的。

在面向对象程序设计中，对象是一个封装数据（属性，静态特征）和操作（服务，动态特征）的实体，是构成系统的基本单元。当用户使用一个对象的时候，只能通过对象所提供的对外界的接口进行访问，而不必知道它的操作方法，就像我们知道的黑匣子一样，你可以使用它提供的接口，却不用知道内部的构造。这就大大方便了用户的使用，使对象的使用变得十分简单。同样，因为在外面看不到对象的内部，更无法对它进行修改，因而也有很高的安全性和可靠性。

2. 类

类是具有相同属性和相同操作的对象的集合，是抽象数据类型的实现。在此定义一个学生类，众所周知，学生具有的属性一般有姓名、学号、性别、成绩等，相应的操作有入学，修改、显示、毕业等。当入学时，需要一定的操作将此学生的信息加入学生类中，而毕业则应该

撤消其相应的信息。具体到每个学生则是学生类的一个实例，也就是一个对象。

可以说，对象的抽象是类，类的实例是对象。在客观世界存在的是类的实例，即对象。比如上例中的学生，我们通常不说抽象的“学生”，而是一个个具体的学生，他们有着自己的姓名、学号等。

在面向对象的程序设计中，总是先声明类，再定义此类的对象，类是创建对象的模板，给出了属于该类的全部对象的抽象定义。就像大批量生产某种产品时，我们总是先把这种商品的模具做好，然后开始生产产品，经过一个模具生产的产品，总有相同的特征。在这里可以把模具看作类，而生产出来的产品看作对象。

3. 封装

所谓数据封装就是指将一组数据和与这组数据有关的操作集合组装在一起，形成一个能动的实体。封装是指把对象属性和操作结合在一起，构成独立的单元，它的内部信息对外界是隐蔽的，不允许外界直接存取对象的属性，只能通过有限的接口与对象发生联系。

在面向对象的程序设计中，封装是把数据和实现操作的代码放在对象内部，隐蔽一些内部细节。类是数据封装的工具，对象是封装的实现。外界要访问时需要通过类，类的访问控制机制体现在类的成员中可以有公有成员、私有成员和保护成员。其中共有成员和保护成员允许访问，私有成员是不允许访问的。对于外界而言，只需要知道对象所表现的外部行为，而不必了解内部实现细节。封装体现了面向对象方法的“信息隐蔽和局部化原则”。

4. 继承

从已有的对象类型出发建立一种新的对象类型就是继承。继承在现实生活中是普遍存在的，比如说我们经常见到的汽车，它就继承了机动车的一些特点，比如都需要燃料、引擎等。同样它也有自己的一些特征，引入继承的一个原因是可以提高代码的重用率，更重要的是对代码的扩充。

面向对象的程序设计中，继承指子类（派生类）可以自动拥有父类（基类）的全部属性和服务。父类和子类是一般与特殊的关系。在定义一个子类时，可以把父类所定义的内容作为自己的内容，并加入若干新的内容。继承是面向对象语言的重要特性，提高了软件的可重用性。

继承分为单重继承和多重继承。单重继承时，一个子类只有一个父类，它继承的是一个基类的特征。多重继承时，一个子类可以有多个父类，它所继承的特征可能不止来自于一个基类。单重继承构成的类之间的关系是树状结构，多重继承构成的类之间的关系是网状结构。

5. 消息

在现实生活中对象都不会是孤立存在的，他们之间都存在着这种或者那种的联系。在面向对象程序中也需要一种机制对这种联系进行描述，这就是消息机制。

首先来解释一下消息，消息是指对象之间在交互通信中所传送的信息，即一个对象对另一个对象发出的请求。一般情况下，我们把发送消息的对象称为发送者或者请求者，接收消息的对象称为接收者或者目标对象。当一个对象向另一个对象发送消息请求某项服务时，接收消息的对象响应该消息，进行所要求的服务，并把操作的结果返回给请求服务的对象。这样就完成了消息的传递。

消息由三部分构成：消息名、接收消息的对象标识和参数。一般它具有以下几个性质：

（1）一个对象可以接收不同的消息，进行不同的响应。

（2）相同的消息被不同的对象接收，做出的响应也可以是不同的。

（3）有些对象可以只接收对象，而不用响应，即对消息的响应并不是必需的。

消息也可以分为两类：公有的和私有的。当有一些消息传递给一个对象时，如果消息是其他对象直接传给它的，这些消息就是公有（public）消息；如果消息是这个对象传给它自己的，这些消息就称为私有（private）消息。

6. 多态性

多态性是指相同的操作、函数或过程可作用于多种类型的对象上并获得不同的结果。多态性也是面向对象系统的重要特征。在面向程序的设计中，多态性是指在基类中定义的属性和服务被子类继承后，可以具有不同的数据类型和表现出不同的行为。当一个对象接收到一个请求进行某项服务的消息时，将根据对象所属的类，动态地选用该类中定义的操作。不同的类对消息按不同的方式解释。

多态性机制不但为软件设计提供了灵活性，减少信息冗余，而且显著提高了软件的可复用性和可扩充性。C++中的多态可以分为四类：参数多态、包含多态、重载多态和强制多态。这些将会在后面的课程中一一进行介绍。

C++支持两种多态性，分别是编译时的多态性和运行时的多态性。在此，编译时的多态性是通过重载实现的，运行时的多态性是通过虚函数实现的。这些内容将在第 6 章详细讲述。

1.2 面向对象程序设计的特点

1.2.1 传统程序设计方法的局限性

用传统的结构化方法开发大型软件系统涉及各种不同领域的知识，在开发需求模糊或需求动态变化的系统时，所开发出的软件系统往往不能真正满足用户的需要。

传统的程序设计方法存在下面几点局限性：

（1）传统程序设计开发软件的生产效率低下。

（2）传统程序设计难以应付日益庞大的信息量和多样的信息类型。

（3）传统的程序设计难以适应各种新环境。

实践证明，用传统方法开发出来的软件，维护时其费用和成本仍然很高，原因是可修改性差，维护困难，导致可维护性差。

用结构化方法开发的软件，其稳定性、可修改性和可重用性都比较差。这是因为结构化方法的本质是功能分解，从代表目标系统整体功能的单个处理着手，自顶向下不断把复杂的处理分解为子处理，这样一层一层地分解下去，直到仅剩下若干个容易实现的子处理功能为止，然后用相应的工具来描述各个最底层的处理。因此，结构化方法是围绕实现处理功能的“过程”来构造系统的。然而，用户需求的变化大部分是针对功能的，因此，这种变化对于基于过程的设计来说是灾难性的。用这种方法设计出来的系统结构常常是不稳定的，用户需求的变化往往造成系统结构的较大变化，从而需要花费很大代价才能实现这种变化。

1.2.2 面向对象程序设计的主要优点

面向对象程序设计既吸取了结构化程序设计的一切优点，又考虑了现实世界与面向对象解空间的映射关系，它所追求的目标是将现实世界的问题求解尽可能简单化。面向对象程序设计

将数据及对数据的操作放在一起，作为一个相互依存、不可分割的整体来处理，并采用了数据抽象和信息隐藏技术。它将对象及对象的操作抽象成一种新的数据类型——类，并且考虑不同对象之间的联系和对象所在类的重要性。

面向对象程序设计优于传统的结构化程序设计，其优越性表现在，它有希望解决软件工程的两个主要的问题——软件复杂性控制和软件生产率的提高。此外它还符合人类的思维习惯，能够自然地表现现实世界的实体和问题，它对软件开发过程具有重要的意义。

在面向对象程序设计中可以用下面的式子表示程序：

程序=对象+对象+…+对象

对象=算法+数据结构+程序设计语言+语言环境

在结构化程序设计中可以用下面的式子表示程序：

程序=数据结构+算法+程序设计语言+语言环境

面向对象的程序设计具有如下优点：

（1）符合人们习惯的思维方法，便于分解大型的复杂多变的问题。由于对象对应于现实世界中的实体，因而可以很自然地按照现实世界中处理实体的方法来处理对象，软件开发者可以很方便地与问题提出者进行沟通和交流。

传统的程序设计技术是面向过程的设计方法，这种方法以算法为核心，把数据和过程作为相互独立的部分，数据代表问题空间中的客体，程序代码用于处理这些数据。

把数据和代码作为分离的实体，反映了计算机的观点，因为在计算机内部数据和程序是分开存放的。但是，这样做的时候总存在使用错误的数据调用正确的程序模块，或使用正确的数据调用错误的程序模块的危险。使数据和操作保持一致，是程序员的一个沉重负担，在多人分工合作开发一个大型软件系统的过程中，如果负责设计数据结构的人中途改变了某个数据的结构而又没有及时通知所有人员，则会发生许多不该发生的错误。

传统的程序设计技术忽略了数据和操作之间的内在联系，用这种方法所设计出来的软件系统其解空间与问题空间并不一致，令人感到难以理解。实际上，用计算机解决的问题都是现实世界中的问题，这些问题无非由一些相互间存在一定联系的事物组成。每个具体的事物都具有行为和属性两方面的特征。因此，把描述事物静态属性的数据结构和表示事物动态行为的操作放在一起构成一个整体，才能完整、自然地表示客观世界中的实体。

面向对象的软件技术以对象（Object）为核心，用这种技术开发出的软件系统由对象组成。对象是对现实世界实体的正确抽象，它是由描述内部状态表示静态属性的数据，以及可以对这些数据施加的操作（表示对象的动态行为），封装在一起所构成的统一体。对象之间通过传递消息互相联系，以模拟现实世界中不同事物彼此之间的联系。

面向对象的开发方法与传统的面向过程的方法有本质不同，这种方法的基本原理是，使用现实世界的概念抽象地思考问题，从而自然地解决问题。它强调模拟现实世界中的概念而不强调算法，它鼓励开发者在软件开发的绝大部分过程中都用应用领域的概念去思考。在面向对象的开发方法中，计算机的观点是不重要的，现实世界的模型才是最重要的。面向对象的软件开发过程从始至终都围绕着建立问题领域的对象模型来进行，即对问题领域进行自然的分解，确定需要使用的对象和类，建立适当的类等级，在对象之间传递消息实现必要的联系，从而按照人们习惯的思维方式建立起问题领域的模型，模拟客观世界。

（2）易于软件的维护和功能的增减。对象的封装性及对象之间的松散组合，都给软件的

修改和维护带来了方便。面向对象的软件技术符合人们习惯的思维方式，因此用这种方法建立的软件系统容易被维护人员理解，他们可以主要围绕派生类进行修改、调试工作。类是独立性很强的模块，向类的实例发消息即可运行它，观察它是否能正确地完成要求它做的工作，对类的测试通常比较容易实现，如果发现错误也往往集中在类的内部，比较容易调试。总之，面向对象技术的优点并不是减少了开发时间，相反，初次使用这种技术开发软件，可能比用传统方法所需时间还稍微长一点。开发人员必须花很大精力去分析对象是什么，每个对象应该承担什么责任，所有这些对象怎样很好地合作以完成预定的目标。这样做换来的好处是，提高了目标系统的可重用性并减少了生命周期后续阶段的工作量和可能犯的错误，提高了软件的可维护性。此外，一个设计良好的面向对象系统是易于扩充和修改的，因此能够适应不断增加的新需求。以上这些都是从长远考虑的软件质量指标。

（3）可重用性好。重复使用一个类（类是对象的定义，对象是类的实例化），可以比较方便地构造出软件系统，加上继承方式的运用，极大地提高了软件开发的效率。用已有的零部件装配新的产品，是典型的重用技术，重用是提高生产效率的一个重要方法。面向对象的软件技术在利用可重用的软件成分构造新的软件系统时，体现出较大的灵活性。它可利用两种方法重复使用一个类：一种方法是创建该类的实例，从而直接使用它；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结构和程序代码，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。由于可以像使用集成电路（IC）构造计算机硬件那样，比较方便地重用对象类来构造软件系统，因此，有人把类称为“软件 IC”。面向对象的软件技术所实现的可重用性是自然和准确的，在软件重用技术中它是最成功的一个。

（4）与可视化技术相结合，改善了工作界面。随着基于图形界面操作系统的流行，面向对象的程序设计方法也将深入人心。它与可视化技术相结合，使人机界面进入 GUI 时代。

1.3 面向对象的系统开发方法

支持部分或绝大部分面向对象特性的语言即可称为基于对象的或面向对象的语言。使用面向对象的语言进行面向对象的系统开发，首先是采用面向对象的概念及其抽象的机制将开发系统对象化和模型化，建立应用系统模型，然后实现系统中的对象。面向对象的开发方法可分为五个阶段：

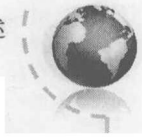
（1）系统调查和需求分析阶段。针对应用系统将要实现的功能以及用户对系统开发的需求进行调查研究。

（2）分析问题的性质和求解问题阶段。在繁杂的问题域中抽象地识别出对象及其行为、结构、属性、方法等。这一阶段称为面向对象分析，简称为 OOA。

（3）整理问题阶段。即对分析的结果作进一步地抽象、归类、整理，最终以规范的形式描述对象和类。这一阶段称为面向对象设计，简称为 OOD。

（4）程序实现阶段。即用面向对象的程序设计语言将第三阶段整理的对象和类的描述映射为应用程序软件。这一阶段称为面向对象程序设计，简称为 OOP。

（5）软件测试阶段。测试是系统开发必须要进行的重要环节，在软件工程项目中占有重要地位，因而倍受重视。



1.3.1　典型的面向对象程序设计语言

（1）Simula 语言。Simula 语言是 20 世纪 60 年代开发出来的，随着结构化程序设计的研究，在计算机科学的诸多领域，提出了面向对象的概念，用类描述一个对象集合的结构和行为。Simula 也支持类继承，用继承按层次组织类，允许实现和结构的共享。

在 Simula 中引入了几个面向对象程序设计语言中最重要的概念和特性，即数据抽象、类和继承性机制。Simula67 是它具有代表性的一个版本，20 世纪 70 年代发展起来的 CLU、Ada、Modula-2 等语言是在它的基础上发展起来的。

（2）Smalltalk 语言。Smalltalk 是 20 世纪 70 年代进一步开发的面向对象程序设计语言。Smalltalk 有几个版本，如 Smalltalk 72、74、76、78、80 等。但最重要和最稳定的是 Smalltalk80。Smalltalk 不仅是一种程序设计语言，而且引入了一个完整的程序设计开发环境和一个基于图形的交互式用户界面。

Smalltalk 语言引入的面向对象的性能有类、继承、对象标识和信息隐藏。Smalltalk 是非类型化语言，Smalltalk 中的变量在说明时不必指定类型，相同程序的相同变量在不同的时间可以具有不同的类型。一个变量的类型由该变量所引用的对象所属的类决定。类好似产生实例的工厂，可以为类实例定义方法并把方法应用到类实例本身。

Smalltalk 面向对象的概念是极为丰富的。Smalltalk 中把每件事都看作是一个对象，包括类和基本数据类型（整数、实数等）。在整个 Smalltalk 的环境中，程序设计包括给对象传递消息。一个消息可以把一个数加到另一个数上，或可以产生一个类的新实例，亦或可以在一个给定的类中引入一个新方法。

Smalltalk 是第一个真正的面向对象程序设计语言，它体现了纯粹的 OOP 设计思想，是最纯的 OOP 语言，它起源于 Simula 语言，尽管 Smalltalk 不断完善，但在那个时期，面向对象程序设计语言并没有得到广泛的重视，程序设计的主流是结构化程序设计。

（3）C++语言。在 19 世纪 80 年代，C 语言成为一种极其流行、应用非常广泛的语言。C++是在 C 语言的基础上进行扩充，并增加了类似 Smalltalk 语言中相应的对象机制的程序设计语言。它将“类”看作是用户定义类型，使其扩充比较自然。C++以其高效的执行效率赢得了广大程序设计员的青睐，在 C++中提供了对 C 语言的兼容性，因此，很多已有的 C 程序稍加改造甚至不加改造就可以重用，许多有效的算法也可以重新利用。它是一种混合型的面向对象程序设计语言，由于它的出现，才使面向对象的程序设计语言越来越得到重视和广泛的应用。

C++以其独特的语言机制在计算机科学的各个领域中得到了广泛的应用。面向对象的设计思想是在原来结构化程序设计方法基础上的一个质的飞跃，C++完美地体现了面向对象的各种特性。

目前，基于 C++语言在不同平台下的程序开发工具有很多种，在 Windows 中的编程环境以微软的 Visual C++ 为主，另外还有 C++ Builder、Dev-C++等，本书的教学内容主要介绍面向对象的 C++语言，并不针对特定的语言开发环境。

（4）Java 语言。Java 语言是一种适用于分布式计算的新型面向对象程序设计语言，可以看作是 C++语言的派生，它从 C++语言中继承了大量的语言成分，抛弃了 C++语言中冗余的、容易引起问题的功能，增加了多线程、异常处理、网络程序设计等方面的支持，掌握了 C++语言，可以很快学会 Java 语言。

Java 吸取了 C++面向对象的概念，将数据封装于类中，利用类的优点，实现了程序的简洁性和便于维护性。类的封装性、继承性等有关对象的特性，使程序代码只需一次编译，然后通过上述特性反复利用。程序员只需把主要精力用在类和接口的设计和应用上。Java 提供了众多的一般对象的类，通过继承即可使用父类的方法。在 Java 中，类的继承关系是单一的非多重的，一个子类只有一个父类，子类的父类又有一个父类。Java 提供的 Object 类及其子类的继承关系如同一棵倒立的树形，根类为 Object 类，Object 类功能强大，经常会使用到它及其它派生的子类。

（5）C#语言。C#是一种现代编程语言。作为编程语言，C#是现代的、简单的、完全面向对象的，而且是类型安全的。重要的是，在类、名字空间、方法重载和异常处理等方面，C#去掉了 C++中的许多复杂性，借鉴和修改了 Java 的许多特性，使其更加易于使用，不易出错，使用起来更加方便。

1.3.2 C++面向对象程序设计流程

面向对象程序设计语言发展到现在，一般都有一个集成的开发环境（Integrated Development Environment，IDE），它是一个将程序编辑器、编译器、调试工具和其他建立应用程序的工具集成在一起的用于开发应用程序的软件系统。近年来，许多软件公司都根据 C++的标准设计了各具特色的编程开发环境，在 Windows 上，应用比较多的是 Borland C++ Builder 6.0 或以上版本、Microsoft Visual C++ 6.0 或者 Visual C++.NET，在 Linux 系统上，比较多的是 G++3.0 和 Borland C++ Kylix3.0 或以上版本，但不论使用怎能样的开发环境，程序开发的流程都是一样的。一个用 C++开发的项目的通用开发过程可以用图 1-1 来表示。

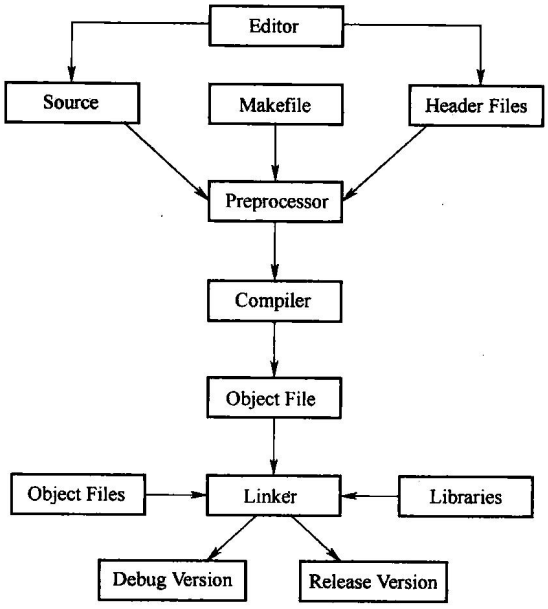
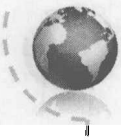


图 1-1 程序开发流程

建立一个项目的基本步骤是：



（1）利用编辑器建立程序代码文件，包括头文件、代码文件、资源文件等。

（2）启动编译程序，编译程序首先调用预处理程序处理程序中的预处理命令（如#include，#define 等），经过预处理程序处理的代码将作为编译程序的输入。

（3）编译对用户程序进行词法和语法分析，建立目标文件，文件中包括机器代码、连接指令、外部引用以及从该源文件中产生的函数和数据名。

（4）连接程序将所有的目标代码和用到的静态连接库的代码连接起来，为所有的外部变量和函数找到其提供地点。

（5）生成一个可执行文件。一般有一个 makefile 文件来协调各个部分产生可执行文件。

可执行文件分为两种版本：Debug 和 Release。Debug 版本用于程序的开发过程，该版本产生的可执行程序带有大量的调试信息，可以供调试程序使用；而 Release 版本作为最终的发行版本，没有调试信息，并且带有某种形式的优化。

不同的集成开发环境中都集成了编辑器、编译器、连接器以及调试程序，覆盖了开发应用程序的整个过程，程序员不需要脱离这个开发环境就可以开发出完整的应用程序。

1.4　程序举例

下面以一个简单的面向对象程序为例，说明面向对象程序的特点及设计、开发和调试的方法。

在教学资源网站 www.aylqs.cn 上下载本章实例 1 程序，在不同的开发环境下进行调试，了解面向对象应用程序的开发方法和过程。

【实例 1】使用面向对象的设计方法，编写一个满足如下要求的程序：

（1）根据输入的日期输出这一日期的前一天和后一天的日期。

（2）根据输出的日期判断是星期几。

```
#include<iostream>
using namespace std;
struct Date
{
    int month;
    int day;
    int year;
};
class TdateType
{
   public:
     TdateType();              //不带参数的构造函数定义
     TdateType(Date b);        //有参数的构造函数定义
     void Next();              //明天的日期成员函数定义
     void Previous();          //昨天的日期成员函数定义
     int Weekday();            //判断是星期几成员函数定义
     void Print();             //打印日期
   private:
```

```
    Date a;                    //日期结构数据成员
    int IsLeapYear();          //私有成员函数,判断是否闰年
    int MonthEnd(int m);       //计算某月的天数
};
//以下为类的成员函数实现部分
TdateType::TdateType()
{
    a.year=1999;
    a.month=1;
    a.day=1;
}
TdateType::TdateType(Date b)
{
   a.month = b.month;
   a.day = b.day;
   a.year = b.year;
}
void TdateType::Next()
{
   a.day ++;
   if( a.day >MonthEnd(a.month) )
   {
      a.day = 1;
      a.month ++;
      if ( a.month > 12)
        {
          a.month  = 1;
          a.year ++;
        }
    }
}
void TdateType::Previous()
{
   a.day --;
   if ( a.day < 1 )
   {
     a.month --;
     if (a.month < 1 )
       {
         a.month = 12;
         a.year --;
       }
     a.day = MonthEnd(a.month);
    }
}
int TdateType::IsLeapYear()
```

```
{
    return ((a.year % 4==0 && a.year % 100!=0 )||(a.year % 400==0));
}
int TdateType::MonthEnd(int m)
{
     switch(m)
     {
          case 1:
          case 3:
          case 5:
          case 7:
          case 8:
          case 10:
          case 12: return 31;
          case 4:
          case 6:
          case 9:
          case 11:return 30;
           case 2:
              if (IsLeapYear())
                  return 29;
              else
                  return 28;
     }
     return 0;
}
int TdateType::Weekday()
{
     long n;
     n=((a.year)-1)*365;                    //直至去年的天数（不考虑闰年）
     n+=((a.year )-1)/4;                    //以下 3 条语句考虑闰年数
     n-= ((a.year)-1)/100;
     n+=((a.year)-1)/400;
     for ( int i=1;i<a.month;i++)           //本年直至上月的天数
         n+=MonthEnd(i);
     n +=a.day;                             //本月的天数
     n %=7;                                 //折算成星期几,若 n 是 0,则为星期日
     return n;
}
void TdateType::Print()
{
   cout <<a.year<<"年" <<a.month<<"月" <<a.day<<"日";
    switch (Weekday() )
    {
       case 0:cout<<"星期日\n";break;
       case 1:cout<<"星期一\n";break;
```