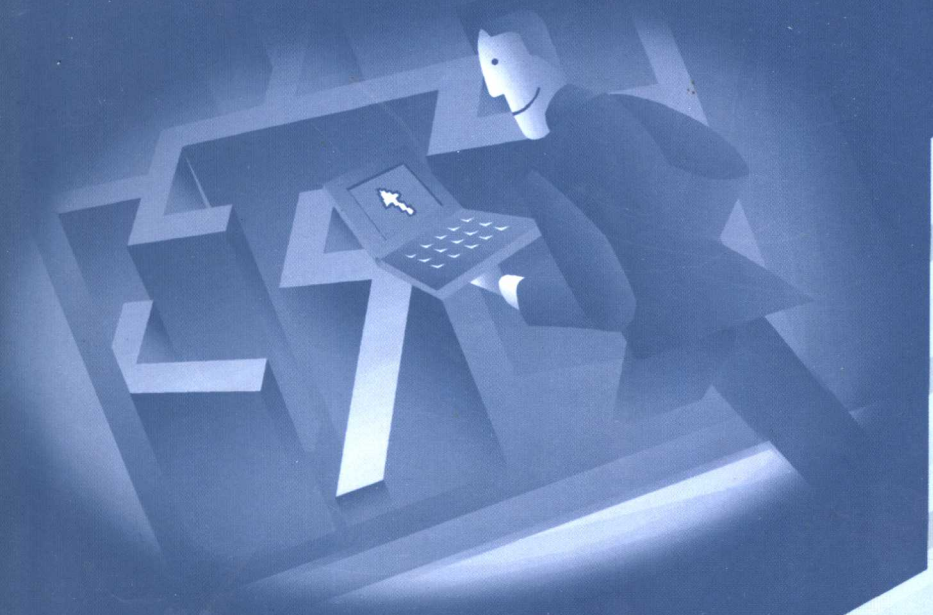
C++面向对象程序设计

C++ MIANXIANG DUIXIANG CHENGXU SHEJI



郝谦 主编

北京邮电大学出版社
www.buptpress.com

C++面向对象程序设计

郝 谦 主编

北京邮电大学出版社
·北京·

内 容 简 介

本书全面系统地介绍了C++面向对象程序设计的基本概念、基本语法和编程方法，全书以大量的实例详尽地讲述了C++面向对象的基本特性：类、对象、派生类、继承、多态性、虚函数、模板、流类库等，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，掌握其基本要领、基本概念和基本方法。

本书以应用为目的，注重培养应用能力，可作为大学本、专科信息类及相关专业学生学习C++程序设计课程使用，也可作为C++语言培训、自学教材或C++软件开发工作者参考用书。

本书例题的源代码与课件随书光盘一并发行。

图书在版编目(CIP)数据

C++面向对象程序设计/郝谦主编. —北京：北京邮电大学出版社，2006

ISBN 7-5635-1204-7

Ⅰ.C... Ⅱ.郝... Ⅲ.C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(2006)第010025号

书　　名：C++面向对象程序设计

主　　编：郝　谦

责任编辑：周　堃

出版发行：北京邮电大学出版社

社　　址：北京市海淀区西土城路10号(100876)

北方营销中心：电话：010-62282185　传真：010-62283578

南方营销中心：电话：010-62282902　传真：010-62282735

E - mail：publish@bupt.edu.cn

经　　销：各地新华书店

印　　刷：北京通州皇家印刷厂

开　　本：787 mm×1 092 mm　1/16

印　　张：16

字　　数：388千字

印　　数：1—3 000册

版　　次：2006年2月第1版　2006年2月第1次印刷

ISBN 7-5635-1204-7/TP·224　　定价：24.00元

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方法。它对降低软件的复杂性、改善软件的重用性和可维护性、缩短软件开发的周期、提高软件的生产效率、降低软件开发的成本，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的一场实质性的革命。

C++语言是在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，掌握其基本要领、基本概念和基本方法，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。

近年来许多高校纷纷将面向对象程序设计及面向对象技术正式列入教学计划的课程设置中，在实际应用领域中也将C++作为软件开发的平台。但许多教材对理论叙述过多，实际应用实例较少，学生学完课程后只会考试，不会利用C++开发软件。

鉴于上述情况，我们在多年的教学和科研的基础上编写了这本教材，旨在使读者迅速跨进面向对象程序设计及C++语言开发平台的大门，掌握C++程序设计的基本思想和基本方法，并能较快地编写出具有良好网络的程序。本书的特点是：通俗易懂、由浅入深、重点实出、便于理解、适于自学。

本教材共分9章，第1章——概述，第2章——C++简单程序设计，第3章——函数和作用域，第4章——指针和引用，第5章——类和对象(一)，第6章——类和对象(二)，第7章——继承与派生，第8章——多态性与虚函数，第9章——C++的I/O流类库。

本教材由郝谦、郑睿颖主编，万红新、周雪梅、杨长红副主编，参加编写及制作课件的还有李宏芳、程山英、曾纯清、段薇、詹莹、郭攀、程琳、邓茹仁、范严、涂伟、李希、汪闰六等同志。最后由郝谦、郑睿颖统一审稿。

本书的总课时为80～100左右，建议其中理论讲述与上机实践课时比为1.5:1～2:1。

在本书编写过程中，得到了华东交通大学李正凡教授、南昌航空学院舒坚教授、北京邮电大学出版社的章剑编辑及出版社其他同志的大力帮助，在此一并表示感谢。

由于水平有限及时间紧张等主客观原因，本书难免有不足或错误之处，恳请读者原谅并给予批评指正。

本书可作为大学本、专科信息类及相关专业学生学习C++程序设计课程使用，也可作为C++语言培训、自学教材或C++软件开发工作者参考用书。

编　者

2006年1月

目　录

第6章 类和对象(二)

第7章 继承与派生

第8章 多态性和虚函数

第1章 概 述

1.1 程序与语言

1.1.1 程序

程序是用能够被计算机理解的一种语言编写的语句的集合。它以某种语言为工具编制出有目的的、预想好的动作序列，表达人的思想。

对于计算机来说，一组机器指令就是程序。当我们说机器代码或者机器指令时，都是指程序，它是按计算机硬件设计规范的要求编制出来的动作序列。对于使用计算机的人来说，程序员用某高级语言编写的语句序列也是程序。程序通常以文件的形式保存起来。所以，源文件、源程序和源代码都是程序。

一个好的程序应该有以下特点：

(1) 正确可靠。不正确的程序不仅不能解决问题，反而会带来不必要的麻烦和损失。

(2) 清晰易读。由于受计算机速度和存储容量的限制，早期的程序往往将程序的效率放在第一位，随着科技发展，程序的可读性和可理解性成为设计的重点考虑内容。

(3) 易维护。当业务需求发生变化时，不需要太多的开销就可以扩展和增强程序的功能。

(4) 可移植性好。编写的程序在各种计算机和操作系统上都能运行，并且运行结果一样。

1.1.2 程序设计语言的发展

程序设计语言的发展经历了机器语言、汇编语言和高级语言等阶段，总的趋势是描述手段越来越高级，越来越接近自然语言或数学语言，越来越贴近客观世界本身。

最早，程序员使用最原始的计算机指令，即机器语言程序。只有机器语言才能为机器所识别和运行。这些指令由一串二进制的数表示。不久，发明了汇编语言，它可以将机器指令映射为一些能被人读懂的助记符，如 ADD，SUB。程序员运行汇编程序，将用助记符写成的源程序转换成机器指令，然后再运行机器指令程序，得到所要的结果。那时，编写程序的都是计算机专业人员，编写程序的语言都是低级的或是较低级的。这些语言的优点是：写出的程序效率较高。缺点是：程序难以设计、理解和维护，难以保证程序的正确性，此外，可移植性不好。

面对上述问题，Fortran，BASIC，Pascal；C 等几十种甚至几百种高级语言应运而生，中间

经历了严酷的优胜劣汰过程,最后剩下的是一些比较优秀的高级语言。高级语言的优点在于:程序容易设计、理解和维护,容易保证程序的正确性,另外,用高级语言编写的程序与采用的具体计算机的指令系统无关,这就容易将程序移植到其他不同型号的计算机中执行。

但是从本质上说,目前的高级语言大都只是在抽象级上比低级语言略高级一些而已,它们大都还是基于冯·诺依曼计算机的计算模型,而采用这些语言就必须按照计算机解决问题的方式来描述解题过程,所以程序设计仍然很困难。因此,人们还在努力设计抽象级别更高的语言或让计算机能够理解自然语言,以便程序员能用更自然的方式来设计程序。

1.2 面向对象程序设计的方法

1.2.1 面向对象方法的由来

"对象"一词在现实生活中经常会遇到,它表示现实世界中的某个具体的事物。

社会的不断进步和计算机科学的不断发展是相互促进的,一方面计算机科学的发展推动了社会的发展,计算机的广泛应用给整个社会生产力带来了勃勃生机;另一方面社会的发展,又给计算机科学提出了许多新的要求,计算机科学只有不断地进行自身提高和自身完善,才能适应不断进步的社会生产力的需要。随着计算机的普及应用,人们越来越希望能更直接地与计算机进行交互,而不需要经过专门学习和长时间训练后才能使用它。这一强烈愿望使软件设计人员的负担越来越重,也为计算机领域自身的发展提出了新的要求。利用传统的程序设计思想无法满足这一要求,人们就开始寻求一种更能反映人类解决问题的自然方法,"面向对象"技术就是在这样的情况下产生的。

"面向对象"技术追求的是软件系统对现实世界的直接模拟,尽量实现将现实世界中的事物直接映射到软件系统的解空间。它希望用户用最小的气力,最大程度地利用软件系统来解决问题。

现实世界中的事物可分为两大部分,即物质和意识,物质表达的是具体的事物;意识描述的是某一个抽象的概念。例如"汽车"和"那辆红色的汽车","那辆红色的汽车"是物质,它是具体的客观存在;"汽车"是意识,它是一个抽象的概念,是对客观存在的事物的一种概括。这些现实世界中的事物可直接映射到面向对象系统的解空间,现实世界中的物质可对应于面向对象系统中的"对象",现实世界中的意识可对应于面向对象系统中的抽象概念——类。汽车在面向对象系统中可用汽车类来表达,那辆红色的汽车在面向对象系统中是一个具体的对象,是汽车类的一个实例。如图 1.1 所示。

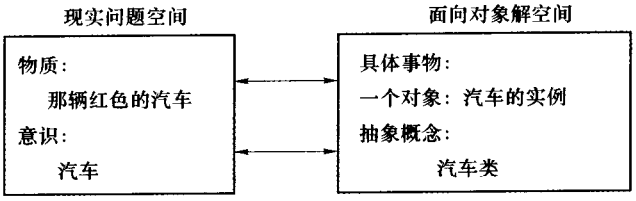


图 1.1 现实世界与面向对象系统之间的对应关系

1.2.2 面向对象的基本概念

面向对象程序设计(Object Oriented Programming)是软件系统设计与实现的新方法,这种新方法是通过增加软件的要扩充性和可重用性,来改善并提高程序员的生产能力,并控制维护软件的复杂性和软件维护的开销。下面先介绍面向对象程序设计的相关概念。

面向对象程序设计是由若干对象构造程序,每个对象由一些数据以及对这些数据所能实施的操作构成;对数据的操作通过向包含数据的对象发送消息(调用对象的操作)来实现;对象的特征(数据与操作)由相应的类来描述;一个类所描述的对象特征可以从其他的类继承。面向对象程序设计的定义包含了下面几个基本概念:

对象:是包含数据和处理这些数据的操作的程序单元,是构成面向对象程序的基本计算单位,由接口、数据及其操作构成。

通信:是指对象间的消息传递,是引起面向对象程序进行计算的惟一方式。

类:描述了一组具有相同或相近特征的对象的结构和行为。

继承:是指对象的一部分特征描述可以从其他的类获得,实现对数据和操作的共享。

1.2.3 面向对象程序设计与结构化程序设计

结构化程序设计(Structure Programming)是由 E. Dijkstra 等人于 1972 年提出的,它建立在 Bohm、Jacopini 证明的结构定理的基础上。结构定理指出:任何程序逻辑都可以用顺序、选择和循环三种基本结构来表示。

按照结构化程序设计的要求,程序在设计中应当采用"自顶向下,逐步求精"和"模块化"原则。"自顶向下,逐步求精"就是先需要对问题本身做出确切描述,并对问题解法做出全局性决策,把问题分解成相对独立的子问题,对每个子问题用抽象数据及其上的抽象操作来描述;然后,再以同样的方式对抽象数据和抽象操作进一步精确化,直到获得计算机能理解的程序为止。在这个过程中,要求程序设计必须先考虑全局,逐步将问题具体化。"模块化"是将一个大的系统按照子结构之间的疏密程度分解成较小的部分,每部分称为模块。分解的原则是模块间应相互独立,联系较少。

结构化程序设计对程序功能的描述比较清晰,所描述的计算过程容易理解,对规模较小的软件,它是非常适用的。但是由于它的数据与操作分离,对于不同的数据格式做相同的处理或是对相同的数据做不同的处理都需要编写不同的程序,随着问题复杂度的提高,数据量和数据类型的空前激增,会导致许多程序的规模和复杂性接近或达到用结构化程序设计方法无法管理的程度。

面向对象的程序设计是在吸收结构化程序设计的一切优点的基础上发展起来的一种新的程序设计方法,其本质是把数据和处理数据的过程抽象为一个具有特定身份和某些属性的自包含实体——对象。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类的重用性。

面向对象程序设计是对问题领域活动的直接模拟,其中的对象往往对应着问题空间中的有形或无形的实体,它使得解题空间与问题空间有自然的对应关系,从而有利于对大型复杂问题给出解决方案,使得程序容易设计、理解和维护。面向对象程序设计的不足之处在

于:对程序的整体功能描述不明显;程序会包含较多的冗余信息,这对小型应用系统有时不合适;程序效率有时不高。

1.2.4 面向对象方法的优点

面向对象系统最突出的特点就是:封装、继承和多态性。

1. 封装

在面向对象的程序设计中,封装是一种数据隐藏技术,它通过把一组数据和与数据有关的操作集合放在一起形成对象来实现。对象通过操作接口与外部发生联系,而内部的具体细节则被隐藏起来,对外不可见。封装机制是结构化程序设计方法难以支持的。

在结构化程序设计过程中,程序的各功能模块和数据之间的关系是由程序员在自己的头脑(或文档)中保持,在程序中这种约束关系没有显式地体现出来。随着将来系统功能的修改或扩充,程序结构可能会变得越来越差,难以维护。

而在面向对象程序设计中,数据及其相关的操作被封装在类或对象中,程序语言直接支持对这种相关性的描述和保护。这样,不仅结构清晰,而且系统各组成部分(对象)之间的独立性好,接口关系简单,交互途径单一——只能通过消息机制这一途径进行通信,这些特点都有利于系统功能的扩充和修改。类的封装机制将数据和代码捆绑在一起,避免了外界的干扰和不确定性。类可以用来创建对象。简单地说,一个对象就是一个封装了数据和操作这些数据的代码的逻辑实体。

2. 继承

面向对象的程序设计中,同样有着继承的机制,通过继承,程序可以在扩展现有类的基础上声明新类——从原有类的基础上派生出来。原有类称为基类,又叫父类,新定义的类称为派生类。在继承基类属性和行为的同时,派生类可以添加自己的数据成员和函数成员。继承把派生类和基类联系起来,派生类对象同时也被认为是基类的对象。利用继承可以方便地重用已经定义的经过测试和调试的高质量的代码,提高软件开发的效率和软件质量。

3. 多态性

多态性是指相同的消息为不同的对象接收到时,可能导致不同的动作。它使得属于同一类的不同对象可以按各自的需要对同一消息做出适当的响应,即为统一管理具有继承关系的不同类的对象提供了方便。使用多态性进行程序设计时,可以为具有继承关系的多个类定义统一的接口,而不同的类对于接口的实现则可各不相同,当通过统一的接口操纵这些对象时,程序可以根据被操纵对象的类型来确定具体该执行什么操作。而在结构化程序设计方法中,函数调用语句就确定了要执行的语句序列,要对不同的对象做不同的操作,就要对这些对象下不同的命令。

由于面向对象程序设计具有上述特点,面向对象方法就可以为软件产品的扩展及质量保证中的许多问题提供解决办法。它能大大提高生产力,并且可以提高软件的质量和降低软件维护费用。下面简单介绍面向对象方法的优点:

(1) 易于建模。允许将问题空间中的对象直接映射到程序中。以数据为中心的设计方法更容易抓住可实现模型的更多细节。

(2) 易于维护。面向对象程序设计的模块性是与生俱来的,其核心是类的设计。数据

隐藏可以保护程序免受外部代码的侵袭，基于对象的工程可以很容易地分割为独立的部分。对象间通信所使用的消息传递技术使得对象与外部系统之间的接口描述更加简单，软件复杂度变得更加容易控制。

(3) 要扩展性好。对象在程序中是一个个相对独立的包含数据和功能的实体，程序员可以向程序中增加一个新类或对象而不会影响到其他类的操作。

(4) 代码重用。继承可以大量减少多余的代码，并扩展现有代码的用途。如果已经有一个具有某种功能的类，则可以很快地扩展这个类，创建另一个具有更多功能的类，而不必一切从头开始，从而减少软件开发时间并提高生产效率。

1.3 程序开发过程

1. 程序设计步骤

程序设计是一门科学，它有规律和步骤可循。程序设计一般遵循以下步骤：

(1) 明确问题

用计算机来解决实际问题，首先要明确解决什么问题，做什么？搞清楚要解决的问题并给出问题的明确定义是解决问题的关键。

(2) 系统设计

明确了问题之后，就要考虑如何解决它。由于计算机解决问题的方式本质上就是对数据进行处理，因此，先对待解决的问题进行抽象，抽取出能够反映出问题本质特征的数据并对其进行描述，即给出数据结构的设计；然后考虑对设计好的数据如何进行操作从而达到解决问题的目的，即进行算法设计。

不同的程序设计在处置数据结构和算法这两者的关系上是有所区别的。在过程式程序设计中，把数据结构设计和算法设计分开考虑，程序由算法构成，数据结构处于附属地位；而面向对象程序设计是把两者结合成对象与类来考虑，程序由对象/类构成。

(3) 用某种语言进行编程

在进行数据结构和算法设计(或对象/类设计)时，往往采用某种与具体程序设计语言无关的语言(伪代码)来进行描述，以避免一开始就陷入某种程序设计语言的一些特殊表示和实现细节中去。过多地涉及实现细节，不利于从较高抽象层次对问题本质的东西进行考虑，会使得对设计过程难以把握和理解。

当然，用伪代码描述的解决方案不能被计算机接受，必须用某种实际的程序语言把它们表示出来，即编程实现。程序员可根据采用的设计方案及实际情况来选择编程语言。如：采用功能分解的设计方案，用某种结构化程序设计语言进行编程比较合适；对于面向对象的设计方案，采用面向对象的程序设计语言来实现就更自然和方便。

选定语言后，就开始编程实现了。但是对于同一个设计，不同的人会写出不同风格的程序。风格有好坏之分，它将影响到程序的正确性和易维护性。程序设计风格取决于编程人员对程序设计的基本思想、技术以及语言精髓掌握的程度。良好的程序设计风格可以通过学习和训练来获得。

(4) 测试与调试

程序写好之后,可能会含有错误。程序错误通常有三种:语法错误、逻辑错误和运行异常错误。语法错误是指程序没有按照语言的语法规则来书写,这类错误可由编译程序来发现。逻辑错误是指程序没有完成预期的功能。运行异常错误是指程序对程序运行环境的非正常情况考虑不足而导致的程序异常终止。后两类错误可能是编程阶段导致的,也有可能是设计阶段或问题定义阶段的缺陷。

程序的逻辑错误和运行异常错误一般可以通过测试来发现。测试方法有静态测试、动态测试等等。静态测试是不运行程序,通过对程序的静态分析,找出逻辑错误。动态测试是利用一些测试数据,通过运行程序观察程序的运行结果是否与预期的结果相符。值得注意的是,不管采用什么样的测试方法,都只能发现程序有错,而不能证明程序是正确的。

(5) 运行与维护

程序通过测试就可以交付使用了。由于测试程序不能保证程序无错,因此,在使用的过程中可能会不断发现程序的错误。需要对程序的错误进行改正,即维护。此外,对程序的需求的改变可能也需要对程序做修改,这也要维护。程序的维护可分为三类:正确性维护、完善性维护和适应性维护。正确性维护是改正程序中的错误;完善性维护是根据用户的要求使得程序功能更加完善;适应性维护是使程序可以移植到不同的计算平台或环境中。

2. C++程序开发过程

C++语言是一个计算机编程语言,利用它编写的程序并不能直接在计算机上运行,而要经过编辑、编译和链接三步生成可执行文件。

程序设计从开始到完成所需的步骤如下:

第一步　程序员进行计算机登录。

第二步　程序员用喜爱的系统的编辑器编辑源程序。

第三步　程序员把编辑好的源程序存入文件中,该文件叫做源代码文件,文件的扩展名习惯上取为CPP。

第四步　程序员用适当的命令指示计算机编译源代码文件。

第五步　编译器将源程序代码翻译成汇编语言格式,然后传给计算机汇编器。

第六步　汇编器将汇编语言代码翻译成可重定位的目标码,转换为OBJ文件,并将这些目标码传送给链接器。

第七步　链接器将程序引用的所有支持例程连接在一起。这些例程存在于运行库中或早已存在于被程序所引用的预编译程序中,然后链接器产生源代码的最终版本,我们把它叫做可执行代码。该代码被传给执行程序的装载器。

第八步　运行程序。

虽然整个处理过程显得很长,但是这些步骤对于程序员来说大多数是透明的。需要程序员做的只是以下几下步骤:

第一步　产生一个包含源代码的文件。

第二步　输入编译命令,得到编译结果。

第三步　运行程序。

用流程图可以表示如图1.2所示。

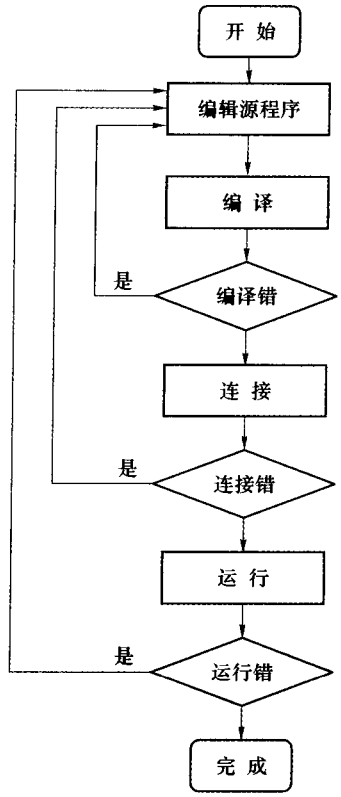


图 1.2　开发 C++ 程序的步骤

习　题　1

1.1　一个好的程序应具备哪些特点?

1.2　什么是面向对象程序设计,它与传统的结构化程序设计有什么不同?

1.3　程序开发的过程是怎样的?