高等学校规划教材

GAODENG XUEXIAO GUIHUA JIAOCAI

C++程序设计

高潮 主编

冶金工业出版社
Metallurgical Industry Press

高等学校规划教材

C++ 程序设计

高　潮　主编

北　京
冶 金 工 业 出 版 社
2011

内 容 提 要

本书分为教程、实验指导和附录三个部分。教程部分包括：概述、C ++语言基础、算法与控制结构、函数及编译预处理、数组与字符串、指针与引用、构造数据类型、类与对象、继承与派生、多态性、输入输出流、C ++的其他几个议题等内容，每章均配有一定量的思考题、选择题、填空题和编程题，供学生课后学习巩固之用。实验指导部分设置了12个与教程相关章节配套的实验项目和一个综合实验项目，以利于编程技能的训练和学习。附录提供了 Visual C ++ 6.0 开发环境及程序调试、C ++的运算符及其优先级、常用库函数和 ASCII 码表等内容。

本书可作为高等院校理工科各专业的程序设计课程教学用书，也可作为各类人员自学程序设计及计算机程序设计的培训教材。

图书在版编目(CIP)数据

C ++ 程序设计 / 高潮主编 . — 北京：冶金工业出版社，2010. 1 (2011. 4 重印)

ISBN 978-7-5024-5128-8

Ⅰ. ①C… Ⅱ. ①高… Ⅲ. ①C 语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2009)第 244137 号

出 版 人 曹胜利

地 址 北京北河沿大街嵩祝院北巷 39 号，邮编 100009

电 话 (010)64027926 电子信箱 postmaster@ cnmip. com. cn

责任编辑 程志宏 廖 丹 美术编辑 李 新 版式设计 葛新霞

责任校对 石 静 责任印制 李玉山

ISBN 978-7-5024-5128-8

北京兴华印刷厂印刷；冶金工业出版社发行；各地新华书店经销

2010 年 1 月第 1 版，2011 年 4 月第 2 次印刷

787 mm × 1092 mm 1/16；20. 5 印张；548 千字；315 页

40. 00 元

冶金工业出版社发行部 电话：(010)64044283 传真：(010)64027893

冶金书店 地址：北京东四西大街 46 号(100010) 电话：(010)65289081(兼传真)

前　言

本书是作为高等院校理工科各专业的第一门程序设计课程的教材来组织和规划的。它把调动学生的学习兴趣、培养学生实际应用能力放在首位，引导学生真正进入程序设计的门槛。它以C++语言为编程工具，介绍程序设计的基本概念和基本方法。C++是从C语言发展演变而来的一种程序设计语言，它的主要特点表现在两个方面，一是全面兼容C，二是支持面向对象的程序设计技术。C++语言是目前最有活力和应用最广泛的程序设计语言之一。

本书分为教程、实验指导和附录三个部分。教程部分包括：第1章概述、第2章C++语言基础、第3章算法与控制结构、第4章函数及编译预处理、第5章数组与字符串、第6章指针与引用、第7章构造数据类型、第8章类与对象、第9章继承与派生、第10章多态性、第11章输入输出流、第12章C++的其他几个议题。教程部分每章均配有一定量的思考题、选择题、填空题和编程题，供学生课后学习巩固之用。实验指导部分设置了12个与教程相关章节配套的实验项目和一个综合实验项目，以利于编程技能的训练、学习和提高。附录提供了Visual C++ 6.0开发环境及程序调试、C++的运算符及其优先级、常用库函数和ASCII码表等内容，这些都是学习C++语言程序设计的重要支撑。

本书具有如下一些特点：

（1）对教学学时的适应性强，对学生知识的起点要求低。在教学中，通过对教材内容的合理组织与取舍以及结合网络教学平台等手段，在保障基本学时和教学内容完整、充实的基础上，本书提供了对教学内容作进一步深入和扩展的灵活性。

（2）淡化面向过程与面向对象两种程序设计方法的分界，注重算法的结构化思想，强调算法在程序设计中的基础性地位，对面向过程程序设计方法仅在第1章中进行说明。这样的安排基于如下的认识：

面向过程和面向对象体现的是两种不同的系统分析方法和系统架构的建立技术，但都是以算法和数据结构的设计为基础的。从面向对象的视角来看，过程是局部的算法问题，对象才是系统架构建立的基础。面向对象程序设计方法是当前程序设计技术的主流，而面向过程程序设计方法则是程序设计方法发展中

的一个历史范畴。作为程序设计的初学者,只需要了解而不应该重复程序设计技术的历史,应该在先进的程序设计思想指导下,首先学习和掌握好算法的结构化、程序的模块化等基础性技术。

(3) 注重解题思路、算法实现和程序设计思想,而不拘泥于语言、不纠缠于细节,尽量避免琐碎的描述和概念的堆砌,在保证概念准确的前提下力求做到语言通俗易懂。

(4) 尽早引入函数概念,以建立程序模块思想,实现对功能(或过程)的封装,并注重建立函数的思考问题的方法,而不仅仅是介绍语法规则;注重在后续章节中以函数(功能模块)来组织程序代码,巩固算法的结构化思想及程序模块思想。

(5) 实验指导与教程内容有机结合。学习程序设计,会编程才是硬道理,因此在教学中必须加强实验指导,提高程序设计学习的实效性。

本书由高潮主编,参与本书编写工作的老师还有:吴明芬(第3章)、李志仁(第4章)、彭腊梅(第6章)、高宏宾(第7章)、郑晓曦(第8章)、白明(第9章和第10章)。五邑大学信息学院计算机系的有关老师对本书的编写提出了许多有益的意见和建议,编者在此一并表示感谢。

读者若对与本书配套使用的PPT电子教案、网络教学及测试平台等有需要,请与编者联系(gordon911@126.com)。

由于水平有限,书中的不妥之处恭请读者批评指正。

编 者

2009年10月

目　录

第1章 概 述

本章首先通过一个现实的例子来说明有关程序的几个概念。然后通过对软件危机的讨论来说明程序与软件的关系及软件开发过程。接着进一步结合现实的例子,介绍面向过程与面向对象的软件开发方法。最后简要介绍 C/C ++ 语言的发展及 C ++ 程序的开发过程。

1.1 程序、算法、数据结构及程序设计语言

我们先看一个现实的例子——“厨师做菜”。

厨师在开始做菜前,他要知道这道菜需要哪些原料,各种原料如何搭配;他要知道如何在恰当的时机对相应的原料采取恰当的方法进行加工和处理。例如该加油炒菜的时候就不能加水,否则就不是炒菜而是煮菜了。厨师所要知道的这些内容,其实就是一道菜的菜谱。厨师按照菜谱操作,最后做出一道可口的菜肴。

以上所谓的“原料如何搭配”说明的是操作的对象;“恰当的时机、恰当的方法”,也就是事先已经确定好的做菜的具体步骤和方法,即做菜的程序。这里的“程序”是人们普遍使用的一个朴素的概念,但也说明了这种思考问题方法的核心是“过程”。

运用计算机来解决一个实际问题,与“厨师做菜”的道理是一样的。首先需要把问题处理的对象搞清楚,处理的具体步骤和方法事先要设计好,然后再利用计算机能够理解的语言写出具体的操作步骤,这就是计算机程序。计算机按照程序运行就可以自动执行程序指令,解决相应的问题,完成相应的任务。

从以上问题的描述,我们可以明确这样几个概念:程序、算法、数据结构和程序设计语言。

(1) 程序(Programs) 程序就是要计算机完成某项工作的代名词,是对计算机完成某项工作所涉及的对象和动作规则的描述。

(2) 算法(Algorithms) 算法体现在对动作规则的描述,它描述了解决一个问题所采取的方法和步骤。

(3) 数据结构(Data Structure) 数据结构体现在对对象(或数据)的描述,它描述了问题所涉及的对象以及对象之间的联系和组织结构。数值计算问题可以用数学方程来描述,但更多的非数值计算问题无法用数学方程加以描述,而是通过表、树和图之类的数据结构来建立数学模型。

1976 年,著名计算机科学家沃思(Nikiklaus Wirth)出版了一本题名为《Algorithms + Data Structure = Programs》的著作,明确提出算法和数据结构是程序的两个要素:

$$\text{程序} = \text{算法} + \text{数据结构}$$

也就是说程序设计主要包括两方面的内容:行为特性的设计和结构特性的设计。行为特性的设计是指完整地描述问题求解的全过程,并精确地定义每个解题步骤,这一过程即是算法的设计;而结构特性的设计是指在问题求解的过程中,计算机所处理的数据及数据之间联系的表示方法。程序设计的关键是构造程序的数据结构,并描述问题所需要的、施加在这些数据结构上的算法。

(4) 程序设计语言 程序最终需要使用计算机能够理解的语言具体写出来,这就是计算机程序设计语言。所谓“语言”就是一套具有语法、词法规则的系统。语言是思维的工具,思维是

通过语言来表述的。计算机程序设计语言(Computer Programming Language)是计算机可以识别的语言,是程序实现的工具。任何一门计算机程序设计语言都需要通过数据类型、运算符与表达式以及控制语句等来定义和实现程序中的数据结构和算法。

1.2　程序与软件及软件开发过程

在计算机应用的初期,人们狭义地认为软件就是程序,软件设计就是程序设计,软件是程序员个人劳动的成果。然而在20世纪60年代以后,随着计算机系统性能的提高,它的应用范围也越来越广泛,计算机软件系统的开发也变得越来越复杂,在开发大型软件的过程中出现了所谓的"软件危机",其主要表现是:开发进度被推迟,成本超出预算,软件产品的可靠性降低等。人们认识到软件开发要比想象的复杂得多。

为什么会产生软件危机呢?其主要原因有三个:

(1) 软件开发者与用户之间对于问题的理解不一致。一方面开发者不熟悉用户问题的领域或没有理解用户的需求,导致设计的软件产品与用户要求不一致;另一方面,用户也难以提出准确、完整、无二义性的软件需求描述。

(2) 软件开发过程更多地体现为设计人员个人的思考过程,这种思考过程没有完整地表现在书面上,因此无法对思考过程进行科学规范及质量管理,软件开发进度也无法控制。

(3) 人的智力在面对越来越复杂的问题时,处理问题的效率会越来越低。因此,在没有找到控制问题复杂性的有效方法时,开发软件所需的时间和费用将随问题复杂性的增加而急剧增加。

经历了"软件危机"之后,人们对软件的概念和软件开发过程有了新的认识。即:软件不仅仅是程序,软件开发也不仅仅就是程序设计。计算机软件是计算机程序和与之相关的文档资料的总和,文档资料包括编制程序所使用的技术资料和使用该程序的说明性资料(使用说明书等),即开发、使用和维护程序所需的一切资料,如图1-1所示。计算机软件开发则是一个包括程序设计在内的可以管理、控制和质量评价的规范的工程。"软件工程"的概念由此而生。

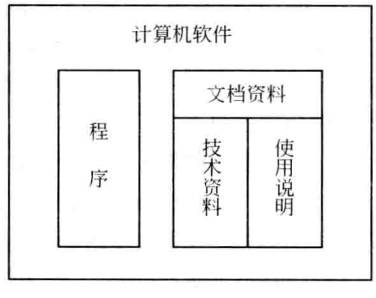

图1-1　计算机软件概念示意图

简单地说,一个大型软件系统的开发,必须经历这样一个过程:

需求分析→系统设计→程序编码以及编辑、编译和连接→系统测试→运行维护

在软件开发过程的每一个阶段,都要有明确的任务和要求,必须产生一定规格的文档资料。

1.3　面向过程与面向对象的软件开发方法

软件开发过程中最关键的是需求分析和系统设计,然后再通过具体程序编码去实现。而系统设计中的一个关键内容是选择合适的软件开发方法。在如何克服"软件危机"的讨论和研究中,诞生了"结构化程序设计"的概念。结构化程序设计是一个面向过程的软件开发方法,它的产生和发展形成了现代软件工程的基础,但是在大型软件开发过程中,面向过程的结构化程序设计方法暴露出了它的不足。20世纪80年代以后,面向对象的软件开发方法得到了重视和快速发展,并已成为当前软件开发方法的主流。

下面分别简要介绍面向过程的结构化程序设计方法和面向对象的程序设计方法以及它们的联系。

1.3.1　面向过程的结构化程序设计

结构化程序设计的基本思想是：自顶向下，逐步求精，将一个复杂的问题分解为若干个易于处理的子问题，从而将整个程序划分成若干个功能相对独立的子模块或过程。子模块又可继续划分，直至最简。每个模块都只有一个入口和一个出口，整个程序则全部可以用顺序、选择、循环这三种基本结构及其组合来实现了。这样的结构化程序设计以解决问题的过程作为程序的基础和重点，是一种面向过程的程序设计方法。

在面向过程的结构化程序中，程序过程是模块化的，模块是分层次的，层与层之间是一种从上往下的调用关系，如图 1-2 所示。面向过程程序的基本构成单位是过程（或者函数）。

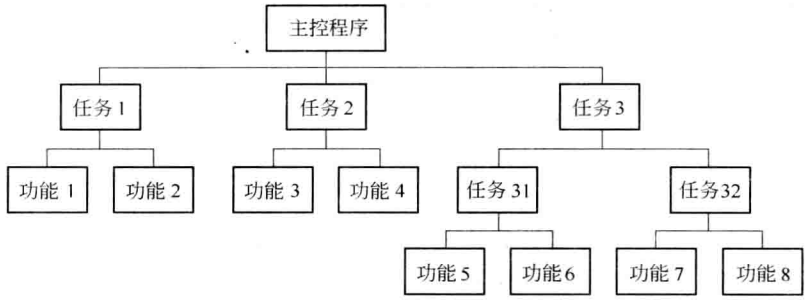

图 1-2　面向过程的程序结构

让我们进一步看看一个餐馆的整个运营过程——“餐馆运营系统”：顾客到餐馆吃饭，服务员负责接待顾客。顾客就座后，点好要吃的饭菜，服务员就去告诉厨师要做哪些菜。厨师做好以后，服务员端上饭菜，顾客就可以品尝这美味佳肴了。最后顾客吃完饭付清账款。

我们可以把这个系统按功能抽象分解为点菜的过程、做菜的过程、品尝菜肴的过程、收付账款的过程，如图 1-3 所示。这样来组织系统的方式是以过程为核心，是面向过程的。

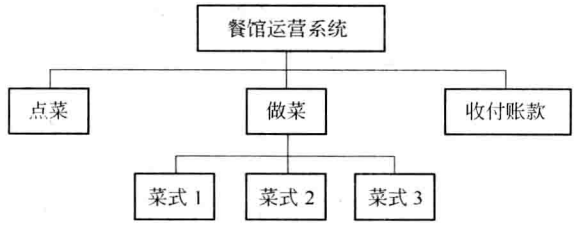

图 1-3　餐馆运营系统（面向过程）示意图

面向过程的结构化程序设计方法有许多优点：

（1）各模块可以分别编写，使得程序更易于阅读、理解、测试和修改。这样不管编多大的程序，不管有多少人参加编写，都可以把他们的模块有效地连接起来。

（2）方便增加新的功能模块。

（3）功能独立的模块可以组成子程序库，有利于实现软件的复用。

但另一方面，由于结构化程序设计方法强调的是模块的功能，是面向过程的，所以数据与处理这些数据的过程是分离的。这样的系统设计方法会带来如下一些问题：

（1）对不同格式的数据作相同的处理，或是对相同的数据作不同的处理，都需要编写不同的程序模块来实现，这使得程序的可复用性大打折扣。

（2）由于过程和数据相分离，数据可能被多个模块所使用和修改，这样很难保证数据的安全性和一致性。

（3）当数据处理的方法或是数据类型有改变时，会导致整个系统的重新设计和编码。

由于上述问题，面向过程的结构化程序设计方法难以适应大型软件的设计开发。

1.3.2　面向对象的程序设计

前面所述的“餐馆运营系统”的分析设计是按照面向过程的方法，把系统按功能分解为点菜的过程、做菜的过程、品尝菜肴的过程、收付账款的过程等。但这种分析设计方法与人们的正常思维习惯并不吻合。

让我们再回头读一下那段对餐馆整个运营过程的描述：“顾客到餐馆吃饭，服务员负责接待顾客。顾客就座后……”。整个系统是以“对象—消息”的方式来描述，如图 1-4 所示。显然这样的描述直接反映了人们的正常思维方式，它首先描述的是在餐馆中的“顾客—服务员—厨师”这些对象之间的联系，然后才是每个对象所具备的属性和能力。因为顾客只要点好要吃的饭菜，而并不需要告诉厨师烧菜的具体步骤，甚至也不知道烧菜的具体步骤；同样地，厨师一般也不需要知道顾客品尝菜肴的过程、服务员收付账款的过程，他只需要按顾客所点的菜做出美味佳肴即可。日常生活中这样的事例还有很多。可以看出，人类成熟而习惯的解决问题的高效率方式就是这种“对象—服务（消息）”的思考问题的方式。这种分析问题方式的核心是对象以及对象之间的联系，是面向对象的。

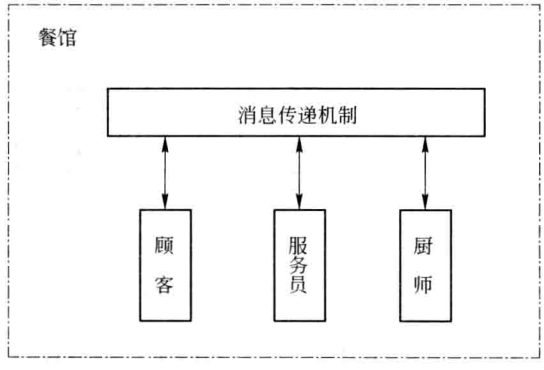

图 1-4　餐馆运营系统（面向对象）示意图

客观世界中任何一个事物都可以看成是一个对象。对象可以是一个物理的实体，如一辆汽车、一间房屋、一只老虎；也可以是一个逻辑的实体，如一个班级、一个连队；甚至一篇文章、一个图形、一项计划等都可视作对象。

可以看到，一个班级作为一个对象时有两个要素：一是班级的静态特征，如班级所属系和专业、学生人数等，这种静态特征称为属性；二是班级的动态特征，如学习、开会、体育比赛等，这种动态特征称为行为。如果想从外部控制班级学生的活动，可以从外界向班级发送一个信息，如听到广播声就去上早操、听到下课铃声就下课等，一般称它为消息。

在面向对象的程序中，作为计算机模拟真实世界的抽象——对象（Object）是一个属性（数据）集及其行为（操作）的封装体。对象与对象之间的联系是通过消息传递的机制来实现的。消息（Message）是对象之间相互请求或相互协作的途径，是要求某个对象执行其中某个功能操作的规范的说明。对象可以向其他对象发送消息以请求服务，也可以响应其他对象传来的消息，完成自身固有的某些操作，从而服务于其他对象。

对象对消息的响应，就是对象调用自身的方法（成员方法或成员函数）去完成相应的任务。方法（Method）描述了对象的行为能力，从程序设计的角度看，它是对象实现功能操作的代码段。方法与消息相互对应，每当对象收到一个消息后，除了知道“做什么”外，还必须知道和决定“怎样做”。方法就是对象中决定“怎样做”的操作代码，是实现每条消息具体功能的手段，它是一段过程代码。

简单地说,面向对象程序的程序结构可以表示如下:

对象 = 算法 + 数据结构

程序 = 对象 + 消息

这里程序是由一个个封装的对象组成,而对象是由紧密联系的算法和数据结构组成,它封装了数据和对数据的操作。对象与对象之间通过消息传递的机制来协调各个对象的运行。如图 1-5 所示。

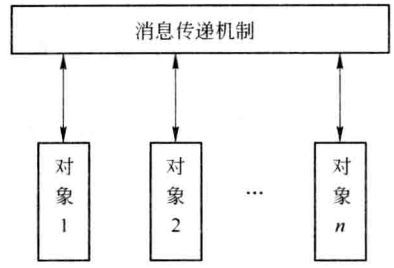

图 1-5 面向对象程序的程序结构

从面向对象的观点来看,现实世界就是由各式各样独立的、异步的、并发的实体对象所组成。每个对象都有各自的内部状态和运动规律。不同对象之间或某类对象之间的相互联系和作用,就构成了各种不同的系统。

在面向对象系统中,对象是构成软件系统的基本单元。但在设计中,人们并不是逐个描述各个具体的对象,而是将注意力集中于具有相同特性的一类对象,通过抽象找出同一类对象的共同属性和行为,形成类。即类(class)是对象的抽象及描述,是具有共同属性和行为的多个对象的相似特性的统一描述体。结合类的继承与多态等性质,可以很方便地实现代码重用,大幅度地提高程序的复用能力和程序开发效率。

1.3.3 面向对象的基本特性及与面向过程的关系

1.3.3.1 面向对象方法的基本特性

面向对象方法具有四个基本特性:抽象性、封装性、继承性和多态性。这几大特性与人类业已形成的对错综复杂的大千世界的认识方法高度吻合,使计算机世界与现实世界更加接近。

(1) 抽象性 所谓抽象,就是从被研究对象中舍弃个别的、非本质的或与研究主旨无关的次要特征,抽取共同性质、实质特征以后形成概念的过程。例如,“马”就是一个抽象的概念,实际上没有任何两匹马是完全相同的,但是我们舍弃了每匹马个体之间的差异,并且通过分类、抽象,在各种动物中抽取出了一种具有共同的本质特征的“马”的动物,形成了“马”这个概念。面向对象方法就是通过类的机制实现其抽象性:类是具体对象的抽象,对象是类的一个实例。

(2) 封装性 封装也称为信息隐藏,是指将对象的内部属性和行为实现的代码封装在对象的内部,从外面无法看见对象的内部信息,更不能随意进行修改。同时,对象向外界提供少量必要的访问接口,外界只能通过对象的接口来访问该对象。这样就将对象的外部特征与内部实现细节清楚地分隔开,从而既有效地避免了外部错误对它的“感染”,又大大减少了内部的修改对外部的影响,有助于最大限度地减少由于需求的改变而对整个系统所造成的影响。

抽象和封装是互补的。面向对象程序设计的主体是类,类是具有相同属性和行为的一类对象的封装体,是对具有相同属性和行为的对象的抽象描述。好的抽象有利于封装,封装的实体则帮助维护抽象的完整性。封装性是软件设计模块化、软件复用和软件维护的一个基础。

(3) 继承性 继承性是面向对象方法中另外一个重要特性,它体现了现实世界中对象之间的独特关系。既然类是对具体对象的抽象,那么就可以有不同级别的抽象,就会形成类的层次关系。就像在现实世界中,我们要描述猫、狗、狼、虎,由于猫、狗、狼、虎都属于哺乳类动物,所以我们可以先定义哺乳类动物。我们把哺乳类动物所共有的特征和习性定义在哺乳类动物中,然后再分别定义猫、狗、狼、虎类动物各自特有的特征和习性。这种由一般到特殊的分类方法是人们总结出的高效率地分类认识世界的方法。同样地,在面向对象方法中,也是用一个类描述一组对象的共性信息,用另一个类描述一组对象的特性信息。前一个类称为父类,后一个类称为子类,