★高等院校21世纪新视野教材★

C++语言及面向对象程序设计

彭江平　黄万晨　主编

湖南大学出版社

C++语言及面向对象程序设计

主　编　彭江平　黄万艮

副主编　蒋炎焱　蒋本立　彭孟良　周光宇　罗庆云

编　委　高守平　张新林　易德成　刘震宇　莫　照

李正华　陈国平　黄天强　羊四清　陈罗湘

罗庆云　彭孟良　蒋本立　蒋炎焱　黄万艮

彭江平　段益群　周光宇

湖南大学出版社

2004年·长沙

图书在版编目(CIP)数据

C++语言及面向对象程序设计/彭江平,黄万艮主编. —长沙:湖南大学出版社,2003.9

ISBN 7-81053-722-9

Ⅰ.C... Ⅱ.①彭...②黄... 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(2003)第 080756 号

C++语言及面向对象程序设计

C++ Yuyan ji Mianxiang Duixiang Chengxu Sheji

彭江平 黄万艮 主编

□ 责任编辑 胡建华
□ 特约编辑 何 晋
□ 封面设计 张 毅
□ 出版发行 湖南大学出版社
社址 长沙市岳麓山 邮码 410082
电话 0731-8821691 0731-8821594
□ 经 销 湖南省新华书店
□ 印 装 长沙鸿发印务实业有限公司

□ 开本 787×1092 16 开 □ 印张 24.5 □ 字数 567 千
□ 版次 2004 年 2 月第 1 版 □ 2004 年 2 月第 1 次印刷
□ 印数 1-5 000 册
□ 书号 ISBN 7-81053-722-9/TP·39
□ 定价 36.00 元

前　言

《C++语言及面向对象程序设计》一书共分上、下两编。上编主要介绍C++程序设计语言、程序结构及过程化的程序设计。在这一部分不仅全面地讲述C++语言的主要内容，而且同时以比较的形式较系统地介绍C语言的基本内容。下编是面向对象程序设计，它建立在C++程序设计基础之上，讲述面向对象的程序设计方法。这一部分对封装、继承、多态、重载与面向对象的错误处理等面向对象程序设计的方法进行了全面的论述，并且对基于统一建模语言(UML)及其在面向对象方法中的应用进行了简要的介绍，为面向对象的系统分析与系统设计作了必要的准备，全面展示面向对象软件工程的概貌。

本书与同类教材比较，有如下特色：

(1) 课程组织模式的创新。C++语言是从C语言进化而来的，是C语言的超集，在程序结构的本质上是与C一致的，过程化的程序设计与面向对象的程序设计之间并无水火不容的矛盾。因此本书作者在多年从事C或C++程序设计语言、面向对象程序设计的教学模式及教学经验总结的基础上，将C++程序设计语言与面向对象程序设计两门课程有机地结合起来：即在C++程序设计语言中对C语言进行简要的介绍，并以C++语言作为过程化程序设计语言的基础，而将面向对象程序设计课程重点放在面向对象程序设计的方法论上。传统的教学模式一般是先讲解C语言程序设计，然后再使用C++或Java等作为面向对象程序设计，因为在面向对象课程中又要讲解C++语言或Java语言，结果使后者也变成一门程序设计课程，很难将面向对象程序设计方法阐述清楚。

(2) 教学模式的创新。随着计算机技术与网络技术的发展，信息管理与计算机专业的招生规模与形式都发生了很大的变化，传统的教学模式已很难满足新形势的需要。因此，本书作者在多媒体教学与网络远程教学的基础上，除整理纸介质的教材外，还配套相应的多媒体教案与课件。现有的教材一般都只有纸介质的课本，或最多是纸介质的电子版本，不能满足多媒体与远程教学的需要。

(3) 从为什么到如何做。本书作者依据多年C或C++程序设计语言、面向对象程序设计的教学实践发现，对讲清楚“为什么”比讲“如何做”更重要，特别是在面向对象的程序设计方法课程学习中更是如此。如果仅讲解“如何做”问题，那么学生虽然明白了当前问题的解决方法，但只有少数善于思考的人才能在碰到具体问题时选择适当的解决方法。因此，在本教材中的许多章节都有“为什么”这一节，这是本书的第三个重要特色。

(4) 突出重点、详解难点。本书作者以多年教学经验为基础，将教学重点内容作了反复的讲解；依据教学中学生提出的难点作了详细的讲解，并列举了大量的实例，并且依据在教学实践中学生们经常可能出现的问题及重点难点问题设计了大量的习题。

(5) 语言简明、概念准确、例题丰富。以通俗的语言讲述了C++语言的基础知识与编程方法，并以大量的例题验证所讲解的内容，读者可以模仿例题并在把握好“为什么”的基础上，去解决形式相仿的问题，据此学会某类而不是某个问题的解决方法。

(6) 练习丰富。本书每章都附有大量的习题，习题内容全面、形式多样。有问答题、填空题、选择题、判断题、分析程序运行结果、编程、模拟试题等，并在电子版中配套有相应的习题解答。通过这些习题，学生可以检验与巩固对学习知识的运用。

本书的每一篇内容都可安排 32 学时的课堂教学与 32 学时的实验上机，每学时 45 分钟。由于 C++语言将逐步代替 C 语言，因此本书的第一部分可以作为 C 语言程序设计课程的替代，使学习者在学习 C++语言的同时兼顾 C 语言的学习。与本书配套的习题解答与 PowerPoint 教案也即将出版。

本书可作为大学本科或专科院校学生的教科书，也可作为教师与学生的参考书，还可作为广大计算机爱好者自学 C++语言及面向对象的程序设计的指导书。

在本书的编写过程中，查阅了有关于 C++语言的中外文资料及一些相关的参考书，谨向这些文献的作者表示感谢。由于时间匆促，在本书编写过程中，难免有错误之处，请读者多提出批评和指正。

感谢所有选择本书和对本书提出批评与指正的读者。

编　者

2003 年 6 月

湖南大学岳麓山

目　　次

上编:C++语言

第 7 章　数组

第 8 章　指针

第 9 章　引用

第 10 章　结构与联合

下编:面向对象程序设计

第 11 章 类

第 12 章 构造函数与析构函数

第 13 章 堆与拷贝构造函数

上编:C++语言

第1章　C++程序设计入门

计算机程序设计技术经过几十年的发展,发生了很大变化。首先,程序语言由机器语言、汇编语言、面向过程的高级语言发展到了面向对象的高级语言;其次,程序设计方法由结构化程序设计发展到了面向对象程序设计;第三,程序开发工具由最初的解释程序、编译程序发展到了集成化及可视化编译环境……C语言是一种支持结构化设计面向过程的高级语言;C++语言是一种支持面向对象程序设计的高级语言,是C语言的发展,目前被广泛应用于计算机科学各领域。学习本章要求掌握程序设计方法的发展、结构化程序设计与面向对象程序设计思想的特点、C语言与C++语言的区别联系、C++语言的特点、C++程序的基本结构以及简单C++程序的编写与运行。

1.1　软件、程序、语言、开发工具

软件、程序、语言和开发工具是人们学习软件编程首先需要理解的几个概念。一般来说,软件和程序使用某种计算机语言利用某种开发工具开发完成。

1.1.1　软件与程序

计算机系统由硬件系统和软件系统组成。硬件是躯体,软件是灵魂。硬件必须在软件的控制下才能运行,完成相应的功能。正是因为人们开发了各种各样的软件,计算机才得以广泛应用于社会生活的各个方面,对社会产生革命性的影响。

程序与软件是一对联系密切的概念,很多时候往往被等同,即认为软件就是程序,但严格地说,二者是有区别的。程序是指用计算机程序设计语言按严格的规范要求写出来的语句序列。程序一般用某种语言编写而成,通常以文件的形式保存,这些文件通常包括源文件、源程序和源代码几种形式。人们利用程序与计算机进行交流,控制计算机完成某种操作。

软件比程序的范围更广泛,它是指能让计算机完成某种功能的程序、数据及文档等的综合体,不仅仅包括程序,还包括数据与文档等等。程序是软件最重要的组成部分,没有一定的程序,软件将无法运行。

随着IT技术和社会生产的发展,软件需求急剧增长,各行各业都要求开发相应的软件以满足生产管理的需要,软件的数量大大增加。与硬件成本快速下降相反,软件费用却急剧上升,软件的稳定性、可维护性、可扩充性已经成为一个很严重的问题。

什么是程序的可读性、易维护、可移植？所谓可读性是指使用良好的书写风格和易懂的语句编写程序。所谓易维护是指当业务需求发生改变时，比较容易扩展和增强程序的功能。所谓可移植是指程序能在各种计算机和操作系统上正确运行。在程序正确的前提下，软件的质量、可读性、易维护性、可移植性对软件开发已提出了严重挑战。人们迫切要求以更快的速度和更低的成本开发出具有很好的稳定性、可维护性、可扩充性的功能强质量高的软件，因此，人们需要不断对软件设计思想和方法进行改进，以降低软件开发成本，加快开发速度，提高程序的可读性、易维护、可移植。这已成为当前软件开发非常关注的问题。为解决这一问题，一种有效的方法是对计算机语言进行改进，采用更高级的语言和更先进的编程思想来开发软件，例如选用C++语言便是一种很好的途径。

1.1.2 语言

程序是用计算机语言编写而成，为了有效提高编程效率，人们先后使用了不同的计算机语言来编写程序。这些语言分为机器语言、汇编语言、高级语言，其中高级语言又分为面向过程的高级语言（如C语言）和面向对象的高级语言（如C++语言），计算机语言的发展有效提高了程序开发的效率。

1. 面向机器语言（低级语言）——机器语言和汇编语言

计算机只能识别“0”与“1”两个代码。最初，人们使用由“0”与“1”组成的代码串编写程序，这样的语言叫机器语言。机器语言是机器惟一能识别的语言，所有的程序最终都必须转化为机器语言才能在计算机上运行。因为由“0”与“1”组成的代码串很难理解和记忆，因此，这样编写的程序很难懂，编程是一种专业性很强的工作，只有专业人员才能编写。

为了减少编程的难度，后来人们将一条条的机器指令映射为一个个能被人读懂的英文助记符，如ADD、SUB等，这样，程序员用助记符来编写源程序，程序编完后再用汇编程序将源程序转化为机器指令程序，再在机器上执行。由这些助记符所构成的计算机语言就叫做汇编语言。

汇编语言的出现大大减轻了编程的难度，提高了编程的效率。因为汇编语言和机器语言一样，直接面向机器硬件，能深入控制计算机系统最底层的硬件，因此，早期计算机系统中需要直接控制硬件的底层程序都使用汇编语言编写而成（如基本输入输出系统BIOS和操作系统DOS）。

下面是一段机器语言和汇编语言写成的程序：

```
00401156   mov [ebp－0x04],0x00000009
0040115D   mov [ebp－0x08],0x00000006
00401164   mov eax,[ebp－0x04]
00401156   sub eax,[ebp－0x08]
00401156   mov [ebp－0x0c],eax
```

这段程序的功能是计算9减6，很显然，程序仍很复杂，不好理解。

2. 面向过程（数据加工过程）的高级语言

虽然汇编语言减轻了编程的难度，但汇编语言仍直接面向机器硬件，只有深入了解计算机的硬件构成和组成原理才能编写程序，同时汇编语言不支持函数和过程调用，不支持结构化程序设计，数据类型单一，程序可移植性仍受很大限制，因此仍存在严重的编程效率问题。当人们需要计算机完成更复杂的操作时，往往需要编写成千上万条指令，用汇编语言来实现很难。

为了更进一步提高编程效率，后来人们又发明了高级语言来编写程序。开始时，这些高级语言只支持面向过程的结构化编程，即通过功能强大的高级语句和将一些经常要用到的特定功能编写成一段可供人们调用的子程序（函数或过程），面向具体的过程来解决实际应用问题。这有效地减轻了编程的难度，减少了代码行的数量，从而使得程序结构清晰，代码可读性好。因此，用高级语言编程比用汇编语言更容易、更快捷。

面向过程的高级语言种类很多，先后有几十种甚至上百种，如 Basic、Fortran、Pascal、C 等等。

3. 面向对象的高级语言

面向过程高级语言通过采用结构化编程方法对提高程序的质量、减轻编程的难度起到了较好的作用。正是因为面向过程的高级语言的出现，才使得普通人员也能使用某种计算机语言编写出实用的计算机程序。但面向过程的高级语言编写的程序必须按严格的处理过程才能正确执行，编程时必须针对一个个的处理过程，数据与对数据的处理过程分开，对处理对象的抽象程度很低，与人们认识世界的思维方法存在较大的距离，因此在解决问题时仍不直观。同时，仍缺少对代码可重用性的有效支持，无法有效地提高代码的可重用率，仍存在编程效率问题，因此，人们需要更高级的计算机语言的支持。这种语言就是现在的面向对象语言，如 C++语言和 Java 语言。

面向对象语言提供定义类与对象的机制。类包括了属性及处理过程，由类可生成子类（派生类），子类继承了父类的属性与处理过程。对象由类生成，是类的实例，具有类的属性和方法。这种编程方法，与现实世界的状态更加接近，有效地提高了代码的可重用性。

4. 高级语言编程与低级语言编程的比较

机器语言是低级语言，汇编语言基本上是低级语言，C 语言是一种支持结构化程序设计面向过程的高级语言，而 C++语言是一种面向对象的高级语言。下面是用高级语言编程与用低级语言编程的比较。

例如：对于 C 或 C++语言的语句：

```
int a=9,b=6,c;    //定义 a,b,c 三个 int 型变量，并令 a=9,b=6
c=a-b;            //表示将变量 a 减变量 b 的值送给变量 c
```

这两条语句的功能如用汇编语言和机器语言来完成则复杂很多，如下所示。

```
00401156   mov [ebp-0x04],0x00000009
0040115D   mov [ebp-0x08],0x00000006
00401164   mov eax,[ebp-0x04]
00401156   sub eax,[ebp-0x08]
00401156   mov [ebp-0x0c],eax
```

程序说明：

第一条命令表示将十六进制数 0x00000009 送入内存的[ebp-0x04]单元（ebp 表示数据段的指针，0x04 是变量的偏移位置）。

第二条命令表示将十六进制数 0x00000006 送入内存的[ebp-0x08]单元（ebp 表示数据段的指针，0x08 是变量的偏移位置）。

第三条命令表示将[ebp-0x04]单元的值放入寄存器 eax 中。

第四条命令表示将 eax 的值减去[ebp-0x08]单元中的值，并存入 eax 中。

第五条命令表示将 eax 中的结果送入[ebp-0x0c]单元中。

由以上比较可看出：语言越低级，编写程序越复杂，指令越难懂；语言越低级，就越靠近机器；语言越高级，就越靠近人的表达与理解。程序语言的发展，就是从低级到高级，从具体到抽象，直到可以用人的自然语言来描述。低级语言必须详尽地描述任何操作，而高级语言抽象表达能力强，只需去完成指定的任务。

5. 可视化、智能化语言

随着计算机技术的发展，面向对象的高级语言仍在继续向前发展。更高级的计算机语言将是一种智能化语言，人们在编写程序时只需简单地拖动鼠标或向计算机发出其他指令，便可自动地生成代码。这种思想在可视化的开发工具（如 VB、PB、C++Builder 等）中已经体现出来。展望未来，计算机语言还将会有更大的发展。

1.1.3 开发工具与应用系统

用计算机语言开发的能满足各种应用需求的软件，称之为应用系统，如人事软件、财务软件等。应用软件必须基于操作系统的基础之上，借助操作系统的支持才能工作。因为应用领域千变万化，为了满足不同领域的需求，人们需要开发各种各样的应用软件。

应用软件一般由高级语言开发而成。不同时期人们往往使用不同的语言和不同的开发工具，以 C 语言为例，就有 Turbo C、Turbo C++、Bloand C++以及目前广泛使用的VC++和C++Builder 等多种开发工具。每个工具又还有不同的版本。此外，还有基于其他高级语言的 Visual Basic(VB)、PowerBuilder(PB)、Java Builder(JB)等等开发工具。现在人们一般使用高级的开发语言和开发工具，来更高效地开发各种应用系统。

1.2 结构化程序设计与面向对象程序设计

程序设计方法经历了面向过程到面向对象程序设计的发展过程。结构化程序设计较好地解决了面向过程进行程序设计的问题，而面向对象程序设计更接近现实世界的本来特征。

C 语言能很好地支持结构化程序设计，而 C++语言能很好地支持面向对象程序设计。在面向对象的程序设计方法中也包含了结构化的程序设计方法。

1.2.1 结构化程序设计

结构化程序设计是产生于 20 世纪 60 年代并在 70～80 年代逐步成熟起来的一种程序设计方法，经过几十年的发展，广为人们接受使用。现在，绝大部分软件都保留了结构化特征。

1. 结构化程序设计的思想与方法

在结构化程序设计方法中，人们采用模块化的思想，将一个大型系统分成多个子模块，每个子模块完成特定的功能，最后再将这些模块组装成一个完整的软件系统。这就像小孩玩积木游戏一样，通过不同功能的积木搭建出不同的物体。如图 1.1 所示。

在结构化程序设计方法中，程序是面向过程的，即把程序看成是处理数据的一系列过程，其主要思想是将程序功能分解并逐步求精。当程序任务十分复杂以至无法描述时，可以将它拆分为一系列较小的功能部件，直到这些完备的子任务小到易于理解的程度。当需要重复地调用某一子功能时，为了提高代码的可重用率，可以将完成这一子功能的这段代码定义成一个过程或函数，程序在需要这一子功能时就调用这一过程或函数。

在结构化程序设计方法中，所有程序都具有三种结构：顺序、分支判断和循环。顺序结构

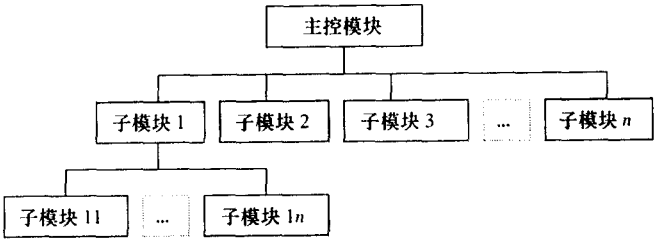

图 1.1　程序模块图

表示按语句的先后顺序依次执行。分支判断表示当程序执行到某一处时，根据条件成立与否，选择执行或不执行某一些语句，如 C＋＋的 if 语句。循环表示当程序重复地完成某种操作时，使其重复执行某一组语句，直到条件成立或不成立为止，如 C＋＋的 for 语句。所有程序都可由顺序、分支判断与循环三种结构组成，通过三种结构的反复使用，人们可以编出功能复杂的程序。

2. 结构化程序设计的优缺点

结构化程序设计成功地为处理复杂问题提供了有力的手段。通过模块化，使得一个功能复杂的程序得以由多个人分别承担完成，程序结构清晰，可读性强，但到了 20 世纪 80 年代末，随着数据信息量的增大，结构化程序设计的缺点也越来越突出。

(1) 数据与处理数据的方法分离使程序难以理解。

在结构化程序设计方法中，过程或函数定义为一个接一个顺序执行的一组指令。数据与程序分别存储，编程的主要技巧在于追踪哪些函数、调用哪些函数，哪些数据发生了变化。当数据量增大时，数据与处理这些数据的方法之间的分离使程序变得越来越难以理解。对数据处理能力的需求越强，这种分离所造成的负面影响越显著。

(2) 代码可重用率低。

采用结构化程序设计方法的程序员发现，当需要解决新的问题时，原来程序的代码并不能提供太多的价值，人们无法重用原来的程序，必须要重新编写新的代码，因此软件的开发效率难以提高，可重用性亟待增强。

1.2.2　面向对象程序设计

面向对象是相对于面向过程而言，面向对象程序设计是面向过程的结构化程序设计的发展。它同样采用了程序模块化和结构化的思想，所有程序都可由顺序、循环、分支三种结构构成，具有结构化程序设计的所有优点，同时又弥补了结构化程序设计中数据与过程分开、代码可重用率低等缺点。它的本质是把数据和处理数据的过程当成一个整体。

在面向对象程序设计中，所有具有相同特征的对象构成类。类是所有同类对象的集合。对象是类的实例。在一个类中不仅包含数据还包括了处理数据的方法。由类可生成子类与实例，子类与实例继承了类的数据和方法，这种思想能有效提高代码的可重用率。

学习面向对象程序设计需要了解对象与类的概念，需要使用封装和数据隐藏技术、继承和重用技术、多态性技术。下面分别介绍这几个概念。

1. 对象

一般而言，对象是现实世界中一个实际存在的事物，它可以是有形的(如一辆汽车、一个学生)，也可以是无形的(如一项计划、学生成绩)。对象是构成世界的独立单位，它具有自己的静

态特征(可以用某种数据来描述)和动态特征(对象所表现的行为或具有的功能)。

在面向对象方法中,对象是系统中用来描述客观事物的一个实体,是用来构成系统的一个基本单位。对象由一组属性和一组行为(方法、过程)构成。属性是用来描述对象静态特征的数据项,行为(方法、过程)是用来描述对象动态特征的操作序列。

2. 类

把众多的事物归纳、划分成一些类,是人类在认识客观世界时经常采用的思维方法。分类所依据的原则是抽象,即忽略事物的非本质特征,只注意那些与当前目标有关的本质特征,从而找出事物的共性。把具有共同性质的事物划分为一类,得出一个抽象的概念,例如,石头、树木、汽车、房屋等都是人们在长期的生产和生活实践中抽象出的概念。

在面向对象方法中,类是具有相同属性和方法(服务、过程)的相关对象的集合。它为属于该类的全部对象提供了抽象的描述。类与对象的关系犹如模具与铸件之间的关系。一个属于某类的对象称为该类的一个实例。

3. 封装和数据隐藏

当人们需要某种器件来完成某种事情的时候,往往只需知道该器件对外提供的性能就行,而对于该器件的内部结构可以不去了解。例如看电视,只需使用电视机面板上或遥控上的按钮就行,并不需要去知道电视机内部是如何工作的;再例如当安装一台电脑时需要一个显卡,技术员并不需要用集成电路芯片和其他材料去制作一个显卡,而是到电脑公司购买一个他所需要的某种功能的显卡就行。技术员只关心显卡的功能,并不关心显卡内部的工作原理。显卡包含了构成显卡的所有电路,自成一体。这种自成一体性称为封装性。不需要知道封装单元内部是如何工作就能使用的思想称为数据隐藏。

显卡的所有属性都封装在显卡中,不会扩展到显卡之外。通过一个与主板的接口,显卡就能与主板连接起来,完成相应功能。因为显卡的数据隐藏在该电路板上,技术员无需知道显卡的工作原理就能有效地使用它。这种思想,对于软件编程同样具有很好的意义。

C++采用了这种思想,通过建立用户定义类型(类)来支持封装性和数据隐藏。当定义一个类后,这个类就可以看成是一个完全封装的实体,被当做一个整体单元使用。类的实际内部工作原理应当隐藏起来,用户不需要知道类是如何工作的,只需要知道如何使用它就行。

4. 继承和重用

当人们需要开发一个新型号的产品时,一般来说,这个新产品并不需要完完全全从头开始,它仍然需要继承原来老产品的一些功能,只不过再增加一些新的功能而已。这种思想便是继承和重用技术。

C++采用继承和重用的思想。当程序定义了一个类以后,可以利用这个类再声明新的子类。原来的类称为父类(基类)。新子类是从现有类派生而来,称为派生类。由派生类还可生成派生类。派生类继承了基类的属性与方法,也可增加新的属性与方法,就好像一台新型号电视机继承了原有电视机的所有属性,并且又在原有型号的电视机上增加若干种新功能一样。

生活中存在大量的类,这些类又可生成子类。如交通工具、车、汽车、小汽车、桑塔纳轿车之间的关系就是父类(基类)与子类(派生类)之间的关系。正是因为面向对象的编程方法更接近于对现实世界的描述,因此具有更高的编程效率。

5. 多态性

通过继承的方法构造类,采用多态性为每个类指定具体的表现行为。C++允许同名函数存在,在程序运行时,如需要调用某一个函数,C++能依据所使用参数的数据类型来确定