高校非计算机专业计算机基础教育改革型教材

C++程序设计语言（第2版）

GAOXIAOFEIJISUANJIZHUANYE
JISUANJIJICHUJIAOYUGAIGEXINGJIAOCAI

◎成　颖　主编

东南大学出版社
SOUTHEAST UNIVERSITY PRESS

C++程序设计语言

（第2版）

成颖　主编

东南大学出版社

·南京·

内 容 提 要

C++在C语言的基础上，增加了对面向对象编程、类属编程、数据抽象等技术的支持，还对C语言进行了非面向对象的扩充。使用C++语言进行程序设计可以获得可重用性、可靠性、连续性、访问控制、继承性以及多态性等优势。

本书是基于第一版的修订。继承了第一版提供完整实例的特点，例子全新并紧贴学生实际。向读者介绍主要支持结构化程序设计的C语言，以及在C语言基础上进行扩展的支持面向对象程序设计的C++语言。在介绍C/C++语言的同时，还介绍了结构化程序设计以及面向对象程序设计的主要内容。

图书在版编目(CIP)数据

C++程序设计语言 / 成颖主编. —2版. —南京：东南大学出版社，2008.2

ISBN 978-7-5641-1094-9

Ⅰ.C… Ⅱ.成… Ⅲ.C语言—程序设计 Ⅳ.TP312

中国版本图书馆CIP数据核字(2008)第007892号

C++程序设计语言(第2版)

出版发行 东南大学出版社
出 版 人 江 汉
社 址 江苏省南京市四牌楼2号(210096)
经 销 江苏省新华书店
印 刷 兴化市印刷有限责任公司
版 次 2008年2月第2版 2008年2月第1次印刷
开 本 787mm×1092mm 1/16
印 张 26
字 数 649千字
书 号 ISBN 978-7-5641-1094-9/TP·181
定 价 38.00元

第一版前言

C语言是AT&T贝尔实验室的Dennis Ritchie和Ken Thompson于1972年设计的一种程序设计语言。C语言具有与ALGOL及Pascal类似的结构化语言风格，同时也具有PL/I语言的诸多特点，其简洁的语法和高效的执行效率使其成为一种流行的系统程序设计语言。现在，C语言已经演变为一种通用型的程序设计语言，可运行于多种系统平台以及多种结构的计算机。同时，C语言也已成为主流的教学语言，本书的介绍以ANSI C为基准。

C++语言是由AT&T贝尔实验室的Bjarne Stroustrup博士设计、实现的，该语言以广泛使用的、适合系统程序设计的C语言为基础，同时吸收了Simula语言在组织与设计方面的特长。C++最初的版本称为"带类的C"(C with classes)，1980年首次投入使用，当时它只支持系统程序设计技术以及数据抽象；1983年，C++语言增加了对基本的面向对象程序设计的支持；1985年，C++第一次走向市场；1987至1989年，增加了对类属程序设计支持。C++的标准化工作于1990年启动，主要由ANSI(American National Standard Institute)以及后来加入的ISO(International Standards Organization)负责，1998年正式发布了第一个国际标准(ISO/IEC 14882)。

C++在C语言的基础上增加了对面向对象编程、类属编程、数据抽象等技术的支持，还对C语言进行了非面向对象的扩充。使用C++语言进行程序设计可以获得可重用性、可靠性、连续性、访问控制、继承性以及多态性等优势。

本书向读者介绍主要支持结构化程序设计的C语言，以及在C语言基础上进行扩展的支持面向对象程序设计的C++语言。在介绍C/C++语言的同时，还介绍了结构化程序设计以及面向对象程序设计的主要内容。第1至8章主要介绍ANSI C的内容，由于C++是C语言的超集，因此，这部分内容也可以认为是C++的内容；第9至第15章是C++的特有内容，介绍了C++对C的非面向对象的扩充以及面向对象的扩充。需要指出的是第14章介绍了C++近几年新增的内容，包括新的强制类型转换运算符、名字空间、运行时类型信息，以及一般教材都不太介绍的异常处理，不过由于这些涉及的内容比较多，因此本书只介绍了主要的部分，一些细节性的内容没有展开。考虑到本书主要的读者对象是初学者，因此，本书的例子大多数都是完整的，并且都已经在Visual C++ 6.0上测试通过。对例子的分析和研究有助于读者了解实际应用中采用面向对象方法解决问题的基本策略。

本书第1章以及第9至15章由成颖编写，第2、3、4章由戴莉苇编写，第5、6、7、8章由曹英编写，各章的习题由张薇薇准备，韩晓静、孙立栋完成了全书的校对工作。全书由成颖修改定稿。

在此，要特别感谢南京大学博士生导师孙建军教授在本书成书过程中给予的帮助和指导，以及就编写策略提出的许多宝贵意见。东南大学出版社张煦老师为本书的编写和出版提供了热情的帮助，在此表示最衷心的谢意。

编　者

2002年10月

第二版前言

本书第一版共15章，基本上囊括了C++语言的所有内容，教学过程中所需学时甚多，由于各高校纷纷将原先两个长学期改为三个学期之后，基本上每门课程的学时数都被削减，因而学时数显得更加不够；第一版在第8以及第15章分别介绍了C语言的I/O函数库以及C++语言的面向对象的I/O类库，现在看来没有必要；此外，在教学过程中，还发现了第一版的其他一些不足之处。为了能够适应学期变短、学时变少的实际，同时让学生在学习过程中能够真正学习到面向对象程序设计的内核，完成了这次修订。对于教材内容的选择有如下的一些基本的思考：

第一，无论哪种风格的程序设计语言都离不开的一些语言元素，比如基本数据类型、运算符、表达式、控制结构、函数以及数组等，这些内容显然应纳入教材之中。

第二，C++语言是支持面向对象的程序设计语言，因而面向对象的核心特点，比如封装性、继承性以及多态性在教材中应突出。

第三，程序设计风格的体现。目前，虽然面向对象的程序设计已经成为主流，不过我们认为这主要体现在宏观层面上，在微观上依然离不开结构化程序设计，比如面对问题域，通过面向对象分析确实可以将其通过n个类来表达，但是在类的微观实现方面依然需要通过结构化程序设计去描述类的行为，因而在具体实例的分析以及实现方面，本书力争做到面向对象程序设计与结构化程序设计的结合。

第四，考虑到篇幅以及学时缩短的现实，本书没有介绍C++在软件工程实践中的重要性，但是只要能够在掌握面向对象基本思想的基础上，稍微花点时间就可以了解以及掌握的内容，本书没有将其纳入，比如C++类库中提供的一些包括Vector、List以及Deque等模板，这些内容读者可以翻阅手册很快就能够使用。

与第一版相比，本书所有的例子都重写了一遍，以使其完全符合最新的C++标准，同时也体现了笔者从编写本书的第一版以来自己学到的一些东西。第一版中的大部分文字都重新进行了检查和修订，在封装性、继承性以及多态性等章节中基本上都围绕一到两个例子以说明这些面向对象的基本属性，避免了第一版中例子的选择能说明问题即可的现象，例子紧贴学生的实际，这样可以增加学生的感性认识，从而增加学生对问题的理解。

本次修订继承了第一版提供完整实例的特点，还是考虑到本书主要的读者对象是初学者，本书的例子都已经在Visual C++ 6.0(SP6)上测试通过，使用VC6的读者需要安装SP6。对完整实例的模仿和分析有助于读者很快入门，并增加对实际应用中采用面向对象方法解决问题基本策略的了解和认识。

本书修订工作由成颖完成，孙海霞准备了各章习题的初稿，全书由成颖修改定稿。

本次修订工作得到了南京大学博士生导师孙建军教授就修订策略与方案方面所提供的指导与帮助。东南大学出版社张煦老师为本次修订提供了一如既往的支持与帮助，在此表示诚挚的谢意。

因水平有限，书中难免存在错漏和不妥之处，望读者批评指正，以利再次修订时改进和提高。

编　者

2007 年 9 月

目 录

1 程序设计概述

程序设计包括三方面，即程序设计语言、程序设计方法和集成开发环境。程序设计语言是用来控制计算机运行的工具，程序的数据与逻辑都包含在程序的源代码中。程序设计方法是指用什么方法来组织程序内部的数据和逻辑。集成开发环境则是用来帮助程序设计人员组织、编译、调试程序的工具。程序设计的发展过程是这三方面的发展过程，三者既有相互独立的一面，也有相互促进的一面。

1.1 程序设计语言概述

语言是交流的工具，人与人之间的交流主要通过自然语言进行，比如汉语、英语以及德语等。限于自然语言处理技术离实际应用尚存在较大的距离，到目前为止的计算机还不能理解自由度很高的自然语言，这样人与计算机以及计算机与计算机之间的交流还需要通过比较严谨、规范的程序设计语言进行。

所谓程序设计语言其实质是一个记号系统，与自然语言一样，程序设计语言主要有语法和语义两方面的定义，有时也可以包含语用信息。语法包括词法规则和产生规则，一个程序设计语言只使用一个有限的字符集作为字母表，词法规则是单词符号的形成规则，产生规则规定了如何从单词符号形成更大的结构(语法单位)的相关规则，也就是说语法规范了构成程序的各个记号之间的组合规则，但不涉及这些记号的特定含义，也不涉及使用；语义表示程序的含义，具体而言就是明确单词以及语法单位的特定含义，但也不涉及使用；语用则表示程序与使用的关系。

程序设计语言的基本成分有：① 数据成分，用于描述程序所涉及的数据；② 运算成分，用以描述程序中所包含的运算；③ 控制成分，用以描述程序中所包含的控制；④ 传输成分，用以表达程序中数据的传输。

程序设计语言按照语言级别可以分为低级语言和高级语言。低级语言有机器语言和汇编语言。高级语言则主要有过程式语言(如 C、Basic 以及 Pascal 等)、面向对象语言(如 C++、Java 等)、应用式语言(如 Lisp)以及基于规则的语言(如 Prolog)。

1.1.1 机器语言

最早的程序设计语言是机器语言，它完全用 0 和 1 组成的代码表示，也是最低层的程序设计语言。用机器语言编写的程序中，每一条机器指令都是二进制形式的指令代码，计算机硬件可以直接识别。机器语言是面向机器的，不同的计算机硬件(主要是 CPU)其机器语言是不同的，因此，针对一种计算机所编写的机器语言程序不能在另一种计算机上运行。

由于机器语言程序是直接针对计算机硬件编写的，因此它的执行效率比较高，能充分发挥计算机的性能。不过，用机器语言编写程序即使是一个简单的算式，也要用很多条指令才能表达出来，因而编写程序的难度比较大，容易出错，效率低，程序的可读性、可移植性也都比较差。

1.1.2 汇编语言

为了克服机器语言可读性差的弊端，人们设计了汇编语言，汇编语言是一些英文助记符号，执行前需要将它翻译成机器语言。因为它几乎与机器语言一一对应，因而执行速度非常快。也正因为它几乎与机器语言一一对应，使得几乎有一种计算机就有一种汇编语言。同样功能的程序在不同的计算机上是不相同的。

汇编语言和机器语言都属于低级语言，这是因为其语言的结构是以面向机器的指令序列，与自然语言距离较远，所以它们的共同缺点是：

(1) 依赖于机器，可移植性差；

(2) 代码冗长，不易于编写大规模程序；

(3) 可读性差，可维护性差。

总之，即使是汇编语言，也没有解决计算机编程难的基本问题。低级语言必然被高级语言取代，没有高级语言就没有今天计算机应用的网络化、多媒体化、智能化等。

1.1.3 高级语言

与汇编语言相比，高级语言更接近自然语言，更易于为人们所理解。它把几条甚至几十条汇编指令浓缩在一起，语言表达更加简洁。只有到了高级语言，特别是微机出现以后，程序设计才从工程设计人员的个人艺术中解脱出来，为普通大众所了解和应用。

现在已经出现成千上万种高级语言，其中最早出现的是 1954－1957 年由 IBM 公司设计的 FORTRAN(FORmula TRANslator)语言，主要应用于需要复杂数学计算的科学和工程领域，目前仍然广为使用。COBOL(Common Business Oriented Language)是一群计算机制造商以及政府和工业界的计算机用户于 1959 年开发成功的，主要用于需要对大量数据进行精确和有效处理的商业领域，目前，许多商业软件仍然是用 COBOL 语言编写的。Pascal 语言几乎是和 C 同时设计的，成为 80 年代计算机科学的主流教学语言，目前其教学语言的地位正逐渐为 C 以及 C++语言所取代。

面向对象程序设计方法提出以后，出现了面向对象程序设计语言。从理论上讲，面向对象是一种方法，可以用任何语言实现，但是支持面向对象程序设计的语言更容易实现面向对象方法。此类语言分为两类：纯面向对象语言和混合型语言。纯面向对象语言中，所有的数据类型都定义成对象，而混合型语言则是在普通程序语言的基础上，增加了对面向对象的支持，纯面向对象语言很容易描述现实世界中事物之间的相互作用，很适合编写大型的应用程序，但当处理具体的过程时，显得过于繁琐。目前，典型的纯面向对象语言有 JAVA 和 SmallTalk，典型的混合型语言有 C++。需要指出的是，语言的高级和低级并不说明语言的优劣，它只是表明了这种语言离硬件的远近，同时也暗含了程序设计的难易和执行效率的高低。

与汇编语言一样，计算机也不能直接理解高级语言，只能直接理解机器语言，所以必须要把高级语言翻译成机器语言，计算机才能执行高级语言编写的程序。

翻译的方式有两种，一种是编译，一种是解释。两种方式只是翻译的时间不同。编译型语言写的程序执行之前需要一个专门的编译过程，把程序编译成为机器语言的文件，比如 exe 文件，以后要运行的话就不用重新翻译了，直接使用编译的结果就可以了(exe 文件)，因为翻译只做了一次，运行时不需要翻译，所以编译型语言的程序执行效率高，本书要学习的 C++语

言就属于编译型语言。

解释则不同，解释型语言的程序不需要编译，省了道工序，解释型语言在运行程序的时候才翻译，比如解释型 Basic 语言，专门有一个解释器能够直接执行 Basic 程序，每条语句都是执行的时候才翻译。这样解释型语言每执行一次就要翻译一次，效率比较低。Java 语言很特殊，Java 程序也需要编译，但是没有直接编译成为机器语言，而是编译为字节码，然后用解释方式执行字节码。

1.2 程序设计方法概述

早期的计算机存储器容量非常小，人们设计程序时首先考虑的问题是如何减少存储器开销，硬件的限制不容许人们考虑如何组织数据与逻辑，程序本身短小，逻辑简单，也无需人们考虑程序设计方法问题。与其说程序设计是一项工作，倒不如说它是程序员的个人技艺。但是，随着大容量存储器的出现以及计算机应用范围的扩大，程序编制越来越困难，程序的大小以算术级数递增，而程序的逻辑控制难度则以几何级数递增，这样人们不得不考虑程序设计的方法。

不同的程序设计方法代表了不同的程序设计范型，所谓“范型”(paradigm)，是指一种做事情的风格，特有的一套办法。程序设计范型是人们在程序设计时所采用的基本方式模型。例如，过程程序设计、结构化程序设计、函数程序设计、逻辑程序设计等都具有不同的程序设计范型。其中，过程式、结构式程序设计范型是选定数据结构、设计算法的过程，对结构式程序设计还要按结构化的思维方式表达算法过程；逻辑程序设计范型是列举事实，定义规则，以提问方式求得解的过程。每一种程序设计范型都与相应的程序设计语言有直接的联系。如过程程序设计与 Fortran、结构化程序设计与 Pascal、函数程序设计与 Lisp、逻辑程序设计与 PROLOG 等。面向对象程序设计范型将计算看作是一个系统的开发过程，系统由对象组成，经历一连串的状态变化以完成计算任务。

1.2.1 手工艺式方法

程序设计就是对解决问题的算法以最终在计算机上能执行的程序代码的形式来描述的过程。对第一代和第二代计算机来说，由于其速度较慢，存储空间较小，加之大多数采用机器语言或汇编语言编写程序，所以要求程序的运行时间尽量短，占用的存储单元尽量少。这样，程序设计是一项技巧性很强的工作，也可以说，采用的是一种手工艺式的设计方法，程序的可读性和可移植性较差，一度使用后即完成使命，寿命一般很短。

自 60 年代到 70 年代，随着计算机硬件技术的发展和计算机应用范围的不断扩大，需要研制一些大型的软件系统，如操作系统。而大型系统的编制周期较长，工作量很大，且由于传统的程序设计方法的局限性，设计出的软件系统往往隐藏着许多错误，这就给软件的使用和维护带来很大困难。一方面，客观上需要研制大量的复杂的软件；另一方面，按照原有方法研制软件周期长，可靠性差，维护困难。这就产生了所谓的“软件危机”。

1.2.2 结构化方法

延长程序的寿命、扩大其使用范围是克服软件危机的一种手段。要使许多人反复长期都能使用，就要考虑到程序必须是易读的，而且要适应环境的变化，必须容易修正和维护。1968