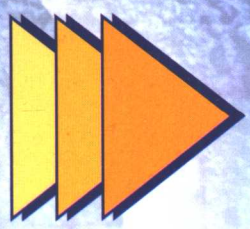
高等学校教材

大学C++程序设计教程

罗建军　朱丹军　顾　刚　刘路放　编著
冯博琴　审



高等教育出版社
HIGHER EDUCATION PRESS

高等学校教材

大学C++程序设计教程

罗建军　朱丹军　顾　刚　刘路放　编著
冯博琴　审

高等教育出版社

内 容 提 要

本书是根据教育部“关于进一步加强高等学校计算机基础教学的意见”组织编写的教材。

本书讲授面向对象C++程序设计。主要内容包括:C++设计基础、控制结构、基本数据类型、表达式、函数、指针、类和对象、继承与派生、多态性、模板、基本算法和数据结构等。在编写体例上,按照本章目标、授课内容、自学内容、程序设计举例、实例编程、编程提示、小结和习题等精心组织内容,条理清楚,逻辑分明。书中还提供了大量有代表性的例题、实例和习题,在强调面向对象理论的同时,突出了实践环节,从而切实使读者的独立编程能力得到提高。

本书可作为高等学校理工类各专业程序设计课程的教材或参考书,也可供应用开发人员学习参考。

本书的支持网站为西安交通大学计算机教学实验中心(http://ctec.xjtu.edu.cn),其上配有本书最新的教学辅助课件、上机实验指导及示例程序源码(请到网站上相关的版块查询),同时它还具有一个全交互性、立体化的网络教学平台(包括教学主要环节、讨论答疑区和最新学习指导信息等),所有内容都在不断更新,可供教师教学和学生自学使用。

图书在版编目(CIP)数据

大学C++程序设计教程/罗建军等编著.—北京:高等教育出版社,2004.8

ISBN 7-04-015505-2

Ⅰ.大… Ⅱ.罗… 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(2004)第078275号

策划编辑 陈红英 **责任编辑** 韩 飞 **市场策划** 刘 茜
封面设计 李卫青 **责任印制** 朱学忠

出版发行	高等教育出版社	**购书热线**	010-64054588
社 址	北京市西城区德外大街4号	**免费咨询**	800-810-0598
邮政编码	100011	**网 址**	http://www.hep.edu.cn
总 机	010-82028899		http://www.hep.com.cn
经 销	新华书店北京发行所		
印 刷	北京鑫海金澳胶印有限公司		
开 本	787×1092 1/16	**版 次**	2004年8月第1版
印 张	22.25	**印 次**	2004年8月第1次印刷
字 数	560 000	**定 价**	26.00元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

前　言

在面向对象程序设计语言中，C++是最流行的语言之一。C++从C语言演变而来，语法上也更严格和安全，而它提供的面向对象功能，使采用C++编写的程序更容易理解和维护，可以更高程度地进行复用，从而更快、更正确、更经济地开发软件。随着国际标准化组织(ISO)和美国国家标准化协会(ANSI)的ISO/ANSI C++语言标准(ISO/IEC FDIS 14882)的推出，C++迅速成为使用最广泛的编程语言。

本教程是C++入门教科书，适用于大学本科类计算机及相关专业学生程序设计能力的培养。为了使基础不高的初学者也能很快地掌握程序设计方法，我们在确定教学目标、设计教学模式、编写教程内容等方面进行了一系列革新探索。本教材具有三个特点：(1) 全书是按照教育部最新的“关于进一步加强高校计算机基础教学的几点意见”文件要求组织编写，并以现代教育理论为指导，适合于“精讲多练”的教学模式；(2) 作者在多年教学实践中发现初学者常常易犯一些小错误，或者不注意培养好的编程习惯，本书将这些带有普遍性的问题以编程提示的形式列举出来，以便初学者少犯错误，少走弯路，尽快掌握C++语言程序设计方法；(3) 为了便于初学者进一步理解和巩固C++基本知识，提高解决实际问题的能力，本书增加了基本数据结构和数值计算两方面的内容。

本教程的目标是使学生掌握使用C++设计应用程序的基本技能，了解面向对象和结构化程序设计的方法，能够编写、调试和运行实用、规范、可读性好的C++程序。本书“从零开始”，在内容组织上循序渐进，不要求学生学过程序设计方面的先修课程。

本教程由16章组成，这16章又可分为三部分：

第一部分为第1～8章，介绍C++编程的基本内容，包括控制结构、基本数据类型、表达式、函数，指针和引用。

第二部分为第9～13章，讲解类与对象、继承和多态性等面向对象程序设计的基础理论以及C++的标准库。

第三部分为第14～16章，介绍基本数据结构和数值计算方法。

为了便于教学，每章按以下主题进行组织：

本章目标　本书的特点是“精讲多练”，因此为教师和学生规定明确的教学和学习目标是非常重要的。

授课内容　是建议教师课堂讲授的内容。一般来说，授课内容是本章所有教学内容的“纲”，起着联系本章所有项目的作用。授课内容部分按2学时组织。第1章的授课内容份量略轻，这是因为在第1章的授课时间中还应划分出部分时间用于介绍上机实验的编辑、调试和运行应用程序项目的基本步骤。

自学内容　“自学内容”和“授课内容”部分一起组成了每一章的基本教学内容。这部分内容通常都是“授课内容”的延伸和继续，由学生在课外时间自学。必须强调的是，自学部分并非不重要，建议不要省略。一般来说，教师应在授课时间中抽出5～10分钟对自学内容略做导引，以便于学生自学。

程序设计举例 为了补充授课内容和自学内容部分的例题，我们设置了程序设计举例部分，提供了大量例题。所有例题均与本章的授课或自学等部分的内容相关，是学生学习和复习本章内容的重要参考资料。同时，本书的所有源代码都是采用标准C++编写的，程序可以使用各种C++编译器进行编译，通常不需要修改或稍做修改即可在所有这些编译器中运行。

实例编程 提供一个比较复杂完善的程序例子，一般是综合多个章节的知识点，解决某个实际问题，它对培养学生分析问题和解决问题的能力很有帮助。这部分一般可以作为习题课或辅导课选讲的内容，也可留给学生自学。

编程提示 介绍一些编程中常见的问题和技巧，帮助学生养成良好的编程习惯，从而编写出正确、清晰易懂、易于测试维护的代码。实践证明，这部分内容对学生提高编程质量是很有帮助的。

小结 是对本章重要知识点的总结，帮助学生复习本章的关键知识。

习题 每章均配有若干习题，供学生上机练习。这些练习题目均为程序设计题目，传统的做法是先编程，再上机。由于C++的特点，也可以在写出较详细的伪代码之后直接上机。“精讲多练”式教学方法的基本特点是上机时数较多，所以这部分的习题量较大，因此，在上机时数不足的情况下可以酌情选做若干题目。

为了保证教学效果，在条件许可的情况下最好采用直接在计算机房进行的联机电化教学。在这种情况下，每个教学单元(即每章)可使用连续的4课时，先由教师讲解授课部分，并对自学部分等内容进行简短的指导(共2学时)，然后学生即可在教师指导下上机练习(2学时)。除此之外，如果能够提供一定数量的课外机时(如20～30小时)则更好。

近年来，西安交通大学计算机教学实验中心在计算机基础教育的理论和实践等方面进行了一系列探索和改革，其成果(“精讲多练”的教学模式是其中之一)荣获了1997年国家级教学成果一等奖。这些成果都是在冯博琴教授(博士生导师、国家级教学名师、教育部非计算机专业计算机课程教学指导分委员会主任委员)的领导下完成的，本课程的建设也不例外。本教程的构思和编写得到了冯博琴教授的多方指导，并由他审核了书稿，在此向冯老师表示深深的谢意。同时，西安交通大学对本书的编写也非常重视，将该书列入了西安交通大学“十五”规划本科生系列教材予以重点支持。参加编写工作的队伍由西安交通大学计算机教学实验中心长期从事程序设计教学和科研的一线教师组成，主要人员有：罗建军(第1～8章、第12章、附录)，朱丹军(第9～11章、第13章)，顾刚(第14～16章)。现远在加拿大的刘路放教授也对本书的编写非常关心，并起到了非常关键的作用。在本书编写过程中，作者亦曾与杨琦、崔舒宁、刘志强、程向前、李波、杨振平和卫颜俊等诸多同事进行了多次交流，受益匪浅。以上同事还提供了一些有用的材料，在此一并表示感谢。 由于作者学识浅陋，编写时间仓促，书中错误在所难免，希望读者不吝指教。本书编者的电子邮件地址是：jjluo@ctec.xjtu.edu.cn；djzhu@ctec.xjtu.edu.cn；ggu@ctec.xjtu.edu.cn。

编 者

2004年3月于西安交通大学

给教师教学的建议

• 本课程建议采取“精讲多练”教学模式。

• 本教程的内容安排采用难易结合、前后呼应的原则，在知识点较多、学生较难掌握的章节之后一般都安排较为简单、有一定弹性的章节，可以重复、强调前面的难点。如第2章(控制结构)学生普遍反映比较难，第3章(基本数据类型)的知识点安排就比较少，但提供了大量的例题供教师选择，这样一方面学生可以有一个喘息机会，另一方面教师可以通过这些例题，帮助学生举一反三，全面回顾掌握前面所学的知识点。

• 本书的结构和其他常见的C++书籍有所区别，特别是第2章和其他章节的前后位置上是有很大的不同的。按照通常的做法，第2章(控制结构)是放在第3章(基本数据类型)和第5章(表达式)之后的，而本书却放在前面，对此有些教师可能不太理解，认为这在学习次序上好像是反的。实际上，我们这样做主要是出于这样的考虑，希望先给学生一个C++程序结构的认识，了解结构化程序设计方法，知道如何将一个实际问题转化为一个合适的控制结构，在此基础上，再从细节上去丰富完善这个程序。

• 要引导学生正确使用本教材，着重培养学生实际上机编程能力，而不要在具体语法细节上纠缠过多。因为本书的知识点是不停重复、前后呼应的，可能前面章节已经用到了后续章节的知识，如果教师和学生在前面就花很多时间和精力来研究某些后面会讲的难点和细节就得不偿失了。例如，第2章的核心例题：验证哥德巴赫猜想问题，教师应该将主要精力放在讲解其结构化编程思想上，而不要太多涉及函数调用、数组斯用等问题，对学生也不要要求其掌握这部分内容，因为后面都会有专门章节讨论这些问题。

• 要引导学生使用最简单、最基本的语句编写出正确的程序，而不要沉湎于编程技巧。

• 设置第14～第16章的目的主要是为了帮助学生总结应用前面学过的编程技术，同时使学生了解一些基本数据结构及算法，教师可根据学生程度和课程学时的不同对该部分内容自行处理。

• 本书所有程序基本都采用标准C++编写，通常不用修改或只需稍加调整即可在目前常用的各种编译器上使用。有关上机实验指导部分内容的材料，教师可以参考本书支持网站上的多媒体课件，同时本书配套的习题与实验指导教材也正在组织撰写中，请留意相关网站和出版社宣传信息。

给学生学习的建议

• 学习程序设计的关键是“多练”，纸上谈兵学不出程序设计本领，只有大量编程、上机调试才能掌握。勤于上机练习是学好程序设计语言的惟一途径，你的编程能力与在计算机上投入的时间成正比。

• 应该指出的是,本书不是自学教材,必须在教师教授和指导下学习,如果学生仅仅靠自己阅读本书来学习C++,其效率会很低。但本书又鼓励学生自学某些内容,主要是要求学生在听完教师提纲挈领的知识点讲授后,扩大知识面,提高编程能力。所以本书的自学部分是指在教师指导下自学。

• 编程序最重要的是掌握编程思想,仔细揣摩别人的程序(如本书中的大量例题)会获得其编程思想。

• 即使你可以很容易地从网上得到本书的所有例题源代码,也建议最好将书里的所有例子程序都亲手输入到计算机上编译调试,并尝试对其进行修改扩充。

• 编程序要肯花精力才能学好,要多看优秀程序,多做上机练习。

• 对已经编写好的程序,请采用某种有效的方式组织保存好,它们也许会成为你今后工作学习中很有用的资源,同时经常回顾以前编写的程序,并尝试用新学到的知识重写它们,这会给你带来新的认识和体会。

• 初学者应尽量使用简单、明了的C++语句完成程序的编写,而不要刻意去追求编程技巧和代码的简练。写出正确的、容易读懂的程序是最重要的。

• 良好的编程风格有助于程序的调试、修改、扩充以及日后的阅读。

• 除了传统的学习方法以外,你还应该充分利用网络平台,提高获取信息、处理信息和交流信息的能力,本书附录3提供了一些有用的网络资源,可以参考使用。

The Process of preparing programs for a digital computer is especially attractive, not only because it can be economically and scientifically rewarding, but also because it can be an aesthetic experience much like composing poetry or music.

– Donald E. Knuth's The art of computer programming (1997)

目　录

第1章　C++语言简介 …… (1)
本章目标 …… (1)
授课内容 …… (1)
1.1　C++程序基本结构 …… (1)
1.2　算法与程序 …… (2)
1.3　C++程序的基本要素 …… (3)
1.3.1　标识符、关键词和标点符号 …… (3)
1.3.2　注释 …… (4)
1.3.3　源程序 …… (4)
1.3.4　文件包含 …… (5)
1.3.5　输入与输出 …… (5)
1.4　输入、编译、调试和运行一个C++程序 …… (6)
自学内容 …… (6)
1.5　程序设计语言的发展 …… (6)
1.6　C++语言的历史、特点、用途和发展 … (8)
1.7　编译预处理 …… (9)
程序设计举例 …… (11)
编程提示 …… (14)
小结 …… (15)
习题 …… (15)
第2章　控制结构 …… (17)
本章目标 …… (17)
授课内容 …… (17)
2.1　程序的基本控制结构 …… (17)
2.2　自顶向下,逐步求精 …… (19)
2.3　C++的控制结构 …… (21)
2.3.1　顺序结构 …… (21)
2.3.2　选择结构 …… (21)
2.3.3　循环结构 …… (23)
自学内容 …… (24)
2.4　结构化程序设计方法的发展历史 …… (24)
2.5　C++的其他控制转移语句 …… (25)
2.5.1　switch语句 …… (25)
2.5.2　goto语句和语句标号 …… (27)
2.5.3　break语句和continue语句 …… (27)
2.5.4　exit()函数和abort()函数 …… (28)
程序设计举例 …… (28)
实例编程　验证哥德巴赫猜想 …… (32)
编程提示 …… (34)
小结 …… (35)
习题 …… (36)
第3章　基本数据类型 …… (37)
本章目标 …… (37)
授课内容 …… (37)
3.1　数据类型 …… (37)
3.1.1　整型数据的表示方法 …… (38)
3.1.2　实型数据的表示方法 …… (38)
3.2　常量 …… (39)
3.2.1　整型常量 …… (39)
3.2.2　实型常量 …… (39)
3.2.3　字符常量 …… (39)
3.2.4　字符串常量 …… (40)
3.2.5　布尔型常量 …… (41)
3.3　变量 …… (41)
3.3.1　变量的声明 …… (41)
3.3.2　变量的初始化 …… (42)
3.4　枚举类型 …… (43)
自学内容 …… (45)
3.5　typedef语句 …… (45)
3.6　类型修饰符 …… (46)
3.7　常量修饰符 …… (46)
3.8　八进制和十六进制常量 …… (47)
程序设计举例 …… (48)
实例编程　模拟仿真 …… (54)
编程提示 …… (57)
小结 …… (57)
习题 …… (58)
第4章　数组与结构体 …… (60)
本章目标 …… (60)
授课内容 …… (60)
4.1　数组 …… (60)
4.1.1　一维数组 …… (61)
4.1.2　二维数组 …… (62)

4.1.3 多维数组 …… (63)
4.2 字符型数组和字符串处理库函数 …… (64)
4.2.1 字符型数组的定义和初始化 …… (64)
4.2.2 字符串的输入与输出 …… (64)
4.2.3 字符串处理库函数 …… (66)
4.3 结构体类型 …… (68)
4.3.1 结构体类型的定义 …… (69)
4.3.2 结构体类型变量的使用 …… (70)
自学内容 …… (70)
4.4 数组和结构体 …… (70)
4.4.1 结构体中的数组 …… (70)
4.4.2 数组中的结构体 …… (71)
4.5 结构体中的结构体(结构体嵌套) …… (71)
程序设计举例 …… (72)
编程提示 …… (77)
小结 …… (77)
习题 …… (78)
第5章 表达式 …… (79)
本章目标 …… (79)
授课内容 …… (79)
5.1 表达式概述 …… (79)
5.2 算术运算符和算术表达式 …… (79)
5.3 关系运算符和关系表达式 …… (80)
5.4 逻辑运算符和逻辑表达式 …… (80)
5.5 赋值运算符和赋值表达式 …… (81)
5.6 自增运算符和自减运算符 …… (82)
5.7 表达式中各运算符的运算顺序 …… (83)
5.8 类型不同的数据之间的混合算术运算 …… (84)
5.9 名字空间 …… (85)
自学内容 …… (87)
5.10 其他具有副作用的运算符 …… (87)
5.11 问号表达式和逗号表达式 …… (87)
5.12 表达式语句 …… (88)
5.13 位运算表达式 …… (89)
程序设计举例 …… (91)
实例编程 Josephus 问题 …… (96)
编程提示 …… (97)
小结 …… (98)
习题 …… (98)
第6章 函数 …… (99)
本章目标 …… (99)
授课内容 …… (99)
6.1 函数概述 …… (99)
6.2 函数的定义 …… (99)
6.3 函数的调用 …… (101)
6.4 函数原型 …… (102)
6.5 函数间的参数传递 …… (103)
6.5.1 值调用 …… (103)
6.5.2 引用调用 …… (104)
6.6 局部变量和全局变量 …… (105)
自学内容 …… (106)
6.7 带有默认参数的函数 …… (106)
6.8 C++的库函数 …… (107)
6.9 变量的存储类别 …… (107)
6.9.1 自动变量 …… (107)
6.9.2 静态变量 …… (108)
6.9.3 寄存器变量 …… (109)
6.9.4 外部变量 …… (109)
6.10 多源程序文件程序中的全局变量说明 …… (109)
6.11 变量使用小结 …… (110)
程序设计举例 …… (111)
编程提示 …… (116)
小结 …… (116)
习题 …… (117)
第7章 指针 …… (118)
本章目标 …… (118)
授课内容 …… (118)
7.1 地址与指针 …… (118)
7.1.1 地址 …… (118)
7.1.2 指针 …… (119)
7.2 指针运算 …… (119)
7.2.1 * 和 & 运算符 …… (119)
7.2.2 指针变量算术运算 …… (122)
7.2.3 指针变量比较运算 …… (122)
7.2.4 指针变量下标运算 …… (122)
7.3 指针与数组 …… (122)
7.4 动态存储分配 …… (125)
自学内容 …… (127)
7.5 指针数组 …… (127)
7.6 指向指针的指针 …… (129)
7.7 结构体与指针 …… (131)
7.8 指针的初始化 …… (131)
程序设计举例 …… (132)
编程提示 …… (134)

小结 …… (134)
习题 …… (135)
第8章　函数与指针 …… (136)
本章目标 …… (136)
授课内容 …… (136)
8.1　递归函数 …… (136)
8.2　函数重载 …… (139)
8.3　指针和函数 …… (140)
8.3.1　指针作为函数的参数 …… (140)
8.3.2　返回指针的函数 …… (141)
8.3.3　指向函数的指针 …… (142)
8.4　带参数的 main()函数 …… (143)
自学内容 …… (144)
8.5　内联函数 …… (144)
8.6　不使用参数的函数 …… (145)
8.7　void 和 const 类型的指针 …… (145)
程序设计举例 …… (146)
实例编程　棋类游戏 …… (150)
编程提示 …… (154)
小结 …… (154)
习题 …… (155)
第9章　类和对象 …… (156)
本章目标 …… (156)
授课内容 …… (156)
9.1　面向对象 …… (156)
9.2　类与对象 …… (157)
9.2.1　类的定义 …… (157)
9.2.2　成员函数的定义 …… (159)
9.2.3　内联成员函数 …… (159)
9.2.4　对象 …… (160)
编程步骤 …… (160)
9.3　构造函数与析构函数 …… (162)
9.4　数据成员的初始化 …… (164)
9.5　对象与指针 …… (165)
自学内容 …… (166)
9.6　C++ 的 string 类 …… (166)
9.6.1　string 类的字符串运算符 …… (166)
9.6.2　string 类的成员函数 …… (168)
9.6.3　字符串流处理 …… (169)
9.7　类的嵌套 …… (169)
程序设计举例 …… (170)
实例编程　职工档案管理系统 …… (173)
小结 …… (176)
习题 …… (176)
第10章　继承 …… (177)
本章目标 …… (177)
授课内容 …… (177)
10.1　基类与派生类 …… (177)
10.1.1　继承 …… (177)
10.1.2　派生类的定义 …… (177)
10.1.3　派生类中的变化 …… (178)
10.2　派生类的继承方式 …… (179)
10.2.1　公有继承 …… (179)
10.2.2　私有继承 …… (181)
10.2.3　保护继承 …… (183)
10.3　派生类的构造函数和析构函数 …… (184)
10.3.1　构造函数 …… (185)
10.3.2　析构函数 …… (186)
自学内容 …… (186)
10.4　接口与实现方法的分离 …… (186)
10.5　显式访问基类成员 …… (189)
10.6　使用 this 指针 …… (190)
程序设计举例 …… (191)
实例编程　象棋类 …… (195)
小结 …… (197)
习题 …… (197)
第11章　多态性 …… (198)
本章目标 …… (198)
授课内容 …… (198)
11.1　多态性概述 …… (198)
11.2　派生类对象替换基类对象 …… (200)
11.3　虚函数 …… (201)
11.3.1　虚函数定义 …… (201)
11.3.2　虚函数的使用限制 …… (203)
11.4　抽象类 …… (204)
11.5　运算符重载 …… (207)
自学内容 …… (208)
11.6　const 修饰符 …… (208)
11.7　静态成员 …… (209)
程序设计举例 …… (210)
实例编程　较完整的日期类 …… (215)
小结 …… (218)
习题 …… (218)
第12章　模板与异常处理 …… (220)
本章目标 …… (220)
授课内容 …… (220)

12.1 模板 …… (220)
12.1.1 函数模板 …… (220)
12.1.2 类模板 …… (222)
12.2 异常处理机制 …… (224)
自学内容 …… (227)
12.3 友元 …… (227)
12.3.1 友元函数 …… (227)
12.3.2 友元类 …… (228)
程序设计举例 …… (228)
实例编程 矩阵类 …… (232)
小结 …… (238)
习题 …… (238)
第 13 章 标准库和输入/输出流 …… (240)
本章目标 …… (240)
授课内容 …… (240)
13.1 标准库概述 …… (240)
13.2 流 …… (241)
13.3 输入/输出流 …… (241)
13.3.1 iostream 类库的头文件 …… (241)
13.3.2 输入/输出流类和对象 …… (242)
13.3.3 输入/输出流的成员函数 …… (243)
13.4 格式化 I/O …… (244)
13.4.1 流格式状态标志和格式化函数 …… (245)
13.4.2 流操纵符 …… (247)
13.5 文件处理 …… (250)
13.5.1 文件和流 …… (251)
13.5.2 打开和关闭文件 …… (251)
13.5.3 文件读/写 …… (253)
自学内容 …… (257)
13.6 对象的输入/输出 …… (257)
13.7 标准模板库(STL)简介 …… (258)
13.7.1 容器 …… (259)
13.7.2 迭代器 …… (260)
13.7.3 算法 …… (260)
程序设计举例 …… (260)
实例编程 电话簿管理程序 …… (261)
小结 …… (268)
习题 …… (268)
第 14 章 线性表 …… (269)
本章目标 …… (269)
授课内容 …… (269)
14.1 线性表 …… (269)
14.2 顺序表类 …… (271)
14.3 链表类 …… (276)
自学内容 …… (282)
14.4 堆栈 …… (282)
14.4.1 堆栈的逻辑结构 …… (282)
14.4.2 顺序栈类 …… (283)
14.4.3 链栈类 …… (283)
程序设计举例 …… (286)
小结 …… (291)
习题 …… (291)
第 15 章 查找和排序 …… (293)
本章目标 …… (293)
授课内容 …… (293)
15.1 查找 …… (293)
15.2 哈希查找 …… (295)
15.2.1 哈希表 …… (295)
15.2.2 哈希表的建立 …… (295)
15.2.3 解决地址冲突的方法 …… (296)
15.2.4 线性探测的哈希查找 …… (298)
15.2.5 链地址法的哈希查找 …… (300)
15.3 排序 …… (301)
15.3.1 排序概述 …… (301)
15.3.2 简单插入排序 …… (302)
自学内容 …… (304)
15.4 二次探测的哈希查找 …… (304)
15.5 基数排序 …… (306)
程序设计举例 …… (307)
小结 …… (313)
习题 …… (314)
第 16 章 数值计算 …… (315)
本章目标 …… (315)
授课内容 …… (315)
16.1 多项式的计算 …… (315)
16.1.1 逐项递推算法 …… (315)
16.1.2 秦九韶算法 …… (316)
16.2 多元一次代数方程组的求根计算 …… (316)
16.2.1 约当消元法 …… (317)
16.2.2 迭代法 …… (319)
16.3 求逆矩阵 …… (321)
16.4 积分计算 …… (324)
自学内容 …… (326)
16.5 梯度法求解非线性方程组 …… (326)

程序设计举例 …… (329)
小结 …… (332)
习题 …… (333)
附录 …… (334)
附录 1　ASCII 码表 …… (334)
附录 2　常用库函数 …… (335)
附录 3　C++ 的技术支持 …… (340)
参考文献 …… (342)

第1章　C++语言简介

本章目标

了解C++程序的基本特点。

授课内容

1.1　C++程序基本结构

计算机系统是由硬件和软件两大部分组成的，功能强大的硬件系统是计算机工作的基础，但如果缺乏必要的软件支持，其作用也将十分有限。

为计算机编写软件需要使用程序设计语言。目前可用的计算机语言有数百种之多，它们各有所长，例如，有些适用于开发数据库应用程序，有些适用于开发科学计算程序，有的简便易学，有的功能全面。本书介绍其中使用最为广泛的C++语言。

首先来看一个简单例子，以便让读者对 C++ 语言编写的程序有一个初步的认识。

例 1-1　第一个C++程序，在计算机屏幕上显示：Hello World!

程序

```
// Example 1-1: 屏幕上显示: Hello World!
#include <iostream.h>                          //包含基本输入/输出库文件
int main()                                     //主函数名
{
    cout << "Hello World!" << endl;            //屏幕显示语句
    return 0;                                  //表示程序顺利结束
}
```

输出

```
Hello World!
```

分析　该程序非常简单，仅由一个主函数构成，在主函数中也只有两条语句。

程序的第一行是注释。注释以“//”开头，直到该行的末尾，用于说明或解释程序段的功能、变量的作用以及程序员认为应该向程序阅读者说明的其他任何内容。可以看到，在该程序中还有一些注释。在将C++程序编译成目标代码时所有的注释行都会被忽略掉，因此即使使用了很多注释也不会影响目标码的效率。恰当地应用注释可以使程序清晰易懂、易于调试，便于程序员之间的交流与协作，所以，在编写程序时，精心撰写注释是一个良好的编程习惯。

第二行是编译预处理，把 iostream.h 库文件插入到程序中相应的位置，当一个程序包含有 iostream.h 时，几个标准流就自动地定义了，包括输入流对象 cin 和输出流对象 cout。

从第三行到最后一行，是主函数。主函数是该程序的主体部分，由其说明部分

```
int main()
```

和用一对花括号{}括起来的函数体构成。

通常，一个C++ 程序包含一个或多个函数，但其中有且只有一个 main()函数，C++ 程序的执行就是从 main()函数开始的。main()函数左边的关键字 int 表示 main()函数返回一个整数值，这是和程序的倒数第二行的 return 0 对应的，有关函数的进一步内容，将在第 6 章介绍，这里只要记住每个程序中都要包含这样的语句就行了。

在函数体内，有两条以分号结束的语句：第一条是输出语句，用于将字符串显示在计算机屏幕上；第二条是 return 语句，它放在函数的末尾，数值 0 表明程序运行成功。

1.2　算法与程序

一般而言，在工程实践中所遇到的问题要比上面的例子要复杂得多，要编写用于解决应用问题的程序，首先必须要彻底了解问题并认真设计解决问题的方案，也就是算法。例如，对于问题：给定两个正整数 p 和 q，如何求出其最大公因数？

古希腊数学家欧几里德（Euclid）给出了一个著名的算法：

步骤 1：如果 $p < q$，交换 p 和 q；

步骤 2：求 p/q 的余数 r；

步骤 3：如果 $r = 0$，则 q 就是所求的结果；

否则反复做如下工作：

令 $p = q$，$q = r$，重新计算 p 和 q 的余数 r，直到 $r = 0$ 为止，

则 q 就是原来的两正整数的最大公因数。

从原则上说，有了算法，人们就可以借助纸、笔等工具直接求解问题了。但如果问题比较复杂，计算步骤很多，则应通过编程，使用计算机解决。

例 1-2　使用欧几里德算法，编写一程序求解任意两个正整数的最大公因数。

程序

```
// Example 1-2: 计算两个正整数的最大公因数
#include <iostream.h>                          //包含基本输入/输出库文件
int main()
{
    // 说明 3 个整型变量 p,q,r
    int p, q, r;
    // 提示用户由键盘输入两个正整数
    cout << " Please input two integer numbers: " << endl;
    cin >> p >> q;
    // 如果 p < q,交换 p 和 q
    if(p < q)
    {
        r = p;
        p = q;
        q = r;
```

```
    }
    // 计算 p 除以 q 的余数 r
    r = p%q;
    // 只要 r 不等于 0,重复进行下列计算
    while(r ! = 0)
    {
        p = q;
        q = r;
        r = p%q;
    }
    // 输出结果
    cout << " The maximum common divisor is " << q << " . " << endl;
    return 0;
}
```

输入与输出

```
Please input two integer numbers:
12 18
The maximum common divisor is 6.
```

分析 可以看出,该程序的主体部分几乎与原算法完全相同,只是增加了一些C++ 语言特有的内容。

从本例也可以看出,在函数体内,除了注释行以外,还有说明语句和执行语句,说明语句用于说明程序中使用的变量、函数等的类型和参数,执行语句实际执行某功能。第五行就是一个类型说明语句

```
int p,q,r;
```

说明在该程序中使用了 3 个整型变量。关于类型和变量等概念,将在第 3 章中详细介绍。

就一般的计算机程序而言,总是要包括 3 个基本内容:数据输入、计算和输出。

该程序的输入部分首先在计算机屏幕上显示一行提示信息(程序第八行):

```
Please input two integer numbers:
```

然后等待使用者由键盘输入两个整数。用户输入的两个正整数由语句

```
cin >> p >> q;
```

分别存入变量 p 和 q。

程序下面的部分就是欧几里德算法的具体实现了。可以看出,C++ 程序类似自然语言,只是结构更加严谨,对照前面的算法不难理解。

程序最后将计算结果显示在计算机屏幕上。

1.3 C++ 程序的基本要素

1.3.1 标识符、关键词和标点符号

标识符是程序中变量、类型、函数和标号的名称,它可以由程序设计者命名,也可以由系统指定。标识符由字母、数字和下划线"_"组成,第一个字符不能是数字。与 FORTRAN 和 BASIC

等程序设计语言不同,C++ 的编译器把大写和小写字母当做不同的字符,这个特征称为“大小写敏感”。各种C++ 编译器对在标识符中最多可以使用多少个字符的规定各不相同,ANSI 标准规定编译器应识别标识符的前 6 个字符。在标识符中恰当运用下划线、大小写字母混用以及使用较长的名字都有助于提高程序的可读性。

在C++ 中,有些标识符具有专门的意义和用途,它们不能当做一般的标识符使用,这些标识符称为关键词,主要有:

asm, auto, bad _ cast, bad _ typed, bool, break, case, catch, char, class, const, const _ cast, continue, default, delete, do, double, dynamic _ cast, else, enum, except, extern, explicit, false, finally, float, for, friend, goto, if, inline, int, long, mutable, namespace, new, operator, private, protected, public, register, reinterpret _ cast, return, short, signed, sizeof, static, static _ cast, struct, switch, template, this, throw, try, type _ info, typedef, typeid, union, unsigned, using, virtual, void, volatile, while

关键词用于表示C++ 本身的特定成分,具有相应的语义。程序员在命名变量、数组和函数的名称时,不能使用这些标识符。

另外,C++ 还使用了下列 12 个标识符作为编译预处理的命令单词:

define, elif, else, endif, error, if, ifdef, ifndef, include, line, progma, undef

并赋予了特定含义。程序员在命名变量、数组和函数时也不要使用它们。

在C++ 字符集里,标点和特殊字符有各种用途,包括组织程序文本、定义编译器或编译的程序的执行功能等。有些标点符号也是运算符,编译器可从上下文确定它们的用途。C++ 的标点和特殊字符有

! % ^ & * ( ) - + = { } | ~

[ ] \ ; ' : " < > ? , . / #

这些字符在C++ 中均具有特定含义。

1.3.2 注释

注释是一种非常重要的机制,没有恰当注释的程序不是一个好程序。

C++ 的注释有两种形式:

(1) //用于单行注释。它以两个斜杠符起头,直至行末。

(2) /* … */用于多行注释。它可以是用斜线星号组合括起的任意文字(注意:这种注释不能互相嵌套)。

注释可以出现在空白符允许出现的任何地方,但习惯上将注释和其所描述的代码相邻,一般可以放在代码的上方或右方,不放在下方。编译器会把注释作为一个空白字符处理。注释应当准确清晰,不要有二义性。恰当使用注释可以使程序容易阅读。

1.3.3 源程序

一个C++ 源程序由一个或多个源文件构成。C++ 源程序中包括命令、编译指示、说明、定义、语句块和函数等内容。为了使程序的结构清晰,通常的做法是在一个源文件中放置变量、类型、宏和类等的定义(称为头文件,后缀为 .h),然后在另一个源文件说明引用这些变量(称为源程序文件,后缀一般为 .cpp)。采用这种方式编写的程序,很容易查找和修改各类定义。

1.3.4 文件包含

文件包含是编译预处理命令中的一种,它是指一个程序将另一个指定文件的内容包含进来,即将另一个程序文件在编译时嵌入到本文件中。文件包含操作的一般格式为

```
# include <文件名>
```

或者

```
# include "文件名"
```

其中"文件名"是指被嵌入的C++ 源程序文件的文件名,必须用双引号或者尖括号(实际上是一个小于号和一个大于号)括起来。通过使用不同的符号可以通知预处理程序在查找嵌入文件时采用不同的策略。如果使用了尖括号,那么预处理程序在系统规定的目录(通常是在系统的 include 子目录)中查找该文件。如果使用双引号,那么编译预处理程序首先在当前目录中查找嵌入文件,如果找不到则再去由操作系统的 path 命令所设置的各个目录中去查找。如果仍然没有查找到,最后再去上述规定的目录(include 子目录)中查找。

原来的源程序文件和用文件包含命令嵌入的源程序文件在逻辑上被看成是同一个文件,经过编译后生成一个目标代码文件,如图 1.1 所示。

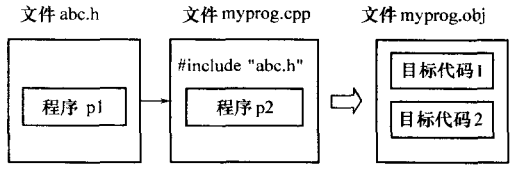


图 1.1 文件包含

1.3.5 输入与输出

程序通常会要求用户提供一些数据,这一过程被称为程序的输入。程序通常也要向用户发出一些数据(如计算结果等),这一过程被称为程序的输出。C++ 程序的输入操作可由系统提供的 cin 标准流对象来完成,而输出操作则可由 cout 标准流对象来完成。

cin 表示输入流对象,它是输入/输出流库的一部分,与 cin 相关联的设备是键盘,其基本用法为

$$cin >> V_1 >> V_2 >> \cdots >> V_n;$$

其中">>"称做提取运算符,$V_1, V_2, \cdots, V_n$ 都是事先定义好的变量。这个语句的执行功能是,程序暂时中止执行,等待用户从键盘上输入数据。用户输入了所有的数据后,应以回车键表示输入结束,程序将用户键入的数据形成输入数据流,用提取运算符 >> 将该数据流存储到各个变量中,并继续执行后续的语句。例如:

```
cin >> p >> q;
```

就是要求用户分别为变量 p 和 q 各输入一个数据。在输入时,应注意用空格或 Tab 键将所输入的数据分隔开。

应说明的是,当用户响应 cin 的要求输入数据时,必须注意所输入数据的类型(C++ 的数据类型将在第 3 章中介绍)应与接受该数据之变量的类型相匹配,否则输入操作将会失败或者得到的将是一个错误的数据。