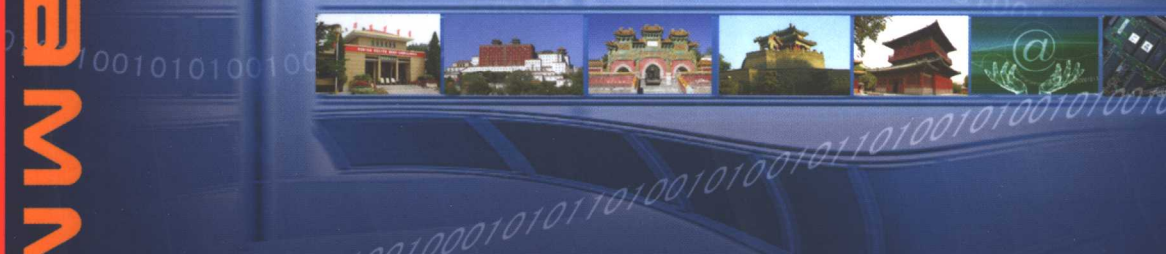
全国普通高校适用教材

C/C++ 程序设计

PROGRAMMING

柴 欣 主编



中国科学技术出版社

全国普通高校适用教材

C/C++ 程序设计

柴 欣 主编

中国科学技术出版社

·北 京·

图书在版编目（CIP）数据

C/C++程序设计/柴欣等主编. —北京：中国科学技术出版社，2006.5
全国普通高校适用教材
ISBN 7-5046-4310-6

Ⅰ.C... Ⅱ.柴... 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022047号

中国科学技术出版社出版
北京市海淀区中关村南大街16号 邮政编码：100081
电话：010-62103210 传真：010-62183872
科学普及出版社发行部发行
北京长宁印刷有限公司印刷
开本：787毫米×1092毫米 1/16 印张：24.25 字数：480千字
2006年4月第1版 2006年4月第1次印刷
定价：26.00元

《C/C++程序设计》

编 委 会

— 序 —

社会发展，科技进步，信息社会初见端倪，预示着人类经济社会生活将发生新的巨大变化，信息越来越成为社会各领域中最为活跃、最有决定性的因素之一。以信息获取能力、信息利用能力和信息甄别能力为主要内涵的信息素养是信息社会中人的综合素质的重要组成部分。信息素养已成为每个社会成员的基本生存能力，更是学习化社会及终身学习的必备素质。

高校是为各级部门输送高级专门人才的重要阵地，培养大学生的信息素养是高等教育的一项重要任务，而且在社会信息化的今天，也是当务之急。信息素养教育在技术层面上主要包括信息知识教育和信息能力教育。信息知识包括信息本身的定义、特点以及信息的测量。有了对信息本身的认知，就能更好的辨别信息，获取、利用信息。信息知识是信息素养教育的基础。信息能力包括信息获取能力、信息加工处理能力和信息技术的利用能力等。这是我们快速、准确地获取信息、加工信息和传播信息的关键所在，也是我们开展计算机基础教育的唯一归宿。计算机基础教育系列教材正是为实现上述目标而编写的。

基于信息技术的快速发展以及国家教育部关于开展技术及基础教学改革的指导思路，我们确定这套系列教材的编写计划与编写体系。教材是有效开展计算机基础教学的首要问题，也是教学过程中的“剧本”。本系列教材编写计划的制定、编写和出版，凝聚了编委会、作者和出版发行部门的心血，是大家多年来在计算机基础教学与研究的成果的体现，呈现出以下主要特点：

1. 内容先进。本系列教材注重将信息技术、计算机技术以及教学研究和科学研究的最新理论、最新成果和最新发展适当地引入教材中来，保持了教材内容的先进性。

2. 适应面广。本系列教材以国家教育部计算机基础教育教学改革要求为依据，兼顾了理、工、农、医、经、管、法、文等各种类型专业教材的要求。本系列教材也适合高职、高专类院校选用。

3. 立体配套。为了适应教学模式、教学方法和教学手段的改革，本系列教材除了文字教材这一形式外，有些教材还配有习题解和上机指导、多媒体电子教案、CAI 课件以及相应网络教学资源库。我们还准备陆续制作并开通相关课程的教学网站，以利于学生自学。

总之，本系列教材的指导思想是内容新颖、概念清晰、先进实用、形式多样。这既是我们多年来在教学实践中逐步形成的创作风格，也是计算机基础教学的必然选择。然而教材建设是一项长期艰巨的系统工程，尤其是计算机科学技术发展迅速、内容更新快，为使我们的教材能够与技术发展同步，我们将密切关注信息技术、计算机技术发展新动向，以使我们的教材编写在内容上不断推陈出新、体系上不断完善成熟、形式上更加新颖实用，不断适应计算机基础教育的需要。

本系列教材得到了教育部高等学校计算机基础教学指导委员会的肯定，并成为其优先推荐教材之一。

计算机基础教育系列教材编委会

2006 年 4 月

目　录

第1章 绪 论

1.1 计算机程序设计概述

1.1.1 程序设计语言的发展

计算机之所以能自动进行计算，是因为采用了程序存储的原理，计算机的工作体现为执行程序。程序是控制计算机完成特定功能的一组有序指令的集合，编写程序所使用的语言称为程序设计语言，它是人与计算机之间进行信息交流的工具。

从 1946 年世界上诞生第一台计算机起，在短短的 60 余年间，计算机技术迅速发展，程序设计语言的发展从低级到高级，经历了机器语言、汇编语言、高级语言到面向对象语言的多个阶段，具体过程如下：

⑴ 机器语言。计算机能够直接识别和执行的二进制指令（也称机器指令）的集合称为该种计算机的机器语言。早期的计算机程序都是直接使用机器语言编写的，这种语言使用 0、1 代码，因此编写出的程序难以理解和记忆，目前已不被人们使用。

⑵ 汇编语言。通过助记符代替 0、1 机器指令以利于理解和记忆，由此形成了汇编语言。汇编语言实际上是与机器语言相对应的语言，只是在表示方法上采用了便于记忆的助记符号来代替机器语言相对应的二进制指令代码，因此也称为符号语言。计算机不能直接识别汇编语言，需要编译后才能识别。这种语言的执行效率较高，但由于难以记忆，因此使用较少。

⑶ 高级语言。机器语言和汇编语言是面向机器的语言，高级语言采用更接近自然语言的命令或语句，使用高级语言编程，一般不必了解计算机的指令系统和硬件结构，只需掌握解题方法和高级语言的语法规则，就可以编写程序。高级语言在设计程序时着眼于问题域中的过程，因此它是一种面向过程的语言，对于高级语言，人们更容易理解和记忆，这也给编程带来很大方便，但它与自然语言还是有较大差别。

⑷ 面向对象语言。面向对象语言是比面向过程语言更高级的一种高级语言。面向对象语言的出现改变了编程者的思维方式，使设计程序的出发点由着眼于问题域中的过程转向着眼于问题域中的对象及其相互关系，这种转变更加符合人们对客观事物的认识。因此，面向对象语言更接近于自然语言，面向对象语言是人们对于客观事物更高层次的抽象。

目前世界上已经设计和实现的计算机语言有上千种之多，但实际被人们广泛使用的计算机语言不过数十种。

1.1.2 程序设计的发展历程

回顾程序设计发展的历史，大体上可以划分为如下几个不同的时期。

20 世纪 50 年代的程序都是用指令代码或汇编语言来编写的，这种程序的设计相当麻烦，编制和调试一个稍许大一点的程序常常要花费很长的时间，培养一个熟练的程序员更需经过长期的训练和实习，这种局面严重影响了计算机的普及应用。

20 世纪 60 年代高级语言的出现大大简化了程序设计，缩短了解题周期，因此显示出强大的生命力。此后，编制程序已不再是软件专业人员才能做的事了，一般工程技术人员花上较短的时间学习，也可以使用计算机解题。这个时期，随着计算机的应用日益广泛地渗透到各学科和技术领域，也发展了一系列不同风格的、为不同对象服务的程序设计语言。其中较为著名的有 FORTRAN、COBOL、ALGOL、LISP、PL/1、PASCAL 等十几种语言。高级语言的蓬勃兴起，使得编译和形式语言理论相应日趋完善，这是该时期的主要特征。但就整个程序设计方法而言，并无实质性的改进。

自 20 世纪 60 年代末到 70 年代初，出现了大型软件系统，如操作系统、数据库，这给程序设计带来了新的问题。大型系统的研制需要花费大量的资金和人力，可是研制出来的产品却是可靠性差，错误多，且不易维护和修改。一个大型操作系统有时需要几千人/年的工作量，而所获得的系统又常常会隐藏着几百甚至几千个错误。当时，人们称这种现象为“软件危机”。“软件危机”震动了软件界，程序设计的传统习惯和工作方式导致了不清晰的程序结构，使得程序的可靠性难以保障；另一方面，程序设计工具的严重缺乏也使得大型系统的开发陷入困境。此时人们开始重新审视程序设计中的一些最基本的问题。例如，程序的基本组成部分是什么？应该用什么样的方法来设计程序？如何保证程序设计正确？程序设计的主要方法和技术应如何规范等等。

1969 年，E. W. Dijkstra 首先提出了结构化程序设计的概念，他强调从程序结构和风格上来研究程序设计。经过几年的探索和实践，结构化程序设计的应用确实取得了成效，用结构化程序设计的方法编写出来的程序不仅结构良好，易写易读，而且易证明其正确性。

到 20 世纪 70 年代末结构化设计方法得到了很大的发展，Niklans Wirth 又提出了“算法+数据结构 = 程序设计”的程序设计方法，他将软件划分成若干个可单独命名和编址的部分，它们被称为模块，模块化使软件能够有效地被管理和维护，能够有效地分解和处理复杂问题。在 80 年代，模块化程序设计方法普遍被人们接受。

虽然几十年来结构化程序设计技术得到了广泛的使用，但有些问题仍未得到很好的解决。由于软件开发是对问题的求解过程，从认识论角度看，软件开发过程包括人们对要解决问题及相关事物的认识和基于认识所进行的描述。而结构化设计方法不能直接反映出人类认识问题的过程。另外，结构化设计方法中，程序模块和数据结构是松散地耦合在一起的，因此，当应用程序比较复杂时，容易出错，难以维护。随着计算机软件的发展，软件系统越来越复杂庞大，结构化程序设计方法已显得力不从心。

在 20 世纪 80 年代，人们又提出了面向对象的程序设计方法。这种方法直接对问题域

中的客观事物建造分析模型中的对象，使得对象的描述与客观事物一致，保持问题域中的单个事物及事物之间的关系的原貌，而面向对象的语言可以直接描述问题域中的对象及其相互关系。用面向对象的程序设计方法，可以使人们对复杂系统的认识过程与系统的程序设计与实现过程尽可能地一致，这是因为它从客观世界中所存在的事物出发，比较符合人们的思维方式。因此，目前这种面向对象的程序设计模式正逐渐取代“数据结构+算法”的面向过程的程序设计模式。

1.1.3 结构化程序设计概述

自提出结构化程序设计的概念后，经过十几年的发展，结构化程序设计已经具有了很广泛的内容，大体上可以归纳为以下几点。

1. 结构化程序的基本结构

结构化程序包含有三种基本结构，人们可以用这三种基本结构来展开程序，表示一个良好的算法，从而使程序的结构清晰、易读易懂且质量好、效益高。这三种基本结构为顺序结构、 选择结构和循环结构。

(1) 顺序结构

顺序结构是一种最简单、最基本的结构，在顺序结构内，各块是按照它们出现的先后顺序依次执行。图 1.1 表示了一个顺序结构形式，从图中可以看出它有一个入口 a 点，一个出口 b 点，在结构内 A 框和 B 框都是顺序执行的处理框。

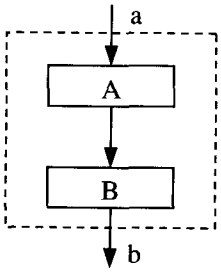


图 1.1 顺序结构示意图

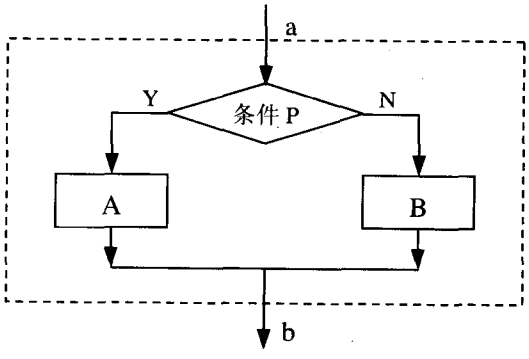


图 1.2 选择结构流程图

(2) 选择结构

选择结构中包含一个判断框，根据给定的条件 p 是否成立而选择执行 A 框或 B 框，当条件成立时，执行 A，否则执行 B。A 框或 B 框可以是空框，即不执行任何操作，但判断框中的两个分支，执行完 A 或 B 后都必须汇合在一起，从出口 b 退出，然后接着执行其后的过程。图 1.2 所示的虚线部分就是选择结构，在选择结构中程序产生了分支，但对于整个的虚线框而言，它仍然只具有一个入口 a 和一个出口 b。

(3) 循环结构

循环结构又称重复结构，是指在一定条件下反复执行一个程序块的结构。循环结构也是只有一个入口，一个出口。根据循环条件的不同，循环结构分为当型循环结构和直到型循环结构两种。

当型循环的结构如图 1.3，其功能是：当给定的条件 p 成立时，执行 A 框操作，执行完 A 操作后，再判断 p 条件是否成立，如果成立，再次执行 A 操作，如此重复执行 A 操作，直到判断 p 条件不成立才停止循环。此时不执行 A 操作，而从出口 b 脱离循环结构。

直到型循环的结构如图 1.4，其功能是，先执行 A 框操作，然后判断给定条件 p 是否成立，如果不成立，再次执行 A 操作；然后再对 p 进行判断，如此反复，直到给定的 p 条件成立为止。此时不再执行 A 框，从出口 b 脱离循环。

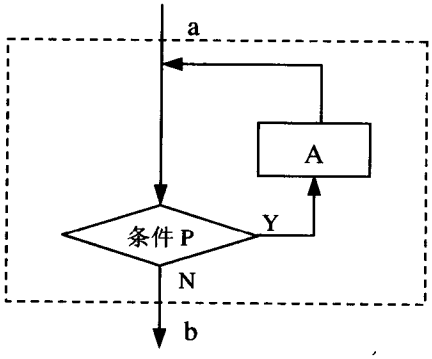


图 1.3 当型循环结构流程图

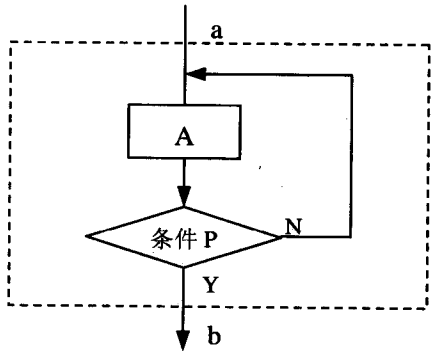


图 1.4 直到型循环结构流程图

由以上三种基本结构构成的程序，称为结构化程序。一个结构化程序，以及三种基本结构中的每一种结构都应具有以下特点：

- 有一个入口；
- 一个出口；
- 没有死语句，即每一个语句都应该有一条从入口到出口的路径通过（至少通过一次）；
- 没有死循环（无限制的循环）。

实践证明，任何满足以上四个条件的程序，都可以表示为由以上三种基本结构所构成的结构化程序；反之，任何一个结构化程序都可以分解为一个个基本结构。

2. 结构化程序的设计方法

结构化程序主要采用自上而下、逐步细化的设计方法，即先全局后局部、先整体后细节、先抽象后具体的设计方法。由于实际问题往往比较复杂，为了提高效率，在进行结构化程序设计时，常常伴随着使用关键部分优先考虑的设计方法。

3. 结构化程序的问题

结构化程序设计技术虽已使用了几十年，但如下问题仍未得到很好的解决：

(1) 面向过程的设计方法与人们习惯的思维方法仍然存在一定的差距，所以很难自然、准确地反映真实世界。因而用此方法开发出来的软件，有时很难保证其质量，甚至需要进行重新开发。

(2) 结构化程序设计在方法实现中只突出了实现功能的操作方法（模块），而被操作的数据（变量）处于实现功能的从属地位，即程序模块和数据结构是松散地耦合在一起的。因此当应用程序比较复杂时，容易出错，难以维护。

1.1.4 面向对象程序设计概述

1. 面向对象方法的起源

由于结构化程序设计的缺陷，结构化程序设计方法已不能满足现代化软件开发的要求，一种全新的软件开发技术应运而生，这就是面向对象的程序设计（Object Oriented Programming，简称 OOP）。

20 世纪 80 年代，在软件开发中各种概念和方法积累的基础上，就如何超越程序的复杂性障碍，如何在计算机系统中自然的表示客观世界等问题，人们提出了面向对象的程序设计方法。面向对象的方法不再将问题分解为过程，而是将问题分解为对象。对象将自己的属性和方法封装成一个整体，供程序设计者使用。对象之间的相互作用则通过消息传递来实现。用面向对象的程序设计方法，可以使人们对复杂系统的认识过程与系统的程序设计与实现过程尽可能地一致。目前，这种“对象+消息”的面向对象的程序设计模式正逐渐取代“数据结构+算法”的面向过程的程序设计模式。

2. 面向对象语言的发展

早在 20 世纪 60 年代，就出现了最早的面向对象语言 Simula67 语言，具有了类和对象的概念。随后又推出了纯面向对象设计语言，如 80 年代美国 Xerox Palo Alto 研究中心推出了 Smalltalk，它完整地体现并进一步丰富了面向对象的概念，还开发了配套的工具环境，为其最终实用化奠定基础。但由于当时人们已经接受并广泛应用结构化设计理论，而不能一下子完全接受面向对象程序设计的所有思想等诸多原因，这些语言并没有能够广泛流行起来。后来人们开始对流行的语言进行面向对象的扩充，曾经推出过许多种类的版本，而主要成功的代表是在当时流行的程序设计语言 C 语言的基础上开发的 C++语言。这时面向对象语言已形成几大类别：一类是纯面向对象的语言，如 Smalltalk 和 Eiffel；另一类是混合型的面向对象语言，如 C++和 Objective C；还有一类是与人工智能语言结合形成的，如 LOOPS、Flavors 和 CLOS；适合网络应用的有 JAVA 语言等。

1.2 C/C++概述

1.2.1 C/C++的发展过程

C 语言的祖先是 ALGOL60 语言，ALGOL60 语言是 1960 年由国际计算机委员会设计的一种面向过程的高级语言。但它不能直接对硬件进行操作，不宜用来编写系统程序。

1963 年，英国剑桥大学和伦敦大学将 ALGOL60 发展成 CPL（Combined Programming Language）语言，该语言已经比较接近于硬件，但规模较大，难以实用。

1967 年，剑桥大学的 Martin Richards 将 CPL 改制成 BCPL（Basic Combined Programming Language），BCPL 比 CPL 大为简化，既具有结构化程序设计语言的特点，也能够直接处理与硬件相关的数据，被软件人员用作系统程序的描述语言。

1970 年，美国贝尔实验室的 Ken Thompson 将 BCPL 修改成 B 语言，并用 B 语言记叙和开发了第一个高级语言 UNIX 操作系统。

1972 年，Ken Thompson 在 UNIX 系统上的合作者 Dennis M. Ritchie 将 B 语言修改设计成 C 语言。C 语言既保持了 BCPL 和 B 语言的精练和接近于硬件的特点，也克服了它们过于简单、数据无类型等缺点。1973 年，Ken Thompson 和 Dennis M. Ritchie 又合作将 1969 年用汇编语言编写的 UNIX 操作系统改用 C 语言编写，其中 C 语言代码占 90%以上，只保留了少量汇编语言的代码。这样就使得 UNIX 操作系统向其他类型的机器上移植变得相当简单。

由上面介绍可以看出，当初 C 语言是为了开发 UNIX 操作系统而研制的，它随着 UNIX 的出名而闻名。随着微型计算机的普及，出现了较多版本的 C 语言系统。尽管 C 语言源程序兼容性很强，但由于没有统一的标准，各种版本之间还是会略有差异。为此，美国国家标准局于 1983 年制定了 C 语言标准，这个标准不断完善，并从 1987 年开始实施，称为 ANSI C。目前在微机上运行的 C 语言版本 Microsoft C，Turbo C，Quick C 等都是与 ANSI C 兼容的版本。

C 语言是一种面向过程的语言，进入 80 年代后，随着面向对象的程序设计概念的日益普及，C 语言在发展的同时，也朝着支持面向对象的编程语言的方向迈出了步伐。1986 年美国 AT&T 公司的贝尔研究所推出了 C 语言的超集——C++语言。

C++语言是 C 语言的扩展，其基础部分除了一些细微的差别外，可以说是 C 语言的超集，它保留了 C 语言功能强、效率高、风格简洁、适合于大多数的系统程序设计任务等优点，使得 C++与 C 之间取得了兼容性，因此，在过去的软件开发中积累的大量的 C 的库函数和实用程序都可在 C++中应用。

另外，C++语言通过对 C 的扩充，克服了原有 C 语言的缺点，特别是引进了面向对象语言的要素，使得 C++语言成为一种面向对象的程序设计语言，它对面向对象程序设计方