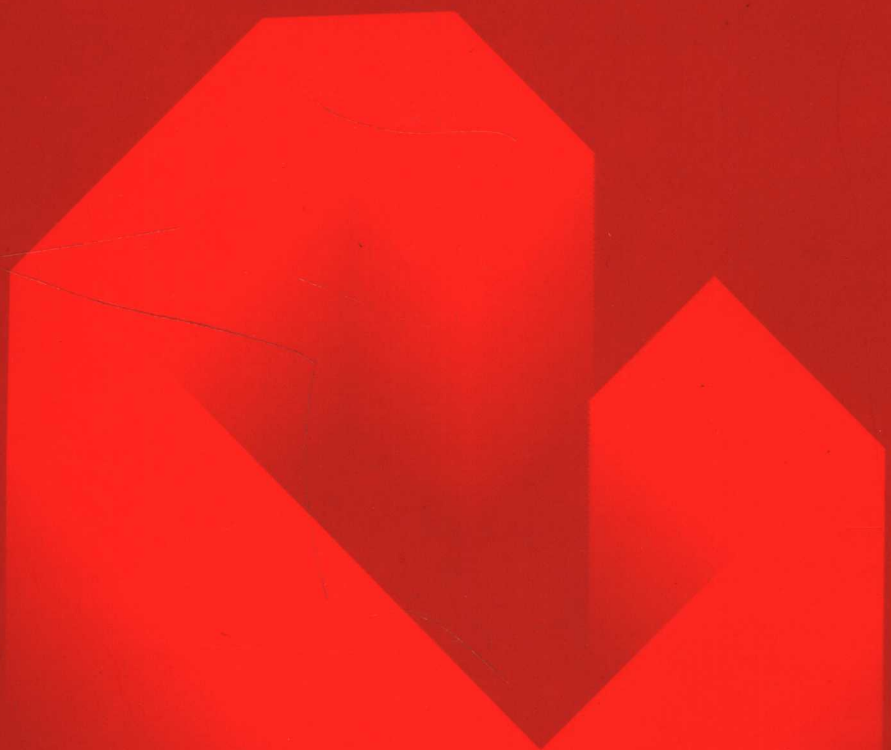
C语言程序设计

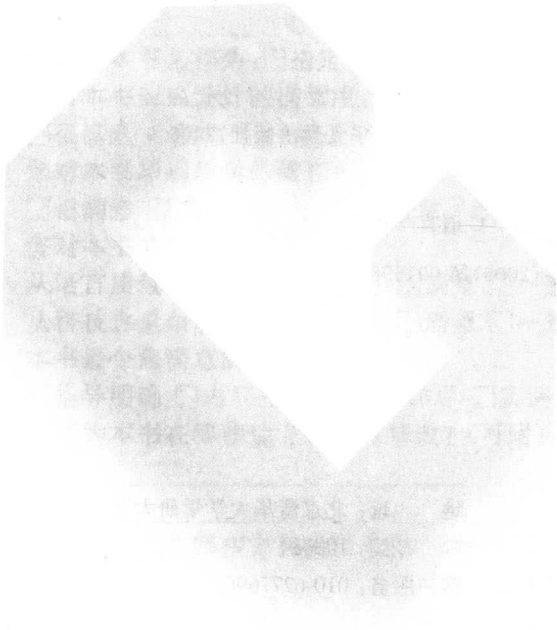
黄保和　江弋　编著



清华大学出版社

C语言程序设计

黄保和　江弋　编著



清华大学出版社
北京

内 容 简 介

本书是一本为高等学校非计算机专业“C 语言程序设计”课程编写的教材。本书有以下特点：(1)突出重点而不是面面俱到，重点介绍基本的、常用的语法，忽略不常用的语法。(2)注重程序设计语言的共性，让学生掌握语言比较学的方法，培养学生自学其他程序设计语言的能力。(3)简单介绍了 C++ 的面向对象的程序设计方法，为学生今后过渡到面向对象的程序设计留下接口。(4)不从理论上讲程序设计。本书以介绍 C 语言的语法为线索，通过一批实例分析，将程序设计的一般方法和技术贯穿其中。

本书也可作为相关科技和工程技术人员学习 C 语言的参考书。

图书在版编目(CIP)数据

C 语言程序设计/黄保和，江弋编著. —北京：清华大学出版社，2006.9

ISBN 7-302-13599-1

Ⅰ. C…　Ⅱ. ①黄… ②江…　Ⅲ. C 语言－程序设计　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2006)第 091698 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦

http://www.tup.com.cn　　**邮　　编**：100084

社 总 机：010-62770175　**客户服务**：010-62776969

组稿编辑：陈国新

文稿编辑：刘　彤

印 刷 者：北京国马印刷厂

装 订 者：三河市化甲屯小学装订二厂

发 行 者：新华书店总店北京发行所

开　　本：185×260　**印张**：21.5　**字数**：534 千字

版　　次：2006 年 9 月第 1 版　2006 年 9 月第1 次印刷

书　　号：ISBN 7-302-13599-1/TP · 8209

印　　数：1～4000

定　　价：28.00 元

前　言

目前，我国高校非计算机专业的计算机教育普遍实施“三个层次”的教学方式，即计算机应用基础、计算机技术基础和结合专业的计算机应用。“C 语言程序设计”属计算机技术基础课程。

因为受到课时数不足、配套课程不够等客观因素的制约，任课老师普遍感到要在非计算机专业的学生中上好“C 语言程序设计”课程不容易，学生也普遍感到学好该课程有一定难度。为了使课程教学达到预期效果，我们认为应该明确并解决好以下问题。

(1) 课程目的。程序设计是计算机技术在各行各业应用的基础。对非计算机专业的学生，在今后的工作中不一定要自己开发应用程序，但一定要会使用计算机程序，也有可能与计算机专业人员合作开发本专业领域的应用程序。因此，学习程序设计的一般原理，掌握基本的程序设计方法和技术具有重要的现实意义。“C 语言程序设计”课程的目的就是帮助学生掌握一种高级程序设计语言(C 语言)，并通过运用 C 语言开展程序设计实践，使学生具备应用高级语言开展程序设计的初步能力。

程序设计必须借助于某种语言。“C 语言程序设计”课程必然应包含“C 语言”和“程序设计”两方面的内容。必须明确，学习程序设计的一般方法和技术是本课程目的，学习 C 语言只是为了实现程序设计，C 语言是程序设计的工具语言。本课程将通过对 C 语言的词法、语法介绍，通过各种基本控制结构的实际案例分析，向学生介绍程序设计的基本方法。通过上机实验，使学生掌握程序调试和测试方法。通过本课程的学习，应使学生应用计算机解决问题的能力得到进一步的提高，为后续的计算机应用课程打下坚实的基础。

(2) 课程内容。对程序设计语言和程序设计技术的熟练掌握和深入理解是计算机系学生的两大基本功。“C 语言程序设计”课程是计算机专业学生 4 年中要学习的系列计算机课程的第一门，属专业基础课，其重要性不言而喻，因此一般安排比较充足的教学课时。况且，计算机系学生还应学习一系列计算机课程，如“数据结构”、“汇编语言”、“编译原理”、“程序设计语言理论”、“算法设计与分析”、“程序设计方法学”、“形式语义学”、“软件工程”等。因此，计算机系学生学习“C 语言程序设计”课程时，主要的课程内容是深入地研究 C 语言的词法和语法，并通过程序设计练习进一步理解程序设计语言。其教学的重点是程序设计语言，程序设计技术是副产品。

在非计算机专业学生中开设“C 语言程序设计”课程，不能全盘复制计算机专业学生的课程内容。一方面课时不够，另一方面应该对本课程的内容做适当的外延。

历史上先后产生了多种高级程序设计语言，而且新的高级程序设计语言也在不断出现。C 语言是目前使用较普遍的一种高级程序设计语言，虽然 C 语言的内容比较庞杂、琐碎，语言描述本身也不如 Pascal 规范，但考虑到它具有较广泛的实用性，比较容易和后续的计算机应用课程衔接，所以笔者认为使用 C 语言作为程序设计的工具语言是合适的。

本课程的基本教学内容应包括：了解软件开发的一般过程，软件开发和软件运行平台的使用，软件工程的基本概念，C 语言的基本词法和语法，结构化程序设计的方法，模块和函

数的概念，顺序、分支和循环3种基本程序控制结构的概念和实现，程序调试和测试的一般方法，面向对象程序设计方法的概念等。

考虑到能提供给本课程的学时有限、学生在学习本课程之前数学基础和专业知识还较欠缺，加上在一、二年级其他功课压力较重等客观因素，具体安排课程内容时应注意以下几点：

(1) 介绍程序设计语言时应有重点，不要面面俱到。C语言语法比较庞杂，有些语句可以相互替代，有些语法不常使用。课程中要重点介绍基本的、常用的语法，忽略重复的、不常用的语法。

(2) 注重程序设计语言的共性。学生以后不一定用C语言写程序，而本课程又不可能介绍所有的计算机语言。所以应该介绍计算机程序设计语言共性的东西，让学生掌握语言比较学的方式方法，使学生具有自学其他程序设计语言的能力。

(3) 应该介绍面向对象的程序设计方法。面向对象的程序设计是今后程序设计的方向，C语言是面向过程的程序设计语言，并不支持面向对象的程序设计方法。但C++是面向对象的程序设计语言，课程中应简单介绍C++的面向对象的内容，为学生今后顺利过渡到面向对象的程序设计留下接口。

(4) 不要从理论上讲程序设计。由于课时的限制，不能安排太多的时间专门讲授程序设计理论，而应以介绍语言为线索，通过一批实例分析，将程序设计的一般方法和技术传授给学生。

长期的计算机基础教学实践，长期的计算机基础教学研究，是本书形成与完善的基础。笔者希望奉献给老师和同学一本适合教、适合学的教材。

本书由江弋和黄保和共同编写，其中江弋编写前6章，黄保和编写后6章。编写期间，得到福建省计算机基础教育研究会的大力支持，得到厦门大学公共计算机教学部各位老师的大力支持，借此表示感谢。

使用本书过程中，如发现错误，恳请与黄保和联系(huangbh@xmu.edu.cn)。

编 者

2006年7月于厦门大学

目　录

第1章 绪 论

C语言是当今世界上使用最广泛的程序设计语言之一。本章主要介绍程序设计语言、程序设计、算法的概念及程序设计的主要步骤，同时介绍C语言的产生和发展以及C程序的构成和书写格式等。

1.1 程序设计语言

人们为了用计算机来解决实际问题，一般总是要编写程序。所谓程序，是指以某种程序设计语言为工具对解决问题的动作序列的描述，它表达了人们解决问题的过程，用于指挥计算机进行一系列操作，从而实现问题的解决。程序设计语言就是用户用来编写程序的语言，它是人与计算机进行信息交流的工具。对于理想的程序设计语言来说，所提供的语法应该能够满足描述算法、数据结构等方面的需要。然而用任何一种高级语言编写的程序(称其为源程序)都要通过编译程序翻译成机器语言程序(称其为目标程序)后计算机才能执行，或者通过解释程序边解释边执行。

程序设计语言是计算机软件系统的重要组成部分，就其发展过程而言，一般可分为机器语言、汇编语言和高级语言。

1.1.1 机器语言

最初的程序是用机器语言编写的，它包括了很多的基本指令，这些机器指令可以由机器直接执行。机器语言编写的程序是由二进制代码组成的代码序列。使用机器指令进行程序设计要求程序设计者具有深入的计算机专业知识，对机器的硬件有充分的了解。这种程序的可读性差，而且由于不同机器的机器指令不同，因此程序的可移植性差，所编写的程序只能在相同的硬件环境下使用，大大地限制了程序的通用性。另外，用机器语言描述问题的处理方式也与人们习惯的思维方式有较大差距。

1.1.2 汇编语言

为了便于记忆，人们很自然地用助记符号来代表机器语言中的01代码，这种用助记符号描述的指令系统称为汇编语言。为了把汇编语言编写的程序转换为机器语言程序，人们开发出了称为“汇编程序”的翻译程序。汇编语言指令与机器语言指令是一一对应的，因此，汇编语言也是与具体计算机硬件相关的。汇编语言除了可读性比机器语言好外，同样也存在机器语言的缺点，尤其是描述问题的方式与人们习惯相差太远。

1.1.3 高级语言

为了提高程序开发的效率，针对机器语言和汇编语言的缺点，20世纪50年代中期，IBM公司的一个编程小组提出了一个设想：能不能使用数学公式来编写人们想要进行的计算，然后再让计算机将这些公式翻译（解释）为机器语言呢？最终该团队于1955年完成了Fortran(Formulation Translation)的最初版本，这是第一个高级编程语言。从那以后，各种高级语言相继涌现，而能引起广泛关注和使用的主要有BASIC、Algol、COBOL 、Pascal、C等。在这些众多的高级语言当中，最为流行，也是最成功的当数C语言，它既可用来写应用软件，又可用来写系统软件（如UNIX操作系统等）。如今时兴的C++和Java也都是基于C语言的。

与低级语言（机器语言、汇编语言）相比，高级语言的表达方式更接近人类自然语言的表述习惯，具有很高的可读性，并且它不依赖于计算机的具体型号，具有良好的可移植性。高级语言的一条语句通常对应于多条机器指令，所以对同一功能的描述，高级语言程序比机器指令程序紧凑得多。

1.2 程序设计的概念

程序设计是一门用计算机解决问题的科学，在掌握程序设计的错综复杂的内容之前，有必要对什么是程序设计有一个感性认识。

1.2.1 程序设计

程序设计的目的就是用计算机解决问题。用计算机解决问题大体上经过两个步骤，首先通过分析问题构造出一个解决问题的算法（即解决问题的方法和步骤）或在解决该问题的备用算法中选择其中之一，这个过程称为算法设计；其次是用一种程序设计语言（如C语言）将该算法表达为程序，这个过程称为编码。程序设计的两个步骤是相辅相成的，作为一个程序设计（或编程）新手，只能从简单的问题入手，而简单问题的解决方法也较简单，所以算法设计阶段不会有什么困难，倒是C语言对于初学者来说是全新的，因此初学程序设计时，编码常常会是两个步骤中较为困难的阶段。但可以确定的是，随着对C语句语法的了解和编程实践的不断增加，编码会变得越来越简单；而与此正好相反的是，随着遇到的问题愈来愈复杂，算法设计会变得更加困难和更具有挑战性。

程序设计既是一门科学，又是一门艺术。就像练习写作一样，你必须不断地编程实践并且大量阅读他人程序，来积累经验，才能形成良好的程序设计风格，而不仅仅只是记住一些规则。只有经过长期编程实践训练，才能不断提高程序设计能力。

1.2.2 程序设计的步骤

对于初学编程的人来说，往往简单地把程序设计理解为编写一个程序，认为能根据实际问题直接用计算机语言编出一个程序就行了，这样理解是不全面的。事实上，程序是程序设

计的最终产品，需要经过中间每一步的细致加工才能得到。如果企图一开始就编写出程序，往往会适得其反，达不到预想的结果。通常应在充分论证数据结构和算法以后才能考虑编写程序，同时编写程序时还需要结合程序设计方法（面向过程的或是面向对象的）和程序设计语言（C语言、C++，Java等）。因此，不提倡一开始就编写程序，特别是对于大型的程序。前面介绍程序设计过程由算法设计和编码两步骤组成只是大略说法，严格地说，程序设计一般应遵循以下步骤。

1. 分析问题

用计算机来解决问题，首先要通过对问题的分析，以便确定在解决这个问题过程中要做些什么？若在没有把所要解决的问题分析清楚之前就贸然开始编制程序，只能起到事倍功半的效果，而且很难得到满意结果。因此，分析问题，弄清楚要解决的问题并给出问题的明确定义是解决问题的关键。

2. 系统设计

在弄清要解决的问题之后，就要考虑如何解决它，即如何做？由于从本质上看，用计算机解决问题的方式就是对数据进行处理，因此，首先要对问题进行抽象，抽取出能够反映本质特征的数据并对其进行描述，即给出数据结构的设计；然后考虑对设计好的数据如何进行操作以获得问题的结果，即进行算法设计。注意，不同的程序设计方式在处理数据结构和算法这两者的关系上是有所区别的。在面向过程程序设计中，把数据结构设计和算法设计分开考虑；而面向对象程序设计是把两者结合成对象来考虑。

3. 用某种程序设计语言编程

在进行数据结构设计和算法设计时，往往采用某种与具体程序设计语言无关的语言（如伪代码或自然语言等）来描述算法。这样做的目的是为了避免一开始就陷入程序设计语言的具体细节中。因为过多地涉及实现细节，不利于从较高抽象层次对问题本质的东西进行考虑，并造成对设计过程难以把握和理解。当然，用伪代码（自然语言或框图）描述的算法是不能被计算机所执行的，必须用具体程序设计语言把它表示出来，即编程实现。因此，用某种程序设计语言编写的程序，本质上也是对问题处理方案的描述，并且是最终的描述。程序文本保存在一个文件或多个文件中。包含程序文本的文件称为源文件，即源程序文件。

4. 测试与调试

源程序文件要经过编译程序把它翻译成等价的目标程序文件，并将一系统列目标文件及库文件链接在一起生成一个可执行文件，程序才能被计算机执行。

程序编写好后，还需要进行调试和测试。只有经过调试的程序才能正式运行。所谓调试，是指找出程序中的错误的位置并改正错误。因此，调试又称查错。而所谓测试，是指利用精心设计的一批测试用例（包括输入数据和与之相应的预期输出结果）来运行程序，看程序的实际运行结果与预期输出结果是否一致，以尽可能多地发现程序中的错误。因此，测试的目的也是为了发现程序中的错误（一般是算法错误而非语法错误），而不是为了证明程序正确。调试和测试往往是交替进行的，通过测试发现程序中的错误，通过调试进一步找出错

误的位置并改正错误。这个过程往往需要重复多次，因此这一步往往是程序设计中最困难的一步。

程序编写好之后，其中可能含有错误。程序的错误通常有3种：语法错误、逻辑错误和运行异常错误。语法错误是指源程序中存在有违反C语法规则的地方，如语句后遗漏了分号“；”或忘记了定义变量等，程序编译时编译器可以发现这类错误，并会给出“出错信息提示”和出错位置，用户可以根据编译器提供的提示信息修改源程序；逻辑错误是指程序没有完成预期的功能，如源程序中错将“c=a%b”写成“c=a/b”或将该用“==”的地方却误用了“=”等，编译程序时，编译器发现不了这类错误，但程序运行时，结果会不正确，甚至根本就无法显示结果。遇到此类错误，用户往往需要通过设置断点，跟踪程序的运行过程等测试手段，才能发现它们。因此程序设计中最致命的错误就是逻辑错误，因为它不太容易被发现，并会导致程序不能正确地解决问题。在程序中找出并修正逻辑错误的过程称为程序调试，它是软件开发中不可缺少的一个环节。俗话说，“三分编程七分调试”，说明程序调试的工作量要比编程的工作量大得多。而善于发现编程中的逻辑错误是通往成功的重要一环。优秀编程者的优秀之处就是他们能尽力将存在于完成程序中的逻辑错误数量减到最少，而不在于他们能够避免犯逻辑错误。运行异常错误是指对程序运行环境的非正常情况考虑不足而导致的程序运行异常终止。逻辑和运行异常错误可能是编程阶段导致的，也有可能是系统设计阶段或问题分析阶段的缺陷。这两类错误一般都要通过测试才能发现。常用的测试方法很多，如静态测试与动态测试。静态测试是不运行程序，而是通过对程序的静态分析，找出逻辑错误。动态测试是利用一些测试数据，通过运行程序观察运行结果是否与预期的结果相符。

需要注意的是，无论采用何种测试手段，都只能发现程序有错，而不能证明程序正确，因为几乎可以肯定，即使程序通过了良好的测试，总还是会有一些隐蔽的错误。一个编程者的任务就是不断找出并改正这些错误，并尽可能地对程序进行彻底测试。在大多情形时，彻底测试可以增加对解决方案正确性的把握。

5. 整理并写出所有的文档资料

有些人认为程序调试成功就万事大吉了，其实这种思想是错误的，因为对于程序设计人员来说，平时的归纳和总结是很重要的。程序员应将平时的源程序和各种文字资料进行归类保存，以便今后查找。最后还要编写使用和维护该程序的说明书，供他人参考。

6. 运行与维护

程序通过测试后就可交付使用了，但程序在使用中还需要不断得到维护。程序维护可分为3种：正确性维护、完善性维护和适应性维护。程序需要维护的原因主要有两个，首先，即使经过大量测试，源代码中依然可能存在逻辑错误，这些错误会在程序的使用过程中不断被暴露；其次，当出现一些不常见的情况或发生之前未预料到的情况时，之前隐藏的异常错误就会使程序运行失败。因此，测试与调试是程序维护的一个基本部分。但它不是最重要的，更为重要的是功能完善。编写程序是为了完成客户需要完成的任务，但程序不可能完成客户想做的每一件事。在一个程序使用了一段时间后，客户会期望程序能做些别的事情，或者用不同的方法做某些事情，或者用更有用的方式显示数据，或者运行得更快一些等。

对于用户提出的完善和扩充程序功能的要求，程序开发者应考虑响应这些要求。无论是要排除逻辑错误还是要完善程序功能，在任何情况下都需要有人查看程序，做出必要的修改，并确保这些修改能使程序更好地工作。

如果对程序的功能进行了较大的更改，应发布程序的新版本。

1.3 C语言的发展和C++简介

1.3.1 C语言发展简述

1972年，为了编写UNIX操作系统，贝尔实验室的D. M. Ritchie设计并实现了C语言。

后来，C语言多次做了改进，逐渐成熟，先后被移植到各种计算机上，现在C语言已成为世界上使用最广泛的程序设计语言之一。C语言在其30多年的发展中，涌现了众多不同的版本，它们都普遍遵守两个重要的标准：一是Brian W. Kernighan和Dennis M. Ritchie于1978年合著的名著《The C Programming Language》，被称为标准C。二是美国国家标准化协会(ANSI)于1983年开始制定并于1988年最终完成的ANSI标准，即“ANSI C”。ANSI C比原来的标准C有了很大的发展。20世纪90年代后，尽管C++、Visual C++开发如火如荼，C语言也并没有停滞不前(更没有被替代)，版本不断更新、升级，但C的特征未变，C仍然是C!。

1.3.2 C++简介

当C语言程序达到一定的规模(代码达到25 000行以上)后，维护和修改显得相当困难。为了满足管理程序复杂性的需要，贝尔实验室的Bjarne Stroustrup博士于1979年开始对C语言进行了改进和扩充，并引入了面向对象程序设计的内容，最初取名为“带类的”C，1983年改名为C++。在经历了3次重大修订后，于1994年制定了标准C++草案，后又经不断完善，成为目前的C++，它具有以下特点：

(1) C++是C语言的超集。C++由两部分组成：一是过程性语言部分，这部分与C语言无本质区别，一般遵守ANSI C标准；二是类和对象部分，这是C语言所没有的，它是面向对象程序设计的主体。

(2) C++充分保持了与C语言的兼容性，绝大多数C语言程序可以不经修改直接在C++环境中运行。

(3) C++仍然支持面向过程的程序设计，是一种理想的结构化程序设计语言，又几乎全部包含了面向对象程序设计的特征。

(4) C++继承了C语言的高效率、灵活性等优点。用Bjarne Stroustrup博士的话来说，C++使程序“结构清晰、易于扩展、易于维护而不失效率”。

(5) C++是一种标准化的、与硬件基本无关的、广泛使用的程序设计语言，具有很好的通用性和可移植性。C++程序通常无需修改，或稍作修改，即可在其他计算机系统上运行。