高等学校计算机专业教材

GAODENG XUEXIAO JISUANJI ZHUANYE JIAOCAI

C语言——程序设计导论

崔雅娟 编

GAODENG XUEXIAO JISUANJI ZHUANYE JIAOCAI
GAODENG XUEXIAO JISUANJI ZHUANYE JIAOCAI
GAODENG XUEXIAO JISUANJI ZHUANYE JIAOCAI

人民邮电出版社
POSTS & TELECOMMUNICATIONS PRESS

高等学校计算机专业教材

C语言——程序设计导论

崔雅娟 编

人民邮电出版社

图书在版编目(CIP)数据

C 语言：程序设计导论/崔雅娟编 .—北京：人民邮电出版社，2002.7
高等学校计算机专业教材
ISBN 7-115-09372-5

Ⅰ.C... Ⅱ.崔 ... 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(2002)第 036025 号

内 容 提 要

本书以程序设计的方法为主线，介绍 C 语言知识及其在程序设计过程中的运用方法和技巧。本书注意程序设计思维方法的培养和训练，并含有结合实际、能够激发学生学习兴趣的实例。全书共分 11 章：程序设计概述，数据类型、运算符与表达式，语句及控制结构，函数和程序结构，预处理命令，数组，指针，结构及其他数据类型，文件，C 语言程序设计务实及实验。

本书可作为高等学校计算机专业教材，也可作为编程爱好者自学 C 语言的参考书。

高等学校计算机专业教材
C 语言——程序设计导论

◆ 编　　崔雅娟
责任编辑　滑　玉

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www. ptpress.com.cn
读者热线　010-67180876
北京汉魂图文设计有限公司制作
北京隆昌伟业印刷有限公司印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：17.25
字数：412 千字　　2002 年 7 月第 1 版
印数：1-6 000 册　　2002 年 7 月北京第 1 次印刷

ISBN 7-115-09372-5/TP・2263

定价：23.00 元

出版者的话

为了适应我国大学本科计算机专业教育发展对教材的需要，我社特邀请教育部所属的中国人民大学、中国地质大学、中国农业大学、北京科技大学、北京林业大学、北京语言文化大学(排名不分先后)6所高等学校的“计算机科学与技术系”系主任及资深教授组成专家组，规划、编写、审订了本套教材。

读者对象 本套教材的主要读者对象是普通高等院校计算机科学与技术专业的学生，兼顾信息、自动化及机电类专业学生学习计算机课程的需要。由于这些专业对学生的培养目标是：掌握计算机软件与硬件的基本理论和方法，能从事计算机应用、软件研制、技术开发和管理工作的高级技术人才，因此，本套教材的内容既注意计算机高级技术人才所应具有的完整知识结构，又适当侧重主要专业知识。

教材特点 本套教材参考了美国IEEE/CS和ACM技术委员会2001年推荐的课程CC2001(Computing Curricula 2001，参见 http://WWW.csab.org/~csab，http://cs.nju.edu.cn/~gchen/teaching/cc2001/overview-bok.html 2001)及我国几十所高等院校计算机专业的2001年教学计划进行规划。教材内容在选择国际上先进的计算机理论、技术的同时，务求符合我国目前教学环境的实际情况，有一定深度，又有较高的实用性。

根据大学教学的特点，本套教材主要包括必修课程、选修课程和辅助课程三类教材。除了每本教材内容自成体系外，还考虑了它在整个教学计划中的安排顺序，适当增加了承上启下的内容。

写作风格 本套教材根据内容需要，沿着“这是什么、有什么用、怎样用、怎样用得更好”的思路编写教材，通过讲解具体知识，传授学习方法，使学生达到掌握理论和技术的目的；同时力求文笔流畅，言简意赅。

教材每一章除基本知识外，还有本章要点、小结、思考与练习题。有些教材还附加教学大纲(包括教学重点、难点，讲授知识点的参考学时数)，操作性较强的课程还配有实验教材(包括上机应具备的软硬件环境和实验内容及方法)。为了方便教师教学，本套教材提供了演示稿，可到人民邮电出版社网站(http://www.ptpress.com.cn)的“教材出版图书出版中心”→“教材附件”→“课件”下载。

本套教材的作者都具有多年教学经验，教材的草稿也都在各自的授课过程中多次使用。这套教材的出版，为计算机专业的师生提供了新的选择。我们希望这套以易教、易学，朴实、实用为特色的教材在培养信息化建设专业人才方面做出应有的贡献。

欢迎广大读者对本套教材的不足之处提出批评和建议。

编 者 的 话

本书是普通高等学校计算机专业的第一门程序设计课程教材。作为计算机专业程序设计的入门课程，采用什么样的教学语言，组织什么样的教学内容，得到什么样的教学效果，这对后续专业课程的学习至关重要，应该引起任课教师的足够重视，值得大家研究和探讨。

作为第一门程序设计课程，应该使用什么样的教学语言？这个问题已经有了答案：使用C语言作为第一门程序设计的教学语言。这样做的主要理由有如下几个。

（1）C语言本身的特点。C语言作为程序设计工具，包含了描述数据和进行程序设计需要的重要机制。以函数为基本单位的程序组织方法特别适于进行模块化和结构化的程序设计。同时，C语言的灵活性提供了在不同层次上进行程序设计的可能性。既可以开发系统软件，又可以开发应用软件。

（2）C语言使用的广泛性。C语言目前已成为使用最广泛的语言之一，用C语言作为教学语言，在学习程序设计方法的过程中，可同时掌握一种实用的程序设计工具。

（3）专业课程学习的连续性。程序设计是计算机专业的基础课程，C语言同时也是许多后续课程的教学语言和描述工具。使用C语言作为第一门程序设计语言，有利于后续课程的学习。

（4）同其他语言相比，用C语言进行程序设计，需要学习者了解一些实现细节，需要更强的控制程序的能力，需要对问题有更周详的考虑。这些设计上的要求，对学习者更深入地理解程序设计的许多问题有很好的促进作用。

本书包括11章，每一章有导读、重点知识归纳和习题。书后附录包括常用字符ASCII代码表、C语言运算符优先级和结合性查阅表、Turbo C 2.0集成开发环境的使用、常见错误、常用函数表。各章节的主要内容如下：

第1章，程序设计概论，为初次接触程序设计的读者建立必要的关于程序设计的概念。主要包括程序设计概念和发展、程序设计方法、程序加工过程、C语言程序组成和Turbo C环境下的上机调试过程。

第2章，数据类型、运算符与表达式，介绍C程序中的数据表示和表达式计算。包括数据类型、变量、运算符、表达式及函数使用等一些重要概念。

第3章，语句及控制结构，介绍C程序语句的使用。包括顺序、选择和循环三种基本结构的实现和控制方法，基本控制结构中常用的控制机制。用实例说明现实问题模型与三种基本结构的对应关系和抽象方法。

第4章，函数和程序结构，介绍C程序的结构化设计问题。包括函数机制、变量作用域和存在期、复杂问题用多程序文件组织项目的实现方法。

第5章，预处理命令，主要介绍宏定义、包含文件和条件编译的概念和作用。

第6章，数组，主要介绍C语言组织相同数据类型数据的机制，包括数组的定义和使用、数组的实现原理和存储方法、多维数组、字符数组和字符串及实际问题中的一些数组应用。

第 7 章，指针，C 语言中通过地址操作对象的机制，主要包括指针的概念、运算和使用，指针和变量的关系，指针和函数的关系，指针和数组的关系。

第 8 章，结构及其他数据类型，主要介绍 C 语言用基本数据对象构造复杂数据对象的其他方法，包括结构、位段、联合、枚举和自定义数据类型等内容。

第 9 章，文件，主要介绍文件的概念和操作方法。

第 10 章，C 语言程序设计务实，将贯穿在全书各处的关于程序设计的要点做总结性的归纳。

第 11 章，根据各章内容设计了 11 个相应的实验。实验的内容与相应教学内容同步，供教师和同学们参考。

这门课的教学目标，首先是建立与程序设计相关的各种基本概念，掌握运用 C 语言进行程序设计的基本方法。包括问题的分析和抽象方法、模型的表示方法、算法设计和描述、程序编码设计和调试。其次，应着眼于掌握一个具体的 C 语言编程环境，了解集成开发环境应具备的基本功能，学习编写、调试程序的一些思路和技巧。本书以标准 C 为基础，原则上任何符合 ANSI C 标准的 C 语言系统都可以作为程序的工作环境，考虑初学者作为入门课练习环境的特点，建议使用比较基本、简单的系统作为学习工具。本书也给出了目前国内使用较多的 Turbo C 2.0 的简单使用方法。书中所有例题的程序，都在 Turbo C 2.0 环境下调试通过。本书某些较复杂的实例中包含了属于 Turbo C 系统的若干库函数，使用处均给出了函数功能的注释信息，而且这部分内容出现在推荐的自学内容当中，供有兴趣的读者在 Turbo C 2.0 环境下完成一些有趣的程序设计题目时参考。第三也是通过基础程序设计课程可以培养学生具备良好的编程习惯。

本书可作为计算机专业的教材或参考书，同样适合自学 C 语言程序设计的读者学习。

关于教学安排，教师可根据学时数酌情选择教学内容，建议课内学时数为 40 ~ 60 学时，上机时间不少于 30 学时。教师应充分考虑计算机专业课程设置的情况，决定讲授内容，如关于“结构”一章链表的操作，“数据结构”中将会有系统的讨论，在基础程序设计课程中，可考虑不讲或少讲。结构的重点也不一定是进行多么复杂的程序设计，可考虑讨论结构这种数据构造方法的特点和好处，比较不同数据表示方法对程序结构和方法的影响。编译预处理的内容可穿插在其他章节中介绍。笔者在教学中遇到的问题，各章都有体现，内容的组织上，也反映笔者对这些问题的思考，希望对老师们有一些参考价值。

这里我要感谢北京语言文化大学计算机系的宋柔教授和卢湘鸿教授，也感谢参加过我的课程的所有同学，他们的思考和提问给过我许多启示。

希望读者将你们的建议及发现的问题及时通知我。我会仔细阅读收到的电子邮件，E-mail 地址是 cyj@blcu.edu.cn。

编　者

2002 年 7 月

目　　录

第 1 章　程序设计概述

本章的主要内容包括：程序设计的相关概念、程序设计的基本方法、C 语言程序组成和上机调试过程。本章中的基本概念较多，在概念首次出现或给出解释的地方有重点符号标识。

1.1　程序、程序设计和程序设计语言

从计算机诞生起，就有了程序，计算机系统在程序的控制下运行。相同或相似的计算机硬件系统可执行完全不同的任务，只因为这些计算机系统运行着不同的程序。人们为计算机设计了丰富多彩的程序，赋予计算机无穷的生命力。在计算机日益普及的今天，能够编写计算机程序是许多计算机爱好者的梦想，让我们从下面的概念起步，开始程序设计之路吧！

程序(Program)：为解决某一问题而设计的一系列指令，这些指令能被计算机识别和执行。程序的表示是静态的，但人们设计程序的最终目的是用它解决问题，因此，程序必须能够运行，否则毫无用处。运行程序中，需要把程序调入内存，按顺序逐条执行程序的指令。

程序设计(Programming)：指设计、书写及检查程序的过程。具体来说是指包括分析问题、确定解决方法、设计程序结构，使问题内容或解题计划变为计算机能够接受的指令或语句序列的过程。程序是程序设计的结果。

语言(Language)：用于传达信息的表示方法、约定和规则的集合，是人们交流信息的工具和媒介。而程序设计语言是人与计算机打交道时交流信息的一类媒介和工具，它由语句(Statement)组成，包含语法和语义两方面。语法(Syntax)定义了构造语言表达式或句子所需的各种规则。语义(Semantic)则是对构成语言成分的含义的定义和说明。以下是程序设计语言发展过程中使用的几类程序设计语言：

机器语言(Machine Language)：计算机直接使用的二进制形式的程序语言或机器代码。它不需翻译即可直接为机器接受，使用绝对地址和绝对操作码，反映机器内部的硬件特点(如存储器的编址方法、运算和逻辑操作、指令表等)。利用机器语言进行程序设计需要设计人员对硬件、指令系统有深入的了解，使用非常不便，书写程序很困难，耗时多，难以保证正确性。

汇编语言(Assembler Language)：一种面向机器的用符号表示的低级程序设计语言。它与机器语言很接近，需经汇编程序(Assembler)翻译成机器语言程序，汇编语言与机器语言基本上是一对一的关系。汇编语言虽然提供了机器指令的助记符号，但仍然摆脱不掉对硬件的依赖，不同系统下的汇编程序没有通用性。

高级语言(High - level Language)：是易为人们所理解的完全符号化的程序设计语言。它

提供数据描述和程序控制机制，而且语法上越来越接近自然语言，为越来越多的人所理解和使用。用高级语言进行程序设计，方便、易读、高效且可通用。因此基于高级语言的程序设计技术发展很快，越来越多的人参加到程序设计的行列中来。1954 年诞生的 Fortran 语言是第一个高级语言，目前广泛使用的有 C、Pascal、Ada、C + + 、Basic、Java 等。

用高级语言编写的程序称为源程序，C 语言源程序文件名字后缀一般必须为“.c”。用高级语言编写的程序，计算机不能直接执行，需要把这个程序转换成二进制代码的机器语言程序。这种转换过程称为“程序加工”，目前通用的加工方法是分两步。①编译(Compiler)，即把用高级语言写的源程序转换为相应的机器语言目标模块(Object Module)。目标模块不是可执行文件，它包括程序及连接程序的控制信息。目标程序文件名的后缀为“.obj”。②连接(Linker)，将目标模块和其他一些必要的功能模块组合在一起，生成可执行文件，执行程序文件的后缀为“.exe”。对广泛使用的语言，不同的系统都提供相应的编译和连接工具，同一种高级语言设计的程序，可在不同的系统下加工，得到不同系统下的程序。这是目前使程序具有通用性的手段之一。

1.2 程序设计中的科学思维方法

进行程序设计的本质，是要用计算机解决某些现实问题。怎样从实际问题出发，运用科学的思维方法，分析、分解和抽象现实问题的模型，采用合理的程序设计技术和工具实现设计意图，是程序设计基础课程应该着重训练的方面，也是这一章的重点。初级课程中，由于问题复杂程度的关系，对数据的描述比较简略，这并不意味着数据的问题不重要。相反，数据是程序的重要组成部分，用于刻画程序处理对象的属性和状态，是程序处理的对象，而程序是对数据施行算法的过程。后继课程(如“数据结构”研究程序设计中数据的关系和运算)将对数据有更深入的讨论。著名的计算机科学家、Pascal 语言的发明者 N·沃思(Niklaus Wirth)教授提出了如下著名公式：

程序 = 算法 + 数据结构

这个公式的重要性在于：不仅要研究数据，还要研究数据之间的关系(数据结构)；不能离开数据结构去抽象地分析程序的算法，也不能脱离算法去孤立地研究程序的数据结构，只能从算法与数据结构的统一上去认识程序。程序就是在数据的某些特定的表示方式和结构基础上对抽象算法的计算机语言的具体表述，程序的实现与数据的表示和存储方式有着密切的关系。大家在本课程的学习和练习中，应注意体会数据组织形式对程序实现方法的影响。

1.2.1 算法

1. 算法概念

任何问题的解决过程都是由一定的步骤组成的。算法(Algorithm)：是对特定问题求解步骤的一种描述。既然算法是解决问题的方法，它处理的对象必然是该问题所涉及到的相关数据。也就是说，算法与数据是程序设计过程中密切相关的两个方面。程序设计中不但要描述程序处理的数据，还要运用一定的算法描述处理数据的过程。程序是数据结构和算法的统一。

我们先看几个例子。

【例 1.1】 A、B 两人各有一桶油，现两人想将各自桶内的油互换，必须借助另外一个空桶，并按如下算法进行(用 *Si* 表示第 *i* 步操作，将 A 的桶命名为 A，将 B 的桶命名为 B，将空桶命名为 M)。

*S*1：将 A 桶中的油倒入 M 桶中；

*S*2：将 B 桶中的油倒入 A 桶中；

*S*3：将 M 桶中的油倒入 B 桶中；

*S*4：结束。

本例中只有四步操作，按顺序依次执行，请注意：经过简单的命名，算法的书写更加简练，含义更好理解。

【例 1.2】 计算 $a+|b|$。

根据公式 $$a+|b|=\begin{cases}a+b, & b\geqslant 0\\ a-b, & b<0\end{cases}$$ 可得如下算法。

*S*1：输入 a、b。

*S*2：判断是否满足条件 $b\geqslant 0$。若 $b\geqslant 0$，则执行 *S*2.1，否则执行 *S*2.2。

*S*2.1：$a+b\rightarrow c$；

*S*2.2：$a-b\rightarrow c$。

*S*3：输出 c。

*S*4：结束。

本例中，*S*2 中又包含了 *S*2.1 和 *S*2.2 两种操作，在 *S*2.1 和 *S*2.2 中，有一个选择，要根据 b 的值决定执行 *S*2.1 还是 *S*2.2，不能两者都执行。符号“→”在以后的算法中表示将“→”左边表达式计算的结果保存在“→”右边的对象内。

【例 1.3】 计算 $1+2+\cdots+100$。

考虑本例题目的求解方法，我们可以按公式逐项累加，也可以按照等差级数前 n 项求和的公式直接来实现。两个算法如下所示。

逐项累加算法

*S*1：输入项数 n（n 为 100）。

*S*2：0→sum。

*S*3：1→i。

*S*4：重复执行下面两步操作，直到 $i>n$。

*S*4.1：$i+\text{sum}\rightarrow\text{sum}$；

*S*4.2：$i+1\rightarrow i$。

*S*5：输出 sum。

*S*6：结束。

等差级数前 n 项求和算法

*S*1：输入 n（n 为 100）。

*S*2：$\dfrac{(1+n)\times n}{2}\rightarrow\text{sum}$。

*S*5：输出 sum。

*S*6：结束。

本例前一种算法中，*S*4.1 和 *S*4.2 被反复地执行，一直延续到“$i>n$”。后一种算法中，不需要进行重复的控制，但计算较前一种复杂些。由此可见，同一问题，可能有多种算法，为了有效地解决问题，可能需要在多个算法之间做决定，最后选择正确、高效的算法。

由上面的例子可以看出，一个算法由一些操作组成，而这些操作又是按一定的控制结构所规定的次序执行的。操作和控制结构是算法的两个要素。

计算机可进行的基本操作包括以下几个方面。

(1) 逻辑运算操作：与、或、非等。

(2) 算术运算操作：加、减、乘、除等。

(3) 数据比较操作：等于、不等于、大于、小于等。

(4) 数据传输操作：输入、输出、赋值等。

一个算法的功能不仅取决于所选的操作，还决定于所选操作的执行顺序，即控制结构。算法的控制结构给出了算法的框架，决定了各操作的执行次序。本书的1.2.2节是关于控制结构的讨论。

2. 算法的特征和要求

根据上面的例子，我们可归纳出算法的以下几个特征。

(1) 有穷性。有穷性指算法必须总是在执行有限步之后结束，且每一步在有限时间内完成，算法不能出现“死循环”。上面的几个例子中，无论算法控制结构如何，我们都可在有限步内得到问题的解。不过，不是所有的程序都要求有穷性(如操作系统)，这也是程序与算法的区别之一。

(2) 确定性。确定性指算法的每一步操作必须有确切的含义，不会产生歧义。或者说任何条件下，算法中只有惟一的路径，相同的输入只能得出相同的结果。这一特性，对描述算法的工具和语言提出起码的要求，使用自然语言很难达到算法的确定性要求。

(3) 输入。输入是算法执行时需要从外界取得的信息。每个算法必须有零个或多个输入。上面算法的第一步都是关于输入的操作，有些算法不需要从外界得到数据，系统自动产生初始值。

(4) 输出。每种算法必须有确定的结果，可产生一个或多个输出。

(5) 有效性。有效性指算法是可行的，算法中的操作都可以通过已经实现的基本操作来实现。如除数为0的除法操作是无意义的，不满足有效性特性。

算法设计应满足下面的基本要求。

(1) 正确性。正确性指算法能满足具体问题的求解需要。算法的正确性是软件领域长久以来的重要课题，这里，我们要求设计的算法达到下面三个层次：①不含语法错；②对几组输入数据能得出满足规格要求的结果；③对精心选择、苛刻、带有刁难性的几组输入数据能得出满足要求的结果。

(2) 可读性。要求设计的算法具有良好的可读性，易于交流、阅读和调试。这是我们在程序设计基础课程中，通过训练可以达到的最基本目标。要求注意程序格式，从最简单的程序开始，培养好的程序设计习惯。

(3) 健壮性。要求算法对非法数据和操作做出反应和处理。

(4) 效率。要求算法要尽可能考虑执行时间和空间的效率问题。

程序设计工作有许多自己固有的特点，我们应该注意程序设计求解过程的多样性。由于设计者对问题分析考虑的方面不同，以及在程序设计过程中的决定或选择不同，人们常常能够得到许多合理的、不同的程序。这些程序各有特点，可能侧重点不同，也可能实现方法上有差异，也可能确实是人们对问题有不同的认识，但都可以解决问题。读者在编程时，应注意同一问题可能存在其他的解决方法，注意比较不同解法的特点。尤其是学习别人的程序

时，应特别注意解决问题的方法，包括如何分析问题，如何抽象、定义问题的模型，如何分解问题成为相对简单的若干部分，如何确定求解方法，如何考虑语言结构等方面。

3. 算法表示

算法设计是由总体到局部，从抽象到具体，逐步细化的过程。同一个算法可以使用不同的程序设计语言来实现。算法描述和表示应该能够适应算法设计过程中不断修改细化的过程，最好不依赖于某种具体的程序设计语言。一般使用以下几种类型的工具描述算法。

(1) 自然语言

自然语言即人们使用的语言，如英语、汉语等，用自然语言描述算法通俗易懂，但它存在着很多缺陷。

① 易产生歧义。往往要根据上下文判别语义，难以满足算法对确定性的要求。

② 语句比较繁琐，且很难清楚表达算法的逻辑流程和控制结构。

③ 计算机还不能处理用自然语言表示的算法。

(2) 专用工具

为了形象地描述算法，人们创造了许多专用工具来描述算法。常用的有流程图、PAD 图和 N－S 图等。这里介绍算法的流程图表示方法，对其他方法有兴趣的读者可参看其他参考书。

流程图采用不同的几何图形来描述算法的逻辑结构，每个几何图形表示不同性质的操作。图 1-1 所示的是 ANSI 规定的常用流程图符号，具体使用规定如下：

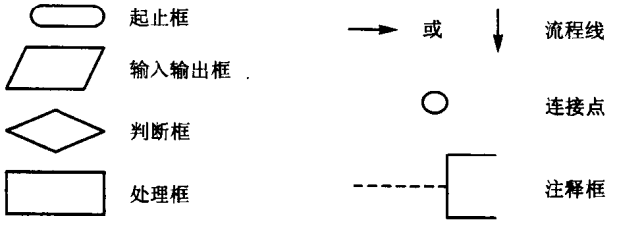

图 1-1　常用流程图符号

① 起止框表示算法的起始、终止，圆圈表示连接，圈内标识同一字母的圆圈均表示流程图中的同一点，通常用于不同页之间流程图的连接或较远距离的流程线连接。

② 平行四边形表示要输入数据和输出结果，框内应填写需要输入或输出的量及相应的操作。

③ 菱形是判断框，框内应填上判断条件。

④ 矩形框是处理框，用于执行计算或赋值，框内用文字或符号表明具体实现的操作。

⑤ 注释框用文字说明注释信息，帮助理解流程图。

⑥ 流程线表示算法中操作的顺序，向下向右可不画箭头，其他方向的箭头必须画出。

图 1-2、图 1-3、图 1-4 是例 1.1、例 1.2、例 1.3 的流程图表示。

由图可见，流程图直观形象，易于理解和查找逻辑错误。对作业中的小程序很合适，适于初学者使用，要求大家掌握。当问题复杂性增加、算法篇幅较多时，流程图就难于处理了。