普通高等学校计算机基础教育“十二五”规划教材·卓越系列

C语言程序设计

C YUYAN CHENGXU SHEJI

徐凤生　黄　超　谢玉华◎编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

普通高等学校计算机基础教育“十二五”规划教材·卓越系列

C语言程序设计

徐凤生　黄　超　谢玉华　编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书是在作者多年教学经验的基础上编写而成，全书共分为 12 章，内容包括 C 语言程序设计基础知识、各种数据类型和常用库函数、各种运算符和表达式、程序控制语句、数组、函数、指针、结构体与共用体、编译预处理、文件、常用算法以及 C 程序设计举例等。本书是山东省省级教学团队项目的研究成果，在内容安排上，注重理论与实践相结合，突出学生编程能力的培养。

本书适合作为高等学校计算机及相关专业“C 语言程序设计”课程的教材，也可作为全国计算机等级考试二级 C 语言程序设计的培训教材。

图书在版编目（CIP）数据

C 语言程序设计 / 徐凤生，黄超，谢玉华编著. —北京：中国铁道出版社，2015.6

普通高等学校计算机基础教育“十二五”规划教材. 卓越系列

ISBN 978-7-113-20472-3

Ⅰ. ①C… Ⅱ. ①徐… ②黄… ③谢… Ⅲ. ①C 语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆 CIP 数据核字(2015)第 115641 号

书　　名：C 语言程序设计
作　　者：徐凤生　黄超　谢玉华　编著

策　　划：刘丽丽　　**读者热线**：400-668-0820
责任编辑：周　欣　冯彩茹
封面设计：一克米工作室
责任校对：汤淑梅
责任印制：李　佳

出版发行：中国铁道出版社（100054，北京市西城区右安门西街 8 号）
网　　址：http://www.51eds.com
印　　刷：北京鑫正大印刷有限公司
版　　次：2015 年 6 月第 1 版　　2015 年 6 月第 1 次印刷
开　　本：787 mm×1 092 mm　1/16　**印张**：16　**字数**：399 千
印　　数：1～3 000 册
书　　号：ISBN 978-7-113-20472-3
定　　价：34.00 元

前　言

FOREWORD

C 语言既具备高级语言的特点，又具有直接操纵计算机硬件的能力，并因其丰富灵活的控制和数据结构、简洁而高效的语句表达、清晰的程序结构和良好的可移植性而拥有大量的使用者。目前，国内几乎所有高校都开设了“C 语言程序设计”课程，该课程不仅是计算机学科的主干课程，也成为理工科专业计算机基础训练的必修课。

本书是在编者多年教学经验的基础上编写而成，共分为 12 章，内容包括 C 语言程序设计基础知识、各种数据类型和常用库函数、各种运算符和表达式、程序控制语句、数组、函数、指针、结构体与共用体、编译预处理、文件、常用算法与以及语言程序设计举例等。

本书具有如下特色：

（1）语言通俗易懂，阐述简洁明了。

（2）遵循“注重理论、强化实践”的核心思想，注重各部分知识的综合应用训练，以提高学生的程序设计能力。

（3）例题丰富，分析透彻，讲解清楚，不仅有效降低了学习难度，而且突出了算法思想设计。

（4）通过典型例题的分析，使学生对所学知识更加系统化和条理化，更易于对所学知识的融会贯通和举一反三。

（5）所有程序示例都在 VC++ 6.0 环境下调试通过。每章都配有上机实验和习题，便于学生课后及时练习。

（6）为了便于教师教学，本书配有电子课件和习题答案，可与出版社直接联系或发送邮件至 xfs@dzu.edu.cn 与作者联系索取。

在编写中我们参阅了许多 C 语言程序设计教材和相关资料，在此向其作者一并表示感谢。本书的出版得到了德州学院教材出版基金和山东省“信息与计算科学”省级教学团队项目的资助。最后，还要特别感谢中国铁道出版社的大力支持，使得本书得以顺利出版。

限于编者水平，书中难免存在疏漏和不足之处，恳请读者不吝指正。

编　者

2015 年 3 月于德州学院

教学建议

教 学 内 容	学习要点及教学要求	课时安排
第 1 章　C 语言概述	了解 C 语言的历史、特点 掌握算法的概念、特征、设计原则及时间复杂度 初步掌握 C 语言的基本语法元素 初步掌握 C 语言的编辑与调试	2
第 2 章　数据类型、运算符与表达式	了解 C 语言的各种数据类型 掌握整型常量、实型常量、字符型常量、字符串常量的表示方法 熟练掌握各种运算符和表达式 掌握变量的定义、初始化 了解各种位运算	4
第 3 章　顺序结构程序设计	掌握基本的输入、输出语句 掌握顺序结构程序设计的基本思想	4
第 4 章　分支结构程序设计	熟练掌握 if 语句、switch 语句的各种形式与应用 掌握 if 语句嵌套的配对规则、break 的应用	6
第 5 章　循环结构程序设计	熟练掌握 3 种循环的形式与应用 掌握循环的嵌套	6
第 6 章　数组	掌握一维数组、二维数组、字符数组的定义、初始化以及数组元素的引用方法 掌握有关处理字符串的库函数的使用方法	8
第 7 章　函数	掌握函数的定义、声明和调用方法 掌握函数调用中数据传递的几种方法 了解递归函数 掌握变量的作用域和变量的存储类型	10
第 8 章　指针	掌握地址、指针、指针变量的概念，能正确定义所需类型的指针变量，能正确将指针变量指向某变量、数组或函数，能正确利用指针变量引用所指向的变量、数组或函数 了解指针数组和多级指针的概念 了解用指针实现内存动态分配的方法	14
第 9 章　结构体与共用体	了解结构体、共用体和枚举类型数据的特点 掌握结构体类型数据的定义方法以及结构体变量、数组、指针的定义、初始化和成员的引用方法 掌握共用体和枚举类型的定义方法和对应变量的定义、引用 掌握用户自定义类型的定义和使用 初步了解线性表的各种基本操作	10
第 10 章　编译预处理	掌握宏定义和宏替换的一般方法 掌握包含文件的处理方法 了解条件编译的作用和实现方法	2

续表

教 学 内 容	学习要点及教学要求	课时安排
第 11 章　文件	了解文件的概念、分类以及文件指针的概念 掌握文件的打开与关闭以及文件的读写操作 了解位置指针、文件定位的相关操作	6
第 12 章　常用算法与 C 程序设计举例	熟练掌握各种常用的算法 初步了解 C 语言程序设计的过程	4（选讲）
总课时	第 1～12 章建议课时	48
	上机实验建议课时	28

说明：

（1）建议采用多媒体教学，实现“讲—演”结合。

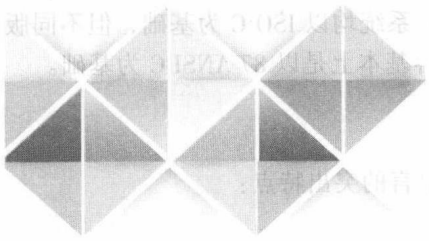
（2）建议教学分为课堂教学和实验教学两大模块。课堂教学重点讲授 C 语言的核心知识及程序设计的方法、步骤。实验教学重点在培养学生分析问题和解决问题的能力，提高编程能力。

（3）建议计算机及相关专业要强化“指针”一章的教学，以便为后续课程“数据结构”的学习打下基础。

（4）不同学校可根据各自的教学要求和教学计划对教学内容进行取舍。

目录

CONTENTS



第1章 C语言概述

程序设计语言是人与计算机之间交流的重要工具，其中C程序设计语言（简称C语言）是高级程序设计语言的典型代表，是国内外广泛使用的一种编程语言。本章将介绍C语言的发展历史与特点、程序设计与算法、C语言的基本语法元素、C语言程序设计求解问题的步骤，以及C语言程序的调试方法等。

1.1 C语言的发展历史与特点

1.1.1 C语言的发展历史

C语言是当今世界最流行的程序设计语言之一，可用于硬件编程，有着和汇编语言相近的高效率，但又比汇编语言形象易懂。C语言既可以编写系统软件如UNIX、Linux，也可以编写应用软件如Matlab。

C语言的发展历程是与UNIX密切相关的。1960年出现的ALGOL 60是一种面向问题的高级语言，但不适宜系统程序的编写。1963年英国剑桥大学推出了CPL（Combined Programming Language）语言。1967年剑桥大学的Matin Richards对CPL语言进行了简化，推出了BCPL（Basic Combined Programming Language）语言。1970年，美国贝尔实验室的Ken Thompson又将BCPL语言进一步简化而设计出B语言（取BCPL的首字母），并用B语言开发了第一个UNIX操作系统，但此时的B语言过于简单，功能有限。1972年至1973年间，贝尔实验室的D.M.Ritchie在B语言的基础上设计出了C语言（取BCPL的第二个字母）。C语言既保持了BCPL和B语言精简、接近硬件的优点，又克服了其过于简单、无数据类型等缺点。1973年，Ken Thompson和D.M.Ritchie合作把UNIX系统软件90%以上的代码用C语言改写。

此后，C语言多次做了改进，但直到1975年UNIX第6版发布后，C语言的突出优点才引起人们的普遍关注。1977年出现了不依赖于机器的C语言编译文本《可移植C语言编译程序》，使C语言移植到其他机器时所需做的工作大大简化。随着UNIX的日益广泛使用，C语言也迅速得到推广。C语言和UNIX可以说是一对孪生兄弟，在发展过程中相辅相成。1978年以后，C语言先后移植到大、中、小、微型计算机上，已独立于UNIX和PDP。C语言很快风靡全世界，成为世界上应用最广泛的计算机语言之一。

1978年，B.W.Kernighan和D.M.Ritchit（合称K&R）合著了影响深远的名著*The C Programming Language*。本书中的C语言成为后来广泛使用的C语言版本的基础，它被称为标准的C。1983年，美国国家标准化协会（ANSI）根据C语言问世以来各种版本对C语言的发展和扩充，制定了新的标准，称为ANSI C。ANSI C比原来的标准C有了很大的发展。K&R在1988年修订了他们的经典著作*The C Programming Language*，按照ANSI C标准重新写了该书。1987年，ANSI又公布了新标准，称为87 ANSI C。1990年，国际标准化组织（International Standard Organization，

ISO）将 87 ANSI C 作为 ISO C 的标准。目前所使用的 C 编译系统均以 ISO C 为基础，但不同版本如 Microsoft C、Turbo C 和 Quick C 等稍有不同。本书的内容基本上是以 87 ANSI C 为基础。

1.1.2 C 语言的特点

C 语言之所以经久不衰，是因为其本身具有不同于其他语言的突出特点：

（1）简洁紧凑、灵活方便

C 语言一共只有 32 个关键字，9 种控制语句，程序书写形式自由，区分大小写。相对其他计算机语言而言，C 语言较容易学习和记忆，源程序较短，编写程序时工作量较少，容易编写和调试。

（2）运算符丰富

C 语言的运算符包含的范围很广泛，共有 34 种运算符。C 语言把括号、赋值、强制类型转换等都作为运算符处理，从而使 C 语言的运算类型极其丰富，表达式类型多样化。灵活使用各种运算符可以实现在其他高级语言中难以实现的运算。

（3）数据类型丰富

C 语言的数据类型有整型、实型、字符型、枚举类型、数组类型、指针类型、结构体类型、共用体类型等，能用来实现各种复杂的数据结构的运算。引入了指针概念，使程序效率更高。另外，C 语言具有强大的图形功能，支持多种显示器和驱动器，且计算功能、逻辑判断功能强大。

（4）表达方式灵活实用

C 语言提供多种运算符和表达式值的方法，对问题的表达可通过多种途径获得，其程序设计更主动、灵活。其语法限制不太严格，程序设计自由度大，如对整型量与字符型数据及逻辑型数据可以通用等。

（5）允许直接访问物理地址，对硬件进行操作

由于 C 语言允许直接访问物理地址，可以直接对硬件进行操作，因此它既具有高级语言的功能，又具有低级语言的许多功能，能够像汇编语言一样对位、字节和地址进行操作，而这三者是计算机最基本的工作单元，可用来写系统软件。

（6）生成目标代码质量高，程序执行效率高

C 语言描述问题比汇编语言迅速，工作量小、可读性好，易于调试、修改和移植，而代码质量与汇编语言相当。

（7）可移植性好

C 语言在不同机器上的 C 编译程序，86%的代码是公共的，所以 C 语言的编译程序便于移植。在一个环境上用 C 语言编写的程序，不改动或稍加改动，就可移植到另一个完全不同的环境中运行。

（8）表达力强

C 语言有丰富的数据结构和运算符，包含了各种数据结构，如整型、数据类型、指针类型和联合类型等，用来实现各种数据结构的运算。C 语言的运算符有 34 种，范围很宽，灵活使用各种运算符可以实现难度极大的运算。C 语言能直接访问硬件的物理地址，能进行位（bit）操作，兼有高级语言和低级语言的许多优点。它既可用来编写系统软件，又可用来开发应用软件，已成为一种通用程序设计语言。

C 语言也有一些缺点，如 C 语言的语法限制不太严格，对变量的类型约束不严格，影响程序的安全性，对数组下标越界不作检查等。但因其上述突出的优点，C 语言仍然是非常优秀的程序设计语言之一。

1.2　程序设计与算法

1.2.1　程序设计

程序设计是给出解决特定问题程序的过程，是软件构造活动中的重要组成部分。程序设计往往以某种程序设计语言为工具，给出这种语言下的程序。程序设计语言分为低级语言和高级语言两大类。低级语言直接面向机器，如机器语言和汇编语言；高级语言独立于机器，用高级语言编写的程序在不同的机器上必须使用不同的翻译程序。C 语言是一种高级语言，它必须被翻译成计算机能识别的语言，即机器语言，才能在计算机上执行。程序设计过程应当包括分析、设计、编码、测试、排错等不同阶段。

一个程序应包括以下两个方面的内容：

（1）对数据的描述。在程序中要指定数据的类型和数据的组织形式，即数据结构（Data Structure）。

（2）对操作的描述。即操作步骤，也就是算法（Algorithm）。

数据是操作的对象，操作的目的是对数据进行加工处理，以得到期望的结果。作为程序设计人员，必须认真考虑和设计数据结构和操作步骤（即算法）。著名的计算机科学家沃思（Nikiklaus Wirth）提出了一个经典的公式：

数据结构+算法=程序

实质上，好的程序设计就是一个好的算法加上一个好的数据结构。算法是程序的灵魂。

1.2.2　算法的特征

具体地说，算法就是对特定问题求解步骤的一种描述，是指令的有限序列。其中，每个指令表示一个或多个操作。

一个算法必须满足以下 5 个重要特性：

（1）有穷性。对于任意一组合法输入值，在执行有穷步骤之后一定能结束，即算法中的每个步骤都能在有限时间内完成。

（2）确定性。对于每种情况下所应执行的操作，在算法中都有确切的规定，使算法的执行者或阅读者都能明确其含义及如何执行。并且，在任何条件下，算法都只有一条执行路径。

（3）可行性。算法中的所有操作都必须足够基本，都可以通过已经实现的基本操作运算有限次实现之。

（4）有输入。作为算法加工对象的量值，通常体现为算法中的一组变量。有些输入量需要在算法执行过程中输入，而有的算法表面上可以没有输入，实际上已被嵌入算法之中。

（5）有输出。它是一组与“输入”有确定关系的量值，是算法进行信息加工后得到的结果，这种确定关系即为算法的功能。

算法与程序是两个不同的概念，两者之间既有联系又有区别。一个程序不一定满足有穷性。例如，操作系统，只要整个系统不被破坏，它就永远不会停止。即使没有作业需要处理，它仍处于动态等待中。因此，操作系统不是一个算法。另一方面，程序中的指令必须是机器可执行的，而算法中的指令则无此限制。算法代表了对问题的解，而程序则是算法在计算机上的特定的实现。一个算法若用程序设计语言来描述，则它就是一个程序。

1.2.3　算法设计的原则

要设计一个“好”的算法，通常应考虑达到以下目标：

（1）正确性。首先，算法应当满足以特定的“规格说明”方式给出的需求。其次，对算法是否“正确”的理解可以有以下 4 个层次：①程序中不含语法错误；②程序对于几组输入数据能够得出满足要求的结果；③程序对于精心选择的、典型、苛刻且带有刁难性的几组输入数据能够得出满足要求的结果；④程序对于一切合法的输入数据都能得出满足要求的结果。通常以第③层意义的正确性作为衡量一个算法是否合格的标准。达到第④层意义的正确性极为困难，因为对所有的输入数据进行验证不太现实。

（2）可读性。算法主要是为了人的阅读与交流，其次才是为计算机执行，因此算法应该易于人的理解；另一方面，晦涩难读的程序易于隐藏较多错误而难以调试。

（3）健壮性。当输入的数据非法时，算法应当恰当地做出反应或进行相应处理，而不是产生莫名其妙的输出结果。并且，处理出错的方法不应是中断程序的执行，而应是返回一个表示错误或错误性质的值，以便在更高的抽象层次上进行处理。

（4）高效率与低存储量需求。通常，效率指的是算法执行时间。对于同一个问题的多个算法，执行时间短的算法效率高。存储量指的是算法执行过程中所需的最大存储空间。两者都与问题的规模有关。如求 100 个数的平均数和求 10 000 个数的平均数所花的执行时间和运行空间有一定的差别。

1.2.4 算法的描述方法

算法的描述方法有多种，常用的方法有自然语言、伪代码、N-S 图、流程图等。这里只介绍算法的流程图描述方法。

流程图是用一些图框来表示各种操作，其常用的流程图符号如图 1-1 所示。

图 1-1　常用的流程图符号

起止框：在框内标注“开始”表示程序开始，在框内标注“结束”表示程序结束，一个完整的流程图始末必须是起止框。

输入输出框：输入框标注输入的数据，输出框标注输出的数据。

判断框：判断框一般有一个入口和两个出口，在条件成立的出口处需注明“是”或“Y”，在条件不成立的出口处需注明“否”或“N”。如果是多分支判断，则可有两个以上出口。

处理框：处理框是用来表示执行赋值、计算、传送运算结果等的图形符号，算法中处理数据需要用到的算式、公式等根据执行顺序分别写在不同的处理框中。

流程线：带箭头的流程线表示执行的先后顺序。

例 1.1　输入两个数，输出其中的较大数。

此算法的流程图如图 1-2 所示，具体执行过程如下：

（1）算法开始。

（2）输入两个数，分别存到变量 a、b 中。

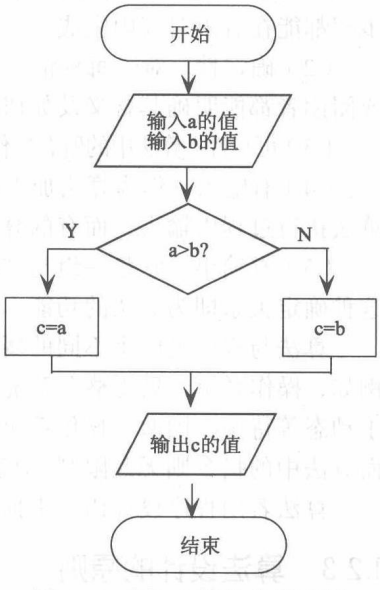


图 1-2　例 1.1 算法的流程图

（3）如果 a 大于 b，则将 a 赋给变量 c；否则将 b 赋给变量 c。

（4）输出变量 c 中的值，即较大数。

（5）算法结束。

可以看出，用流程图表示算法不仅形象直观，逻辑清晰，且易于理解。当算法不太复杂时，采用流程图进行描述不失为一种好方法。

1.2.5　算法的时间复杂度和空间复杂度

所谓算法的时间复杂度，是指执行算法所需要的计算工作量。一般情况下，算法中基本操作重复执行的次数是问题规模 n 的某个函数 $f(n)$，算法的时间度量记作 $T(n)=O(f(n))$。它表示随着问题规模 n 的增长，算法执行时间的增长率和 $f(n)$的增长率相同，称为算法的渐近时间复杂度（简称时间复杂度）。例如：

（1）{a=3; b=4; sum=a+b;}

（2）for(i=1;i<=n;i++)s+=x;

（3）for(i=1;i<=n;i++)

　　for(j=1;j<=n;j++)x=x+2;

3 个程序段的时间复杂度分别为 $O(1)$、$O(n)$、$O(n^2)$。

有些情况下，算法中基本操作重复执行的次数随问题的输入数据的不同而不同，例如，对 n 个整数进行排序，此时取其平均时间复杂度。但当平均时间复杂度无法计算时，则取其最坏情况下的时间复杂度。

算法的空间复杂度一般是指执行这个算法所需要的内存空间，记作 $S(n)=O(g(n))$，其中 n 为问题的规模，它表示随着问题规模 n 的增大，算法运行所需存储量的增长率与 $g(n)$的增长率相同。一个上机执行的程序除了需要存储空间来寄存本身所用指令、常数、变量和输入数据外，也需要一些对数据进行操作的工作单元和存储一些实现计算机所需信息的辅助空间。若输入数据所占空间只取决于问题本身，和算法无关，则只需要分析除输入和程序外的辅助变量所占的额外空间，否则应同时考虑输入本身所需的空间（和输入数据的表示形式有关）。若所需的额外空间相对于输入数据量来说是常数，则称此算法为原地工作。若所需存储量依赖于特定的输入，则通常按最坏情况考虑。

1.3　C 语言程序示例

首先给出一个例子，使读者对 C 语言程序有一个初步的认识。对程序内容的具体含义、语法与功能等则不必深究，相关详细内容将在以后的章节中重点介绍。

例 1.2　输入一个正整数 n，求 $n!$。

```
#include<stdio.h>                          /*编译预处理命令*/
void main()                                /*主函数*/
{
  int n;                                   /*定义变量*/
  int factorial(int);                      /*函数声明*/
  printf("请输入一个正整数: ");
  scanf("%d",&n);                          /*输入一个整数*/
  if(n>0)
    printf("%d!=%d\n",n,factorial(n));
```