

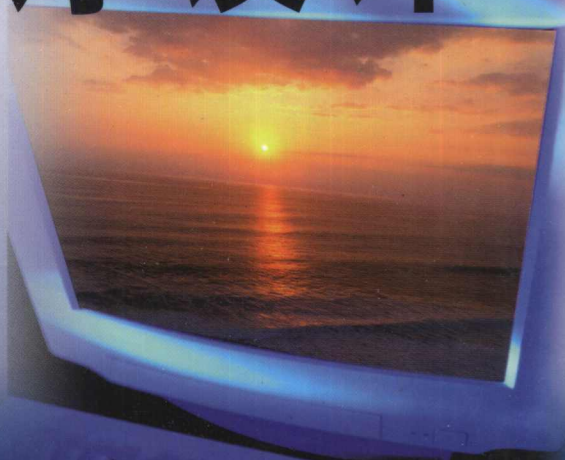
高职、高专计算机系列教材



C语言程序设计

● 刘天印 贾学斌 主编

华中科技大学出版社



高等职业技术教育计算机系列教材

C 语言程序设计

主 编 刘天印 贾学斌
副主编 胡大威 祁文青 黄国军
杜华兵 唐 波 王秀章
余云霞

华中科技大学出版社

图书在版编目(CIP)数据

C 语言程序设计/刘天印 贾学斌 主编
武汉:华中科技大学出版社, 2002 年 3 月
ISBN 7-5609-2652-5

I. C…

Ⅱ. ①刘… ②贾…

Ⅲ. C 语言-程序设计-高等学校:技术学校-教材

Ⅳ. TP312

C 语言程序设计

刘天印 贾学斌 主编

责任编辑:曾 光

封面设计:刘 卉

责任校对:陈元玉

责任监印:张正林

出版发行:华中科技大学出版社

武昌喻家山 邮编:430074 电话:(027)87545012

录 排:华中科技大学出版社照排室

印 刷:武汉首壹印刷厂

开本:787×1092 1/16

印张:16.25

字数:354 000

版次:2002 年 3 月第 1 版

印次:2002 年 3 月第 1 次印刷

印数:1—5 000

ISBN 7-5609-2552-5/TP·458

定价:20.80 元

(本书若有印装质量问题,请向出版社发行部调换)

内 容 简 介

本书介绍了 C 语言的基础知识、算法的各种描述以及结构化程序设计方法；程序的顺序结构、分支结构和循环结构；数组、函数和变量的存储类型；指针型数据类型；结构体、共用体、枚举及自定义类型；文件的基本概念，文件的各种操作方式和 C 语言的编译预处理、位运算的特色。书末附有常用字符与 ASC II 代码对照表、运算符的优先级和结合性及常用的 C 库函数。

本书内容详尽，由浅入深，范例经典，重点突出，可满足不同层次读者的需求。

本书既可作本科院校相关专业的教材，也可作高职高专相关专业的课程教材，还可供读者自学使用。

面向 21 世纪计算机教材出版指导委员会

主任 陈火旺（中国科学院院士）

沈绪榜（中国科学院院士）

邹寿彬（华中科技大学博士生导师）

委员（以姓氏笔画为序）

王长胤 韦 敏 卢开澄 卢正鼎

张 峰 何炎祥 苏锦祥

秘书 沈旭日

高职、高专计算机系列教材编委会

顾问 胡金柱 瞿 坦

编委（以姓氏笔画为序）

王绍卜 毛芳烈 王前新 叶远谋 刘小芹 向显智

张学礼 张桂宁 张栢勤 李家瑞 邹依琴 唐铸文

黄元山 黄东峰 程时兴 覃晓康 黎秋萍

秘书 曾 光 谢燕群

前 言

C 语言是介于高级语言与汇编语言之间的中级语言,因为 C 语言既具有高级语言的特点,又具有低级语言的功能,其功能强大,使用广泛。因此,我国绝大部分高等院校都以 C 语言作为学生程序设计入门的一种语言,同时,C 语言也为进一步学习 C++以及 Visual C++奠定了基础。

本书共有 13 章和 3 个附录。第 1 至第 3 章介绍了 C 语言的基础知识、算法的各种描述以及结构化程序设计方法。第 4 至第 5 章介绍了程序的控制结构:顺序结构、分支结构和循环结构。第 6 至第 8 章介绍了数组、函数和变量的存储类型。第 9 章介绍了指针型数据类型,这是学习 C 语言的一个难点。第 10 章介绍了结构体、共用体、枚举及自定义类型。第 11 章介绍了 C 语言文件的基本概念,文件的各种操作方式。第 12 至第 13 章是 C 语言的两大特色:编译预处理和位运算。3 个附录分别是:常用字符与 ASCII 代码对照表、运算符的优先级和结合性及常用的 C 库函数。

本教材由刘天印、贾学斌主编。第 1 章由祁文青编写;第 2 至第 3 章由王秀章编写;第 4 章由唐波编写;第 5 至第 6 章由胡大威编写;第 7 至第 8 章由贾学斌编写;第 9 章由黄国军编写;第 10 章由余云霞编写;第 11 章由杜华兵编写;第 12 至第 13 章及附录由刘天印编写。以上各编者都是具有丰富教学经验的老师。

本教材叙述精练、重点突出、体系合理、便于教学、通俗易懂。每章中的例子以及习题都经过了精心的筛选,富有启发性,有助于读者深入理解所学内容。

由于编者水平有限,时间仓促,书中不足之处在所难免,恳请读者和专家提出宝贵意见。最后,我们衷心地希望读者在本教材的指导下取得更大的收获。

编 者
2002 年 1 月

目 录

第 1 章 C 语言概述.....	(1)
1.1 C 语言简史	(1)
1.2 C 语言的特点	(1)
1.3 C 语言的基本程序结构	(2)
1.4 C 语言的编写	(4)
1.5 C 语言的编译和运行	(5)
习题一	(7)
第 2 章 算 法.....	(8)
2.1 算法的概念	(8)
2.1.1 基本概念	(8)
2.1.2 简单算法举例	(9)
2.2 算法的特性	(11)
2.3 算法的描述	(12)
2.3.1 用自然语言描述	(12)
2.3.2 用流程图表示算法	(12)
2.3.3 三种基本结构和改进的流程图	(15)
2.3.4 用 N-S 流程图表示算法	(16)
2.3.5 用 PAD 图描述算法	(18)
2.3.6 用伪代码表示算法	(19)
2.3.7 用计算机语言实现算法	(20)
2.4 结构化程序设计方法	(21)
习题二	(22)
第 3 章 基本数据类型及运算	(23)
3.1 基本数据类型及运算	(23)
3.2 常量与变量	(23)
3.2.1 标识符与关键字	(23)
3.2.2 常量和符号常量	(24)
3.2.3 变 量	(25)
3.3 整型数据	(26)
3.3.1 整型常量的表示方法	(26)
3.3.2 整型变量	(27)

3.3.3 整型常量的类型.....	(31)
3.4 实型数据.....	(31)
3.4.1 实型常量的表示方法.....	(31)
3.4.2 实型变量.....	(31)
3.4.3 实型常量的类型.....	(33)
3.5 字符型数据.....	(33)
3.5.1 字符常量.....	(33)
3.5.2 字符变量.....	(34)
3.5.3 字符数据在内存中的存储形式及其使用方法.....	(34)
3.5.4 字符串常量.....	(36)
3.6 变量赋初值.....	(36)
3.7 运算符和表达式.....	(37)
3.7.1 运算符简介.....	(37)
3.7.2 算术运算符和算术表达式.....	(38)
3.7.3 赋值运算符和赋值表达式.....	(40)
3.7.4 关系运算符和关系表达式.....	(41)
3.7.5 逻辑运算符和逻辑表达式.....	(42)
3.7.6 逗号运算符和逗号表达式.....	(44)
3.7.7 运算优先级与结合性.....	(45)
3.8 数据类型转换.....	(46)
3.8.1 各类数值型数据间的混合运算与类型转换.....	(46)
3.8.2 赋值运算中的数据类型转换.....	(46)
3.8.3 强制类型转换.....	(48)
习题三.....	(50)
第 4 章 数据的输入输出.....	(51)
4.1 格式输出函数 printf.....	(51)
4.1.1 printf 函数的一般格式.....	(51)
4.1.2 格式字符.....	(52)
4.2 字符输出函数 putchar.....	(57)
4.3 格式输入函数 scanf.....	(58)
4.3.1 scanf 函数的一般格式.....	(58)
4.3.2 格式说明.....	(59)
4.3.3 scanf 函数执行中常见错误.....	(60)
4.4 字符输入函数 getchar.....	(61)
4.5 程序举例.....	(62)
习题四.....	(63)

第 5 章 控制语句	(65)
5.1 概述	(65)
5.2 if 语句	(66)
5.2.1 if 语句的三种形式	(66)
5.2.2 if 语句的嵌套	(69)
5.2.3 例题	(71)
5.2.4 条件运算符	(74)
5.3 switch 语句	(74)
5.4 while 语句	(78)
5.4.1 while 语句	(78)
5.4.2 例题	(79)
5.5 do-while 语句	(80)
5.5.1 do-while 语句	(80)
5.5.2 while 语句和 do-while 语句的区别与联系	(81)
5.6 for 语句	(83)
5.6.1 for 语句	(83)
5.6.2 例题	(86)
5.6.3 三种循环的比较	(87)
5.7 循环嵌套	(87)
5.8 break 语句、continue 语句和 goto 语句	(89)
5.8.1 break 语句	(89)
5.8.2 continue 语句	(91)
5.8.3 goto 语句	(91)
5.9 程序举例	(92)
5.9.1 例题	(92)
5.9.2 常见错误	(95)
习题五	(97)
第 6 章 数组	(99)
6.1 一维数组	(100)
6.1.1 一维数组的定义	(100)
6.1.2 一维数组的引用	(101)
6.1.3 一维数组的初始化	(102)
6.1.4 例题	(103)
6.2 二维数组	(106)
6.2.1 二维数组的定义	(106)
6.2.2 二维数组的引用	(107)
6.2.3 二维数组的初始化	(108)

6.2.4 例题	(109)
6.3 字符数组	(110)
6.3.1 字符数组的定义	(110)
6.3.2 字符数组的初始化	(111)
6.3.3 字符数组的引用	(112)
6.3.4 字符串和字符串结束标志	(112)
6.3.5 字符数组的输入输出	(113)
6.3.6 字符串处理函数	(114)
6.3.7 例题	(115)
6.4 程序举例	(116)
6.4.1 例题	(116)
6.4.2 常见错误	(118)
习题六	(119)
第 7 章 函数	(121)
7.1 函数的定义	(121)
7.1.1 函数的结构	(121)
7.1.2 函数的定义	(122)
7.2 函数的参数与返回值	(123)
7.2.1 返回语句	(123)
7.2.2 函数的参数	(123)
7.3 函数的调用	(125)
7.3.1 函数调用的一般形式	(125)
7.3.2 函数调用的方式	(125)
7.3.3 被调函数的说明	(127)
7.4 函数的嵌套调用	(129)
7.5 函数的递归调用	(130)
7.6 函数与数组	(133)
7.6.1 一维数组名作实参	(133)
7.6.2 二维数组名和指针数组作实参	(134)
7.7 程序举例	(135)
习题七	(137)
第 8 章 变量的存储类型	(138)
8.1 自动变量 (auto)	(138)
8.2 外部变量 (extern)	(140)
8.3 静态变量 (static)	(144)
8.4 寄存器变量	(147)
8.5 变量的初始化	(148)

8.6 程序举例.....	(149)
习题八.....	(151)
第 9 章 指针	(153)
9.1 指针的概述.....	(153)
9.2 指针变量.....	(155)
9.2.1 指针变量的类型说明.....	(155)
9.2.2 指针变量的赋值.....	(155)
9.2.3 指针变量的引用——指针操作符和指针表达式	(156)
9.2.4 指针变量作为函数的参数.....	(161)
9.3 数组与指针	(164)
9.3.1 指向数组元素的指针.....	(164)
9.3.2 指针和数组的关系	(165)
9.3.3 指针的下标	(167)
9.3.4 数组名和指针变量作函数参数	(169)
9.4 字符串与指针	(171)
9.4.1 字符串指针变量的定义说明与使用	(171)
9.4.2 使用字符串指针变量与使用字符数组的区别.....	(174)
9.5 函数与指针	(175)
9.5.1 函数指针变量	(175)
9.5.2 指针型函数	(176)
9.6 指针数组和指向指针的指针	(177)
9.6.1 指针数组.....	(177)
9.6.2 指针数组作为 main 函数的参数	(179)
9.6.3 指向指针的指针.....	(180)
习题九.....	(184)
第 10 章 结构体、共用体、枚举及类型定义.....	(185)
10.1 结构体的概念与定义.....	(185)
10.2 结构体数组.....	(186)
10.2.1 结构体数组的定义.....	(186)
10.2.2 结构体数组的初始化	(187)
10.3 结构体与函数.....	(188)
10.3.1 结构体变量作为函数参数	(188)
10.3.2 返回结构类型值的函数.....	(190)
10.4 结构体与指针	(191)
10.4.1 指向结构体变量的指针.....	(191)
10.4.2 指向结构体数组的指针.....	(193)
10.4.3 用指向结构体的指针作函数参数	(195)

10.5 用指针处理链表	(196)
10.6 共用体	(198)
10.6.1 共用体的概念	(198)
10.6.2 共用体变量的引用方式	(199)
10.7 枚举类型	(200)
10.8 定义函数——typedef	(201)
习题十	(203)
第 11 章 文件	(205)
11.1 文件概述	(205)
11.1.1 文件的概念	(205)
11.1.2 缓冲文件系统和非缓冲文件系统	(206)
11.2 文件(FILE)类型指针	(207)
11.3 文件的打开与关闭	(208)
11.3.1 文件的打开(fopen 函数)	(208)
11.3.2 文件的关闭(fclose 函数)	(210)
11.4 文件的顺序读写	(210)
11.4.1 输入和输出一个字符	(210)
11.4.2 输入和输出一个字符串	(211)
11.4.3 格式化的输入和输出	(211)
11.4.4 按“记录”的方式输入和输出	(212)
11.5 文件的定位	(212)
11.6 文件操作的出错检测	(213)
11.7 程序举例	(214)
习题十一	(219)
第 12 章 编译预处理	(220)
12.1 编译预处理的概念	(220)
12.2 宏定义	(220)
12.2.1 不带参数的宏定义	(220)
12.2.2 带参数的宏定义	(223)
12.3 文件包含	(226)
12.4 条件编译	(228)
习题十二	(230)
第 13 章 位运算	(231)
13.1 位运算概述	(231)
13.2 位运算符与位运算	(231)

13.2.1 “按位与”运算	(232)
13.2.2 “按位或”运算	(233)
13.2.3 “按位异或”运算	(233)
13.2.4 “按位取反”运算	(234)
13.2.5 “左移”运算	(235)
13.2.6 “右移”运算	(235)
13.2.7 位运算赋值运算符	(235)
13.3 位运算应用	(235)
13.4 位段	(238)
习题十三	(239)
附录 I 常用字符与 ASCII 代码对照表	(240)
附录 II 运算符的优先级和结合性	(241)
附录 III 常用的 C 库函数	(242)
参考文献	(246)

第 1 章

C 语言概述

1.1 C 语言简史

C 语言与 UNIX 操作系统互相依存，二者相互促进并得以发展。UNIX 操作系统是美国贝尔实验室的 K. Thompson 和 D. M. Ritchie 于 1971 年设计成功的，第 1 版是在 GE635 机器上产生并通过纸带把可执行代码传送到 PDP-7 上的，后又以汇编语言编写，由于汇编语言不可移植，描述问题效率不高，且可读性差，故 K. Thompson 决定开发一种高级语言来描述 UNIX 操作系统。这就是 C 语言产生的历史背景。

1972 年 C 语言研制成功并投入使用，1973 年 K. Thompson 和 D. M. Ritchie 把 UNIX 操作系统用 C 语言进行改写。UNIX 操作系统由于使用了 C 语言而取得成功，几乎成为 16 位微机的标准操作系统，C 语言一诞生就由于编制 UNIX 操作系统的成功而引起人们的关注。

1973 年之后，C 语言的发展相当迅速：1975 年 UNIX 第 6 版公布，1977 年又研制成功不依赖具体机器的 C 语言编译文本——“可移植 C 语言编译程序”，推动了 C 语言在各种机型上的广泛应用；UNIX 第 7 版在 1978 年研制成功。以其中的 C 编译系统为基础，美国国家标准化协会于 1983 年制定了一个 C 语言标准草案，即 83 ANSI C，1987 年又公布了 87 ANSI C。

目前，C 语言编译系统有多种版本，在微机上常用的有 Microsoft C，Turbo C，Quick C 等。不同的版本各略有差异，因此，读者应了解所用计算机系统 C 语言编译的特点和规定（可参阅有关手册）。

美国 Borland 公司在 Turbo C 的基础上，研制成功了面向对象的程序设计语言 Turbo C++，并应用于 IBM PC 上。Turbo C++ 与 Turbo C 高度兼容，但在功能方面前者有更多的扩充。

1.2 C 语言的特点

C 语言是一种高级语言，同时它又提供了类似于汇编语言的低级语言的功能，如它可以直接访问物理地址并能进行位操作，为编写系统程序提供了可能。

C 语言程序生成的目标代码质量高，效率只略低于汇编语言，所以执行效率高。

C 语言程序可移植性好，即容易从一种类型的计算机系统移植到另一种类型的计算机系统。

C 语言设有预处理功能，它设置了预处理命令，如#define、#include 等，在编译源程序时先处理这类命令，以提高程序的可读性和可移植性，并且为调试程序提供了方便。

C 语言源程序是由一个或多个函数组成的，函数段之间是相对独立的。C 语言还提供了丰富的库函数，包括图形函数等，可供用户调用。用户自编函数可使用系统提供的九种控制语句。程序结构清晰、易编、易读。

C 语言提供的数据类型丰富，拥有可以实现现代化语言的各种数据结构，如指针类型使用十分灵活，用它可以构成链表、树、栈等。

C 语言提供了 40 多种运算符，包括位运算符，可以完成其它高级语言难以完成的运算。

综上所述，C 语言是一种功能很强的语言。但是，它也有一些不足之处：C 语言语法限制不严谨，对于熟练的程序员编程灵活，但安全性低；运算符丰富，完成功能强，但难记、难掌握。因此，学习、使用 C 语言不妨先学基本部分，先用起来，用熟练后再学不规范的语法规则，进而全面掌握 C 语言。

1.3 C 语言的基本程序结构

① 程序的总体结构。一个 C 语言源程序是由一个主函数或者再加上若干个函数段组成的，程序的开始部分可以根据程序的需要，写出以“#”为首字符的编译预处理行和全程序都使用的变量说明，其后是若干函数段。每个段的最前面是函数名。主函数名是 main，其它函数段由程序员命名。函数名后是由圆括号“()”括起来的形式参数表，圆括号的后面是每一个形式参数的类型说明。再后就是由花括号“{ }”括起来的函数主体了。

② 函数段内的内容。函数段内可以有两部分内容：说明部分和可执行部分（语句部分）。说明部分中可以包括多个说明项，用以说明函数中使用的变量和需调用的其它函数等。语句部分可以包括多个可执行语句。

③ 各语句之间用分号“;”分隔开；并列的标识符或项之间用逗号“,”分隔；两个关键词相邻时，中间用空格（至少一个）相间。

④ C 语言在书写时是比较自由的，几个说明项或几个语句可以写在一行，一个语句可以分为几行写，但是一个词或一个数不能分两行写。至于格式主要考虑程序易读和便于程序的维护。

⑤ 为了使程序易读，在程序中可设置注释部分，注释的内容写在/* 和 */之间，程序运行时系统忽略其间的内容。

例 1.1 一个不执行任何功能的程序。

程序如下：

```
main()
{
}
```

程序运行时，什么也不做。

例 1.2 编写 C 语言程序，打印一行字符。

程序如下：

```
main ()
{
    printf("I WISH YOU SUCCESS! ");
}
```

运行这个程序将输出双引号中间的一行字符。printf 是函数库中的一个函数，此处是调用库函数完成输出功能。

例 1.3 输入 a、b、c 三个整型量，输出其中最大者。

```
#include <stdio.h>          /* 编译预处理：标准输入输出头文件 */
void main()
{
    int smax(int x,int y,int z);
    int a,b,c,max;
    scanf("%d,%d,%d",&a,&b,&c); /* 标准输入函数调用 */
    max=smax(a,b,c);
    printf("a,b,c=%d,%d,%d;max=%d\n",a,b,c,max); /*标准输出函数调用*/
}
int smax(int x,int y,int z)    /* 用户自定义函数 */
{
    int k;
    if(x>y) k=x;
    else k=y;
    if(z>k) k=z;
    return(k);
}
```

运行结果：

输入：8, 4, 9 （按回车键）

输出：a, b, c=8, 4, 9; max=9

本程序中包括两个函数：main 和 smax。在函数 main 中，前两行是说明部分，后面的多行是可执行的语句部分。在函数 smax 中，在括号中是形式参数 x、y 和 z 的类型说明，花括号中第一行是变量的类型说明，后面四行是可执行的语句部分。

从以上三个程序可以看出，要学习使用 C 语言编程序，首先要学习程序的结构：如何将函数组成一个完整程序，程序中使用的参数（形式参数和实在参数）、变量如何说明，还要学习使用各种可执行的语句，了解它们如何控制计算机完成运算。

1.4 C 语言的编写

在 1.3 节中举出的三个用 C 语言编制的程序虽然繁简不同，但都称为 C 语言源程序、C 语言程序或 C 语言的用户程序。下面将介绍程序员编制 C 语言的用户程序的过程，随着读者对语法规则的学习，将逐步加深对此步骤的理解。

需要用 C 语言编制程序来处理某一问题时，无论问题的繁简，都应经过以下几个步骤：

① 问题定义：首先要回答的关键问题是“要解决的问题是什么”、“要处理的对象（数据）是什么”。即使是初学者的练习题，也要把这些问题搞清楚。如果是用户提出的问题，程序员还需要和用户协调，甚至多次的协调以达到共识。要将问题写成清楚的文字材料，这是程序文档的第一部分。

② 可行性研究：对前面所确定的问题“是否有行得通的解决办法”。这个阶段只需在高层次、较抽象级上进行研究，研究此问题是否可能解决和是否值得解决，并估计研制程序或软件系统的成本效益。

③ 需求分析：在上述 1、2 步骤的基础上，程序员得出问题的逻辑模型，通常用功能图表和文字说明形成“文档”。所谓“文档”就是程序的有关文字资料和程序。

④ 一般设计：这个阶段由程序员考虑应该“如何解决这个问题”。有的人说：“计算机是万能的，什么问题都可以解决，怎么还要我来想解决问题的办法呢？”这种说法是不正确的，因为即使计算机处理数据速度快，存储量大——记忆好，可以处理各种数据，解决很多问题，但这一切必须在程序的控制下工作，而程序是由程序员编制的。

程序员设计程序时必须具备如下条件：熟悉 C 语言语法，熟悉可能利用的库函数，熟悉待解决的问题。

下面介绍考虑具体解决方案时的指导思想：

① 如果要解决的问题比较简单，则用一个模块（一个函数）即可以解决。如果要解决的问题比较复杂，则考虑分为多个模块（例如在例 1.3 中分为 main 和 smax 两个函数），根据题目的复杂程度，还可以继续分下去，这就是目前推荐的逐步求精的程序设计方法。这样设计的程序符合结构化程序设计的思想，得到的程序结构性好。

② 现代程序要求要有友好的人机界面，要充分利用库函数中的输出函数和图形函数等，使程序在运行过程中能给用户醒目的提示和必要的干预，并且输出中间结果和最终结果。

③ 在设计一个系统时，设计方案不止一种，功能要求越完美，程序将越复杂，软件成本越高。在设计实际程序系统时，必须估计各种方案的成本和效益，充分考虑到成本/效益比，权衡利弊，选取一种最佳方案。

④ 详细设计和编码。根据第 3 步中确定的方案，进一步解决“具体怎么实现”。这一步，包括设计出程序的详细规格说明和编写程序，并对编写的每个模块进行测试。

⑤ 综合测试。这一阶段主要是通过总体测试以解决该程序是否能完成预期的功能。这一阶段是把已通过了单元测试的模块按既定的策略装配起来，在装配过程中对程序进行必要的测试，即集成测试。最后，按系统功能说明书由用户（或在用户参与下）进行验收