高等教育系列教材
（计算机与信息管理类）

高级语言程序设计

high-level language programming

总主编　彭　波
本书编著　彭　波　等
崔永普
策　划　水木时代

电子科技大学出版社

高等教育系列教材(计算机与信息管理类)

高级语言程序设计

彭　波　　总主编
彭　波
崔永普　等　编　著
水木燕园　策　划

电子科技大学出版社
2003·成都

图书在版编目(CIP)数据

高级语言程序设计/彭波,崔永普等编著. —成都:电子科技大学出版社,2003.3

高等教育系列教材(计算机与信息管理类)

ISBN 7-81094-034-1

Ⅰ.高… Ⅱ.①彭… ②崔… Ⅲ.高级语言-程序设计-高等学校-教材 Ⅳ.TP34

中国版本图书馆 CIP 数据核字(2003)第 02340 号

高等教育系列教材(计算机与信息管理类)

高级语言程序设计

彭 波 崔永普 等编著

出 版:电子科技大学出版社(成都市建设北路 2 段 4 号 邮编 610054)

责任编辑:张致强

发 行:新华书店

印 刷:安徽省蚌埠市方达印刷厂

开 本:787×960 1/16 **印张**:19.75 **字数**:356

版 次:2003 年 3 月第一版

书 号:ISBN 7-81094-034-1/TB·21

印 数:1—5000 册

定 价:29.80 元

编审说明

C言是近年来在国内外得到迅速推广应用的一种现代语言。C语言功能丰富，表达能力强，使用灵活方便，应用面广，目标程序效率高，可移植性好，既具有高级语言的优点，又具有低级语言的许多特点，因此特别适合编写系统软件。C语言诞生后，许多原来用汇编语言编写的软件，现在可以使用C语言来编写了。而学习和使用C语言要比学习和使用汇编语言容易得多。

作者根据多年的教学和写作经验，从程序设计的角度出发，将C语言写得深入浅出，易于掌握。本书共分十二章内容，主要包括：概述，程序设计基本方法，数据类型、运算符及表达式，逻辑运算和判断选取控制，循环控制，数组，函数，编译预处理，C语言指针，结构体和共用体，位运算及文件。

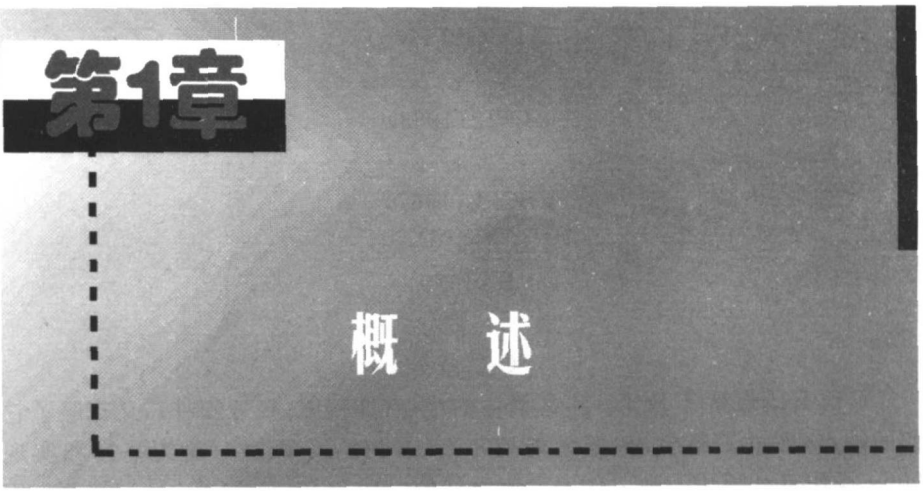
本书体系合理，概念清晰，例题丰富，逻辑性强，文字流畅，通俗易懂，是一本学习C语言的好教材。经审定，本书可以作为高等院校计算机与信息管理类专业教材，也可供广大计算机爱好者自学使用。

由于编写时间仓促及作者学识有限，本书难免存在疏漏和不妥之处，敬请有关专家和广大读者不吝赐教，以便不断修订完善。

高等教育系列教材编审指导委员会

2003年3月

目　录



1.1 C语言的发展历史

C语言是在B语言的基础上发展起来的，它的根源可以追溯到ALGOL 60语言。1960年出现的ALGOL 60语言是一种面向问题的高级语言，它离硬件比较远，不宜用来编写系统程序。1963年，英国的剑桥大学推出了CPL(Combined Programming Language)语言。CPL语言在ALGOL 60语言的基础上更接近硬件一些，但是规模比较大，难以实现。1967年，英国剑桥大学的Matin Richards对CPL语言作了简化，推出了BCPL(Bootstrap Combined Programming Language)语言。1970年，美国贝尔实验室的Ken Thompson(来自伯克里加洲大学University of California Berkley)以BCPL语言为基础，又作了进一步简化，设计出了很简单的而且很接近硬件的B语言(取BCPL的第一个字母)，并且使用B语言写了第一个Unix操作系统，在PDP－7上实现。1971年，在PDP－11/20上实现了B语言，并且写出了Unix操作系统。但是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语言的家谱如下所示：

ALGOL(1960)
↓
CPL (1963)
↓
BCPL(1967)
↓
B (1970)
↓
C (1972)

C语言既保持了BCPL和B语言的优点(即精练，接近硬件)，又克服了它们的缺点(即过于简单，数据无类型等)。最初的C语言只是为描述和实现Unix操作系统提供一种工作语言而设计的。1973年，D. M. Ritchie把原来用汇编语言写成的Unix系统中90%的部分改由C语言重写，并在PDP－11型小型计算机上用C语言及少量汇编语言完成了沿用至今的Unix操作系统。Unix系统改用C语言写成后，既有利于Unix系统向其他不同类型计算机的移植，使各种类型的计算机都能够使用Unix操作系统，同时，又促进了C语言的不断发展和普及。

此后，C语言作了多次改进，但主要还是在贝尔实验室内部使用。直到1975年Unix第6版公布后，C语言的突出优点才引起人们的普遍注意。1977年出现了不依赖于具体机器的C语言编译文本《可移植C语言编译程序》，使C移植到其他机器时所需做的工作大大简化了，这也推动了Unix操作系统迅速地在各种机器上实现。例如VAX、AT&T等计算机系统都相继开发了Unix。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。

尽管C语言是与Unix系统同时发展起来的，但它已经是一种不单是能在Unix环境下工作的语言了。1978年以后，C语言先后移植到大、中、小、微型计算机上，独立于Unix和PDP，发展成为能在多种机型、多种操作系统，尤其是能在微型计算机及其操作系统的支持下进行程序设计的一种功能极强的通用计算机语言。现在C语言已风靡全世界，成为世界上应用最广泛的几种计算机语言之一。

以1978年发表的Unix第7版中的C编译程序为基础，Brian W. Kernighan和Dennis M. Ritchie(合称K&R)合著了影响深远的名著《The C Programming Language》，在这本书中介绍的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年修改了他们的经典著作《The C Programming Langnage》，按照ANSI C标准重新写了该书。1987年，ANSI又公布了新的标准：87ANSI C。1990年，国际标准化组织ISO(International Standard Organization)接受87ANSI C为ISO C的标准(ISO 9899—1990)。以前流行的C编译系统都是以它为基础的。

本篇的叙述基本上以ANSI C为基础。目前广泛流行的各种版本C语言编译系统虽然基本部分是相同的，但也有一些不同。在微型计算机上使用的有Microsoft C，Turbo C，Quick C，Borland C等。

1.2 C语言的特点

C语言是一种相对比较“低级”的语言，它具备一些过去只有汇编语言才能够实现的功能。起初，C语言主要是用来替代汇编语言编写操作系统、编译器、数据库以及机械控制程序等。随着它自身的不断强化和完善，更由于它具有处理速度快、操作方便等优点，今天已经在科学及工程应用领域中受到了普遍的重视和欢迎，并且在这些领域中发挥了积极的作用。归纳起来，C语言主要具有如下特点：

1.2.1 语言简洁紧凑，使用方便灵活

C语言简洁、紧凑，使用方便、灵活。C语言的关键字很少，标准情况下只有29个，是一个比较小的语言体系，是现有程序设计语言中规模最小的语言之一。小的语言体系往往能够设计出较好的程序。

1.2.2 运算符丰富，使用方法独特

C语言运算符丰富，使用方法独特，如位操作、移位操作等；多个语句(例如间接地址计算等)还可以并入一条语句或一个表达式中。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。

1.2.3 具有极强的数据定义方法

C语言具有极强的数据定义方法，几乎可以定义所有类型的数据；并且数

据类型之间的相互转换也十分自由；能够用来实现各种复杂的数据结构(例如链表、树、栈等)的运算。

1.2.4 具有完善的流程控制结构

C语言具有完善的流程控制结构，从而保证了程序的结构化、简易化，并且易于更新、维护。

1.2.5 以函数作为程序的模块单位

C语言以函数作为程序的模块单位。这样，编程者就可将一个大型程序分割成若干部分且分别由不同的人员同时编写。因此C语言也是一种具有结构化和模块化特性的语言。

1.2.6 同时具有高级语言及低级语言的功能

C语言中可以使用指针，允许直接对硬件进行操作，能进行位(bit)操作，这就使它具备了其他高级语言都没有的近似于汇编语言的功能。因此，C既具有高级语言的功能，又具有低级语言的许多功能，可用来写系统软件。

1.2.7 允许使用递归函数结构且具有宏定义功能

C语言允许使用递归函数结构，还具有宏定义功能，这一特性极大地丰富了C语言的表达能力。

1.2.8 生成目标代码质量高且程序执行效率高

C语言生成目标代码质量高，程序执行效率高，一般只比汇编程序生成的目标代码效率低10%～20%。

1.2.9 允许用户将自己定义的新命令加入语言之中

C语言允许用户将自己定义的新命令加入语言之中。实际上，用C语言编写的程序完全是由模块化的函数集合所组成的，用户对函数的定义本身就可以看做是对语言的一种扩充。

1.2.10 容易移植

C语言是非常容易移植的。这主要是因为C语言本身比较小，并且C语言从一开始是产生于小型计算机，同时，还由于C语言本身没有输入输出功能，输入输出是通过库函数来实现的。

当然，C语言并不是完美无缺的，例如C语言没有自动检查数组边界的功能，有些符号，诸如“*”和“=”具有多重用途等等，因此，在使用中往往容易引起一些不必要的错误，稍不注意就可能导致意想不到的结果。

1.3 C语言程序的开发过程

一般说来，开发汇编语言程序有5个步骤。

1.3.1 设计

根据任务要求，设计程序结构、算法，画出流程图。

1.3.2 编辑

根据设计方案创建C语言源程序。该文件一般以.C作为扩展名。它由程序员通过文本编辑器来完成。这里的文本编辑器是任何流行的字处理器或编辑器，只要它们可以产生纯ASCII码文本文件，不带有任何特殊控制码和格式码。DOS下常用的有Edit、PE2、WordStar等。

1.3.3 编译

将C语言源程序转化为机器可以执行的目标程序。C编译器用于完成这步工作。编译器可以发现源程序中的语法错误，如格式错误、非法调用、数据类型不匹配等，提示程序员对源程序加以修改，并重新编译。编译后生成以.OBJ为扩展名的目标文件。

1.3.4 连接

目标程序(.OBJ)必须经过连接器与C的运行库连接，生成附带重定位信息的可执行程序(EXE文件或COM文件)才能运行。

连接器任务之一就是将目标模块(.OBJ)与C的运行库连接，产生一个运行模块。连接器的第二个任务是可将几个目标模块结合成一个可执行的模块，并同样使其具有可重定位性。这样，你可以一次编写程序的一部分，通过结合，将几个目标模块合并，产生一个完整的运行模块。

1.3.5 调试

完成设计、编辑、编译、连接以后得到的可执行程序中可能存在一些错误(Bug),这些错误可能是设计缺陷或者编辑时的笔误,它们可能使程序运行出现结果错误、死机甚至其他无法预测的后果。我们称这类错误为逻辑错误,它们是编译器和连接器都无法发现的。这类错误只能靠程序员凭借调试器对可执行程序进行耐心细致地跟踪调试,才能够逐一排除。

从20世纪80年代起,一些先进的计算机软件公司推出了集成开发环境IDE(Integraded Development Environment)的C编译系统。将编辑、编译、连接、运行、调试集中到一个的界面里,使C语言的开发过程更为简便。其中,典型的系统有Borland公司的Turbo C/C++、Borland C/C++,Microsoft公司的MSC/C++6.0、Visual C/C++等。

1.4 C语言程序的基本结构

C语言程序的基本结构可以归结为如下所示的形式:

```
[全局变量定义]
main ([形式参数])
[形式参数说明]
{
    [局部变量定义]
    程序语句
}
[全局变量定义]
函数名([形式参数])
[形式参数说明]
{
    [局部变量定义]
    程序语句
}
……
```

方括号([ ])里的内容表示可省略项。

可见,C 语言的程序完全是由函数构成的,其中 main()代表一个特殊的函数,程序将首先从这里开始执行。

程序中的全局变量是指在程序的首部(函数的外面)或部分函数的前面所定义的变量或常量。在程序首部定义的全局变量可供程序中所有函数共同使用,而在部分函数之前定义的全局变量则从定义之处起对后面所有的函数有效。

局部变量是指在函数内部或以"{}"括起来的功能块内部定义的变量,函数(功能块)内定义的局部变量只在该函数(功能块)内有效。局部变量允许与全局变量同名,但在这种情况下,局部变量的优先级较高,即全局变量在该功能块里暂时被屏蔽。

C 语言的程序中的函数可以嵌套调用,但不能嵌套定义。

1.5 C 语言程序的书写格式

C 语言对程序的书写格式几乎没有任何限制,可以采用完全自由的形式,对语句位置的唯一要求就是预处理命令必须放在程序的最前面。

尽管如此,为使程序易于理解、具有良好的可读性、便于程序的维护和修改,在编程中还是应该遵循一定的格式。

下面我们通过一个最简单的例子来看看应该如何书写 C 语言程序。

【例 1-1】 打印输出"Hello World !"的程序

```
#include   <stdio.h>                    /*第1行*/
main()                                  /*第2行*/
{                                       /*第3行*/
    printf ("Hello World !");           /*第4行*/
}                                       /*第5行*/
```

本程序的结果是输出以下一行信息:

```
Hello World!
```

1.5.1 预处理命令

通常,程序的开头是预处理命令,例如第一行的 #include<stdio.h>。stdio.h 被称为"头文件",它包含有程序在编译时的必要信息。该预处理命令行通知编译器在编译时先读入 stdio.h 文件一起进行编译。一般 C 编译系统都会提供若干个不同用途的头文件。

1.5.2 main( )函数

程序的正文，由 main( )函数开始，这是一个特殊的函数。C 语言规定程序的执行从 main( )函数开始，函数体由花括号“{”和“}”括起来，函数中每条语句结束时都要加上分号(;)进行标识。

1.5.3 注释

在程序中加入适当的注释会有助于对程序的阅读理解和维护。C 的注释由“/ * ”开始，由“ * /”结束，具有如下所示的形式：

```
/*    注释    */
```

也可以写成多行，例如：

```
/*
注释
……
注释
*/
```

由于在编译时编译器会自动跳过注释部分，因此注释可以放在程序中的任何位置。

1.5.4 大写字符与小写字符

在 C 语言程序中既可以使用大写字符，也可以使用小写字符。一般的习惯是在正常情况下采用小写字符，而对一些具有特殊意义的变量或常数则采用大写字符。但是必须注意，大写和小写是有区别的，即在 C 语言中同一字符的大写和小写会被认为是两个不同的字符，例如 Abc 和 abc，在编译时将被解释成两个不同的变量。

1.5.5 编程格式

前面已经介绍，C 语言程序的书写方式十分自由，也就是说，可以采用任意的方式编写程序。但是，如果在编程时不注意程序的书写格式，即使是本人，恐怕在过了一段时间之后也难以明白原来程序的内容，更不用说要让他人来阅读或修改程序了。

为了使程序便于理解，设计时应该注意以下几个问题：

(1)使用模块化结构,将不同的处理功能分别做成不同的函数。

(2)一个函数的处理内容不宜过多,应保持函数的"小型化"。

(3)在一个函数内部,按功能块分层次"缩格"书写。

1.5.6 编程风格

良好的编程风格有助于程序的阅读、理解和维护,所以一开始就应该养成良好的编程风格。

C是一种"自由式"的语言,其意是说,只要语法正确,用什么格式、用什么风格都没有什么关系。但是,某些风格习惯却使编出来的C程序易于阅读。

用花括号把语句分成组,并作函数开始和结束的标志。适当地缩进排版和适当地设置花括号的位置使C程序易于阅读和排错。

对一个函数来说,左花括号设置在下面,并与函数说明的开始部分对齐,例如在例1-1中:

```
#include <stdio.h>                    /*第1行*/
main ()                               /*第2行*/
{                                     /*第3行*/
    printf ("Hello World !");         /*第4行*/
}                                     /*第5行*/
```

左、右两个花括号都和main()函数的字母m对齐。

通过使语句缩进排版形成了main()函数的函数体。因此我们很容易通过看左、右花括号看出函数main()从什么地方开始、在什么地方结束、哪些语句构成函数体。当程序的复杂性增加时,这样做对程序的可读性大有好处。

总之,一是要注意编程风格;二是在养成良好的风格之后应长期坚持下去。

1.6 C语言的关键字

关键字也称保留字,是指在C语言中被赋予了严格定义的独特标识符。在程序文本中,关键字不允许被重新定义或作其他用途使用。表1-1按类别列出了C语言的关键字。