# C程序设计语言基础

姚传胤　编

# C 程序设计语言基础

姚传胤　编

哈尔滨工业大学出版社

## 内 容 提 要

本书详细介绍C语言程序设计的基本方法。全书共分十章，并附有C语言参考手册。第一章概述C语言的基本概念。第二章至第八章介绍C语言的基本格式和功能。第九章为帮助初学者上机提供了在CROMEMCO A档机、IBM-PC/XT和PDP-11/23计算机上运行C语言程序的范例。第十章选编了六个典型程序。

本书适于初学者和有关工程技术人员使用，并可作大专院校相应专业的教材或教学参考书。

**C程序设计语言基础**

姚传胤 编

*

哈尔滨工业大学出版社出版

新华书店首都发行所发行

哈尔滨工业大学印刷厂印刷

*

开本850×1168 1/32 印张9.875字数25.4000

1987年1月第1版 1987年1月第1次印刷

印数 1—3,000

书号 15341·36 定价 2.05元

# 前　言

C语言是通用的程序设计语言，是当今世界上最有影响的程序设计语言之一。

C语言简单、灵活、易于掌握。它具有先进的控制流和数据结构，丰富的数据类型和多种运算符以及实用表达式。它适于描述操作系统以及编制各种系统软件。它的通用性及可移植性使我们能够在不同机器上方便地移植C语言程序。C语言的编译程序简单、紧凑，由它编译的目标质量可与汇编程序的相比。现在C语言已可以取代目前使用的汇编语言。例如，著名的UNIX操作系统以前是用汇编语言编写的，现在其主要部分都改用了C语言。

C语言不具备字符串处理、输入/输出等功能，但需要时，这些功能可由不同系统提供的程序库来实现。

最早的C语言是1968年提出的CPL (Combined Programming Language) 程序设计语言。由于追求功能完备而使本身过于复杂，CPL最终未能推广。此后，在CPL的基础上以计算机科学为中心，设计了BCPL (Basic CPL) 语言。BCPL使用较广，分别用于大型机IBM370、小型机TI990、NOVA、INTERDATA、PDP－11等。1970年，Ken·汤普生以BCPL语言为基础研制出B语言，并用该语言编写了PDP-7机上的第一个UNIX操作系统。1974年，D·M·里奇将多次扩充的B语言改名为C语言，并用它编写了在PDP-11上实现的UNIX操作系统。1978年，贝尔实验室的B·W·克尼范和D·M·里奇编写了《程序设计语言C》。最近贝尔实验室又发表了面向大学的C语言扩充版$C_{++}$。

本书从使用角度阐述C语言的基本格式和功能，叙述力求通俗易懂。为便于理解和自学，除在第十章选编若干较大的典型程序以外，还在其它各章节例举了适当的程序。为帮助初学者上机

操作，第九章提供了在几种机器上运行C语言程序的范例。附录选录了UNIX第七版用户手册中的“C语言参考手册”供读者参考。

在本书的编写过程中，参考了〔美〕B·W·克尼范和D·M·里奇合编的《程序设计语言C》、〔日〕石田晴久编的《C程序设计》以及国内各兄弟院校编写的有关书刊和讲义。哈尔滨工业大学计算中心李红同志为本书编写了第四章的一部分及第八章的初稿。日本大阪大学工学部长津和秀夫教授为编写此书提供了大量的资料和帮助，在此表示衷心感谢。

本书适于初学者和有关工程技术人员使用，并可作大专院校相应专业的教材或教学参考书。

由于经验水平有限，缺点和错误在所难免，恳切希望广大读者批评指正。

编　者

一九八六年九月十八日

# 目　录

# 第一章 概 述

C 语言是随著名的 UNIX 操作系统的发展而发展起来的，它是由许多函数定义的集合形成的一种语言。用它编写的程序清晰、紧凑，而且执行效率高。在这一章我们以实际的程序例说明该语言的基本概念和形式，并介绍其中使用的变量、常数和运算符等等。

## § 1.1 C 语言的特点

下面扼要说明C 语言的特点及其仍存在的一些问题。

C 语言的主要特点：

1. 程序使用英文小写字母

例如 最简单的打印程序

```
main ( )
{
   printf ("abcdefg") ;
}
```

其中 main、printf 等都用小写字母。这比用大写字母 MAIN、PRINTF 容易书写和阅读。

2. 程序由函数 (Function) 组成[1]

譬如一个程序可由下列 函 数 (或更多的函数) main( ) 、power( ) 等组成。

1. 关于函数的概念在 §1.3 节说明。

```
main( )
{
  ……
}
power( )
{
  ……
}
```

就是说，C 语言的程序是由许多函数集合而成的，是模块化的程序结构。而且在程序中这些函数是相互独立的，可以分别进行编译。其中 main ( ) 为主函数，每一个 C 语言程序一定要有一个称为 main 的主函数，而 power （或其它函数）为被调用函数。它们之间的排列顺序是任意的。

3. 可使用指针 (pointer) [1]

C 语言中有一种变量，它含有的值是另一变量的地址，此种变量称为指针。指针具有机器语言间接寻址方式的特点。用指针编写的程序，其执行效率高。它可代替汇编语言。

4. 具有丰富的运算符

C 语言中有许多简洁的运算符。例如赋值运算 y=y+x，可用 y+=x 代替；增1运算 y=y+1，可用 y++ 代替。除此还有按位运算符（&、|、∧）、二进制移位运算符（ ≫、≪ ）以及逻辑运算符（!、&&、||）等。

5. 具有先进的控制流

所谓控制流，就是程序中规定的计算顺序。C 语言中计算顺序的表示是十分紧凑的，例如表示程序控制流的循环语句

```
while ((C=getchar( )) != EOF)
  {
    ……
  }
```

1. 指针的概念在第四章中详细说明。

```
for (i = 0 , j = 100 ; i<j ; i++, --j )
    {
        ……
    }
```

以及 if (…) 、else (…) 、switch (…) 、do-while (…) 等语句。

上例中花括号{ }表示函数中语句（以及复合语句）的开始和结束，它类似于 Algol 中的 begin—end 和 PL/1 中的DO—END。

6. C 编译中有预处理程序

这种功能的意义在于通过预处理程序可扩充某些语言功能。最常用的有宏替换和文件包含。

宏替换：例如

```
#define EOF -1
```

的定义是宏替换中的一种形式，表示在程序中遇到名字 EOF 时用字符串 -1 来替换。

文件包含：为了便于处理一组#define 和说明，C 语言有包含文件的能力。即凡写有

```
# include "mysource.c"
```

的控制行，在程序中都用标有 mysource.c名字的文件内容来替换。

7. C 语言的编译程序非常方便

UNIX 中C 语言的编译程序所用的命令（cc）是很方便的。它可以连接已编译的目标文件及库函数，还可以在 cc 中同汇编连接。此外源程序经修改，重新编译后的连接效率也是很高的。

上面简要地叙述了C语言的特点。从这种语言在编写程序中应注意的事项来看，还存在以下几方面的问题。

1. C 语言的表示方法非常紧凑，但其弊病是易出现费解的地方。例如

虽然这种表示是正确的，但很难理解。编写程序时应注意这类问题。

2. C 语言的最大优点之一是使用指针。但指针容易出错，常常会指向意想不到的地方。使用时应注意不要给错指针的值。

3. C 语言的另一个问题是出错检验的能力较弱。因此应避免数据类型不一致、函数之间参数个数和类型不一致的错误，以及数组下标的超出、数的溢出和下溢等问题的发生。

4. 语法上某些地方可能会引起误解。例如

$$a \gg 2 + 3$$

的运算结果并非 $(a \gg 2) + 3$，而是 $a \gg 5$，即把 a 向右移 5 位。又如

$$4 > 5 > 6$$

会被看成运算结果为“真”。因为 $5 > 6$ 条件为“假”，结果为 0，由此得出：$4 > 0$ 为“真”，即总的结果为“真”。这种二义性虽然可由 UNIX 中的 lint 程序检验，但最好用括号区别，以便辨认。

5. C 语言中没有数据的字长规定（位数、精度）。如整型数（int）的字长在个人微型机上一般只有 16 位，而在 32 位的小型机上可以用到 32 位。此外字符类型的变量（char）可以按整型数处理，其中由于机种不同，数值的范围也各不相同。有的是 0 ～255，也有的是 -128～127。

综上所述，C 语言具有类似于汇编语言的程序设计能力，灵活性较强，但程序容易出错，在编写时应十分谨慎。

## § 1.2 程 序 例

图 1-1 是字符输入和输出的程序。它的作用是，如果在键盘上一次输入一个字符，那么在输出设备（如CRT）每次就显示

一个字符。为说明方便，程序的左端标注了行号1～12。而实际的源程序没有行标号。

```
1;      /* copy input to output */
2;      main ( )
3;      {
4;              int c ;
5;
6;              c=getchar ( ) ;
7;              while (c ! =-1)
8;              {
9;                  putchar (c) ;
10;                 c=getchar ( );
11;             }
12;     }
```

〔执行结果〕

```
a a b b c c d d e e f f g g
```

图 1-1 字符输入和输出程序

1：注解。

为便于理解和阅读，程序中间可以加文字注解。这部分与程序无关，可随意添写，放在空白或换行符出现的地方，但一定要在/*和*/符之间。对注解行，编译程序不进行编译。

2：函数的定义。

表示定义一个函数名为 main 的函数。main ( ) 除代表一个函数外，还有另一特殊意义，即表示程序从此函数开始执行。在C语言中函数的名字和数目是任意的，但这个函数（相当于主程序）不能省略。

3：和12：花括号｛和｝。

表示函数 main ( ) 的范围。左花括号表示函数的开始，右花括号表示函数的结束。

4：说明语句。

C语言语法规定，任何变量在使用前都必须加以说明。这一

行说明变量 c 属于 int (integer的缩写) 类型的变量。 int 是一个关键字[1],它代表数据类型，表示一个带有正负号的整型数(16位或32位)，并指出在下面可执行语句中出现的变量 c 都是整型变量。

在 FORTRAN 中没有说明的变量都被看成是 INTEGER 的，而这里不允许没有说明的变量。

5： 空白行。

这一行的目的是突出说明语句与下面可执行语句的界限。一般来说，C 语言的描述格式是很自由的，可以没有空白行，甚至可以几行并在一起。例如

```
main( )
{  int c  ;  c=getchar ( ) ;
```

的写法也是允许的。

6： 赋值语句。

c=getchar ( ) ; 表示将 getchar ( ) 内容赋值给变量 c。而 getchar ( ) 及 9：行的 putchar (c) 都是标准库函数，这是对它们的调用。 getchar( ) 的功能是每调用一次即取下一个输入字符，并作为此函数值返回。经过赋值，变量 c 就得到了输入的一个字符。 putchar(c) 的作用是把 c 的内容在终端（如显示器或打印机）上输出。

7：～11： 循环语句 while 。

循环语句的一般表示形式为

```
while (表达式)
   语句
```

其中括号里的表达式是 while 的循环条件，下面的语句是它的循环体。若满足表达式的条件，则执行该循环体，且一直循环到表达式的条件不满足为止。

---

1. 所谓关键字，即在语法中其本身用途固定，程序员不得为其它目的使用的字。

本例的循环条件是 c != -1 。其中 != 是比较运算符，表示不相等。这里用 -1 代表字符输入的结束。若输入的字符不是 -1 ，即不是结束标志时，执行循环体。否则循环即停止，继续执行循环体外的后续语句。这里的循环体是花括号括起的二行语句 putchar( ) 和 c=getchar( ) (称为复合语句)。执行这个循环体时，首先调用 9: 行的 putchar(c)函数，将取来的字符输出，然后再调用 10: 行的 getchar( ) 函数，取来下一个字符。

12: main( ) 函数的结束。

也是整个程序的终止。

下面分析这个程序的执行过程。如图 1-1 所示，在键盘上输入一个字符a (字符下面标有横线的表示是由键盘输入)时，6: 行函数 getchar( ) 即将此字符作为函数的值返回，并赋值给 c 。接着执行下面循环语句。此处首先验算 7: 行中的表达式。因 a 不等于 -1，满足循环条件，故执行 9: 行 putchar(c) ，在终端显示字符 a 。然后执行 10: 行，由 getchar( ) 重新取一字符 b，再回到 7: 行验算表达式条件。如此循环，直至取完所有输入字符。当取来的字符为 -1 时，跳出循环转向下一行语句。因为本例 12: 行是右花括号，表示函数的结束，所以至此整个程序执行完了。

最后终端上显示

```
a a b b c c d d e e f f g g
```

这里应指出，上述循环语句中出现了数字 -1，这对理解程序不利，应尽力避免。为此可利用 §1.1 节讲过的预处理功能。其用法如图 1-2 所示。

```
#define EOF -1
main ( )
{
    int c ;
```

```
    c = getchar ( ) ;
    while (c  != EOF)
    {
        putchar(c);
        c = getchar( ) ;
    }
}
```

图1-2 预处理功能#define

在程序前面， main( ) 的上一行加入一行

#define EOF -1

即把一个符号名 EOF[1] 定义为字符串 -1。此后在函数中可用 EOF 代替 -1，而编译程序则在出现 EOF 的地方自动用 -1替换。本例将 7: 行的循环条件改写为

while (c  != EOF)

为区别小写的变量名等，此处符号名用大写字母。

预处理功能的好处在于，使程序不仅易读、便于理解，而且也便于修改。譬如 EOF (Eend of file) 是一个文件结束符，在程序中它比 -1 更能明确地表示输入的结束。此外如果程序中几处都使用相同的数字，需要修改时仅变更 #define 一个地方就可以了。

将 EOF 定义为 -1后，若从键盘输入文件结束符 EOF 时，则在 UNIX 操作系统可同时按 CTRL 和 d 两个键，其代表的值为 -1。此时即表示文件输入结束。

## § 1.3 函数及其成分

前面已多次提到函数。函数在 C 语言中占有极重要的地位。C 语言程序实际上是由许多小的独立的函数组成，各种复杂的运算过程都可以隐含在函数之中。这使得整个程序非常清晰，便于

1. 符号名又称宏名，见 §7.3节。

阅读和修改。因此编写C语言程序，实际是实现和完成每一个小的函数的过程。为进一步了解函数的这些概念，我们通过图1-3程序说明函数的有关问题。

这是一个两个数的求和并打印输出的程序。

```
/* …… add·c…… */
main( )
{
    int x, y, z;
    x=33051;   print(x);
    y=62941;   print(y);
    z=x+y;     print(z);
}
/* …… print …… */
print(a)
    int a;
{
  printf("%d\n",a);
}
```

〔执行结果〕

```
33051
62941
95992
```

图 1-3 求和程序

如图 1-3 所示，在函数 main( ) （或称主函数）中将加数、被加数及和分别赋值给 x、y 及 z，而由函数

print(a)

依次将它们输出。

1. 函数的一般形式

函数的一般表示形式为

函数名（参数表）

参数表说明