计算机科学大众丛书

DZ

C语言程序设计及应用实例

杨 潮 编著

電子工業出版社

C语言程序设计及应用实例

杨 潮 编著

电子工业出版社

内容简介

本书注重实用性和简洁性，不过分追求语法细节，而力求通过大量实例来说明C语言的特点。书中的例子思路清晰，可读性强，许多例子可以直接引用。本书对于C语言中一些较难的问题，采用了较多的图形解释，以期达到易于理解的目的。

本书内容深入浅出，通俗易懂，非常适合于自学，中专以上文化的读者都可以通过学习本书而掌握C语言。对于具有一定C语言编程基础的读者来说，本书也是一本很有价值的参考书。

本书带有学习软盘，同书一起发行。

C语言程序设计及应用实例

杨　潮　编著

责任编辑：宋玉升

*

电子工业出版社出版

北京市海淀区万寿路173信箱(100036)

电子工业出版社发行　各地新华书店经销

中国科学院印刷厂印刷

*

开本：787×1092毫米 1/16　印张：16　350千字

1995年10月第一版　1995年10月北京第一次印刷

印数：6000册　　定价：20.00元

ISBN7-5053-3134-5/TP.1116

前　言

C 语言是当今世界上最流行的程序设计语言之一，它的应用覆盖了从操作系统、语言处理、数据库、CAD、图像处理、网络、通讯直到多媒体技术等各个领域。八十年代起，C 语言开始在国内流行，成为最热门的语言。C 语言得到广泛使用的直接的原因正是因为程序员喜欢它。虽然现在我们大多已转向用 C++ 编程，但对 C 仍有着深刻的感受。

作者一直在用 C(C++)从事软件开发，并讲授 C 语言课程，教授对象既有已具备其他高级语言甚至汇编语言的学生，也有计算机编程的初学者，本书就是在上述实践基础上编写而成。

本书也像“计算机科学大众丛书”的其他书一样，适合于有中专以上文化的读者自学。把 C 语言程序设计写得浅显易懂，对作者来说并非易事，许多精采的例子由于过大或涉及范围过多而只能舍弃。本书力求通俗易懂，深入浅出，选用的例子多在教学中积累并精选的，许多概念尽量用图示方式说明，并在各章的末尾给出易犯的错误及如何防止的提示。

本书在编者过程中得到许多同仁及学生的帮助，特别是专家们的大力支持。北京印刷学院刘兆毓教授审读了全书并提出许多宝贵意见，北方交通大学丁嘉仲教授对本书的结构和体例提出了很好的建议，在这里对他们表示衷心的感谢。由于作者学识有限，书中错误在所难免，敬请读者改正。

作者

1995,5　北京

目　录

第一章 绪 论

本章学习重点

本章介绍C语言的发展历史以及C语言的特点，然后通过简单例子展示C语言的风貌，最后告诉你如何在计算机上运行C程序。

第一节 C语言的产生及特点

C语言是一种得到广泛重视并普遍应用的计算机程序设计语言，也是国际上公认的最重要的几种通用程序设计语言之一。它适合作系统描述语言，既可用来写系统软件，也可用来写应用软件。

C语言是美国贝尔实验室的Dennis. M. Ritchie于1972年设计实现的。C语言直接来源于B语言，但它的根源可以追溯到ALGOL60。ALGOL60结构严谨，其设计者非常注重语法、分程序结构，因此对于后来许多重要的程序设计语言，如PASCAL，PL/I，SIMULA67产生过重要的影响。但它是面向过程的语言，与计算机硬件相距甚远，不适合编写系统软件。1963年英国剑桥大学在ALGOL60的基础上推出更接近硬件的CPL语言，但CPL太复杂，难于实现。1967年，剑桥大学的Matin Rinchards对CPL语言作了简化，推出了BCPL语言。1970年贝尔实验室的Ken Thompson以BCPL为基础，设计了更简单也更接近硬件的B语言(取BCPL的第一个字母)。B语言是一种解释性语言，功能上也不够强，为了很好地适应系统程序设计的要求，Ritchie把B发展成称之为C的语言(取BCPL的第二个字母)。C语言既保持了BCPL和B的优点(如精练，接近硬件)，又克服了它们的缺点(如过于简单，数据无类型等)。1973年K. Thompson和D. M. Ritchie用C改写了UNIX代码，并在PDP－11计算机上加以实现，即UNIX版本5，这一版本奠定了UNIX系统的基础，使UNIX逐渐成为最重要的操作系统之一。

C语言是D. M. Ritchie1972年设计实现的

图1-1展示了C语言的由来。

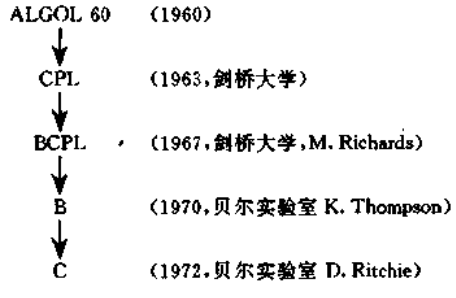

图1-1 C语言的由来

C 设计语言的目的是为描述和实现 UNIX 操作系统提供一种工具语言。但 C 并没有被缚束在任何特定的硬件或操作系统上，它具有良好的可移植性。1977 年出现了不依赖于具体机器的 C 语言编译文本，使向各种机器移植 C 变得更加简单，这也推动了 UNIX 系统的广泛实现。随着 UNIX 的日益普及，又反过头来带动了 C 语言的迅速推广，使它先后被移植到各种大、中、小、微型计算机上。

1978 年，贝尔实验室的 Brian. W. Kernighan 和 Dennis. M. Ritchie（合称 K&R）合著了《The C Programming Language》一书，并在附录中提供了 C 语言参考手册，这本书成为以后广泛使用的 C 语言的基础，被人们称作非官方的 C 语言标准。1983 年美国国家标准化协会（ANSI）开始制定新的标准，这就是 ANSI C 标准。1990 年，C 语言成为国际标准化组织（ISO）通过的标准语言。

C 语言是作为描述系统的语言而设计，但随着其日益广泛的应用，特别是 80 年代以后各种微机 C 语言的普及，它已经成为众多程序员最喜爱的语言，它的使用覆盖了几乎计算机的所有领域，包括操作系统、编译程序、数据库管理程序、CAD、过程控制、图形图象处理等等。

C 语言如此成功是有其自身特点的：

1. C 语言是比较“低级的”语言。

有人把 C 语言称为“高级语言中的低级语言”，也有人称它是“中级语言”。这样说不是说它功能差或难于使用，而是指它具有许多通常只有象汇编语言才具备的功能，如位操作、直接访问物理地址等等，这使 C 语言在进行系统程序设计时显得非常有效，而过去系统软件通常只能用汇编语言编写。事实上，C 语言的许多应用场合是汇编语言的传统领地，现在用 C 来代替汇编，使程序员得以减轻负担、提高效率，而写出的程序具有更好的可移植性。

C 语言具有更多的接近硬件操作的功能，而不提供直接处理复合对象，如作为整体看待的字符串、数组等的操作，这些较高级的功能必须通过显式调用函数来完成。这看起来是个缺陷，但这使语言的规模较小，更容易说明，学习起来也快。比如说，C 语言只有 32 个关键字，而一些微机上的 BASIC，关键字多达 100 个以上。

2. C 语言是结构化的语言。

C 语言在结构上类似于 ALGOL60、PASCAL 等结构化语言。这大概与它源于 ALGOL60 以及 60 年代末 70 年代初结构化程序设计方法的兴起有关。C 语言的主要结构成分是函数——C 的子程序。函数允许一个程序中的各任务分别定义和编码，使程序模块化，在函数的外部只需了解函数的功能，而将实现的细节隐藏起来，设计得好的函数能够正确地工作而对程序的其他部分不产生副作用。C 语言还提供了多种结构化的控制语句，如用于循环的 for、while、do-while 语句，用于判定的 if-else、switch 语句等，以满足结构化程序设计的要求。

3. C 语言是程序员的语言。

看到这个提法，你可能会奇怪：难道还有不是程序员的程序设计语言吗？确实有许多程序设计语言不是为专业程序员设计的，比如说：FORTRAN 是为工程师，COBOL 是为商业人员，PASCAL 是为学生，而 BASIC 是为非程序员设计的。C 是为专业程序员设计的语言。Ritchie 是专业程序员，而 C 最初是为了他自己写 UNIX 操作系统而设计的。C 语言实现了程序员的期望：很少限制，很少强求，程序设计自由度大，方便的控制结构，独立的函数，紧凑的关键字集合和较高的执行效率。用 C 编写程序可获得高效的机器代码，其效率通常只比

汇编语言生成的机器代码低10～20%，而同时C又具有PASCAL那样的结构，这就难怪有大量的程序员喜欢它。

C语言的语法限制不太严，例如，对数组下标越界不做检查，整型、字符型数据可以通用，不专设逻辑型数据而以整型来代替等。较少的限制给程序员带来较大自由，这就要求程序员在编程时应确实明白自己在做什么，而不要把检查错误的工作仅寄托于编译程序。当然这会带来一些麻烦，但作为程序员应该考虑好再开始编码。其实，有时候让计算机惩罚一下粗枝大叶的程序员也不一定是什么坏事。

C语言得到广泛使用的主要原因可能就是因为程序员喜欢它。但C也不是专业人员的专利，非专业人员经过实践也会熟练地掌握它。现在有许多不同专业的非计算机专业人员正在使用或正在转向C语言。C语言是我们大家共同的财富。

第二节　用C语言编写程序

学习一种程序设计语言唯一的有效途径就是用它编写程序。下面几个简单的程序将带你进入C的世界，开始我们的学习。

第一个程序总是输出一个字符串，这里，选择著名的"hello, world"程序，以表示对K&R的敬意。

例1.1

```
main()
{
    printf("hello,world\n");
}
```

这个程序的运行结果是输出字符串

```
hello,world
```

上面的程序称C语言源程序，简称C程序。为了让它运行，必须先找一个编辑程序输入它，然后编译，装入，最后执行。

现在让我们来看看程序本身。main是一个函数名，表示"主函数"。一个C程序，不管简单或复杂，总是由一个或多个函数组成，由这些函数实现要做的操作。函数名可以按照你的喜好去取，但main是一个特殊的名字。C程序总是从main函数开始执行，这意味着，一个程序必须在某个地方有一个main函数。

花括号{}括起构造函数的语句，称函数体。函数体中只有一条函数调用语句

```
printf("hello,world\n");
```

其功能是将双引号""中的内容输出。printf是C语言中的库函数，它来自计算机厂家提供的输入输出库，你可以直接去调用它。双引号中的内容是调用printf时提供的参数，表示为一个字符串。字符串中的\n是C语言中的换行字符，在输出时，它将终端的光标移动到下一行的开始。如果不使用\n，输出时就会发现，光标停在输出字符串的后面。printf不提供自动换行的功能，每当在程序中需要执行换行操作时，都必须写出\n。例如，我们把例1.1改写一下，

```
main()
{
    printf("hello,") ;
    printf("world") ;
    printf("\n") ;
}
```

执行结果也是一样的。如果要将程序改成

```
main()
{
    printf("hello,\n") ;
    printf("world\n") ;
}
```

你会发现输出变为：

```
hello,
world
```

在这里我们可以看到，C 语言倾向于把怎样处理的权利交给程序员。

printf 语句的最后跟着一个分号“;”，它表明一个语句的结束。

例 1.2

```
/* 求两个整数之和 */
main()
{
    int a, b, sum ;
    a = 12 ;
    b = 34 ;
    sum = a + b ;
    printf("sum is %d\n", sum) ;
}
```

本程序的功能是求两整数之和。/* … */部分是注释，是为了便于阅读程序而安排的，对程序的编译和执行没有影响。注释可以加在程序的任何位置。注释可以是任意一串字符，但不能再含有/* … */，也就是说，注释是不能嵌套的(也没有必要)。这里的注释是用汉字给出的，你的计算机可能没装汉字，可用英文或汉语拼音来写注释。本书中采用汉字注释完全是为了阅读的方便。

程序中的

```
int a, b, sum ;
```

是变量定义部分，这里我们用 int 定义变量 a、b 和 sum 为整型。C 语言规定使用任何变量之前都要先定义。

```
a = 12 ;
```

```
    b = 34 ;
    sum = a + b ;
```

是三个赋值语句，“=”是赋值运算符，它的作用是把其右部表达式的值赋给左部的变量，这里是把值 12 赋给变量 a，把值 34 赋给变量 b，把 a+b 的结果赋给变量 sum。

```
    printf("sum is %d \n",sum);
```

中的"sum is"和"\n"我们已经熟悉，%d 是一个输出转换说明，意思是：输出时用一个整型值来代替它，这里，这个整型值是变量 sum 的值。于是本程序的输出结果为：

```
    sum is 46
```

例 1.3

```
main()/*求两个数中较大者*/
{
    int a, b, c ;
    printf("a,b=") ;
    scanf("%d,%d", &a, &b) ;
    c = max(a, b) ;
printf("max=%d\n", c) ;
}
int max(a, b)   /*返回 x,y 中较大者*/
int x, y ;
{
    int  z;
    if ( x > y )
            z = x;
    else
            z = y ;
    return(z) ;
}
```

本程序定义了两个函数：main 和被调函数 max。在 main 中，语句

```
    printf("a,b=") ;
```

输出

```
    a,b=
```

然后光标停在=之后。下一句

```
    scanf("%d,%d",&a, &b) ;
```

用于接收两个整型数据到变量 a 和 b 中。scanf 是 c 输入输出库中提供的输入函数，它要求按照转换字符(这里是%d，表示整型)将相应类型的数据输入到指定变量的存储单元中去，&a 表示取变量 a 的地址。

语句

```
    c = max(a, b) ;
```

是将函数 max 的值赋给变量 c，其中 a 和 b 是参数(称为实在参数)，执行时将变量 a 和 b 的值传给被调函数 max。max 的作用是将 x、y 中较大的数送 z，并通过语句

```
return(z) ;
```

将其作为函数值返回给主调函数。函数头

```
int max(x, y)
int x, y ;
```

表示 max 带有两个形式参数 x 和 y，在 max 被调用时，这两个参数的值由主调函数中对应的实在参数(本程序中是 a 和 b 的值)传过来，函数体中求出 x、y 中较大的数值送 z，并返回主调函数，语句

```
if( x > y)
    z = x;
else
    z = y;
```

是 C 中的一种判定控制语句结构，它根据 x 是否大于 y 来选择执行

```
z = x
```

或

```
z = y
```

本程序的执行情况如下：

```
a,b=4,6↓
max=6
```

例 1.4

```
#include "stdio.h"
main()
{
    int c ;
    while(( c = getchar() ) ! = EOF )
            putchar(c);
}
```

这个程序的功能是将输入的内容复制到输出，第一行

```
#include "stdio.h"
```

是 C 语言中的文件包含，stdio.h 是一个头文件，通过文件包含命令 #include，程序把 stdio.h 包含在自己所在的文件中。stdio.h 是关于标准输入输出的头文件，其中包括使用标准输入输出库函数的许多信息。在本程序中，由于使用 getchar 和 putchar 函数(实际上是宏，将在第九章中介绍)，因此需要用命令包含，将存有必要信息的 stdio.h 包含进来。符号 EOF 也在 stdio.h 中定义，实际中，它定义为

```
#define EOF -1
```

意思是在程序中遇到 EOF 就用 -1 代替。EOF 是文件结束标志，对于任意一个文件，系统会自动在其尾部加一个 EOF 标志。当文件是从键盘上输入时，在微机 DOS 操作系统下，打入

^z(ctrl+z)后，系统自动为输入流加上一个 EOF(UNIX 系统下，可打入^d)。函数 getchar 用于读一个字符，putchar 用于输出一个字符，语句 while 表示当括号内的表达式值为真时，反复地执行后面跟着的语句，因此

```
while((c=getcharc())! =EOF)
    putchar(c);
```

的意思是：当读入的字符不是 EOF 时，就输出它，直到读入 EOF，所以我们可以这样运行这个程序

```
abc1234  xyz789^z(输入)
abc1234  xyz789  (输出)
```

在本章的例子中，我们让每个语句单独占据一行的位置，并且书写时作了适当的缩进，这些都不是 C 语言语法所要求的，而是为了阅读方便，如例 1.4 也可以写成

```
#include "stdio.h"
main(){int c;while((c=getcharc())! =EOF)putchar(c);}
```

其结果也是一样的。C 语言不是靠回车，而是靠分号来标识一个语句的结束，回车的功能与一个空格是相同的。虽然可以把程序写成上面那个样子，而且可以正确地执行，但实际中几乎没有人那么写，即使是很短的程序。计算机界有句名言：程序是给人读的。看到这个命题，你也许会奇怪，程序不是为了在计算机上运行的吗？确实如此，但不仅仅如此，你写一个程序以后可能要修改，别人也可能会阅读或修改你的程序，在没有完全读懂的情况下做修改，将会带来许多麻烦，即使不改，一个表达清楚的程序也会给人带来愉快。再者，连你自己都不能清楚地表达思路的话，还能指望计算机作什么呢？

为了清晰而正确地写程序，首先要使程序具有良好的结构，如使用将实现细节隐蔽得很好的函数，少用外部变量，不采用非结构化的控制结构等。同时，也要注意书写风格，如每个语句独立占一行，适当的缩进，尽量选择有实际意义的标识符，适当的加入注释等等，这些在我们今后的讨论中还会反复提到。总之，从一开始写程序就要培养好的风格。

第三节　C 程序上机运行

学习编程，上机是非常重要的。在计算机上编程并运行，从结果中了解程序的运行过程，进而了解程序的结构，比只看书要有效得多。另外，你可以对已有的程序做各种修改，进而了解语言更多的知识，也可以向别人学到很多东西。上机可以提高你编程的兴趣，也可以提高自信心，当看到自己编写的程序在计算机上正确地运行时，你会感到自己的创造力。

本节中，我们介绍 turbo　C 上机的简单步骤，如果你使用其他的 C，或希望对 turbo　C 有更多的了解，请参考相应的用户手册。

用 turbo C 上机的步骤：

先将 turbo C 软件安装到机器的硬盘上。

1. 调入 turbo　C

只要打入 tc，然后回车，屏幕顶部将显示出 turbo C 的命令菜单(见图 1-2)

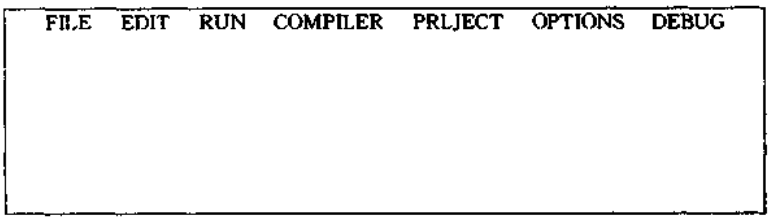

图 1-2 命令菜单

用键盘上的"←"和"→"键可以移动屏幕上的光标，如果光标不在顶部的命令菜单行中，可以打 F10 键将光标移到顶部。将光标移到"FILE"处，打回车键，FILE 下面出现一个子菜单，如图 1-3，用"↓"键将光标移到 Load(或 New)处打回车键，表示要输入源程序，屏幕上又出现一个小窗口，如图 1.4 所示，要求你输入文件名，此时可打入

file1.c ↵

如果原来没有这个名字的文件，将建立一个新文件，如果已有此文件，则将此文件调入并在屏幕上显示，然后自动转为编辑(EDIT)状态。

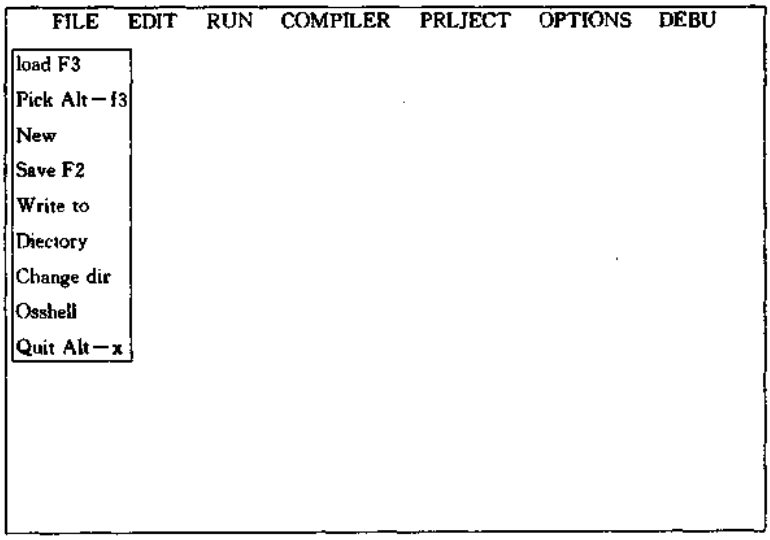

图 1-3 子菜单

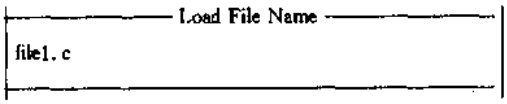

图 1-4 文件名输入框

你也可以直接进入编辑状态，这时在 DOS 命令行中打入

tc file1.c

系统就自动进入编辑屏幕。也可不打 c，这时系统取缺省值.c。

2. 编辑源文件。

你可以根据需要输入或修改源程序。

3. 编译源程序

只需要简单地打 F9 键，就可以进行源程序的编译和连接，并在屏幕上显示有无错误。如有错误，按任一键后，屏幕显示源程序，光标停在出错的位置上。屏幕的下半部分将显示出

有错误的行和错误原因。根据提供的信息可以修改错误，这时按下“F6”键后就可修改错误，修改完后再按“F6”键，接着打“↓”键，光标就停到下一个出错的行。所有错误修改完后，再按“F9”进行编译，如此反复，直到没有编译错误为止。

4. 执行程序

你只要打入 ^ F9(按下 ctrl 键的同时按 F9 键)即可执行程序。如果程序中需要输入数据，就在此时将所需数据输入，接着程序执行，输出结果。

如果发现结果不对，可再按步骤 2、3、4 修改程序、编译、执行程序，直到得到正确的结果。

5. 保存文件

编辑完源程序或准备退出 turbo C 时，也可打“F2”键将源程序存在硬盘上。

6. 退出 turbo C

同时按下 Alt 和 X 键，便退出 turbo C，回到操作系统命令状态。此时可以用系统命令来显示源程序或运行程序。如

type file1.c ↓(显示源程序清单)

file1 ↓(执行程序 file1.exe)

如果想再输入或修改源程序，可以再从步骤 1 开始。

以上的步骤是对一个程序只放在一个源文件中时采用的，如果一个程序放在多个文件中，就要使用 PROJECT 项生成所谓 prj 文件。考虑到初学时，程序通常都较小，不必放在多个文件中分别编译，这里就不多介绍了。

最后讲一下本书附带磁盘的使用方法。本书中全部例子的源程序附在一张磁盘上，文件名是字母 E 开头，后跟章、例号，如例 1.1 为

E1－1.c

你可以直接调入 turbo C 编译并执行这些例子，有些例子有多个程序，这时文件名中再含有一个标识版本的数字放在最后，如例 1.1 有两个版本的话，程序名分别是

E1－1－1.c

和

E1－1－2.c.

书中有些例子只给出了一个被调函数，而没有 main，为了便于上机操作，盘中都给了用于测试这些例子的主调函数，上机时也可以直接编译运行。