C语言应用程序设计

徐君毅 吴 京 高海锋 于 玉 朱 鹰 编

LANGUAGE FOR APPLICATION PROGRAMMING

复旦大学出版社

C语言应用程序设计

徐君毅　吴　京
高海锋　于　玉　朱　鹰　编

复旦大学出版社

(沪)新登字 202 号

内 容 提 要

本书主要介绍如何使用C语言进行应用程序设计，书中大部分篇幅主要介绍菜单和窗口设计、图形的设计、串行口通信程序设计以及C语言与数据库的接口程序设计。

本书以微机上的C语言为背景，全书的程序均可在Turbo C环境中运行，读者掌握了C语言的基础知识及DOS的有关知识即可阅读本书。

本书可作为广大计算机应用工作者的参考书，也可作为高等学校开设应用软件课的教材和参考书籍。

责任编辑 陆盛强 孙未来

责任校对 马金宝

C 语言应用程序设计

徐君毅 吴 京
高海峰 于 玉 朱 鹰 编

复旦大学出版社出版

（上海国权路 579 号）

新华书店上海发行所发行 复旦大学印刷厂印刷

开本 787×1092 1/16 印张 19.5 字数 484,000

1993 年 12 月第 1 版 1993 年 12 月第 1 次印刷

印数 1— 10,000

ISBN7-309-01148-1/T・87

定价：15.00元

前　言

C语言是一种通用的程序设计语言。它不仅能方便地编写系统程序，也能很好地编写应用程序。近年来在国外已获得广泛的应用，在我国也已十分流行，尤其是微机上C语言的应用已十分普及。为帮助广大计算机应用工作者能尽快地提高编写各种应用程序的能力，我们编写了这本书，书中有关内容是作者多年来用C语言开发有关应用程序的一些体会。

本书是针对微机上的C语言编译器来编写的，考虑到目前在微机上使用最广泛的C语言编译版本，因此，书中程序均能在Turbo C（2.0）语言下运行，对于Microsoft C只要稍加修改也能运行。

本书共分五章：

第一章简略介绍了C语言的基础知识，这部分的内容是带有复习性的，对于已经掌握C语言基础的读者可以不必阅读。

第二章介绍C语言在菜单和窗口技术中的应用，这部分的内容对于编写一个应用程序所需要具有的良好的用户界面是不可缺少的。

第三章介绍C语言在图形学中的应用，介绍了用C语言绘各种实用图形的有关技术。

第四章介绍了用C语言进行串行口通信的程序设计。

第五章介绍了C语言与数据库的接口，讨论了如何使用C语言对广泛使用的dBASE和FoxBASE的数据库文件作进一步处理的技术。

由于C语言应用的方面很多，在书中不可能面面俱到，希望能在阅读本书的基础上做到举一反三的作用。本书由五人合作编写，第一章由徐君毅执笔，第二章由高海锋执笔，第三章由吴京执笔，第四章由朱鹰执笔，第五章由于玉执笔，全书由徐君毅定稿。

本书是一本计算机应用书，书中对C语言基础、库函数以及DOS调用知识未作详细介绍，读者可查阅有关参考资料。

由于编者水平有限，书中错误在所难免，欢迎广大读者批评指正，以备再版时修正。

编　者

目　录

第一章　C 语言概述

§1.1　简单程序举例

C 语言是一门简洁、高效的中级语言，它兼有高级语言所有的可移植、结构化、易于使用、具有复杂的数据结构，以及低级语言所有的快速、紧密、高效、能进行地址和位运算等特点。为尽快了解 C 语言的各种成分，我们先介绍 C 语言的一个简单的例子。

例 1.1　C 语言的一个简单的例子。

```
/*   A simple example */
main () {
    printf (" A simple example\n");
}
```

C 语言程序完全由一系列函数组成，这些函数可以放在一个文件里，也可以放在多个文件里。在这一系列的函数中必须有一个且只有一个以 main 为名的函数，这个函数称为**主函数**，整个程序从它开始执行，并且，一般来说，在它里面结束。

在上例这个 C 语言程序中，只有一个主函数，函数名 main 后面的一对圆括号是函数的标志，是不可以省略的。主函数中只有一条语句，printf 是一个格式输出函数，它在屏幕上显示括号内的字符串。在 C 语言中，字符串是用双引号括起来的。反斜线“\”可与后面跟随的一个字符合并在一起代表一个字符，如\n 表示换行、\t 表示水平制表符。

C 语言允许注解书写在程序的各个部分，注解由斜线后随星号开始，并以星号后随斜线结束。

/* 注解的内容 */

程序中出现的一对花括号，称为程序体括号。用它们括起来的部分是函数体，程序的可执行部分都在函数体内。

C 语言中，分号是语句的终结符，因而每一个可执行的 C 语句后面都必须带有分号。

C 语言是自由格式的语言，只要符合语句的形式，多条语句可以在一行中连续书写，一条语句也可以写几行。

§1.2 变 量 说 明

变量和常量取值含义由其类型来决定。类型决定了能施加的运算及内存中存放空间的大小。C 语言规定，任一变量在使用前必须说明。对于每个变量都必须确定其数据类型。说明就是用来规定一组变量的数据类型的。

简单类型说明的形式如下：

类型区分符 标识符 1，标识符 2，……，标识符 n；

其中，类型区分符用以指明类型，后面的标识符即为要说明的变量名。类型区分符有 int、float、double 和 char，分别表示整型、浮点型、双精度型和字符型。例如：

```
int i, j, k;
float length, height;
char c;
```

分别说明变量 i、j 和 k 是整型的，变量 length 和 height 是浮点型的，变量 c 是字符型的。

在函数中，说明必须出现在任何可执行语句之前。变量说明起两个作用：①分配适当的存贮空间；②正确执行对应操作。变量标识符在同一层内不允许作二次说明。

在说明的同时，可以对变量进行初始化。下面的例子中，整型变量 a 和 b 分别被赋予初始值 3 和 4，浮点变量 x 和 y 分别被赋予初始值 12. 4 和 41. 2，而变量 c 和 z 由于未进行初始化，因此它们的初始值不定。

```
main ()
{
    int a=3, b=4, c;
    float x=12.4, y=41.2, z;
        .
        .
        .
}
```

§1.3 基本数据类型和常量

C 语言提供了四种基本数据类型，它们是整型 (int)、浮点型 (float)、双精度型 (double) 和字符型 (char)。

一、整型常量

C 语言中整常量分十、八、十六进制。十进制整常量 0～9 组成的并且不以数字 0 开头的数字序列，前面的" －" 号表示该值是负值。八进制整常量以数字 0 打头，该值的数字

必须是0～7之中的一个。十六进制整常量以一个数字0和字母x（或X）引导，后面紧跟的是十六进制数字串，即由0～9数字和a～f字母（大小写均可）组成。如果整常量后面加上后缀L（或l），这个整常量就是long型的；如果后缀是U（或u），就是unsigned型。unsigned long型常量是在整常数后面同时加上字母U（或u）和L（或l）构成的。

例 1.2

```
main ()
{
    printf (" \n %d %x %o", 19, 19, 19);
    printf (" \n %d %x %o", 017, 017, 017);
    printf (" \n %d %x %o", 0x1c, 0x1c, 0x1c);
    printf (" \n%x", 7151);
    printf (" \n%d", 0x1BEF);
    printf (" \n\n");
    }
```

程序产生的结果如下：

```
19 13 23
15 f 17
28 1c 34
1bef
7151
```

二、浮点型（float）和双精度型（double）常量

float型的常量和double型的常量没有特别的区别。我们根据习惯将他们统称为**浮点常量**。它们的形式为

±数字串. 数字串 E±数字串

有些成分是可以省略的，但是小数点和E两者不能全省，小数点的左右必须有一边有数字，而E的左右两边都必须有数字。如345.、3. 14、0. 0、－. 123、2. 1e4、. 516和15. 5都是合法的浮点常量的例子。

三、字符型（char）

字符常量是由括在单引号中的一个字符构成。屏幕上无法显示的字符可以通过相应的转换字符或转换字符序列表示，如'\n'可用来表示换行，'\t'可用来表示水平制表。有关转换字符可见表1. 1。从表中可以看出，我们可以用'\''表示单引号。

表 1.1 转换字符

转换代码	表示意义	解　　释
\0	NUL	字符串结束符
\b	退格	把光标向左移动一个字符
\n	换行	把光标移到下一行的开始
\r	CR	把光标返回到当前行的开始
\f	换页	换到下一页
\t	TAB	把光标移到下一个制表位
\\	\	引用反斜杠字符
\"	"	在字符串中出现的双引号
\'	'	单引号字符

还可用位方式\ooo 和\hhh 来表示字符，ooo，hhh 分别是三位八进制数和十六进制数，它对应于字符的编码。如'\007'为 ASCII 码中的 BELL。

例 1.3

```
main ()
{
    printf ("%d %c %c\n", 'A', 'A', 65);
    printf ("%d %d\n", '0', '\0');
    printf ("%c %c %c\n", '0', '0'+1, '0'+9);
}
```

打印的结果是

```
65 A A
48 0
0 1 9
```

四、字符串常量

字符串常量是用一对双引号括起来的零个或多个字符序列，"" 称为**空串**。

字符串可看成数组，其长度不限，以'\0' 结尾。如字符串中出现双引号，必须表示成\"；转换序列和\ooo、\hhh 也可作为单个字符出现在字符串中，如

```
printf (" \007Error! \n");
```

字符串不可跨行书写，如要跨行要用“\”，如，语句

```
printf (" This is a char\ (回车换行符)
acter string.");
```

打印出

```
This is a character string.
```

注意，字符“\”后面必须紧跟回车换行符。

C 语言将几个相互之间除了空格、制表符和回车符外没有其他字符的字符串看成一个

字符串，例如

" abc" " def"

" hij"

被看成

" abcdefhij"

§1.4 类 型 转 换

一、隐式类型转换

当混合数据类型在一个表达式（特别是算术表达式）中时，就会自动地进行类型转换，如下次序，这一次序就决定了哪些运算分量必须进行类型转换

char＜int＜long＜float＜double

右边的类型" 高于" 左边的类型，一般地，当不同类型的变量或常量在一个运算符两边参加运算时，在运算之前将" 较低" 类型提升为" 较高" 类型，结果是较高类型。

二、显式类型转换——强制

强制类型转换的形式为

（类型）表达式

其中（类型）可以是C语言中任何可能的类型，表达式是C语言中任何合法的表达式。强制类型转换仅改变表达式所产生的值的类型，而不改变这一表达式的实际值。

三、类型转换的数量关系

在较高类型的值向较低类型转换时，值和符号都得以维持。在较低类型的值向较高类型转换时，如果较高类型的值在较低类型的范围内，值和符号都得以维持，如果较高类型的值超出较低类型所表示的范围，会将较高类型的值截断，值的情况随机器而定。

§1.5 运 算 符

一、运算符和优先级

C语言的表达式由运算符和运算分量构成，运算符的运算次序由优先级和结合性决定，C语言将其运算符分成15个优先级，分别以数1～15来代表，数字越大，优先级越高。见表1．2。

二、基本算术运算符

基本算术运算符有7个，分单目运算符和双目运算符两类。单目运算符有两个，它们分别是取负运算符“－” 、取正运算符“＋”。双目运算符有5个，它们是“＋”、“－”、“＊”、“/”和“%”，它们分别表示加、减、乘、除及取模运算。运算符“%”只能对非浮点数进行运算，它表示两个整数相除后的余数，例如5%3为2。“%”和“/”的关系如下：

a%b=a－（a/b）＊b　a、b整型数且b≠0

表 1.2　运算符的优先级和结合性

优先级	运　算　符	结合性
15	()　[]　.　→	→
14	!　～　++　--　－　+　&　＊（类型名）sizeof	←
13	＊　/　%	→
12	+　－	→
11	<<　>>	→
10	<　<=　>　>=	→
9	==　!=	→
8	&	→
7	^	→
6	|	→
5	&&	→
4	||	→
3	?:	←
2	=　＊=　/=　%=　+=　-=　>>=　<<=　&=　^=　|=	←
1	,	→

单目“－”、“＋”的优先级是14，右结合；双目运算符都是左结合，它们的优先级如下：

“＋”　“－”　12

“＊”　“/”　“%”　13

三、关系运算符和相等性比较运算符

关系运算符有四个：“>”、“<”、“<=”、“>=”，分别表示“大于”、“小于”、“小于等于”和“大于等于”，他们的优先级为10。相等性比较运算符有等于“==”和不等于“!=”两个，优先级为9，这两组运算符都是双目运算符，都是左结合的。表达式的值是整型，如表达式成立，值为1，否则为0。

四、移位运算符

“>>”和“<<”两个双目运算符分别表示右移和左移。它们的优先级为11，运算分量都是整型。表达式E1<<E2表示E1左移E2位，左移是比较简单的，移入的位均为0。E1>>E2，表示E1右移E2位，它把E1的各位向右移E2位。如果E1是无符号的，进行的是算术移位；如果E1是带符号的整数，可能是算术移位，也可能是逻辑移位。

五、按位运算符

按位运算符有下面几个：“&”、“|”、“^”、“～”，它们分别表示按位“与”、按位“或”、按位“异或”和按位求反。它们的优先级分别为8、6、7、14。

六、逻辑运算符

逻辑运算符为“&&”、“||”和“!”，分别为逻辑合取、逻辑析取和逻辑非。优先级分别为5、4、14。运算结果是int型的。对于“&&”，计算时只有当两个运算对象都不为0时表达式的值为1，其他情况表达式的值都为0；对于“||”，计算时只有当两个运算对象都为0时表达式的值为0，其他情况表达式的值都为1。“&&”和“||”是严格从左至右进行运算的，当运算到一定程度，表达式的值可以被确定时，就不再运算下去。

七、增量运算符

增量运算符有两个，它们分别是“++”和“--”，它们都是右结合的单目运算符，优先级为14。其运算分量是左值。假定E是一个可以在赋值号左边出现的C语言表达式，那么表达式

++E 使得E增加一个单位，再用它的值作为表达式的值；

E++ 用E的值作为表达式的值，然后使得E增加一个单位；

--E 使得E减少一个单位，再用它的值作为表达式的值；

E-- 用E的值作为表达式的值，然后使得E减少一个单位。

八、条件运算符

条件运算符“?:”，是三目运算符，优先级为3，右结合。由它组成的表达式的形式为

表达式1? 表达式2 : 表达式3

计算条件表达式时，首先计算表达式1，如果表达式1的值非0，计算表达式2，条件表达式的值为表达式2的值；否则，计算表达式3，条件表达式的值为表达式3的值。条件表达式值的类型总是表达式2和表达式3的类型中较高的类型。必要时要进行类型转换。

九、赋值运算符

赋值运算符是

=，+=，-=，*=，/=，%=，>>=，<<=，&=，|=，^=

它们的优先级都为2，右结合。表达式" V op= E" 的意义是 V=V op (E)。

十、逗号运算符

逗号“,”是一个左结合的双目运算符，它的优先级为1，逗号运算符严格自左朝右运算，其值和类型分别为第二个计算分量的值和类型。

十一、和地址有关的运算符

和地址有关的运算符有两个，它们互为逆运算，运算符“*”（单目）是取指针所指单元的内容，表达式的类型是指针的基准类型，运算符“&”（单目）的作用是取存贮单元的地址，其结果的类型是指向该存贮单元类型的指针。这两个运算符虽然和乘法运算符及按位与运算符相同，但目数不同。它们的优先级为14，右结合。

十二、sizeof 运算符

sizeof 运算符用来取存贮单元或类型的字节数。它的优先级为 14，右结合。

§1.6 语　　句

一、表达式语句和复合语句

在一个 C 语言的表达式后面加上一个分号“;”就构成一个表达式语句。空语句是特殊的表达式语句。

用花括号“{”和“}”将一串语句括起来就成为复合语句，复合语句在语法上看成一个语句。

二、if 语句

C 语言所提供的一个选择语句是 if 语句，该语句的最简单的形式为

if（表达式）语句

执行该语句时，先计算括号内“表达式”部分的值，如果表达式的值为真（即不为 0），此时条件成立，就执行“语句”部分；若表达式的值为假（即为 0），条件不成立，就跳过“语句”部分执行后面的语句。

例 1. 4　将输入的字符全部转换成大写输出

```
#include <stdio. h>
main ()
{
    int c;
    while ( (c=getchar ())! =EOF) {
      if (c>='a' && c<='z')
        c+='A'-'a';
      putchar (c);
    }
}
```

if 语句的另一种形式为

```
if（表达式）      语句 1
else              语句 2
```

执行此语句时，先计算括号内“表达式”部分的值，如果表达式的值为真（即不为 0），此时条件成立，就执行“语句 1”部分；若表达式的值为假（即为 0），条件不成立，就执行

“语句 2”部分，在“语句 1”部分或“语句 2”部分执行完毕以后，执行后面的语句。C 语言规定，else 和它前面出现的离它最近的 if 配对。

三、循环语句

C 语言的循环语句有三类，分别是 for 语句、while 语句和 do—while 语句。

for 语句的格式如下

for（表达式 1；表达式 2；表达式 3）语句

其执行过程见图 1.1。

for 语句中的三个表达式都可省略，其做法是：只要略去某个所要求的表达式，而在该位置上用一个分号标记。若省略表达式 2，循环不能正常退出，要提供一些其他可以从该循环中退出的方法。

while 语句的格式是：

while（表达式）语句

执行过程见图 1.2。

do 循环语句。句法如下：

```
do
  语句
while (表达式);
```

执行过程见图 1.3。

例 1.5　计算 x^n

```
main ()
{
    float x, p=1. 0;
    unsigned int n;
    scan ("%f", &x);
    scanf ("%d", &n);
    while (n--) p *=x;
    printf ("%f\n", p);
}
```

程序的第一第二条可执行语句是用来读入 x 和 n 的，最后一条语句是用来打印结果的。中间的就是 while 语句。其中条件（n--）相当于（n--！=0）每执行一次 n 减 1，直到 n 为 0 为止。这时循环进行了正好 n 次，p 中存放的正好是 x 的 n 次连乘积。

例 1. 6　下面是一将输入拷贝到输出的程序

```
#include <stdio. h>
```

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

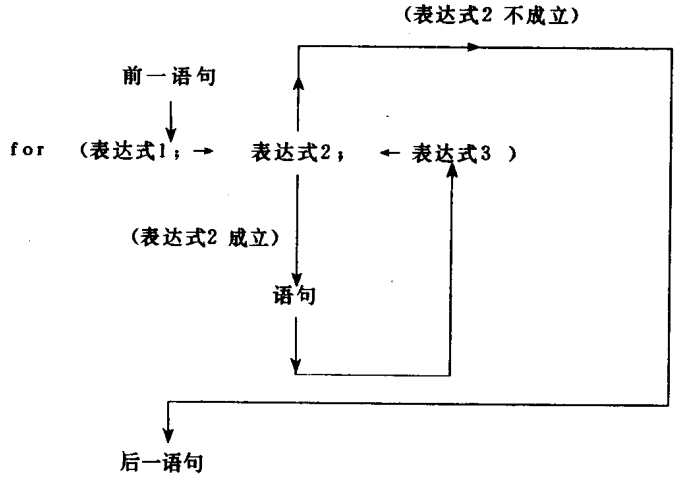

图 1. 1 for 循环语句执行过程

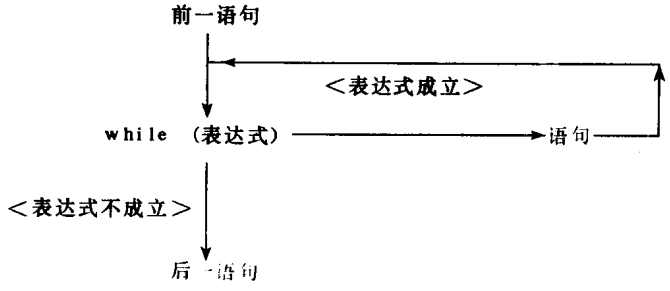

图 1. 2 while 循环语句执行过程

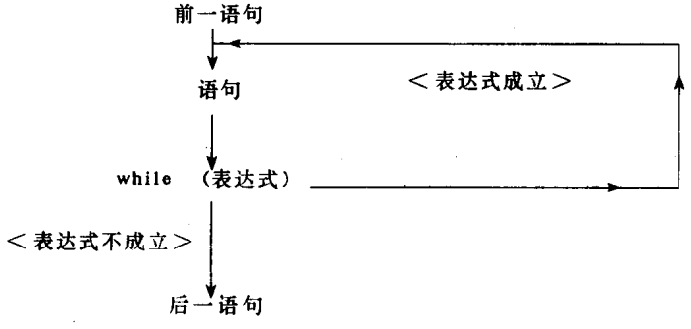

图 1. 3 do－while 循环语句执行过程

四、switch 语句

switch 语句的一般格式如下：

```
switch（表达式）{
  case  整常量表达式 1：  语句序列 1
  case  整常量表达式 2：  语句序列 2
  …
  case 整常量表达式 n：  语句序列 n
  default：              语句序列 n+1
}
```

执行 switch 语句时，先计算表达式，再与每个 case 后面的常量表达式比较，如果表达式的值等于某个常量表达式的值，就执行该常量表达式后面出现的第一条语句，然后一直执行下去，一直到遇到转出此条 switch 语句的语句，或 switch 语句中此后的所有语句都依次执行完以后（即执行到 switch 语句结尾处的"}"）。如果没有相匹配的常量表达式，就去执行以 default 为前缀的语句，直到遇到转出此条 switch 语句的语句或 switch 语句中此后的所有语句都依次执行完以后。如果既没有相匹配的常量表达式，也没有 default 前缀，这条 switch 语句什么事都不干，它相当于一个空语句。执行完 switch 语句后，就执行 switch 语句后面的语句。

五、break 语句

break 语句导致包含它的最小的循环语句或 switch 语句的终止，控制传递到被间断的语句的下一语句。

例 1．7　编写一个程序，打印 3 到 100 之间的所有质数，每行打 10 个。

```
main ()
{
    int i, j, k;
    k=0;
    for (i=3; i <=100; i++) {
      for (j=2; j <=i-1; j++)
          if (i%j==0) break;
      if (i==j)
          printf ("%2d%c", i,
                ++k%10==0 ? '\n' :'');
    }
    printf (" \n");
}
```

程序输出：

```
3 5 7 11 13 17 19 23 29 31
```