



YUYAN YINGYONG CHENGXU SHEJI YU GONGJU KAIFA

# 应用程序设计与工具开发

主 编 刘亚秋 赵化启  
副主编 孙维连 李 满  
主 审 王霓虹

东北林业大学出版社

# C 语言应用程序设计与工具开发

主 编 刘亚秋 赵化启  
副主编 孙维连 李 满  
主 审 王霓虹

东北林业大学出版社

---

图书在版编目 (CIP) 数据

C 语言应用程序设计与工具开发/刘亚秋, 赵化启主编. —哈尔滨: 东北林业大学出版社, 2007.6

ISBN 978-7-81131-031-3

I. C… II. ①刘… ②赵… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2007) 第 097307 号

---

责任编辑: 冯 琪

封面设计: 彭 宇



NEFUP

**C 语言应用程序设计与工具开发**

Cyuyan Yingyong Chengxu Sheji Yu Gongju Kaifa

主编 刘亚秋 赵化启

东北林业大学出版社出版发行

(哈尔滨市和兴路 26 号)

哈尔滨市工大节能印刷厂印装

开本 787 × 960 1/16 印张 17.75 字数 318 千字

2007 年 6 月第 1 版 2007 年 6 月第 1 次印刷

印数 1—1 000 册

ISBN 978-7-81131-031-3

TP·80 定价: 28.60 元

# 前言

C 语言对于那些要求很高的效率、良好的实时性,或者与操作系统内核紧密关联的程序来说,C 语言是很好的选择。鉴于 C 语言良好的可移植性以及现有的很多程序,如语法分析器、GUI Builder 等,都可以产生非常好的 C 代码,C 语言对于程序员来说具有无可替代的价值。就我们熟知的大多数语言而论,只要你发掘得足够深,到最后就会看到它们的内核都是用纯正的、可移植的 C 语言写成的。C 语言最出色的地方还在于其高效和贴近机器,C 语言有益于帮助你在硬件体系的层次上思考问题。随着计算机技术的飞速发展,C 语言在单片机、嵌入式、DSP、IC 验证、V.O.S 技术等领域的应用非常活跃。

C 语言是一种通用的程序设计语言,它不仅能方便地编制各种系统程序,也能很好地编写应用及工具程序。为帮助计算机爱好者尽快提高编写系统程序、贴近硬件体系的应用程序与工具软件开发的能力,我们特意编写了这本《C 语言应用程序设计与工具开发》。

书中内容是作者多年从事计算机教学与科研过程中,开发有关应用程序与工具软件的一些体会及实例。本书针对微机 C 语言编译系统而编写,书中的程序均能在 TURBOC 2.0 下编译运行。

有些用户在使用计算机的过程中经常会遇到一些技术问题,需要自己去解决。如,研究文件系统的结构、硬盘的组织结构、数据恢复等问题时,需要自己去编制一些工具软件来解决相关的技术问题。这些问题往往要运行在实模式下,确保读写大硬盘的顺利进行。为使工具软件人机交互性好,就应在工具软件中使用汉字且在非汉字系统下,这样就更方便用户使用。本书正是基于这种思想,使早期的 C 语言,能在我们的手下大放异彩,解决了许多技术问题,如 TURBOC 2.0 对大硬盘的读写技术等等。

我们在进行计算机安全技术研究和教学过程中,深刻体会到工具软件的开发与使用绝不能脱离计算机去空谈,而必须结合使用的计算机进行具有可操作性和实战性的应用。这样学生才能很好地理解与消化,便于他们将来能从事更广泛的研究。各种实战环境的建立,也需要一些工具软件来完成。为此,我们把本书相关的汉字技术、加/解密、文件管理、接口应用、硬盘维护工具技术等结合起来,完成实战环境的建立与实战演练,收到良好的效

## 2 C语言应用程序设计与工具开发

果。由于C语言应用广泛,希望读者在阅读本书的基础上做到举一反三。本书由刘亚秋、赵化启任主编,孙维连、李满任副主编。其中,东北林业大学刘亚秋编写了第1章、第4章及第5章,佳木斯大学赵化启编写了第2章及第6章3~4节,黑龙江科技学院李满编写了第3章,佳木斯大学孙维连编写了第6章的1~2节并做了全书的统稿工作,石金飞、贺雷参加了插图、植字与排版工作,全书由东北林业大学王霓虹教授主审。

由于编者水平有限,书中错误与不足在所难免,欢迎广大读者批评指正,以备再版时修改。

编 者

2007年1月

## 目 录

|                            |       |
|----------------------------|-------|
| 1 C语言的基础知识及基本语句 .....      | ( 1 ) |
| 1.1 C语言的起源 .....           | ( 1 ) |
| 1.2 C语言的特点 .....           | ( 1 ) |
| 1.3 C语言的一般介绍 .....         | ( 2 ) |
| 1.4 C语言的语句 .....           | (16)  |
| 1.5 预处理命令 .....            | (21)  |
| 1.6 C语言的标准库函数 .....        | (29)  |
| 1.7 C程序的运行 .....           | (37)  |
| 2 汉字嵌入技术 .....             | (39)  |
| 2.1 图形系统的初始化 .....         | (39)  |
| 2.2 基本图形函数 .....           | (40)  |
| 2.3 颜色选择函数 .....           | (45)  |
| 2.4 封闭图形填色函数及画图函数等函数 ..... | (46)  |
| 2.5 创造独立的图形程序 .....        | (51)  |
| 2.6 综合例子 .....             | (52)  |
| 2.7 汉字的编码与显示 .....         | (53)  |
| 2.8 关于汉字菜单 .....           | (57)  |
| 2.9 建立小型专用汉字库 .....        | (60)  |
| 2.10 综合应用实例 .....          | (67)  |
| 2.11 直接嵌入程序 .....          | (73)  |
| 2.12 汉字信息转换 .....          | (77)  |
| 2.13 汉字字模内嵌在应用程序中的实例 ..... | (79)  |
| 3 软盘格式化与加密 .....           | (91)  |
| 3.1 软盘标准格式化 .....          | (91)  |
| 3.2 加密 .....               | (96)  |
| 3.3 一些加密磁道的制作思路 .....      | (101) |
| 3.4 常驻程序 (TSR) .....       | (119) |
| 4 文件管理技术 .....             | (143) |
| 4.1 文件 .....               | (143) |

## 2 C语言应用程序设计与工具开发

|                                         |              |
|-----------------------------------------|--------------|
| 4.2 实例练习 .....                          | (162)        |
| <b>5 接口应用 .....</b>                     | <b>(188)</b> |
| 5.1 CMOS 端口应用 .....                     | (188)        |
| 5.2 使用物理扇区读写技术编写硬盘锁 .....               | (192)        |
| 5.3 串口通信 .....                          | (196)        |
| <b>6 硬盘维护与工具开发技术 .....</b>              | <b>(210)</b> |
| 6.1 DOS 在硬盘上的存放及硬盘的体系结构 .....           | (210)        |
| 6.2 应用基本 INT 13H 读写 8GB 以下硬盘的开发技术 ..... | (231)        |
| 6.3 应用扩展 INT 13H 读写大硬盘的开发技术 .....       | (236)        |
| 6.4 应用汉字技术与扩展 INT 13H 读写大硬盘的开发技术 .....  | (253)        |
| <b>参考文献 .....</b>                       | <b>(278)</b> |

# 1 C 语言的基础知识及基本语句

## 1.1 C 语言的起源

C 语言是近年来在国内外得到迅速推广使用的一种现代化语言。C 语言功能丰富，表达能力强，使用灵活方便、应用面广，既具有高级语言的优点，又具有低级语言的许多特点，因此是成功的系统描述语言，又是通用的程序设计语言。

C 语言是一种编译型程序语言，也是 UNIX 系统的主力语言，它与 UNIX 系统有着相互依存、休戚与共的关系。C 语言的前身是 Martin Richards 在 20 世纪 60 年代开发的 BCPL 语言。BCPL 语言是软件人员在开发系统软件时使用的。1970 年，Ken Thompson 继承和发展了 BCPL 语言的优点，进而提出了“B 语言”，当时最新型的小型机中的 UNIX 就是用 B 语言开发的。此后在对 UNIX 操作系统进行更新和开发工作中，Dennis·M·Ritchie 和 Brian·W·Kernighan 对 B 语言作了进一步的充实和完善，于 1972 推出了新型的程序语言——C 语言，并首次在 UNIX 操作系统的 DEC PDP-11 计算机上使用。早期的 C 语言是附属于 UNIX 系统的，且在 PDP-11 上实现的。但目前 C 语言却独立于 UNIX 系统，它适用于各种计算机。从 8 位的微型机到巨型机都有 C 语言的踪迹，与 FORTRAN、Pascal、BASIC 等语言一样成了微、小、超小、大、超大和巨型机上共同使用的语言。随着微型计算机的日益普及，出现了许多 C 语言的版本。由于没有统一的标准，使得这些 C 语言之间出现一些不统一的地方。为了改变这种情况，美国国家标准研究所（ANSI）为 C 语言制定了一套 ANSI 标准，成为现行的 C 语言标准。

## 1.2 C 语言的特点

C 语言的特点是多方面的，人们从不同的角度总结出众多的特点。归纳起来有以下几点：

(1) 运算符较多。有 34 种运算符，包括括号、赋值、强制类型转换等，从而可以实现在其他高级语言上难以实现的运算。

(2) 数据结构丰富。有各种数据类型, 可以处理各种复杂的数据结构, 具有较强的数据处理能力。

(3) 具有结构化的语句。结构化语句的显著特点是代码及数据的分隔化, 即程序的各个部分彼此独立。这种结构化方式可使程序层次清晰, 便于使用、维护以及调试。

(4) 程序设计模块化。C 语言的函数结构, 有利于把整体程序分割成相对独立的功能模块, 为模块间相互调用和数据传输提供了便利; 也为多人同时进行集体开发的软件工程方法, 提供了有力的支持。

(5) 允许直接访问物理地址, 能实现汇编的大部分功能。

(6) 可移植性好。在一个环境中, 用 C 语言编写的程序不加改动或稍加改动就可移植到另一个环境中去。

(7) C 语言本身简洁, 编译程序小。C 语言除了在表示法上尽可能简洁外, 语言的许多成分都是通过显示函数调用来完成的。

(8) C 语言功能齐全。C 语言具有各种各样的数据类型, 并引入了指针的概念, 可使程序的效率更高。

### 1.3 C 语言的一般介绍

#### 1.3.1 C 语言程序的组成

像其他语言一样, C 语言也有自己固定的格式和一些语句。一个简单的 C 语言程序如例 1.1。

##### 例 1.1

```
/* example program of c */
#include "dos.h" /* 包含文件说明 */
main()          /* 主函数定义 */
{
    int i;
    /* 定义局部变量 */
    for(i = 0; i < 16; i++)
    {
        int j;
        for(j = 0; j < 8; j++)
        {
```

```

char c = 14;
int k = 0, l = 0;
printf("%cs%d, %d)", c, l, j);
k = (5 * i) + 1;
l = j + 1;
g(k, l);
printf("%d%d", l, j);
}
}
oo();
}
oo()
/* 用户定义的子函数 */
{
int l, j;
char c = 14;
printf("\n 请输入颜色值(前景色 背景色):");
scanf("%d%d", &l, &j);
printf("%cs%d, %d)", c, l, j);
}
g(x, y) /* 用户定义的子函数 */
int x, y;
{
union REGS r;
r.h.ah = 2
r.h.dl = x
r.h.dh = y
r.h.bh = 0
int86(0x, &r, &r);
}

```

由此可看出, C 源程序主要有以下几个特点:

- (1) 程序一般用小写字母书写;
- (2) 每个语句结尾都要用终止符“;”;
- (3) 每个程序必须有一个主函数 main ();
- (4) 每个程序体(主函数和子函数)必须用一对花括号“{”和“}”括

起来;

(5) 注释部分包含在 “/\*” 和 “\*/” 之间;

(6) 完整的程序大致包括: 包含文件、变量定义、主函数、子函数及程序体。

### 1.3.2 变量、数据类型与数组

像其他语言一样, C 语言的变量在使用之前必须先定义其数据类型, 未经定义的变量是不能使用的。定义变量的类型应在可执行语句前面。

C 语言有七种内部数据类型, 如下所示:

|        |              |
|--------|--------------|
| 字符型    | char         |
| 短整型    | short int    |
| 整形     | int          |
| 无符号整形  | unsigned int |
| 长整形    | long int     |
| 浮点型    | float        |
| 双精度浮点型 | double       |

变量名是由一个或几个字符组成的字符串, 下划线也可以作变量名的一部分。例如: back\_color。又如, TEST 和 test 代表的是两个不同的变量。

所有变量在使用之前必须说明, 说明的一般形式是:

类型 变量名

例如, 要说明 x 是一个浮点型, y 是整型, 应按如下格式书写:

```
float x;
```

```
int y;
```

```
char z, 说明 z 是一个字符型变量。
```

另外, 对于结构和联合类型可以用 struct 和 union 的组合来建立, 也可以用 typedef 生成变量类型的新名称。

数组的下标是以 0 开始的, 如 y[0] 就是数组 y 的第一个元素。

要说明一个有 100 个元素的整形数组 y, 可以写成

```
int y[100]
```

这样就建立了一个有 100 个元素的数组, 第一个元素的标号是 0, 而最后一个标号是 99。如语句:

```
for (x=0; x<100, x++) y[x] = x
```

其作用是把数字 0 到 99 送到数组 y 中。

多维数组的说明是将附加维的大小放在另一对括号中。例如, 可用如下

形式说明一个 10 乘 20 的整形数组

```
int x [10] [20]
```

上边建立了一个行数为 10、列数为 20 的 2 维数组，但是要注意它的第一个元素是 `x [0] [0]`，最后一个元素是 `x [9] [19]`。

另外，C 语言中的字符常数可直接用单引号括起来表示，如 ‘A’，‘9’ 等。有些常用的字符有如表 1.1 的特殊规定。

表 1.1 常用字符

| 规定符    | ASCII 码          | 含 义    |
|--------|------------------|--------|
| ‘\f’   | ‘\x0c’           | 换页     |
| ‘\r’   | ‘\x0d’           | 回车     |
| ‘\t’   | ‘\x09’           | 制表键    |
| ‘\n’   | ‘\x0a’           | 换行     |
| ‘\ ’   | ‘\x5c’           | 反斜杠符 \ |
| ‘\ ’   | ‘\x27’           | 单引号符 ‘ |
| ‘\ ’   | ‘\x22’           | 双引号符 “ |
| “\ddd” | 一到三位 8 进制数代表的字符  |        |
| “\xhh” | 一到两位 16 进制数代表的字符 |        |

表中最后两行是用 ASCII 码表示一个字符，如 \101 代表 ASCII 码为十进制数 65 的字符 ‘A’。请注意，\0 是代表 ASCII 码为 0 的控制字符，即“空操作”字符，它将作用在字符串中。

转义字符的使用如例 1.2。

### 例 1.2

```
main()
{
    printf(" ab c \t de \r f \t g \n");
    printf("h \t i \b \b j k");
}
```

程序中没有设字符变量，用 printf 函数直接输出双引号内的各个字符。请注意其中的“转义字符”。第一个 printf 函数先在第一行左端开始输出“ab c”，然后遇到“\t”，它的作用是“跳格”，即跳到下一个“制表位置”，在我们所用系统中一个“制表区”占 8 列。“下一制表位置”从第 9 列开始，

故在第9~11列上输出“de”。下面遇到“\r”，它代表“回车”（不换行），返回到本行最左端（第1列），输出字符“f”，然后遇“\t”再使当前输出位置移到第9列，输出“g”。下面是“\n”，作用是“使当前位置移到下一行的开头”。第二个printf函数先在第1列输出字符“h”，后面的“\t”使当前位置跳到第9列，输出字母“i”，然后当前位置应移到下一列（第10列）准备输出下一个字符。下面遇到两个“\b”，“\b”的作用是“退一格”，因此“\b\b”的作用是使当前位置回退到第8列，接着输出字符“jk”。

程序运行时在打印机上得到以下结果：

```
fab c    gde
```

```
h        jik
```

注意在显示屏上最后看到的结果与上述打印结果不同，是：

```
f        gde
```

```
h        j k
```

这是由于“\r”使当前位置回到本行开头，自此输出的字符（包括空格和跳格所经过的位置）将取代原来屏幕上该位置上显示的字符。所以原有的“ab c”被新的字符“f g”代替，其后的“de”未被新字符取代。换行后先输出“h i”，退两格后再输出“j k”，j后面的“k”将原有的字符“i”取而代之。因此屏幕上看不到“i”。实际上，屏幕上完全按程序要求输出了全部的字符，只是因为输出前面的字符后很快又输出后面的字符，在人们还未看清楚之前，新的已取代了旧的，所以误以为未输出应输出的字符。而在打印机输出时，不像显示屏那样会“抹掉”原字符，留下了不可磨灭的痕迹，它能真正反映输出的过程和结果。

### 1.3.3 运算符

C语言的运算符非常丰富，主要分为以下几类：算术运算符、关系与逻辑运算符、位运算符、指针运算符和杂项运算符。

#### (1) 算术运算符

C语言有七种算术运算符：

- 减法

+ 加法

\* 乘法

/ 除法

% 取模 即取余数！ $11\%3=2$  而  $11/3=3$

-- 自减 1 如果 P 等于 2, 则  $P-- = 1$

++ 自加 1 如果 P 等于 2, 则  $P++ = 3$

这些运算符的优先级是:

最高      ++    --

         \*   /   %

最低      +    -

同级运算符从左到右运算。

## (2) 关系与逻辑运算符

关系运算符和逻辑运算符是用来产生 TRUE/FALSE (真/假) 结果, 两者经常在一起使用。在 C 语言中任何非零数产生 TRUE 值, 它的关系和逻辑运算符用数字 1 代表 TRUE, 用数字 0 代表 FALSE。

关系运算符如下:

>          大于

> =        大于或等于

<          小于

< =        小于或等于

= =        等于

! =        不等于

逻辑运算符如下:

&&        AND (与)

$(2 > 1) \&\& (3 > 2) == \text{TRUE}$

||        OR (或)

$(1 > 2) || (2 > 1) == \text{TRUE}$

!        NOT (非)

$! 1 == \text{FALSE}$

运算符的优先级是:

最高级    !

         > = , < =

         =

         ! =

         &&

最低级    ||

下面的结果为 TRUE:

$(150 < 200) \&\& 10$

## (3) 位运算符

C语言还提供了位运算符，它们能对变量内部实际上的位进行操作。对于整形和字符型变量，可使用如下运算符：

&      AND (与)  
 |      OR (或)  
 ^      XOR (异或)  
 ~      反码  
 >>    右移  
 <<    左移

## “&amp;”与操作

如果两个相应的2进制位都为1，则该位结果为1，否则为0。

即  $0 \& 0 = 0$ ;  $0 \& 1 = 0$ ;  $1 \& 0 = 0$ ;  $1 \& 1 = 1$ ;

## “|”或操作

两个相应的2进制位中只要有一个为1，该位的结果即为1。

即只有  $0 | 0 = 0$ ，其他时候值都为1。

## “^”异或

异或的意思是判断两个相应的位值是否为“异”，为“异”（值不同）就取1，否则为0。

假设有 01111010 想使其低4位翻转，即1变0，0变1。可以将它与 00001111 进行 ^ 运算。

```
01111010
^ 00001111
= 01110101
```

在程序中，“&”、“|”和“^”可以像其他运算符一样使用，如：

## 例 1.3

```
main ()
{
    char x, y, z;
    x = 1; y = 2; z = 4;
    x = x & y; /* 结果 x = 0 */
    y = x | z; /* 结果 y = 5 */
}
```

反码运算符“~”是对字节中的每个位求反。例如，字符变量 ch 的位模式如下：

0011 1001

那么执行语句

ch = ~ ch;

将把位模

1100 0110

存入 ch 中。

右移和左移运算符可根据指定的位数将一个字节或字节中的所有位向右或向左移动。当位移时空出的位添入零。移位运算符右边的数字说明了位移的数目。移位运算符的一般形式是

变量 > 移位数

变量 < 移位数

例如, 有如下位模式

0011 1101

右移一位变成

0001 1010

右移相当于除以 2, 左移相当于乘以 2。如对 X 中的值首先乘以 2, 然后再乘以 2。可以用如下程序段来完成:

```
int x
```

```
x = 10
```

```
x = x << 1;
```

```
/* X = 100 */
```

```
x = x >> 1;
```

```
/* X = 10 */
```

位运算的优先级如下:

最高级 ~

>

<

&

^

最低级 !

#### (4) 指针运算符

指针运算符在 C 语言中是很重要的。它们允许把串、数组和结构传递给函数, 也允许函数修改它们的调用参数。两个指针运算符是 "&" 和 "\*"。

“&”运算符加在变量前面时，将给出该变量的地址。如整形变量 X 在内存的地址是 2200，那么

```
y = &x; /* 把 X 的地址送给 y */
```

是把值 2200 送给 y。

“&”可读做“取……的地址”。

“\*”运算符取其后面变量的值，并将该值作为内存中的信息的地址。

例如：

```
y = &x
y * = 100;
```

其结果是把值 100 送入地址 y 中。“\*”可读作“在地址”。“\*”运算符也可以用在赋值语句的右边。例如：

```
y = &x;
y * = 100;
z = * y/10;
```

其结果是将值 10 存入 z 中。

指针如例 1.4 所示。

#### 例 1.4

```
main()
```

```
{
    char * p, a[ ] = "hello world"; /* 定义一个指针 P 和一个字符数组 */
    p = a; /* 把字符数组 a 的首地址赋给指针 p */
    while( * p) /* 当 p 指向的字符不为 \0 时 */
        printf("%c", * p++ ); /* 打印出数组 a */
}
```

#### (5) 赋值运算符

在 C 语言中，赋值运算符是一个等号。例如：

```
x = x + 10;
```

```
y = y/z;
```

也可以写成如下的简写形式：

```
x + = 10;
```

```
y / = z;
```

#### (6) ? 运算符

“?”运算符是个三元运算符，它可用来代替一般类型的 if 语句。

if 表达式 1 then x = 表达式 2