



C 程序设计及应用软件开发技术

齐从谦 著

```
/* 主程序 */
extern PROG prog[200];
void Draw(void)
{
    int i;
    float draw-x, draw-y, draw-x-next, draw-y-next;
    draw-x-next=x1; draw-y-next=y1;
    while(1)
    {
```

```
        pal=cos((x2-x1)/Co-to-Co);
        pal=sin((y2-y1)/Co-to-Co);
        if(fabs(r1)<fabs(r2) && r1+r2>0)
        { pal=cos=-pal-cos; pal=sin=-pal-sin; }
        draw-x-next=x1+fabs(r1)*pal-cos;
        draw-y-next=y1+fabs(r1)*pal-sin;
        if(!i)
        {
            if(x0+M=draw-x, y0-M=draw-y, x0+M=draw-x-next, y0-M=draw-y-next);
            Map-L=prog(i, draw-x, draw-y, draw-x-next, draw-y-next); j++;
            xx=draw-x+draw-x-next; yy=draw-y+draw-y-next; }
        else {
            x-a=draw-x-x1; y-a=draw-y-y1;
            start-a=Map-angle(x-a, y-a);
            x-a=draw-x-next-x1; y-a=draw-y-next-y1;
            end-a=Map-angle(x-a, y-a);
            if(r1<0)
            { temp=start-a; start-a=end-a; end-a=temp; }
            arc(x0+M=x1, y0-M=y1, start-a, end-a, M=fabs(r1));
            Map-C=prog(i, j, draw-x, draw-y, draw-x-next, draw-y-next); j++;
            draw-x=draw-x-next; draw-y=draw-y-next;
        }
```

中国科学技术出版社

C 程序设计及应用软件开发技术

本书可作为高等院校计算机专业及相关专业教材，也可供从事计算机工作的工程技术人员参考。
本书可作为高等院校计算机专业及相关专业教材，也可供从事计算机工作的工程技术人员参考。
齐从谦 著

中国科学技术出版社

·北京·

内 容 提 要

C 语言是一种模块化、结构化的计算机程序设计语言。近几年来,C 语言以它的功能丰富、表达能力强、使用灵活、应用面广、目标代码率高、可移植性强,且能对硬件直接进行操作等多方面的优点而深受广大科技工作者的欢迎,愈来愈多的人已经和正在选择使用 C 语言,市面上出现了大量的关于 C 语言的教材和著作。据调查很多科技期刊上凡是采用计算机语言来表达的内容,几乎 90%以上是用 C 语言编写的。

本书根据作者近十年来在国内外从事教学和研究工作中应用 C 语言的体会和经验编写而成。全书分基础篇 and 开发篇两部分,既向读者介绍了 C 语言程序设计的基础知识,又从提高的角度论及 C 高级编程的理论和技巧,并通过大量来自工程实践的范例介绍如何用 C 语言开发应用软件。

本书内容殷实,叙述流畅,编排合理,易于阅读。书中的例题均在 TURBO C2.0 环境下编译通过,其源代码和可执行代码都存放在本书所提供的软盘中,随时可供读者使用。软盘中还收录了“用户界面”,“图形编程”,“汉化菜单”,“数控编程”等多个实用的、商品化的软件,可供有关专业的读者参考使用。

C 程序设计及应用软件开发技术

齐从谦 著

责任编辑:张秀智

黄 冰

特约编辑 席庆义

*

中国科学技术出版社出版(北京海淀区白石桥路 32 号)

新华书店北京发行所发行 各地新华书店经售

安徽省总工会义兴印刷厂印刷

*

开本:787×1092 毫米 1/16 印张:19.75 字数:455 千字

1995 年 8 月第 1 版 1995 年 8 月第 1 次印刷

印数:1—5000 册 定价:20.00 元

ISBN 7—5046—1356—8/TP·76

目 录

基 础 篇

第一章 C 语言概述	(1)
1 C 语言发展的历史	(1)
2 一个简单的 C 程序	(2)
3 C 语言的特点	(5)
4 C 程序的结构和风格	(7)
5 源程序的编辑、编译、连接和运行	(12)
第二章 C 语句及其要素——数据、运算符、表达式	(13)
1 语句	(13)
2 数据类型描述	(13)
3 常量	(16)
4 变量及其定义和赋初值	(19)
5 数据类型的转换	(23)
6 算术运算符	(24)
7 赋值运算符	(25)
8 关系运算符与逻辑运算符	(27)
9 移位运算符	(31)
10 条件运算符和逗号运算符	(32)
11 优先级和结合率	(33)
12 数据的输入和输出	(34)
第三章 C 程序的流程控制	(39)
1 流程控制语句	(39)
2 if...else 结构	(39)
3 switch 结构	(43)
4 循环控制结构	(45)
5 其它流程控制语句	(55)
第四章 函数及变量的作用域	(59)
1 函数的定义	(59)
2 函数的说明	(61)

3	函数的调用	(62)
4	隐含参数	(66)
5	变量的存储类型及其作用域	(68)
6	动态变量	(68)
7	静态变量和外部变量	(70)
8	函数的重载及 inline 函数	(73)
9	宏与文件包含	(74)
第五章	数 组	(79)
1	数组的基本概念	(79)
2	字符数组和字符串	(80)
3	数组与函数的参数	(84)
4	二维数组和多维数组	(85)
5	数组应用综合举例	(87)
第六章	指 针	(92)
1	指针的概念	(92)
2	指针变量的定义和使用	(93)
3	指针与函数	(95)
4	指针与数组	(102)
5	指向指针的指针	(108)
6	main()函数中的参数	(111)
第七章	结构、联合及枚举	(113)
1	结构的基本概念及定义	(113)
2	结构变量的初始化及使用	(116)
3	结构数组	(120)
4	指向结构的指针	(125)
5	结构与函数	(127)
6	关于动态存储分配的若干基本概念	(132)
7	联合	(139)
8	枚举	(144)
9	用 TYPEDEF 定义类型	(145)
第八章	文 件	(148)
1	文件概述	(148)
2	顺序文件的操作	(152)
3	按记录方式的输入输出	(159)
4	随机文件的读写操作	(162)
5	文件操作的出错检测	(166)

开 发 篇

第九章 TURBO C 及其集成开发环境	(167)
1 TURBO C 概述	(167)
2 TURBO C 集成开发环境	(167)
3 集成调试程序	(174)
4 集成环境下的 C 程序基本调试方法	(175)
5 TURBO C 库函数	(177)
第十章 C 语言图形编程	(183)
1 图形初始化与关闭	(183)
2 图形方式下的颜色	(186)
3 线型、填充型式及字型	(189)
4 视区设置与清屏	(194)
5 视区中的坐标与字符串输出	(195)
6 线、方框、立体条形图	(197)
7 图形的存取	(198)
8 动画技术	(201)
9 绘图机驱动软件	(219)
第十一章 汉字处理技术	(224)
1 CC-DOS 概述	(224)
2 汉字信息处理基础	(225)
3 西文操作系统下汉字显示技术	(227)
4 自造实用小型汉字库	(234)
第十二章 应用软件中的人机界面	(239)
1 弹出式窗口系统的基本原理	(239)
2 弹出式窗口系统的实现	(240)
3 以屏幕为对象的接口程序	(252)
4 下拉式菜单系统	(254)
第十三章 数控线切割机自动编程软件开发	(271)
1 数控编程概述	(271)
2 数控自动编程软件的数学模型	(278)
3 数据结构及其它函数	(284)
4 MAP 软件的应用	(305)
后 记	(310)

第一章 C 语言概述

1 C 语言的发展历史

C 语言是美国贝尔(Bell)实验室于 1972 年开发出的一种简捷而又高效的程序设计语言。最初由 Dennis Ritchie 设计编写,后来与 Ken Thompson 一起用它来重写了 Unix 操作系统,从此建立起的 C 语言便逐步成为程序员的一种有力工具。

在最近的几年里,C 语言发展很快,十分流行,被认为是当今结构和语言都最简单且通用性最强的程序设计语言之一。属于强调简捷、易用、规则性强而又可靠的结构程序设计语言系列。C 语言是一种高效率、可移植的语言,用它在一个系统上写好的程序,要想在另一个系统上运行,只需对程序稍作修改或不加修改。若要用于系统软件的设计,如编写编译程序、操作系统、文本处理程序等,C 语言是最好的一种开发工具。由于 C 语言具有描述系统程序的属性,从而引起许多软件工作者在开发商务应用、数据库应用、空白表格软件、通讯包软件等方面发挥了 C 语言的潜在优势。C 语言提供具有汇编语言通常所具有的存取操作,因此,程序员能使他们的程序取得最大的效率。

C 语言的编译系统还包括可供 C 程序员充分有效使用的 50 ~ 200 个内部函数。此外,C 语言还允许用户对其功能进行扩充,即根据自己的需要设计并编译通过的标准函数,然后按系统规范添加到系统标准函数库中去。以短小、简单而又直接的方法来解决普遍而复杂的程序设计问题,C 语言是行之有效的程序设计语言之一。

可以说 C 语言反映了当前计算机的能力,是一种高效实用、灵活的软件开发工具。尽管它的首次开发使用是在 Unix 操作系统上,但今天它已相当普遍地应用于个人计算机(PC)的领域。由于越来越多的开发人员和用户使用 C 语言,所以必须有一套保证 C 程序员使用相同语言工作的工具。通常人们把以 B. W. Kernighan 和 Dennis Ritchie 于 1978 年所著的《The C Programming Language》一书中提出的 C 语言版本为基础,并称该版本为标准 C,1983 年美国国家标准局(ANSI)语言标准化委员会对 C 语言问世以来的若干发展、补充进行了较系统的归纳和总结,并进一步作了修改订正后,公布一个 C 语言标准草案即 83 ANSI C,1987 年又推出了 87 ANSI 标准 C。此后一些在世界上颇有影响的软件公司如 Microsoft、Borland International 等相继在 87 ANSI 标准 C 的基础上又陆续开发了更为实用、灵活、功能丰富的 C 编译系统,如 Microsoft C 1.0—6.0 版本、Turbo C 1.0—2.0、Turbo C++、Borland C++等。

2 一个简单的 C 程序

首先通过一个非常简单的 C 语言程序,使读者初步了解 C 程序的概貌和最基本的特征。

例 1—1 求三个数的平均值。

```
1:    #include <stdio.h>
2:    /* 一个简单的程序例 */
3:    main()
4:    {
5:        float a,b,c,average;
6:        a=5.6; b=4.7; c=8.3;
7:        average=(a+b+c)/3.0;
8:        printf(" average=%f" ,average);
9:    }
```

这个小程序共有 9 行:

第 1 行:告诉计算机本程序需要的有关信息包含在头文件内。

第 2 行:这是一个注释行,常用来说明程序的主要功能、变量的含义或其他说明。当我们了解如何在 C 语言中使用汉字后,也可以用汉语予以注释。对 C 语言来说,使用“/*”和“*/”把注释的内容夹在其间;对于 C++ 来说,可以来用更简捷的方式,即把每行注释的内容放在符号“//”之后,编译系统在遇到“/*”和“*/”之间的内容,或“//”之后的内容时,将分别跳过而不予编译,也就是说注释的内容是仅供程序员阅读、分析程序作参考的。

第 3 行:main()表示一个名称为main的函数,在 C 程序中,凡是在一个标识之后带有一对圆括号“(”、“)”,就表示是一个函数,括号内含有参数或缺省。函数是 C 程序中的基本模块,这里的main()是一个特殊的函数,称之为主函数,在每一个 C 程序的源文件中只能有一个main()。

第 4 行:这个左花括号表示组成函数的语句开始,函数的具体内容就是在左花括号和最后的右花括号之间的部分。

第 5 行:这里说明 a,b,c,average 四个变量为实型变量。

第 6 行:给三个实型变量分别赋以具体的数值。

第 7 行:求三个数的平均值,并把结果赋给 average。

第 8 行:在屏幕上显示average的数值,这里printf()是一个标准函数,它是由编译系统提供的,它的功能包含在第一行中的“stdio.h”文件中。

第 9 行:以右花括号结束 main()和整个程序。

例 1—1 无疑是一个正确的程序,经编译、运行后程序会在屏幕上显示结果:

average=6.2

但这显然是一个不太理想的程序,因为 a,b,c 三个实型数在程序中是固定的,如果想求另外三个数的平均值,必须对程序进行修改,还要对源程序重新进行编译。为此,对例 1—1

稍作改进,以实现求任意三个数的平均值。

例 1—2 输入三个实型数,并求它们的平均值。

```
#include <stdio.h>

main()
{
    float a,b,c,average;
    printf(" 请输入三个实型数:\n" );
    scanf(" a=%f b=%f c=%f" ,&a,&b,&c);
    average=(a+b+c)/3.0;
    printf(" average=%f" ,average);
}
```

这个程序中增加的 `printf(" 请输入三个实型数:\n" );` 一行语句,是利用标准输出函数在屏幕上原封不动地把双引号中的“请输入三个实型数:”显示出来,用以提醒操作人员,并把光标返回到下一行的第一列位置上。所增加的另一行语句中的 `scanf()` 也是一个包含在“`stdio.h`”文件中的标准函数,它的功能通过键盘接受操作者输入的数据。

经过改进的程序可以使操作人员输入任意三个合法的实型数并求解它们的平均值。

这个例子毕竟十分简单。通常用 C 或 C++ 所设计的解决实际问题的程序往往要复杂得多,这时,我们希望写在 `main()` 中的内容愈简洁愈好,所以总是把那些实现某些特定功能的语句另外写成一个相对独立的函数。而在 `main()` 内只要通过具体的参数去调用这些函数就可以了。按照这一思路,对例 1—2 作进一步的改动。

例 1—3 通过函数调用,求三个数的平均值。

```
#include <stdio.h>

float ave(float x,float y,float z);

main()
{
    float a,b,c,average;
    printf(" Enter 3 real numbers: \n" );
    scanf(" %f %f %f" ,&a,&b,&c);
    average=ave(a,b,c);
    printf(" average=%f" ,average);
}

float ave(float x,float y,float z)
{
    float aver;
    aver=(x+y+z)/3.0;
    return(aver);
}
```

我们看到在 `main()` 里并没有进行求三个数的平均值的运算,这个运算是在名为 `ave`

()的函数内实现,然后通过 `return()` 函数,把所求得平均值即 `aver` 的值返回到主函数中,再把它赋值给变量 `average`。由于 `main` 函数要调用 `ave` 函数,而 `ave()` 函数定义在 `main` 函数之后,所以需要在 `main` 函数之前对 `ave` 函数进行说明(或声明)。一是说明 `ave` 是一个函数,二是说明 `ave` 的函数返回值是实型,三是对 `ave` 函数中的参数进行说明。

最后通过一个稍微复杂的例子来描述 C 程序的基本结构。

例 1-4 输入三个实型数,分别求它们的平均值,平方和及均方根值。

```
#include <stdio.h>

float ave(float x,float y,float z);
float sum_square(float x,float y,float z);
float square_root(float x,float y,float z);
main()
{
    float a,b,c,average,s,sn;
    printf(" Enter 3 real numbers: \n" );
    scanf(" %f %f %f" ,&a,&b,&c);
    average=ave(a,b,c);
    s=sum_square(a,b,c);
    sn=square_root(a,b,c);
    printf(" average=%f\n" ,average);
    printf(" sum_square=%f\n" ,s);
    printf(" square_root=%f\n" ,sn);
}

float ave(float x,float y,float z)
{
    float aver;
    aver=(x+y+z)/3.0;
    return(aver);
}

float sum_square(float x,float y,float z)
{
    float s_abc;
    s_abc=x * x+y * y+z * z;
    return(s_abc);
}

float square_root(float x,float y,float z)
{
    float s_abc;
    s_abc=x * x+y * y+z * z;
    return(sqrt(s_abc));
}
```

```

float sn_abc;
sn_abc=sqrt(x * x+y * y+z * z);
return(sn_abc);
}

```

在最后一个函数 `sum—square()` 中 `sqrt()` 函数也是一个标准函数,它被包含在一个“`math.h`”文件中。

3 C 语言的特点

通过以上介绍的几个简单程序,已对 C 语言有了初步的印象,现在可以对 C 语言的特点作一初步的归纳。C 语言和 C 语言在功能上更完善,它基本上包括了 C 语言的功能,所以这里先简单介绍 C 语言的共同特点,至于 C++ 语言的特点将在后面介绍。

(1) C 语言是一种模块化的程序设计语言。C 语言程序是按函数形式进行装配的,可以说 C 语言程序是函数的集合,其中的每个函数都能实现特定的功能。函数有两大类,一类是由标准函数库所提供的,象上节中已出现的 `printf()`、`scanf()` 及 `sqrt()`。C 语言编译系统能提供极其丰富的标准函数,程序员可以把它们作为标准部件那样用来装配自己的程序。另一类函数需要自行设计,设计自己的函数时,同样可以用前面一类标准库函数来装配,自行设计的函数一旦经过编译调试通过并运行证明它是正确无误之后,也可以把它按照一定的规则转化为标准函数,这就是对标准函数库的扩充。以后就可以象使用标准部件那样使用它。这些函数应组织成层次结构。最顶层的一个函数为主函数:`main()`,一个 C 程序文件中只能有一个 `main()`。

(2) C 语言有预处理功能,预处理命令如 `#include`(文件包含) `#define`(宏定义)等是在编译时预先处理的命令,它告诉编译系统在那些系统文件中包含了程序中所用到的标准库函数,说明一些特殊变量的含义。采用预处理命令可以提高程序的可读性和可移植性,并给程序调试提供了方便。

(3) C 语言要求对程序中的数据和函数都要先作说明,然后才能使用。即在使用数据和函数之前必须先声明和定义它们的类型,稍后我们将会看到,C 语言提供有丰富的数据类型,能满足现代程序设计的要求。变量的定义语句和说明语句可以指定其作用域。变量一经定义,同时也就进行了说明。

(4) 运算符种类丰富。汇编语言中使用的运算大多数都能作为 C 语言的运算符来定义。除了在通常的高级语言中使用的算术运算符、逻辑运算符等以外,还定义了按位逻辑运算符、移位运算符、赋值运算符、逗号运算符等。指明函数用的括号“(”和“)”以及数组中所用的括号“[”和“]”等等也被定义成运算符。因此 C 语言的运算多样化并具有较强的数据处理能力,不仅能完成一般高级语言的运算功能,也能实现低级语言的许多功能,既适宜于编写系统软件,也适用于编写面向机器硬件的应用软件。

(5) C 语言主要使用英文字母且大小写敏感。所使用的保留字全部由英文小写字母构成,变量等标识符也由英文小写字母构成。因此,大写和小写字母是当作不同符号来处理的,故称之为大小写敏感。在字符串中可以使用汉字等特殊符号,但只有在汉字能够受到

正确处理的情况下方能使用汉字。

(6) C 语言源程序书写灵活,可以每行一句,也可以一行数句,但语言间须用“;”隔开,且可任意换行,能给程序设计者以较大自由,程序非常简练。C 语言语法检查不严格,如对数组下标溢出不作检查。因此使用 C 语言编程对程序设计人员的要求比较高,所以 C 语言往往为熟练的程序人员使用。

(7) 表达式必定有值。一个常数、一个变量、一个函数都可称为表达式。而由一个或一个以上的表达式进行各种运算的式子也称为表达式。表达式后面若跟着分号(;)的话,便成了一个执行语句,执行一个表达式就是求出该表达式的值。表达式执行后所得到的值叫作评价值。

(8) 有高级的控制结构。C 语言源程序原则上是按顺序执行,其控制流程通过条件分支、多路开关、循环等实现。C 语言的条件分支和循环等控制功能丰富,可以编制出易于理解且便于修改和扩充的结构化程序。

(9) 指针(地址)可作为数据使用。指示数据存放场所的数据称作地址。其值为地址的变量称作指针。地址、指针这些概念在汇编语言中并不少见,由于 C 语言可以处理与汇编语言级别相当的数据,因而可以实现与汇编语言同等的功能。用在函数间传递地址的方法可以编出一些通用的函数,同时可以高速地进行某些复杂的处理,而用通常的高级语言是不能进行这些处理的。不过,地址和指针的概念难以理解,使用起来易出错,且这些错误在多数情况下很难查出。

(10) 可以处理由基本数据组合而成的数据。作为基本数据而提供的有字符型和整型等几种,可以定义并使用由这些基本数据自由组合而成的集合数据。集合数据中有数组和结构,特别是结构对复杂的数据处理尤为重要,在 C++ 语言中它被扩充后起着重要的作用。

(11) 没有字符串处理功能。字符串和字符有着明确的区别,字符串是作为字符的序列集合而处理的。对作为基本数据字符进行运算的运算符,却不存在对作为集合数据的字符串(一般说来对数组)进行运算的运算符,像 BASIC 语言那样用运算符进行字符串的赋值、比较等运算是做不到的。所有这些运算均利用函数进行。但在 C++ 语言中可以自由定义并使用对字符串进行处理的运算符。

(12) 没有输入输出功能。其他高级语言必备的输入输出功能, C 语言中却没有。所有的输入输出均由函数进行。由于那些使用复杂得多的编译程序来实现的文件处理功能均由函数实现,编译程序变得非常紧凑。又由于编译中不含那些受硬件影响较大的输入输出功能,程序在不同机种间的可移植性很强。这也是 C 语言得以广泛普及的原因之一。

(13) C 语言允许把一个程序分割成若干部分进行处理。C 语言可以把一个程序分割成多个模块进行编译,对它们分别进行编译、调试,最后再把它们联接成一个可执行的程序文件。分割后的文件可以仅含有说明语句和定义语句,也可以仅含有函数。利用这种分割功能和变量等的有效范围特征,便可以由多人同时对一个程序进行开发,这样有利于开发大型程序。不过,虽然一个文件所含的函数个数没有限制,但不管哪个函数都必须在一个文件中结束(把一个函数分散在多个文件里是不容许的)。

(14) 采用 C 编译系统的命令行编译功能可以方便地构成命令,在启动用户编制的程序时,可以把命令行的参数传递给执行程序。被传递的参数在程序内进行解释,并据此进行适当的处理,因此可以用用户自编的程序对操作系统(OS)的功能进行扩充,这些追加的程序实际上成了与 OS 所提供的系统命令用法相同的可执行命令,即外部命令。

从以上介绍不难看出,仅仅利用 C 语言的功能已能显示出迄今为止的语言所不具有的优良特色,用扩充后的 C++ 语言将能编制出更为漂亮的程序来。

C++ 语言基本上包含了 C 语言的功能。因此,在下面的介绍中,凡说到“C 语言”的时候,意为 C++ 语言均成立。至于那些只对其中一种语言成立的情形,均特别加以声明。

4 C 程序的结构和风格

这一节主要从 C 语言源程序书写的角度来介绍 C 语言程序的结构和风格。前已提及, C 语言程序书写格式自由,一行内可写几个语句,一个语句也可以写在多行上,无须换行符,而且 C 语言程序没有行号,也不像 FORTRAN 和 COBOL 语言那样严格规定了语句必须从某一列开始;除少数例外情况, C 语言各语法成分间可以空格,总之 C 语言的灵活性为我们的程序设计带来很多方便。但不能因此而随心所欲,在程序设计中应养成良好的风格,尽量采用清晰、明朗、规范的书写方法,既便于阅读理解,又便于分析调试。

4.1 程序书写格式

C 程序的书写格式要求遵守结构化方式,不赞成一行写多句,应养成良好的程序风格。

例如下面的程序,尽管可以通过运行,但让人读起来感到非常杂乱无序,除了节省版面,别无任何好处。

```
main(){unsigned int albatross,marlin,whale;
char sen_string[81];double stars_insky,fish_in_sea;
while(stars_in_sky>fish_in_sea){save_whales(& albatross,& marlin,& whale);
which_greater(& stars_in_sky,& fish_in_sea,
sea_string);}printf(" %s" ,sea_string);}
```

把它改写成例 1—5 的书写格式,给人一种整齐规范,井然有序的感觉,符合结构化程序设计的要求,阅读起来一目了然,这个主函数包含了三种类型数据变量的定义,然后根据如满足一个条件表达式来循环调用另外两个函数,否则即显示一个字符串数组的值。

例 1—5 正确的书写格式。

```
main(){
    unsigned int albatross,marlin,whale;
    char sen_string[81];
    double stars_insky,fish_in_sea;
    while(stars_in_sky>fish_in_sea){
        save_whales(& albatross,& marlin,& whale);
        which_greater(& stars_in_sky,& fish_in_sea,
```

```
    sea_string);}
    printf(" %s" ,sea_string);
}
```

4.2 采用缩进格式

所谓缩进格式主要是指每一行的语句从那一行开始写起。程序书写的格式是完全自由的,从什么地方开始写都无关紧要。通常把 tab(严格地讲应为水平 tab)键和 return 键视同于 space(空白)。以下把 tab、return 和 space 统称为空白字符。因此,在变量名和常数等单词之间,凡可以写空白的地方,原则上都可以自由地进行换行。由于随意换行和加空,可能使写出来的程序很难读,但是,象例 1-6 那样,在每一行的开头加入适当的空白字符,可以使程序显得更便于理解。这种在每一行的开头加适当的空白的方法就叫作缩进。此外,为提高程序的可读性,在功能模块之间也可插一些空行。在字符串数据内部直接插进 return 是不行的,加进 tab 和 space 则被视为不同的字符串数据。下面的例子就是用缩进格式来重新书写例 1-4,为了更具有一般性,可对例 1-4 的功能略作修改,在输入三个任意实型数后,再由键盘输入一个功能选择值。即当输入值分别为 1、2、3 时,程序将自动地选择求三个数的平均值,或平方和,或均方根值。

例 1-6 输入三个实型数,根据开关值执行相应的运算。

```
switch=1      求三个数的平均值
switch=2      求三个数的平方和
switch=3      求三个数的均方根值
```

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

float ave(float x,float y,float z);
float sum_square(float x,float y,float z);
float square_root(float x,float y,float z);

/* 这是一个空行,把预处理命令与程序文本隔开 */
main()
{
    float a,b,c,average,s,sn;
    int i;
    printf(" 请输入 3 个实型数: \n" );
    scanf(" %f %f %f" ,&a,&b,&c);
    printf(" 请输入选择符: " );
    scanf(" %d" ,&i);
    if(i==1)
    {average=ave(a,b,c);
      printf(" average=%f \n" ,average);
    }
}
```

```

    else if(i==2)
    {
        s=sum_square(a,b,c);
        printf(" sum_square=%f\n",s);
    }
    else if(i==3)
    {
        sn=square_root(a,b,c);
        printf(" square_root=%f\n",sn);
    }
}

/* 这是一个空行,把主函数与其它函数隔开 */
float ave(float x,float y,float z)
{
    float aver;
    aver=(x+y+z)/3.0;
    return(aver);
}

/* 空行,把函数与函数之间也隔开 */
float sum_square(float x,float y,float z)
{
    float s_abc;
    s_abc=x * x+y * y+z * z;
    return(s_abc);
}

/* 空行 */
float square_root(float x,float y,float z);
{
    float sn_abc;
    sn_abc=sqrt(x * x+y * y+z * z);
    return(sn_abc);
}

```

采用缩进格式书写,程序中的每个功能模块显得相对独立,流程清晰,作为每个模块开始和结束的花括号“{”、“}”相互对应,即使出现错误或丢失,也容易查找。

4.3 用现代风格来说明和定义函数

函数的说明是对程序中所用到的函数的特征——返回值的类型、函数参数的个数和类型——进行必要的说明,编译系统以函数声明中给出的信息为依据,对调用表达式进行检测,以保证调用表达式与函数之间的正确传递参数。函数的定义是对函数的功能予以具体的表达,是由变量定义部分和可执行语句组成的独立实体。函数的说明与定义之间要有

一定的对应关系,这里所说的对应关系主要是指函数的参数在说明和定义中的一致性,由此而产生两种不同的风格,请看下面的例子。

例 1—7 函数的说明和定义

```
main()
{
    ...

    float new_style(int a,float x); /* 函数说明 */

    ...
}

float new_style(int a,float x) /* 函数定义 */
{
    函数体
}
```

我们注意到函数的说明与函数的返回值的类型,函数名,函数的参数个数及类型之间的严格的一一对应关系,所不同的是函数说明不包含函数体,因而不要求分配内存空间,函数定义则要求给它分配内存单元,用来存放经编译后的函数指令,这里的函数说明和定义所采用的风格称现代风格,对于上例,传统的方式如下:

```
main()
{
    ...

    float new_style(); /* 函数说明 */

    ...
}

float new_style(a,x) /* 函数定义 */
int a;
float x;
{
    函数体
}
```

这种说明方式在函数说明中未对函数的参数加以说明,在函数的定义中对函数的参数是分两步加以定义的。ANSI 约定,函数的说明和定义均采用现代风格,使用 Turbo C、Turbo C++、Borland C++ 也必须使用现代风格说明和定义函数,否则编译可能不予通过。

4.4 关于注释

一个完整的 C 源程序应该包含恰当得体的注解内容,这对于程序的阅读、分析、调试和移植都起到积极作用,尤其是对于那些由多个程序设计人员分别完成不同的模块,最后

综合而成的大型程序来说,简明扼要的注释能使程序设计人员之间沟通信息,便于相互理解彼此思路。在 Turbo C 中由“/*”和“*/”所界定的任意字符串被当作注释处理,编译时被忽略。在一个程序中,凡可插入空白字符(space、tab 和 return)等的地方均可加适当的注释。注释跨行书写也行,但注释的嵌套(即在注释中加注释)一般是不行的。在 C++ 语言中一行内结束的注释可用“//”打头,该行结束的地方也就是注释结束的地方。虽然以“/*”和“*/”界定的注释是不能嵌套的,但是在这样的注释中容许指定以“//”打头的注释。在以“//”打头的注释中也可以指定以“/*”和“*/”界定的注释。

注释不是写给计算机看的,而是写给人看的,凡是与程序有关的各种信息均可写在注释中。利用注释可以记载注意事项,说明处理内容等等。但是插入注释时,应注意保持程序的可读性。例如,

```
int /* 注释不当一例 */ a;
```

这样的注释,使得程序读起来很不舒服。

例 1-8 注释举例。

/* 说明注释用法的程序例子

这种类型的注释

可以跨行书写

*/

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

```
main(){ // 执行始于此
    int a; // 变量定义
    printf(" : " ); // 提示数据输入
    scanf(" %d" ,&a); // 数据输入
    printf(" %d%d\n" ,a,a * a); // 显示结果
}
```

这一程序例子的执行内容与例 1-1 相同。以“/*”和“*/”界定的字符串作为注释被编译程序忽略。在 C++ 语言中,“//”以下的字符串直到该行的结束均视为注释,如果用的是 C 语言,可把以“//”打头的注释改写成以“/*”和“*/”界定的注释。

4.5 C 语言中的汉字

C 语言中的汉字通常是在两种情况下出现:一是在注释中,采用中文注释能够使使用汉字的程序设计人员更容易阅读和理解程序的结构和思路;二是用在用户界面中,用汉字设计出的图文并茂的人机界面能给使用汉字的程序操作者一种亲切感,尤其在设计用于各种不同行业,不同层次的应用软件,有了汉字界面和汉字提示,能使操作者很容易很方便掌握和操作这类软件系统。所以不少软件设计人员都致力于汉化式菜单,用户界面的研究开发工作。

一个最简单的办法是在中文操作系统下使用 C 或 C++ 语言编写程序,如由我国电子部六所在 PC-DOS 的基础上,为 IBM-PC 及其兼容机开发的 CC-DOS 汉字操作系统,在该系统的支持下,使用 C 或 C++ 开发应用软件,就可以直接输入汉字,类似的操作系统还有 2.13 汉字操作系统、SPDOS、UCDOS、TW 操作系统等,对于汉字的使用有两