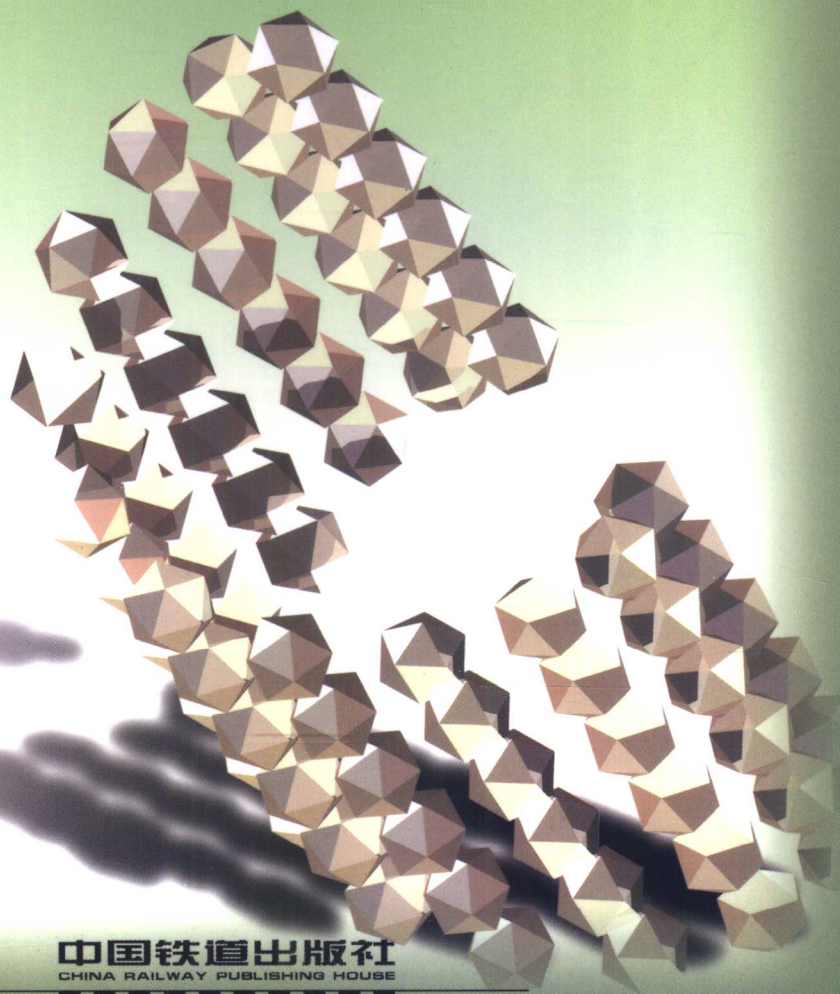




高等院校新课程体系计算机基础教育规划教材

C 语言程序设计

苏长龄 刘威 主编



中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高等院校新课程体系计算机基础教育规划教材

C 语言程序设计

苏长龄 刘 威 主 编

王北星 于秀霞 副主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书是一本学习 C 语言程序设计的实用教材。教材的内容包括数据类型及其表达式、3 种基本结构的程序设计、数组、函数、构造数据类型、指针、文件和位运算等。全书内容符合教育部计算机课程教学指导委员会颁布的大纲要求。本书安排的教学内容具有很强的实用性和可操作性, 可作为高等学校及高职高专各专业学生相关课程或参加计算机等级考试的教材, 也可作为高等院校成人教育的培训教材或教学参考书。

本书配套有《C 语言程序设计习题解答与上机指导》供读者学习时选用, 并为授课教师备有多媒体电子教案。

图书在版编目 (C I P) 数据

C 语言程序设计/苏长龄, 刘威主编. —北京: 中国铁道出版社, 2006. 2

高等院校新课程体系计算机基础教育规划教材
ISBN 7-113-06882-0

I. C... II. ①苏... ②刘... III. C 语言—程序设计
—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2006) 第 013255 号

书 名: C 语言程序设计

作 者: 苏长龄 刘 威 王北星 于秀霞

出版发行: 中国铁道出版社 (100054, 北京市宣武区右安门西街 8 号)

策划编辑: 严晓舟 秦绪好

责任编辑: 严 力 崔晓静 文 正

封面设计: 薛 为

封面制作: 白 雪

责任校对: 刘 洁

印 刷: 北京新魏印刷厂

开 本: 787×1092 1/16 印张: 17.75 字数: 426 千

版 本: 2006 年 3 月第 1 版 2006 年 3 月第 1 次印刷

印 数: 1~5 000 册

书 号: ISBN 7-113-06882-0/TP·1715

定 价: 24.00 元

版权所有 侵权必究

凡购买铁道版的图书, 如有缺页、倒页、脱页者, 请与本社计算机图书批销部调换。



高等院校新课程体系计算机基础教育规划教材

编
委
会

主 任： 何 桥

副 主 任： 吕英华 邢满堂 苏长龄

委 员： （按姓氏字母先后为序）

卞 清 崔荣一 何 桥 李 平

吕英华 宋绍成 苏长龄 滕国文

王北星 邢满堂 尹相一 于凡华

于江涛 张凌雯 张 强 朱丽莉

为进一步推动高等学校的计算机基础教学改革,提高教学质量,适应新世纪对高级人才知识的需求,掌握一门计算机语言已经成为应用计算机必备的条件之一。目前,不论是计算机专业还是非计算机专业的学生,都将C语言作为学习程序设计语言的入门语言。因为C语言功能丰富,表达能力强,使用灵活方便,应用面广,目标程序效率高,可移植性好,既具有高级语言的优点,又具有低级语言的许多特点。学习和掌握C语言已成为许多学生的迫切需要。

现组织长期从事计算机基础教学工作的教师编写了《C语言程序设计》一书,编写内容的选择上充分考虑计算机学科发展快、更新快的特点,力图反映新内容,使之具有先进性,同时又兼顾了高等学校计算机语言教学的实际情况使之具有现实可行性,尽量做到少而精,力求通俗易懂。本教材的主要特点是:由浅入深、循序渐进地讲解C语言程序设计的思想和方法。全书在编写过程中,力求做到概念准确、内容正确、由浅入深、循序渐进、繁简适当。每章都有小结和习题,通过实例和习题加深基本概念的理解和掌握,提高计算机操作的水平。为进一步满足本书的教学和实验要求,我们还编写了与本书配套的《C语言程序设计习题解答与上机指导》供读者学习时选用。

本书可作为高等学校及高职高专各专业学生相关课程或参加计算机等级考试的教材,也可作为高等院校成人教育的培训教材或教学参考书。

本书由苏长龄、刘威主编,王北星、于秀霞副主编。刘威编写第1章~第5章,于秀霞编写第6章和第7章,苏长龄编写第8章和第9章,王北星编写第10章和第11章,全书由苏长龄统稿。感谢在本书的编写和出版过程中,中国铁道出版社给予的大力帮助和支持。

由于编者水平和经验有限,书中难免有不足之处,恳请读者提出宝贵意见和建议。

编者

2006年2月

第1章 C语言概述	1
1.1 概述	1
1.1.1 C语言及其特点	1
1.1.2 C语言程序的组成和结构特点	3
1.2 基本符号	5
1.2.1 基本符号集	5
1.2.2 标识符	6
1.3 算法及其描述方法	7
1.3.1 算法的概念	7
1.3.2 算法的描述方法	8
1.4 程序设计方法简介	11
小 结	13
习题一	14
第2章 数据类型及其表达式	15
2.1 数据类型	15
2.2 常量	16
2.2.1 整型常量	16
2.2.2 实型常量	17
2.2.3 字符常量	18
2.2.4 字符串	19
2.2.5 符号常量	20
2.2.6 转义字符	21
2.3 变量	22
2.3.1 变量的概念	22
2.3.2 变量说明	23
2.3.3 变量地址	23
2.3.4 变量的初始化	24
2.3.5 数据类型	25
2.4 运算符和表达式	27
2.4.1 运算符和表达式	27
2.4.2 赋值运算	28
2.4.3 算术运算	30
2.4.4 逗号运算	32
2.4.5 关系运算	33
2.4.6 逻辑运算	34

2.4.7 条件表达式	35
2.4.8 运算符的结合律和优先级	36
小 结	37
习题二	39
第 3 章 顺序结构的程序设计	43
3.1 顺序结构的语句	43
3.1.1 表达式语句、空语句和赋值语句	43
3.1.2 复合语句	44
3.2 字符数据的输入和输出	45
3.2.1 字符输出函数 putchar()	45
3.2.2 字符输入函数 getchar()	46
3.3 输出函数 printf()	48
3.4 输入函数 scanf()	51
小 结	53
习题三	54
第 4 章 选择结构的程序设计	57
4.1 条件选择结构	57
4.2 条件分支的嵌套	60
4.3 开关选择结构	63
4.4 无条件选择结构	70
小 结	71
习题四	71
第 5 章 循环结构的程序设计	74
5.1 用 goto 语句循环结构	74
5.2 while 语句	75
5.3 do...while 语句	78
5.4 for 语句	80
5.5 循环嵌套结构	84
5.6 循环结构中使用 break 和 continue 语句	85
5.6.1 break 语句	85
5.6.2 continue 语句	87
5.7 循环语句小结	88
5.8 程序举例	88
小 结	90
习题五	91
第 6 章 数组	94
6.1 数组和数组元素	94

6.2 一维数组	95
6.2.1 一维数组的定义和使用	95
6.2.2 一维数组的初始化	97
6.2.3 一维数组程序举例	97
6.3 多维数组	102
6.3.1 二维数组的定义和使用	102
6.3.2 二维数组的初始化	104
6.3.3 二维数组程序举例	106
6.4 字符数组	108
6.4.1 字符数组的定义和使用	108
6.4.2 字符数组的初始化	108
6.4.3 字符串的输入和输出	109
6.4.4 用于字符处理的库函数	111
小 结	112
习题六	114
第7章 函数	118
7.1 C 语言程序的组成	118
7.1.1 概述	118
7.1.2 C 语言程序的组成	119
7.1.3 存储类型和变量的作用域	120
7.2 函数的参数	123
7.2.1 带参数函数定义的一般形式	123
7.2.2 形式参数和实际参数	124
7.2.3 数组作为函数的参数	125
7.3 函数的调用	130
7.3.1 函数的原型	130
7.3.2 函数调用的一般形式	132
7.3.3 函数的值	133
7.3.4 函数的递归调用	134
7.4 程序编译预处理	138
7.4.1 宏定义	138
7.4.2 文件包含	140
7.4.3 条件编译	140
小 结	143
习题七	144
第8章 构造数据类型	148
8.1 结构体	148
8.1.1 结构体类型定义及结构体类型变量说明	148

8.1.2	结构体类型变量的使用.....	152
8.1.3	结构体变量的初始化.....	154
8.1.4	结构体数组	155
8.1.5	结构体和函数	157
8.2	共用体	158
8.2.1	共用体类型的定义和共用体变量的说明	158
8.2.2	共用体成员的使用.....	159
8.3	枚举类型	161
8.3.1	枚举类型的定义和枚举变量的说明	161
8.3.2	枚举类型数据的使用.....	162
8.4	用 typedef 定义类型.....	163
小 结		165
习题八		166
第 9 章	指针.....	171
9.1	指针变量的概念	171
9.1.1	变量和地址	171
9.1.2	指针变量和指针的类型.....	172
9.2	变量的指针与指针变量.....	172
9.2.1	指针变量的定义及使用.....	172
9.2.2	指针变量的初始化.....	175
9.2.3	指针运算	176
9.3	指针与数组	178
9.3.1	指向数组的指针.....	179
9.3.2	字符指针与字符数组.....	183
9.3.3	多级指针及指针数组.....	186
9.3.4	指针与多维数组.....	190
9.4	指针与函数	194
9.4.1	函数参数为指针.....	194
9.4.2	函数的返回值为指针.....	197
9.4.3	指向函数的指针.....	198
9.4.4	命令行参数	202
9.5	指针与结构体	205
9.5.1	结构体指针与函数.....	205
9.5.2	结构体指针与指向结构体数组的指针	207
9.6	链表.....	209
9.6.1	链表和动态存储分配.....	209
9.6.2	用包含指针项的结构体变量构成结点	211
9.6.3	用于动态存储分配的函数.....	214

9.6.4 链表应用举例	215
小 结	221
习题九	223
第 10 章 文件	229
10.1 概述	229
10.1.1 文件概述	229
10.1.2 缓冲文件系统和非缓冲文件系统	230
10.1.3 标准输入输出库函数	230
10.1.4 标准设备文件及 I/O 改向	233
10.2 缓冲型文件输入输出系统	234
10.2.1 文件 (FILE) 类型结构及文件指针	234
10.2.2 文件的打开与关闭	235
10.2.3 文件的读写	236
10.2.4 文件的定位	243
10.2.5 出错的检测	245
10.3 非缓冲型文件输入输出系统	245
小 结	249
习题十	250
第 11 章 位运算	253
11.1 基本位运算符与位运算	253
11.1.1 按位与运算符 (&)	253
11.1.2 按位或运算符 ()	254
11.1.3 按位异或运算符 (^)	255
11.1.4 按位取反运算符 (~)	256
11.2 位移运算符与位移运算	257
11.2.1 左移运算符 (<<)	257
11.2.2 右移运算符 (>>)	258
11.3 位运算的复合赋值运算符	258
11.4 位段	259
11.4.1 位段的定义	259
11.4.2 位段的引用	261
11.5 位运算应用举例	262
小 结	264
习题十一	265
附录 A C 语言常用库函数	267
附录 B ASCII 码表	272
参考文献	273

第 1 章 C 语言概述

本章主要介绍 C 语言及其特点、C 语言程序的组成和结构特点、C 语言的基本符号和算法及其描述方法,最后,对结构化程序设计做了简明阐释。

1.1 概述

1.1.1 C 语言及其特点

1. 程序设计语言的发展

随着计算机的诞生,用于计算机的程序设计语言随即产生,人们利用程序设计语言编制程序以便更好地使用计算机。程序设计语言经历了机器语言、汇编语言、高级语言的不同发展阶段。机器语言是计算机能够识别和直接执行的二进制语言,用机器语言编写的程序不直观、难懂、易出错。下面这两行就是某计算机的机器指令,用于计算十进制数 63 与 56 相加之和:

```
0111010000011111    (数 63 送入寄存器 A)
0010010000111000    (寄存器 A 的内容 63 与 56 相加,其和 119 送回 A)
```

为了克服机器语言的不足,人们又发明了汇编语言,汇编语言用助记符来表示机器语言,但计算机不能直接执行,需通过汇编系统将汇编翻译成机器语言。机器语言与汇编语言都是面向机器的语言,一般称为低级语言。下面就是计算十进制数 63 与 56 相加之和的汇编指令:

```
MOV    A, #63          ;数 63 送入寄存器 A
ADD     A, #56          ;寄存器 A 的内容 63 与 56 相加,其和 119 送回 A
```

随着计算机的发展,人们发明了接近自然语言的程序设计语言,这些语言能直接表达计算式和逻辑式,如 FORTRAN、Pascal、BASIC、COBOL、C 以及 C++等,被称为高级语言或算法语言。“A=63+56”就是用 C 语言计算 63 与 56 相加之和的语句。

2. C 语言简介

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,也是国际公认的最重要的几种通用程序设计语言之一,它既可用于编写系统软件也可用来编写应用软件。

1972 年,C 语言是由贝尔实验室的 Dennis Ritchie 和 Brian Kernighan 根据 Thompson 的 B 语言设计的,而 B 语言又是由一种早期的编程语言 BCPL (Basic Combined Programming Language) 发展演变而来的。BCPL 的根源可以追溯到 1960 年的 ALGOL 60 (Algol Programming Language), ALGOL 60 是一种面向问题的高级语言,离硬件较远。1963 年,英国剑桥大学推出 CPL (Combined Programming Language) 语言, CPL 修改了 ALGOL 60,使其能够直接做较低层次的操作。1967 年英国剑桥大学的 Martin Richards 对 CPL 做了改进,推出了 BCPL 语言。

最初的 C 语言是为描述和实现 UNIX 操作系统提供的一种工具语言。但 C 语言并没有被束缚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——《可移植 C 语言编译程序》,用该程序编写的 UNIX 系统迅速在各种机器

上实现, UNIX 系统支持的 C 语言也被移植到相应的计算机上。C 语言和 UNIX 系统在发展过程中相辅相成, 得到了广泛应用, 使 C 语言先后被移植到各种大、中、小、微型计算机上。

以 1978 年发表的第 7 版 UNIX 系统中的 C 语言编译程序为基础, B.W.Kernighan 和 D.M.Ritchie 合著了《The C Programming Language》。这本书中介绍的 C 语言成为后来广泛使用的 C 语言版本的基础, 被称为标准 C 语言。1983 年, 美国国家标准化协会 (ANSI) 根据 C 语言问世以来的各种版本对 C 语言的发展和扩充制定了新的标准, 称为 ANSI C。1990 年, C 语言成为国际标准化组织 (ISO) 通过的标准语言。本书介绍的 C 语言部分以 ANSI C 为基础。书中的程序全部在 Turbo C 2.0 系统中调试通过。

3. C 语言的特点

一种语言之所以能存在和发展, 是因为它有不同于其他语言的特点。C 语言也是如此, 它的特点是多方面的, 人们从不同的角度可总结出众多特点, 但从全面考虑可归纳为以下几点。

(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。它具有许多通常只有汇编语言才具备的功能, 如位操作、直接访问物理地址等, 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 语言代替汇编语言, 减轻了程序员的负担, 提高效率, 而且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、数组等操作, 这些较高级的功能必须通过调用函数来完成。这看起来是个缺陷, 但这种语言规模小, 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 BASIC 语言, 关键字多达 100 个以上。

(2) C 语言是结构化的程序设计语言。C 语言主要结构成分是函数, 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。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, 十分便于采用自顶向下、逐步细化的结构化程序设计技术。因此, 用 C 语言编制的程序容易理解、便于维护。

(3) C 语言具有丰富的运算能力。在 C 语言中除了一般高级语言使用的算术运算及逻辑运算功能外, 还具有独特的以二进制位 (bit) 为单位的位与、位或、位非以及移位操作等运算。并且 C 语言具有如 a++、b--等单项运算和 +=、-=等复合运算功能。

(4) C 语言数据类型丰富, 具有现代语言的各种数据类型。C 语言的基本数据类型有整型 (int)、浮点型 (float)、字符型 (char)。在此基础上按层次可产生各种构造类型, 如数组、指针、结构体、共用体等。同时还提供了用户自定义数据类型。用这些数据类型可以实现复杂的数据结构, 如栈、链表、树等。因此, C 语言具有较强的数据处理能力。

(5) C 语言具有预处理能力。在 C 语言中提供了 #include 和 #define 两个预处理命令, 实现对外部文件的包含以及对字符串的宏定义。同时还具有 #if...#else 等条件编译预处理语句。这些功能的使用提高了软件开发的工作效率并为程序的组织和编译提供了便利。

(6) C 语言可移植性好。目前, C 语言在许多计算机上实现, 大部分是由 C 语言编译移植得到的。C 编译程序的可移植性使 C 语言程序便于移植。

C语言的优点很多,但也有一些不足。如语法限制不太严格、类型检验太弱、不同类型数据转换比较随便,这就要求程序员对程序设计的方法和技巧更熟练,以保证程序的正确性。总之,C语言已成为国内外广泛使用的一种编程语言,并且非常适合用于程序设计语言课程的教学。

1.1.2 C语言程序的组成和结构特点

C语言是函数型语言,函数是构成C语言程序的基本单位。下面我们通过一个例子来分析C语言程序的组成和结构。

【例 1.1】C语言程序的组成和结构。

```
main()                                /* 主函数 */
{
    int a,b,sum;                      /* 定义 a、b 和 sum 共 3 个变量 */
    a=3; b=4;                        /* 为 a,b 赋值 */
    sum=add(a,b);                    /* 调用函数 add,将得到的值赋给变量 sum */
    printf("sum=a+b=%d\n",sum);      /* 屏幕输出 sum 变量的值 */
}

int add(int x,int y)                 /* 定义 add 函数和形式参数 x,y */
{
    int z;                           /* 定义 z 变量 */
    z=x+y;                           /* 将变量 x 与 y 相加的和送给 z */
    return(z);                       /* 返回 z 的值送给 add */
}
```

运行结果:

sum=a+b=7

从本例中可以看到,C语言程序有如下特点:

1. C程序由函数组成

C语言源程序由若干个函数组成,函数是C程序的基本单位。组成程序的若干函数中必须有且仅有一个名为main的函数。例1.1中包含两个函数:main()和add()。因为在main()函数中调用add(),所以main()为主函数,add()为被调用的函数。被调用函数可以是系统提供的库函数,如例1.1中的printf(),也可以是用户根据需要自己定义的函数,如例1.1中的add()。一个C程序可以包含零个到多个用户自定义函数。

2. C语言函数由函数首部和函数体两部分组成

(1) 函数首部

这部分包括函数名、函数类型、参数名和参数类型。

如例1.1中函数add()的说明部分:

int	add	(int	x,	int	y)
↓	↓	↓	↓	↓	↓
函数类型	函数名	参数类型	参数名	参数类型	参数名

函数名后必须有一对圆括号“()”,这是函数的标志,函数类型是函数返回值的类型。参数类型就是形参类型。形参可以有也可以省略,形参省略时函数名后的一对圆括号不能省略,如函数main()就没有参数。如果有参数,放在圆括号中,如int add(int x,int y)。

参数类型的说明也可以放在圆括号外，是传统的函数说明形式，如：

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

这种参数类型的说明形式和放在圆括号中的参数说明形式 `int add(int x,int y)` 作用一样。

(2) 函数体

函数体由函数首部下面最外层的一对大括号中的内容组成。一个函数如果有多对大括号，则最外层一对中的内容为函数体的范围。

函数体一般包含变量定义（变量说明）和执行语句两部分。在例 1.1 中函数 `main()` 中的 `int a,b,sum;` 是变量定义部分，其余是语句执行部分。

3. 函数 `main()`

C 程序必须有函数 `main()`，习惯上称其为主函数。C 语言程序总是从函数 `main()` 开始执行，并且在函数 `main()` 中结束，这与函数 `main()` 在程序中的位置无关，函数 `main()` 可以在整个程序的任意位置，通常把函数 `main()` 放在程序中其他函数的前面。

4. C 程序书写格式自由

C 程序没有行号，书写格式自由，一行内可写多条语句，且语句中的空格和回车符均可忽略不计。一个语句也可以写在多行上，用“\”做续行符。

5. 程序中的每个语句后必须有一个分号

分号“;”是 C 语句的一部分。例如，`sum=a+b;` 中的分号不可少，即便是程序的最后一个语句也应包含分号。

6. C 语言本身没有输入输出语句

输入和输出的操作是由库函数 `scanf()` 和 `printf()` 等来完成的。C 语言对输入输出实行“函数化”。

7. 可以在 C 程序的任何部分加注释，以提高程序的可读性

注释使程序变得清晰，能帮助用户阅读和理解程序。给程序加注释是一个良好的编程习惯。C 语言注释部分由“/*”开始，至“*/”结束，应包括在“/*”和“*/”之间，“/”和“*”之间不允许留有空格。注释部分允许出现在程序中的任何位置上。注释可为若干行，但不允许在“/*”和“*/”中间又出现“/*”和“*/”注释。

下面介绍几个简单的 C 语言程序，以便对 C 程序结构有进一步的了解。

【例 1.2】最小的 C 程序例。

```
main()
{ }
```

这是一个最小的 C 程序，什么也不做，但这是符合 C 语言函数规定的程序，由 `main()` 组成的函数的首部和一对大括号中无任何内容的函数体，在计算机上运行时没有错误，因为函数体是空的，这个 C 程序什么也不做。

【例 1.3】函数 C 程序例。

```
main()          /*主函数*/
{
    printf("This is a C program.");
}
```

运行结果:

This is a C program.

这个程序的函数 `main()` 的函数体中有一个输出函数 `printf()`，在计算机上运行时 `printf()` 按照原样向显示屏幕输出双引号中的字符串，本程序的字符串是英文句子 This is a C program。

【例 1.4】 编写 C 语言程序求键盘输入的两个数中较小的数，并且将其输出到屏幕上。

```
main()                                /* 主函数 */
{
    int a,b,c;                        /* 说明 3 个变量 a,b 和 c */
    printf("Input two integers: "); /* 输出一行提示信息 */
    scanf("%d,%d",&a,&b);            /* 键盘输入两个数值送给 a 和 b 变量 */
    c=min(a,b);                      /* 调用 min 函数,运行结果送给 c 变量 */
    printf("min=%d",c);              /* 输出 c 变量的值 */
}

int min(int x,int y)                 /* 定义 min 函数,x,y 为形参 */
{
    int z;
    if(x<y) z=x;                    /* 条件语句: 如果 x 小于 y 则 z=x,否则 z=y */
    else z=y;
    return(z);                      /* 将 z 的值返回,通过 min 带回调用处 */
}
```

当运行上面这个程序时，首先，屏幕上显示一条提示信息：

Input two integers:

要求用户从键盘输入两个整数。如果用户输入“3”和“5”，即：

Input two integers: 3,5 ✓

这里，符号“✓”表示按一下回车键，显示输入的结束。此时屏幕显示运行结果：

min=3

本程序包括主函数 `main()` 和被调用的函数 `min()`。程序执行 `scanf()` 时，操作员由键盘输入两个整数值送给 `a` 和 `b` 变量。程序执行 `c=min(a,b)` 时，调用函数 `min()`，将 `a` 的值送给 `x`，`b` 的值送给 `y`，程序转到 `min()` 函数执行，`min()` 函数中 `if` 语句的作用是将 `x` 变量和 `y` 变量中的较小值赋给 `z` 变量。`return` 语句的作用是将 `z` 变量的值返回给函数 `min()`，同时程序返回主函数执行，将 `min()` 函数值再送给 `c` 变量。最后 `printf()` 输出 `c` 变量的值到屏幕。

1.2 基本符号

1.2.1 基本符号集

C 语言的基本符号是指在 C 程序中可以出现的字符，主要由 ASCII 字符集中的字符组成，

包括阿拉伯数字、大小写英文字母、特殊符号、转义字符和键盘符号等。这些字符多数是可以见到的，对于不可见的字符（如回车键）C 语言规定用转义字符来表示，转义字符将在本书 2.2 节介绍。C 语言的基本符号具体是以下几部分。

- (1) 阿拉伯数字 10 个：0, 1, 2, 3, ..., 9。
- (2) 大小写英文字母各 26 个：A, B, C, ..., Z, a, b, c, ..., z。
- (3) 下划线：_。
- (4) 特殊符号，主要是指运算符和操作符，通常由 1~2 个特殊符号组成：+、-、*、/、%、<、<=、>、>=、==、!=、&&、||、!、,、&、|、~、=、++、--、? :、<<、>>、()、[、]、.、->、+=、-=、*=、/=、%=、&=、^=、|=、^、#、sizeof。

1.2.2 标识符

标识符是一个在 C 语言中作为名字的字符序列，用来标识变量名、类型名、数组名、函数名和文件名等，其实标识符就是一个名字，C 语言规定了标识符的命名规则。C 语言的标识符可分为用户标识符、保留字和预定义标识符共三类，有些教材称保留字为关键字。

1. 用户标识符

用户可以根据需要对 C 程序中用到的变量、符号常量、自己的函数或文件指针进行命名，形成用户标识符，这类标识符的构成规则如下。

(1) 一个标识符由英文字母、数字、下划线组成，且第一个字符不能是数字，必须是字母或下划线。例如，a、_A、aBc、xlz、y_3 都是合法的标识符，而 123、3_ab、#abc、!45、a*bc 都是非法标识符。

(2) 标识符中大、小写英文字母的含义不同。例如，SUM、Sum 和 sum 代表 3 个不同的标识符，这一点一定要注意。

(3) C 语言本身并没有要求标识符的长度，不同的 C 编译系统允许包含的字符个数有所不同，通常可以识别前面 8 个字符，但在任何机器上，所能识别的标识符的长度总是有限的，有些系统可以识别长达 31 个字符的标识符（如 VAX-11 VMSC），而有些系统只能识别 8 个字符长度的标识符，这意味着即使第 9 个字符不同，只要前 8 个字符一样，系统也认为是同一个标识符，如 Category1 和 Category2。因此，为了避免出错和增加可移植性，最好令标识符前 8 个字符有所区别。

(4) 用户取名时，应当尽量遵循“简洁明了”和“见名知义”的原则。一个好的程序，标识符的选择应尽量反映出所代表对象的实际意思。如表示“年”可以用“year”，表示“长度”可以用“length”，表示加数的“和”可以用“sum”等，这样的标识符增加了可读性，使程序更加清晰。

2. 保留字

保留字是 C 语言编译系统固有的，用做语句名、类型名的标识符。C 语言共有 32 个保留字，每个保留字在 C 程序中都代表着某一固定含义，所有保留字都要用小写英文字母表示，且这些保留字都不允许作为用户标识符使用。C 语言的保留字如表 1-1 所示。

表 1-1 C 语言保留字

描述数据类型定义	typedef void
描述存储类型	auto extern register static volatile
描述数据类型	Char const double float int long short signed struct union unsigned enum
描述语句	break case continue default do else for goto if return sizeof switch while

注意：

- (1) 所有保留字都用小写字母表示。
- (2) 用户自定义的常量名、变量名、函数名和类型名不能使用上述保留字。

3. 预定义标识符

这些标识符在 C 语言中都具有特定含义，如 C 语言提供的编译预处理命令#define 和#include，C 语言语法允许用户把这类标识符作其他用途，但这将使这些预定义标识符失去系统规定的原意，鉴于目前各种计算机系统的 C 语言已经把这类标识符作为统一的编译预处理中的专用命令名使用，因此为了避免误解，建议用户不要把这些预定义标识符另做它用或将它们重新定义。

1.3 算法及其描述方法

1.3.1 算法的概念

1. 算法

在日常生活中，人们每做一件事，都要遵循一定的方法，比如要到某一个地方，需要按照某一条路线去，乘什么车，在什么地方下车，这就是解决问题的方法即算法。程序设计也是如此，编写一个程序，首先要设计算法，依据此算法进行编程。什么是算法呢？著名计算机科学家沃思（N.Wirth）对程序有如下的描述：

程序=数据结构+算法

说明一个程序由以下两部分组成。

- (1) 对数据的描述和组织形式，即数据结构（Data Structure）。
- (2) 对操作或行为的描述，即操作步骤，也就是算法（Algorithm）。

数据是操作的对象，操作的目的是对数据进行加工处理。编写一个程序的关键就是合理地组织数据和设计算法。所谓算法，就是一个有穷规则的集合，其中规则确定了一个解决某一特定类型问题的运算序列。简单地说，算法就是为解决一个具体问题而采取的确定的有限操作步骤。这里的算法指的是计算机算法。

2. 算法的特性

算法必须具备如下 5 个特性。

- (1) 有穷性：一个算法必须总是在执行有限步之后结束。有穷性也称为有限性，就是指算法的操作步骤是有限的，每一步骤在合理的时间范围之内完成，如果计算机执行一