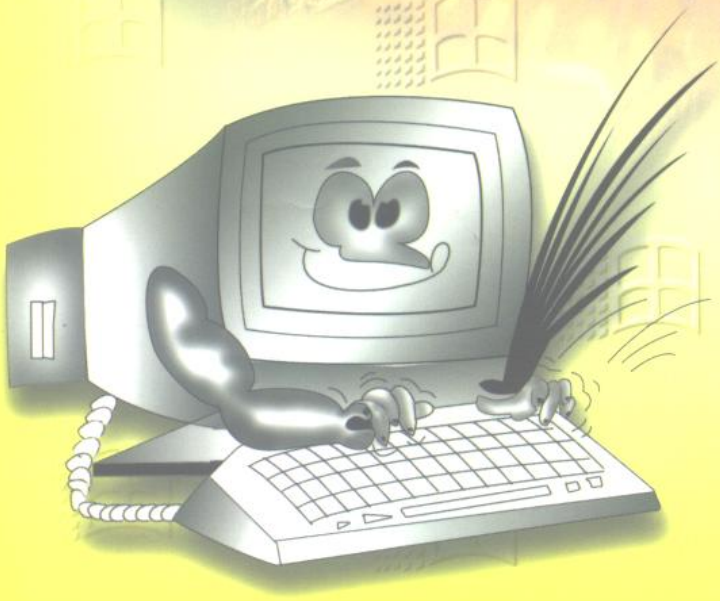
万水计算机培训系列丛书

C语言编程入门

王路敬 主编
满伟 董占山 李文炬 编著



中国水利水电出版社

万水计算机培训系列丛书

C语言编程入门

王路敬　主编

满　伟　董占山　李文炬　编著

中国水利水电出版社

内 容 提 要

作者根据教学经验和实践编写了《C语言编程入门》一书。全书共分两大章，第一章为C语言编程入门，通过大量应用实例对C语言基本知识作了深入浅出的说明；第二章为C语言程序设计应用实例，共有27个典型的C语言程序设计应用实例。每一个实例均按照问题的提出、程序设计思路、程序设计方法、源程序清单的方式展开，对初学C语言程序设计的读者很有帮助。

本书的特点是实例多，所有程序均调试通过，可供C语言初学入门的读者使用。

图书在版编目(CIP)数据

C语言编程入门/王路敬主编.—北京:中国水利水电出版社,1998.5
（万水计算机培训系列丛书）
ISBN 7-80124-493-1
Ⅰ.C… Ⅱ.王… Ⅲ.C语言—程序设计 Ⅳ.TP312
中国版本图书馆CIP数据核字(98)第15363号

书　　名	C语言编程入门
作　　者	王路敬　主编　　满　伟　董占山　李文炬　编著
出版、发行	中国水利水电出版社（北京市三里河路6号　100044） 北京万水电子信息有限公司（北京市车公庄西路20号　100044）
排　　版	北京万水电子信息有限公司
印　　刷	北京市天竺颖华印刷厂
规　　格	787×1092毫米　16开本　18印张　415　千字
版　　次	1998年4月第一版　1998年4月第一次印刷
印　　数	0001—5000册
定　　价	25.00元

前　言

C 语言是一种通用程序设计语言，它具有表达简洁、控制流与数据结构先进和操作功能丰富等特点。它原来是在 UNIX 操作系统上发展起来的，并且 UNIX 本身就是用 C 语言写成的。然而 C 语言并不局限于某一操作系统或机器，因此很容易写出在支持 C 语言的任何机器上运行而不需要作修改的程序，就是说它具有优良的“可移植性”。C 语言不但常用来编写系统程序，而且对于写数据处理程序也同样很有用。由于 C 语言具有上述特点，因此它在微型计算机上也流行起来，成为微机上有影响的一种程序设计语言。为了帮助初学 C 语言的读者学习使用 C 语言进行编程，我们根据教学经验和实践编写了《C 语言编程入门》一书。

全书共分两大章，第一章为 C 语言编程入门。其内容共 16 节，该章通过大量的应用实例，就 C 语言基本知识作了深入浅出的说明，为进行 C 语言编程打下初步基础。第二章为 C 语言程序设计应用实例。该章在第一章介绍 C 语言编程知识的基础上，举例说明运用 C 语言进行程序设计。该章一共列举了 27 个典型的 C 语言程序设计应用实例。每一个实例都按照问题的提出、程序设计思路、程序使用方法、源程序清单这种方式展开，力图给用 C 语言进行编程的读者一个清晰的概念和可供参考借鉴的源程序，对初学 C 语言程序设计的读者很有帮助。

本书的突出特点是实例多，所有的程序都调试通过。不管是介绍 C 语言的基础知识和基本操作还是介绍程序设计实例，都是通过一个个的程序来实现的。这样学习起来容易接受，缩短了学和用之间的距离，对 C 语言编程入门的读者来说该书是一本实用性比较强的参考书。

参加本书编写的有满伟、董占山、李文炬等同志，全书由王路敬主编。由于编者水平有限，缺点与错误在所难免，敬请读者批评指正。

编写者

1997 年 12 月

目　　录

第一章　C 语言基础知识

第一节　C 语言概述

一、C 语言的特点

C 语言是一种通用程序设计语言，用它既可编写计算机系统软件，也可用来七发各种应用软件。C 语言之所以能够得到快速的发展，受到用户的欢迎，是由于它具有下列一些突出的特点。

1. C 语言是结构化的语言

C 语言是以函数为模块来编写源程序的，所以形成的 C 程序是模块化的。

2. C 语言的语句简洁、灵活，使用方便

C 语言一共有 32 个关键字，9 种控制语句，程序书写形式自由。

3. C 语言具有丰富的运算符

C 语言一共有 34 个运算符，其中有很多运算符对应于常用的机器指令，比如++等可直接编译成机器代码。使用起来简单精练，生成的机器代码质量高、内存占用少、运算速度快。

4. C 语言的数据类型丰富，可进行各种复杂的数据结构运算，链表、树、栈的运算

很多关系型数据库是用 C 语言开发的，比如 ORACLE 、 INFORMIX 、 dBASE-III、FoxBASE 等。

5. C 语言的位(bit)操作，可直接对计算机硬件的物理地址进行访问

C 语言有汇编语言的很多功能，既是高级语言又兼有低级语言的功能。用 C 语言可编写出操作系统、数据库、编译器和一般的应用软件。因此说 C 语言既是系统的描述语言，又是通用的程序设计语言。

6. C 语言具有较好的可移植性

C 语言源程序可直接在各类计算机和操作系统下编译后执行（有的程序需稍加修改）。

以上对 C 语言的一般特点进行了介绍，至于 C 语言的其他更深入的特点，将在以后各节中结合程序进行说明。大家在学习和使用 C 语言的过程中，会对它的特点有更多、更深刻的体会。

二、几个简单的 C 程序

程序 1-1-1.c：

```
main()
{
printf("Good morning.\n");
}
```

运行结果：

输入　　输出

Good morning.

如果操作系统是汉化的，可以这样写这个程序:

```
main()
{
printf("早上好。\n");
}
```

运行结果：

输入　　输出

早上好。

在程序中， main()表示主函数。一个 C 源程序可以有多个函数但只能有一个主函数，也必须有一个主函数。在 main()下面的一对大括号内是函数体，即函数体是以大括号开始到大括号结束。在函数体内有一条语句 printf("Good morning.\n");这条语句的作用是在屏幕上打印出一条信息:Good morning. 双引号表示在屏幕输出由双引号括起来的字符串。\n 是光标换行符。分号表示一个语句的结束。程序 1-1-1.c 的函数体中只有一条语句。 printf()是标准输出函数，在后面将详细介绍，在这里只分析一下 C 程序的构成。

程序 1-1-2.c：

```
main()
{
int sum; /*定义一个整型变量 sum*/
sum=125+75; /*为 sum 赋值*/
printf("sum is %d\n",sum);
}
```

运行结果：

输入　　输出

sum is 200

main()为主函数，在大括号中函数体内有三条语句。第一条语句定义了一个整型局部变量，关于局部变量在以后各节中详细介绍。在这条语句后边的/*...*/为注释行，由两对斜杠加星号括起来，是对程序的说明，可增强程序的可读性。但过多的注释会增加编译时间。

第二条语句是赋值语句，等号后是算术表达式，经过计算后将值赋给 sum 。

第三条语句是标准输出语句，表示在屏幕上输出 sum 的值。双引号内 sum is 是原样在屏幕上输出，而%d 是输出格式代码，表示输出整型数据。 printf()函数的格式码很多，在后

面有关节中将详细介绍。在%d 的后面\n 为换行符。在逗号后面的 sum 是变量，是 printf() 要输出的整型变量。

程序 1-1-2.c 说明了 C 程序在使用变量前一定要先定义说明，否则编译器不认识这个变量，比如 sum 整型变量的说明。第二个内容说明了赋值语句的使用，在等号后可以是值或是一个表达式计算后的值。第三个内容说明了 C 程序的注释行是由/*...*/括起来的内容。在/号和*之间不能有空格。

下面这个源程序 1-1-3.c 有两个源文件： 1-1-3.c 和 1-1-3-1.c 。

程序（源文件） 1-1-3.c ：

```
#include <stdio.h> /*标准库函数头文件*/
extern int max(int,int); /*说明外部函数*/
main()
{
int w=10,x=30,y=60;
int z=0;
z=max(x,y); /*调用 max 函数*/
w=max(z,w); /*第二次调用 max 函数*/
printf("max=%d\n",w); /*打印最大数*/
}
```

源文件 1-1-3-1.c ：

```
 int max(a,b)
int a,b; /*形式参数说明成整型变量*/
{
if(a>b) /*判断最大值*/
return(a);
else
return(b);
}
```

编译和链接：

```
c:\tc>tcc 1-1-3 1-1-3-1
```

生成可执行文件 1-1-3.exe

运行结果：

输入	输出
	max=60

上面这个源程序包括两个源文件，每个源文件有一个函数，但只有一个主函数。前面

曾经说过 C 语言是结构化的。一个应用问题可划分成若干小模块来处理，每个模块可编制成一个函数。C 语言应用程序一般由许多小函数组成，各模块独立性强，功能单一，便于修改和维护。除了有结构化的语句外，最主要的是函数。C 语言就像一个个构件或砖堆砌而成的建筑物，而函数就是一块块砖或构件。

一个 C 源程序允许有多个源文件，可一次编译生成可执行程序，也可分别编译生成目标代码(在 DOS 操作系统下是.obj 文件，在 UNIX 操作系统下是.o 文件)，最后再连接成可执行程序。

程序 1-1-3.c 的作用就是取出两个数中的最大数。第一个源文件 1-1-3.c 中的第一条语句是以#include 开头，表明这是预处理命令，将包含某函数说明的头文件嵌入 C 源文件。本程序中，由于调用了库函数 printf()，所以在源文件头需有#include <stdio.h>来说明包含标准库函数的头文件 stdio.h 。第二条语句说明了外部函数 max()，在前面要加上 extern 。第七、八两行是两次调用 max()函数，第一次调用是取出 x 、 y 中最大者并赋给 z ，第二次调用是取出 w 、 z 中最大者并赋给 w ，第九行是打印最大数 w 。

1-1-3-1.c 文件中的 max()函数是判断最大值并返回给调用函数。在形式参数说明上是经典 C 的写法。而 ANSI 规定现代 C 的形式参数说明应放在小括号内。示例如下:

经典 C 形式参数说明为:

```
int max(a,b)
int a,b;
  {
    .  ;
    .  ;
    .  ;
  }
```

现代 C 形式参数说明为:

```
int max (int a ,  int b)
{
   .  ;
   .  ;
   .  ;
}
```

老版本的编译器不认识现代写法。新版本的编译器两种写法均认识。如以下各版本的 C 编译器两种写法均承认。

① Microsoft c 4.0 ， 5.0 ， 6.0 或 C++ 7.0

② Borland C++ 3.0 ， 3.1 ， 4.0

③ UNIX: SCO UNIX/386 3.2V2.0
SCO UNIX/386 3.2V4.2
USL UNIX/386 4.2

④ XENIX-V/3862 、 3.4

三、C 程序的生成步骤

建立C源程序→编译C源程序→连接有关库函数→生成可执行程序→在计算机上运行→输出正确结果。

1. 建立 C 源程序

在 DOS 操作系统下使用的编译器可以是 WS 、 EDLIN 、 WPS 等。

在 UNIX 或 XENIX 操作系统下用的编译器可以是 vi 、 ed 等。

无论在什么操作系统下，用什么编译器， C 源程序的扩展名全是.c 。

2. 编译 C 源程序

⑴ 在 IBM PC/XT 计算机上(CPU 是 8088 芯片)只能安装低版本 C 编译器： Lattice C2.14 。

⑵ 在 286 、 386 、 486 高档微机上(CPU 是 80286 ， 80386 ， 80486 芯片)应当安装 Microsoft C 5.0 以上版本编译器。在安装时可建立组合库，因此编译非常方便。只需一个命令即可完成编译、连接、生成可执行文件。

例① c:\>cl 1-1-1.c 生成 1-1-1.exe 可执行文件

② c:\>cl 1-1-2.c 生成 1-1-2.exe 可执行文件

③ c:\>cl 1-1-3.c 1-1-3-1.c 生成 1-1-3.exe 可执行文件

高版本的编译器 Microsoft C 6.0 、 C++7.0 ， Borland C++ 3.0 、 3.1 、 4.0 还可使用编译器提供的程序员工作台，它可编辑源文件、编译、连接和调试源程序，是一个窗口菜单式的集成软件，可直接生成可执行文件。

⑶ 在 286 、 386 、 486 计算机上安装的是多用户多任务操作系统 UNIX 或 XENIX 。它的 C 编译器是 CC 命令。

格式:CC [选择项].c 文件名

选择项:-c　　不连接只生成.o 文件

　　-o 文件名　　生成指定文件名可执行文件

　　-lm　　连接数学函数库

例① 1-1-1.c

$ cc 1-1-1.c 生成 a.out 可执行文件

或 $ cc -o 1-1-1 1-1-1.c 生成 1-1-1.exe 可执行文件

② 1-1-3.c ， 1-1-3-1.c

$ cc 1-1-3.c 1-1-3-1.c 生成 a.out 可执行文件

或 $ cc -o 1-1-3 1-1-3.c 1-1-3-1.c 生成 1-1-3.exe 可执行文件

或 $ cc -c 1-1-3-1.c 生成 1-1-3-1.c 表示文件

$ cc -o 1-1-3 1-1-3.c 1-1-3-1.c 生成 f3 可执行文件

3. 执行 C 程序

编辑、编译、链接并生成可执行文件后就可执行 C 程序了。

如： c:\tc>1-1-1↵

屏幕显示： Good morning.

在 UNIX 或 XENIX 操作系统下生成的是 a.out 或 1-1-1.exe 可执行文件。

如：$ a.out↵

屏幕显示：Good morning.

或　$ 1-1-1↵

当一个C程序经过以上几个步骤的工作，执行后达到了设计的要求，输出正确的结果，则完成了C程序的设计。如果前面编译虽然通过，但执行C程序输出的结果不正确，说明程序在逻辑上有问题，应当重新编辑进行修改，然后编译、执行。这样反复若干次直到输出正确的结果，才算完成了C程序的设计工作。

第二节 数据类型

要编好C程序，首先要熟悉所处的环境，这个环境就是计算机上安装的操作系统。比如计算机的操作系统是DOS，要学习C语言编程则要求熟炼使用DOS的一些基本命令，还要会使用DOS环境下的任一种正文编辑器，如：WS、WPS、EDLIN、EDIT等。

如果计算机的操作系统是XENIX或UNIX多用户操作系统，同样要求C语言的编程者会使用XENIX或者UNIX的一些基本命令，同时要会使用vi、ed等正文编辑器。

有了这个条件，就可设计相应的程序算法，当数据模型设计好后，就可以着手编写C程序了。

一个C语言程序等于数据结构加上算法。

在C语言中，数据结构是以数据类型的形式来描述的，因此C程序的开始部分一般是程序中要使用的数据类型的说明、定义。

一、C语言的数据类型

1. *基本数据类型*

(1) 字符型 char

(2) 整型 int

(3) 浮点型

① 单精度 float

② 双精度 double

(4) 无值型 void

2. *扩展数据类型*

(1) 数组类型

(2) 函数类型

(3) 指针类型

(4) 结构类型

(5) 联合类型

(6) 位域类型

(7) 枚举类型

二、常量与变量

在C程序的开头说明和定义的数据，又有常量和变量之分。

1. 常量

在程序运行过程中，值不变的量为常量。一般常量的说明、定义是在程序一开头用#define定义一个符号等于一个整型或浮点型数据，比如:#define AB 50或#define CD 2.31等。

在C程序中，常量可以是任一基本数据类型。除了上面介绍的十进制常量外，还有八进制和十六进制常量、字符常量、字符串常量、反斜线字符串常量。下面分别说明：

(1) 八进制常量

八进制常量必须以0(零)为前缀。比如: int a1=015;int b2=035; 等。

(2) 十六进制常量

十六进制常量必须以0x为前缀。如:

int HE=0x70; int HH=0x96;等。

(3) 字符常量

字符常量要用单引号括起来，如：'a'、'A'、'b'、'e' 等。在程序中用单引号括起来的字符，表示取这个字符的ASCII码值。例：'A'的ASCII码值是65，'a'的ASCII码值是97。

(4) 字符串常量

用双引号" "括起来的字符序列称为字符串常量，比如："a"、"AB"等。用双引号括起来的串中包含一个隐含的字符'\0'(NULL)，这是字符串的结束符。"a"实际上是两个字符： a和\0。

(5) 反斜线字符常量

C语言规定了一系列反斜线字符常量，每个反斜线字符常量都含有特殊意义见表1-2-1。在程序中用反斜线码替代等价的ASCII码，增加了程序的可移植性。反斜线字符常量又称为转义序列。

表1-2-1 转义序列表

反斜线码	意义(ASCII码)	十六进制值	键
\0	空NULL	0X00	Ctrl/I
\a	报警BEEP	0X07	Ctrl/G
\b	退一格BS	0X08	Ctrl/H
\t	水平制表HT	0X09	Ctrl/I
\n	换行LF	0X0A	Ctrl/J
\v	垂直制表VT	0X0B	Ctrl/K
\f	走纸换页FF	0X0C	Ctrl/L
\r	回车CR	0X0D	Ctrl/M
\"	双引号"	0X22	"
\'	单引号'	0X27	'
\?	问号?	0X3F	?
\\	反斜线\	0X5C	\
\ddd	三位八进制数d		
\xd	十六进制数d		

程序 1-2-1.c：

```
#define PRICE 50/*定义了常量 PRICE 等于 50*/
main()
 {
int num,total;
printf("请输入整数:");
scanf("%d",&num); /*键盘接收整数*/
total=num*PRICE; /*计算总价钱*/
printf ("总价钱=%d\n",total);
 }
```

运行结果：

输入	输出
	请输入整数:
10↵	总价钱=500

在程序的一开头就用预处理指令#define 定义了一个常量 PRICE 等于 50 。这样在下面凡是出现 PRICE 的地方就代表整型常量 50 。这样比直接在程序中使用 50 要显得清晰，程序的可读性好，如果是大的程序也便于维护。比如价钱(PRICE)要修改为 100 ，这样只需要修改开始的常量定义即可。如果不用常量定义而直接用 50 ，将使得程序的可读性降低，同时修改起来也麻烦，需要将多处的 50 改变为 100 等等。

2. 变量

值可以改变的量称为变量。在 C 程序中，变量是先定义、说明，然后才可以使用，否则在编译时将出错。

变量的定义、说明在不同的位置有不同的性质，因此变量的定义、说明在程序中的位置非常重要。变量在程序中有三个不同的位置：在函数内部说明、定义的变量，称为局部变量；在函数名后的小括号内说明、定义的或是函数名与大括号间说明、定义的是形式参数；在所有函数的外部说明的变量叫全局变量。下面分别介绍这三种变量。

(1) 局部变量。在函数体内部，也就是在函数大括号内说明的变量称为局部变量。

例：

```
fu(void)
   {
   int x;
   int y;
   char ch;
    ...
   }
```

上面说明的 x 、 y 、 ch 全是在函数大括号内说明的变量称为局部变量，又称为自动变量，亦可在前面加上修饰符 auto 。局部变量只在定义它的函数内有效，或者说局部变量的作用域是该变量所在的函数，而且它的值可以不断变化，不断地赋值更新，出了该函数体

则该变量无效。

局部变量在内存中是动态的。当使用它时才分配内存空间，它不像全局变量那样在内存中分配了固定的空间。因此应多使用局部变量，尽量少使用全局变量。这样可以节省计算机的内存空间，同时程序的通用性好并增加了程序的结构化。

(2) 形式参数。调用函数时给的参数叫实际参数，简称实参。比如语句： nn(sc); 其中分号表示一个语句的结束，小括号内的变量 s 和 c 是实参，它们在前面的语句中已赋了值，也就是调用 nn()函数时给的值。

被调用函数中说明的参数叫形式参数，简称形参。它的说明要同调用函数的实参一一对应。形参的说明位置是在函数名之后和函数体之前，这是总的原则，但在很多 C 程序中写法不一，这是因为大量的 C 程序是 UNIX 的经典写法，但后来制定了标准 C(ANSI)。因此形式参数的说明形式有两种，现代的高版本编译器对两种说明形式全兼容。也就是哪一种写法都可编译，比如:Microsoft C 4.0 、 5.0 、 6.0 、 7.0 ； Borland C++ 3.0 、 3.1 、 4.0 ； UNIX ： SCO UNIX/386 3.2V2.0 、 3.2V4.2 ； USL UNIX/386 4.0 、 4.2 等。

例如：在第一节中我们给出了 1-1-3-1.c 源文件，其中 max()函数的形参说明如下：

现代标准 C 说明形参：

```
int max(int a,int b) /*在小括号内说明形参*/
{
if(a>b)
return(a);
else
return(b);
}
```

经典 C 说明形参：

```
max(a,b)
int a,b; /*在函数名与大括号间说明形参*/
 {
if(a>b)
return(a);
else
return(b);
}
```

(3)全局变量

在所有函数外部说明的变量称为全局变量。它在整个源程序中有效。在内存 RAM 中，全局变量放在代码段之上。在整个程序执行过程中都占用内存，这样不但内存开销大，同时使程序的通用性差，因为结构化的语言注重数据和代码的独立性，而全局变量的大量使用将破坏结构化，导致程序出错。局部变量和函数的使用将加强结构化，节省内存空间，使代码独立性强。

3. 变量的存储类别

编译器在编译 C 源程序时，根据变量的存储类别存放变量。变量的存储类别说明符有

以下四种：

extern ——外部变量说明符。

static ——没有找静态变量说明符。

register ——寄存器变量说明符。

auto ——自动变量说明符。。

(1) 外部变量—— extern 。一个大型 C 源程序有很多源文件，在一个源文件中定义了全局变量，其他源文件中要引用这个全局变量应在变量说明符前面加上 extern 存储类别说明符。

例如:

```
f2-3.c                    f2-3-1.c
int x,y;                  extern int x;y;
char ch;                  extern char ch;
main()                    fu2 ()
  {                         {
;                          x=y%10;
}                           }
fu1()                     fu3 ()
{                          {
...                       ...
x=15;                      y=20;
}                          }
```

在源文件 f2-3.c 中，说明了两个整型全局变量 x 、 y 和一个字符型全局变量 ch 。在另一个源文件 f2-3-1.c 中要引用全局变量，则应重新说明并冠以 extern 。 extern 将告诉编译器，这个全局变量不再分配存储空间，连接时将正确装配这个变量，使外部变量在整个源程序中有效。

(2)静态变量—— static 。如果 static 加在全局变量前，这个变量就成为全局静态变量。全局静态变量在一个源文件中有效，在这个源程序中，它的值是不变的。而全局变量的作用域是整个源文件（一个源程序有多个源文件）。如果 static 加在局部变量前，这个变量就成为局部静态变量。它的作用域是这个函数，在这个函数中，其值不变。

静态变量的引用使变量的作用域又出现了另一个限定。在管理一个大型的源程序时非常有用，它可以使变量限制在一个源文件、一部分函数内有效。比如在生成库函数的程序中，可建立通用库函数。

(3)寄存器变量—— register 。 register 是寄存器变量说明符，它通知 C 编译器把这个变量的值放在 CPU 寄存器中，这样对寄存器变量的操作就比一般放在内存中的自动变量要快，它减少了访问内存的时间。 C 语言只允许局部变量和形参说明成寄存器变量，不允许全局变量说明成寄存器变量。

寄存器变量的个数同 CPU 有关。如果定义的寄存器变量的个数超过了 CPU 中寄存器的个数， C 编译器会自动将多余的变量转换成非寄存器变量。