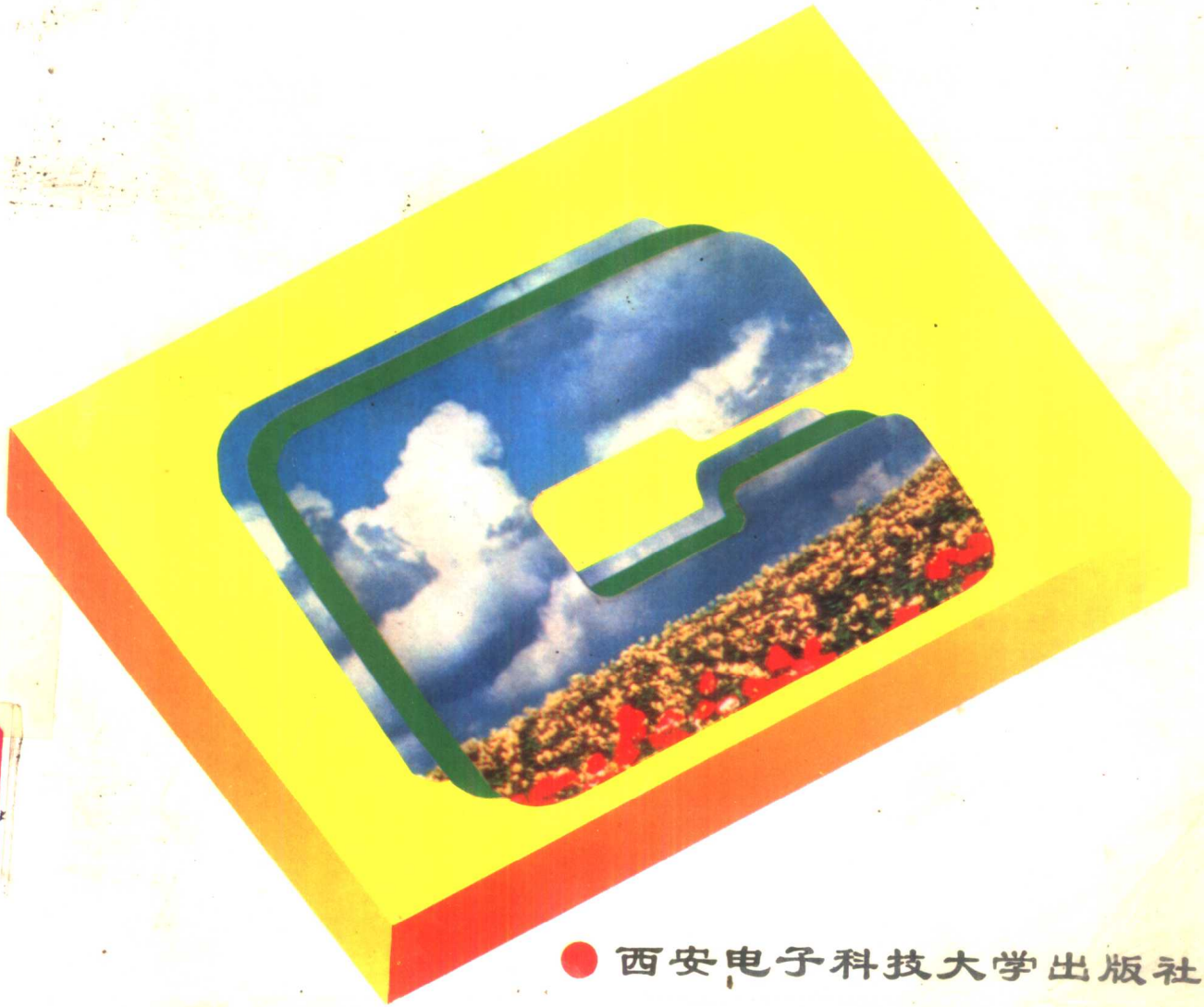
标准C语言

程序设计及应用

龚杰民　金益民　殷　勇　编著



西安电子科技大学出版社

标准C语言程序设计及应用

龚杰民　金益民　殷　勇　编著

西安电子科技大学出版社

1995

(陕)新登字 010 号

内 容 简 介

C 语言是现代最流行的通用程序设计语言之一，并能广泛用于系统软件设计及应用软件开发，包括测控软件的开发。

本书按照 C 语言国际标准与国家标准，全面地、系统地阐述了 C 语言的基本概念、语法规则及用 C 语言进行程序设计的方法。本书概念准确、体系合理、深入浅出，凡学过一门高级语言的读者通过学习本书就可掌握 C 语言的主要内容。本书共有 150 多个程序实例。各章节的实例注重阐述 C 语言的基本用法，最后一章为综合应用实例，实用性很强。每章末附有习题。书后有附录，给出标准 C 语言的语法摘要及库函数的原型声明等有用资料。

本书是在作者 1986 年出版的《C 语言程序设计及其应用》一书的基础上进行扩充与修订，并作了不少更新。由于本书是按 C 语言标准写的，各种编译器均能实现其大多数内容，因而本书应用范围广，不受各种编译版本和上机环境的限制。同时本书对典型的 C 语言版本（如 Turbo C、Unix C、Microsoft C）的上机方法作了简要介绍。

本书可作为高等院校及计算机培训班的教材，也可供广大科技人员自学使用。

标准 C 语言程序设计及应用
龚杰民 [illegible] 勇 编著
责任编辑 王绍菊

西安电子科技大学出版社出版发行
陕西省富平县印刷厂印刷
新华书店经销
开本 787×1092 1/16 印张 17 8/16 字数 414 千字
1994 年 7 月第 1 版 1995 年 9 月第 2 次印刷 印数 8001—16000

ISBN 7-5606-0313-0/TP·0113 定价：15.80 元

前　言

C语言自70年代问世以来，由于它具有功能丰富、简洁、灵活、实用、代码紧凑、目标程序效率高和可移植性好等特点，已成为当今国内外最流行的程序设计语言之一。它是一种支持结构化程序设计的高级语言，通用性强，编程方便；同时，它又具有低级语言的特点，可以用来编写系统软件和面向硬件的应用软件。它的设计者最初用它来编写UNIX操作系统及其支持的应用软件，显然比过去用汇编语言写系统程序容易与方便得多。随着计算机应用的普及，C语言被更多的人所了解与赏识，它的应用范围也更为广泛。

80年代中期，C语言在我国的推广趋势才初露端倪，当时国内的C语言教材寥寥无几。于是我们编著了《C语言程序设计及其应用》一书，由西北电讯工程学院出版社在1986年2月出版。该书受到了广大读者的欢迎与鼓励，已印刷多次。自该书出版至今已八年，在此期间，一方面C语言的推广极为迅速，要求学习C语言的人员大为增加；另一方面，C语言本身也在不断地完善和演化，特别是标准化工作进展很大。1989年国际标准化组织(ISO)确定以美国国家标准C(即ANSI C)为基础，制定国际标准。已于1990年12月正式公布了ISO/IEC 9899《信息处理标准　程序设计语言C》。我国的程序设计语言标准化工作委员会于1992年11月最终完成了C语言国家标准的文本审定，已报国家技术监督局，预期1994年开始实施，而目前国内尚未有按照C语言正式标准来撰写的书。我们感到有必要根据我们在主持起草C语言国家标准过程中的经验和体会，按照标准来修改更新原书。本书的宗旨是按照标准撰写一本进行C语言程序设计的教材。本书较全面、系统地阐述了当前标准C语言的所有成分，不仅包括C语言的数据类型、语句、存储类别等，并较详细地叙述了预处理程序与文件处理。对于标准C语言明显区别于传统C语言的部分极为重视，例如函数原型的概念与风格；结构型变量作为整体进行赋值，作为参数传递与函数返回值；类型限定词等等。在本书附录中还列出了C语言标准的语法摘要及与传统C语言的主要差别，以使读者在遇到传统C程序时不至感到困惑。我国的C语言标准是等同采用国际标准，当然在名词术语上是中国化的。本书一般使用国家标准中的术语，但有少量名词兼顾了人们的习惯，例如“文件”、“读写”、“说明”、“换行”等词分别与标准中使用的“文卷”、“输入输出”、“声明”和“新行”通用。

信息处理环境的国际化已提到议事日程。在标准C语言中已有了一些对支持汉字等需多个字节表示一个字符的文字的基本机制，但还不够充分。由于ISO/IEC 10646编码字符集的标准发布不久，国际标准C语言中扩充对多字节字符处理支撑能力的补篇尚未最终定稿，目前也还没有支持多字节字符处理的C编译程序商品，因此在本书中对标准C语言中的宽（多字节）字符类型和5个相关的处理函数没有作介绍。

本书的对象是学过一门以上程序设计语言的高校学生和广大科技人员。书中由浅入深地阐述C语言的程序结构与语言成分，着力讨论初学者较陌生的内容，如指针类型、聚集类型及预处理程序等。在讨论方法上，充分考虑初学者的思维特点，遵循从具体到抽象的原则。通常从程序设计的角度提出问题，举出引例，然后逐步展开，系统地阐明概念与语法规则，指出难点与易混淆之处，并通过较多的实例来说清概念与示范如何应用。学习程序设计语言与学习自然语言有相似之处，不能只学语法，还须大量实践。因而多读一些程

序实例是十分必要的。为此，本书中不仅在各章都给出中、小型例题，第十章又集中给出了综合应用的实例。这些实例取材于我们自己科研和教学的实践，是工作中很有用的程序，如文件加密与解密程序、设备与资产的折旧计算、通讯录的排序与检索、数值计算、人机界面设计等等。本书实例均在 DOS 环境下用 Turbo C 和 Microsoft C 调试通过，并给出运行结果。由于绝大多数例子都是采用标准 C 语言编写，因而在 Unix、Xenix 环境下也可通过。

本书的§1.1、§9.1 与第七章的大部分由金益民同志编写；§8.3 至§8.5，§9.5 和§9.6 由殷勇同志编写，其余各章节由龚杰民同志编写。

我校的多名研究生和本科生参加了本书若干程序的编写、调试和书稿打印工作，他们是：舒珏、赵旭晟、徐鑫炎、郑文彬和郭健强。肖宏明同志为程序的美化打印做了不少工作，我们在此一并致谢。

由于作者水平和经验的限制，加上时间紧促，书中错误和缺点在所难免，敬请各位专家和广大读者批评指正。

作　者

于西安电子科技大学

目　录

第一章　C语言概述和简单的C程序

§1.1　C语言的历史与特点

1.1.1　C语言的发展史简介

60年代初，随着计算机科学的形成和发展，高级程序设计语言的研究得到了蓬勃的发展。但是，当时得到广泛应用的高级程序设计语言中缺乏用于书写操作系统和编译程序等系统程序的工具，系统程序设计仍然依赖于汇编语言。为改变这种局面，人们开始探索能用于系统程序设计，有足够的表达能力，面向机器体系结构并且高效的高级程序设计语言。作为这种探索工作的结果之一，剑桥大学的 Martin Richards 在 1969 年设计并实现了 BCPL (Basic Combined Programming Language)语言。同年稍后至次年，Bell 实验室的 Ken Thompson 设计了一种类似于 BCPL 的语言，命名为 B，并用它书写了在 PDP—7 机上实现的第一个实验性的 UNIX 操作系统。接着在 1972 年至 1973 年间，Bell 实验室的 Denis Ritchie 在 B 的基础上重新设计了一种语言，首先在 PDP—11 机上实现，用它重写了 UNIX。由于该语言是 BCPL 和 B 的后继，因而取名为 C。BCPL 和 B 是无类型的语言，而 C 语言与它的前驱不同，能支持多种数据类型，因而更能反映大部分计算机系统的体系结构，直接而且清晰地表现客观世界的各种对象；C 语言引进了更为全面的控制结构，为编写具有良好结构的实用程序奠定了基础。参考文献［12］列出了 C、BCPL 和 B 语言的主要异同处。

为了使 C 语言具有高度的可移植性，在最初的 PDP—11 C 编译程序的基础上，Bell 实验室的小组作了进一步的努力。在麻省理工学院的 Alan Sander 根据与 Bell 实验室的合同，于 1974 年实现的可移植的 C 编译程序原型基础上，Bell 实验室的 S. C. Johnson 于 1977 年实现了一个成功的可移植的 C 编译程序，简称为 PCC，其核心思想是把编译程序中与机器有关和与机器无关的部分分离。结果编译程序中只有约 25%与机器有关，在移植时需作改动，其余 75%的代码无须修改。以后实现的大多数编译程序都采用了 PCC。

在短短的几年间，C 语言得到了不断的修改、完善和扩充，并随着 UNIX 一起走出实验室，到 1983 年，C 语言已基本成熟。C 语言和 UNIX 操作系统的开发者一起获得了 1983 年度的 ACM 图灵奖，评审委员会对 C 语言的高度评价，确定了 C 语言的学术地位。现在从大型机到微型机，无论机器使用何种操作系统，都配有 C 编译程序，而由于表达能力、高效等特点和高度的可移植能力，C 语言的应用范围也已扩展到了其开发者最初并未考虑到的领域。

随着 C 语言的普及和应用领域的扩展，每个编译程序厂商都以各自不同的方式扩充 C 语言，方言的增多将严重危害 C 语言应用程序的可移植性。标准化工作开始提到议事日程上。美国国家标准院在 1983 年组成 X3J11 技术委员会，开始起草 C 语言的美国国家标准，并在 1989 年得到美国国家标准院的正式批准，即现在流行的 ANSI C。国际标准化组织 (ISO)从 1985 年开始组建 C 语言的标准化工作组 ISO/IEC JTC1/SC22 WG14，确定以美国

国家标准为基础，制定 C 语言的国际标准。在1990年正式公布了 ISO/IEC 9899《国际标准程序设计语言 C》。我国的 C 语言标准化工作组于1988年成立，按照等同采用国际标准的原则，于1992年11月最终完成了国家标准的文本审定，并于1993年报国家技术监督局，预期1994年开始实施。

1988年由我国代表首先提出的“在程序设计语言中增加对非 ASCII 字符的支撑”的动议在国际标准化组织程序设计语言分技术委员会华盛顿年会上被接受并列入大会决议，开创了程序设计语言国际化的进程。ANSI C 和 ISO/IEC 9899中都提供了对多八位字符(由多个八比特字节表示的字符，如汉字)的基本支持，但支撑能力较弱。为了使 C 语言能更好地支持对全世界各国语言文字的处理，适应新的编码字符集国际标准 ISO 10646，ISO/IEC JTC1/SC22 WG14工作组正在为 C 语言中扩充对多八位字符支撑能力的设施标准进行紧张工作，预期将在1993年底或1994年初发布对这一部分扩充的 C 语言国际标准补篇。

目前的 C 语言标准与由 B. W Kernighan 和 D. M. Ritchie 所著的《The C Programming Language》一书中描述的 C 语言(简称为传统 C)的主要差别在附录二中列出。

1.1.2　C 语言的主要特点

70 年代初设计的 C 语言在很多方面继承和发扬了60年代出现的许多高级程序设计语言的成功经验和特色。如较丰富的数据类型、多种存储类别、自由格式、一定程度的模块化结构、参数的传值方式、结构化的控制、分别编译等。但由于它最初是作为一种系统程序设计的工具语言设计的，设计者将高效率、灵活性等特性放在第一位考虑，因而决定了 C 语言具有许多特殊性。与几乎同时出现的 PASCAL 语言相比，由于它们各自的设计目标不同，差异较大。熟悉 PASCAL 语言的程序员在初次接触 C 语言时往往会感到 C 程序难以调试。这是因为 PASCAL 为保证程序的正确性而对语言规定了许多限制，特别是类型一致性的限制。在程序编译时通过检查并报告程序是否违反了这些限制以帮助程序员检测程序设计中的错误。而 C 语言则为了使语言有较大范围的可用性，尽可能少作限制，因而不是一种强类型的语言。C 语言鼓励使用表达式的副作用，大量使用指针。这种以灵活性和效率为主要设计目标的结果，使得较为难以对 C 程序作静态验证。标准 C 语言已在一定程度上对传统 C 语言作了改进。在程序设计时，应用显式类型转换手段保证类型一致性等程序设计方法论研究的成果可以有效地弥补 C 语言在这方面的不足。另外，C 程序的查错和调试工具目前已十分普遍，使用这些工具，可大大减轻调试 C 程序的负担。

C 语言的主要优点是语言规模小，相对简单，表示法简洁，高度灵活，运行效率高，容易移植。

(1) C 语言基本上不提供直接处理复合对象，如作为整体对待的字符串、集合、表或数组等的操作，C 语言本身也没有提供输入输出手段。这些都由运行时的宿主环境(如 UNIX、DOS、VMS等)以库函数的形式提供。C 语言也只提供单线的控制流结构，而不提供多任务、并发与同步等操作。因此其编译程序就比较简洁。

(2) C 语言的一些操作直接对应于实际机器所执行的动作，是一种较“低级”的语言，在许多方面具有汇编语言的特性。因而能充分反映机器硬件的功能，其代码效率高。统计表明用 C 语言编写的程序编译后仅比用汇编语言直接编写的代码增加15%～20%的开销。所以在编写最讲究效率的软件，如操作系统、编译程序等时，C 语言足以代替汇编语言。最明

显的是 UNIX 系统本身和 MicroSoft Windows，它们从核心到外层几乎全部是用 C 语言写的。

(3) 虽然 C 语言能与计算机的许多能力相适应，但它是独立于具体计算机体系结构的。只要稍加小心便能书写出便于移植的系统程序。只要尽可能遵从标准，严格控制对方言特征的使用，那么用 C 语言编写的应用程序在不同的宿主系统间有高度的可移植性。

§1.2　C 程序的书写形式与程序结构

C 语言的源程序是什么样的呢？让我们看几个简单的 C 程序。

例 1－1　显示字符串的 C 程序。

我们要求终端屏幕上出现如下两行字符串：

```
welcome to the computer world!
* * * * * * * * * * * * * * * * * * * * *
```

用 C 语言来写源程序就是

```
main( )
{   printf("welcome to the computer world !\n");
    printf(" * * * * * * * * * * * * * * * * * * * * *\n");
}
```

这里 main 表示“主函数”。任何 C 程序都由一个主函数和若干(或零)个其它函数组成。主函数的名字必须用 main，其它非标准函数的名字由编程者命名(须符合 C 语言中标识符的语法规则，见下文)。函数名后面紧跟着的圆括号用来放参数。参数传递是函数之间数据联系的一种方式，这与 FORTRAN、PASCAL 等语言有相似之处。例 1－1的 main ( ) 表示主函数为无参函数，但圆括号不能省略，因为它们是函数的标志。

花括号{ }是函数体的界限，也可用作语句括号。它们和 ALGOL、PASCAL 中的 begin、end 的作用类似。函数体内通常有说明部分和语句部分。例 1－1没有任何变量，因而不需要说明部分，仅由两个语句构成。每个语句须以分号结尾。书写方式为自由格式，一行可以写多个语句，一个语句也可分成几行写，但一个标识符不能分成两行。

例 1－1的两个语句都表示按格式输出。printf 是标准 I/O 库提供的“格式化输出函数”的名字，这里调用该函数两次。函数名后面同样必须有括号，其中放参数，也称“参数表”。双引号中的字符序列叫做字符串或字符串常数。其中要特别注意反向斜线\，\n 代表换行字符，应看成只代表一个字符，它使光标移动到下一行的左端，如果例 1－1中的 \n 都去除，则输出就不分行，且运行结束时，光标也不移到下一行左端。

例 1－2　定义一个函数来显示字符串的 C 程序。

```
main( ) /* It is main function */
{   printf("welcome to the computer world!\n");
    stars( );
}
stars( ) /* The function is to print a lot of stars */
{   printf(" * * * * * * * * * * * * * * * * * * * * *\n");
}
```

该程序的运行结果与例 1－1相同。但它们打印一行星号是由函数 stars( ) 来实现的。主函数中出现调用 stars( )，就转到该函数的定义段；stars( ) 的函数体内只有一个语句——调用 printf，当执行完这语句，又返回到主函数，程序结束。C 程序中各函数的书写次序可以是任意的，main( )也可以放在 stars( )函数体结束后，而且各函数可以分布在几个文件中，并可分别编译。但运行时总是从主函数的语句开始。

/* 和 */ 是一对注释的开始符号与结束符号。程序中可以有任意个注释，并可出现在允许空格出现的任何地方，因为一个注释等价于一个空格。注释用于帮助读者阅读和理解程序，在编译时被忽略掉，即并不产生目标代码。注释可用汉字或英文，本书例题中均有示范。

例 1－3　三个整数相加的程序。

```
main( ) /* The sum of three integers */
{   int a,b,c,sum;   /* 局部变量说明 */
    a = 1; b = 2; c = 3;   /* 给变量赋值 */
    sum = a + b + c;   /* 求和,并赋给变量 sum */
    printf("sum is %d\n", sum);   /* 输出 */
}
```

与 PASCAL 等语言类似，C 语言程序中的每个变量必须在使用前进行类型说明或定义（此外还有存储类的说明，见第六章），使编译程序便于对变量分配存储单元。变量的基本类型有四种：

int	代表整数类型；
char	代表字符型；
float	代表单精度浮点型；
double	代表双精度浮点型。

上述表示类型的四个名字都属于关键字，也就是说它们具有专门的意义和用途，用户不能用它们来表示其它变量或函数的名字(即自定义的标识符)，见附录一。

变量的类型说明（也可称“类型声明”）的形式为：

类型名　变量名1，变量名2，……，变量名 n；

这里的 n 个变量具有同一类型，它们也可以等价地表示为多个说明，即

```
类型名　变量名1;
类型名　变量名2;
……
类型名　变量名 n;
```

本例中出现了赋值语句，请注意赋值号(=)与 FORTRAN 相同，而不同于 PASCAL 中的:=。本例中函数调用 printf 具有两个参数，以双引号为界的是第一个参数，称为控制串，逗号后的 sum 是第二个参数。格式字符串中每出现一个 % 符号就表示后面的其它参数须用 % 后指示的数据类型与格式输出(或输入)，称为格式说明。如 %d 表示“十进制整数类型”，%f 表示“浮点整数类型”。因而本例中的变量 sum 以整数类型格式输出。控制串中除了格式说明外，其余的字符(本例中的 sum is)按它原样输出。

本例的运行结果为：

```
sum is 6
```

请注意：printf 的调用方式为：printf(控制串,变元1,变元2,…,变元 n)；

本例中的控制串为：

" sum is	%d	\n"
普通字符	格式说明	换行的转义符

例 1－4 计算圆面积。

```
float process(float r);  /* 函数 process 的原型,即函数声明 */
main ( )                 /* 主函数的定义由此开始 */
{  float radius,area;  /* 主函数中的局部变量定义 */
   printf("Radius = ?");  /* 提示用户输入 */
   scanf("%f",&radius);  /* 输入半径值 */
   area=process(radius);  /* 调用 process 函数 */
   printf("Area = %f",area);  /* 输出计算结果 */
}

float process(float r)  /* 函数 process 的定义由本行开始 */
{   float a;        /* 局部变量定义 */
    a = 3.14159 * r * r;      /* 计算并赋值 */
    return(a);      /* 返回计算结果 */
}
```

运行结果：

```
Radius = ?  键入1.1 【换行】
Area = 3.801324
```

在 scanf 的参数表中第一个参数必定是带引号的控制串。其格式说明与 printf 中相似，但后面的各参数必须是地址参数，详见1.6.3。

本程序由主函数 main 和进行运算处理的函数 process 组成。主函数通过调用库函数 scanf 接受用户输入的半径值 radius，在调用函数 process 时将此值传给对应的形式参数 r(简称"形参")。函数 process 按此值计算 a=3.14159 * r * r；由 return 语句把 a 值返回给主调函数。这返回值是通过函数名 process 带回到主函数中调用处。主函数中赋值语句使该值赋给变量 area，然后由函数 printf 输出结果。程序第一行声明该程序中将有自定义函数 process，并说明其类型以及参数类型。这称为"函数的原型"。关于函数的定义、调用及原型等概念，我们在第四章将详细讨论。现小结如下：

(1) C 程序由一系列函数构成。其中必须有一个主函数，名字为 main。各函数定义的书写顺序是任意的，但不能嵌套定义。每个函数完成某个特定的功能，函数之间通过参数传递以及外部变量进行通讯(外部变量将在第六章讨论)。

(2) 每个函数的定义由一行说明及函数体构成。函数体内通常包括变量定义及语句序列。图1.1为单个程序构成的 C 程序一般形式的示意图。但这仅是一个粗框架，实际程序中还有一些其它成分，以后再叙述。例 1－4中，变量 a 是函数 process 中使用的局部变量，因而需在该函数体内进行说明。(float a 表示 a 为浮点数类型)。若某函数内不用任何变量时也就

无需变量说明，如例1－1和例 1－2那样。甚至，还可以有空函数，例如

```
nothing( )
{       }
```

它不做任何事，但语法上是合法的。

```
外部变量定义或说明
函数的原型
main( )
{ 局部变量定义
  语句序列
}
类型名 function_1(…)
{ 局部变量定义
  语句序列
}
  ……
类型名 function_n(…)
{ 局部变量定义
  语句序列
}
```

图 1.1　C 语言程序的一般形式

(3) C 编译程序的实现者已编写了许多常见的通用函数，供用户在程序中调用。用户不必另行定义。前面例子中已经用到了标准 I/O 库中的格式化输出函数 printf 和输入函数 scanf。§ 1.6 将进一步介绍用法。

(4) C 程序书写格式很自由，这点与 PASCAL 程序相似——一行内可写多个语句，也可以把一个语句分写在多行。为使源程序可读性好，往往把程序中同一层次的花括号按列对齐，而内层的花括号向右缩进，并在必要处添加注释。注释可以插在程序的任意部分。C 语言的关键字都是小写，如 int 是关键字，而 INT 就不是。

(5) 每个语句及变量定义均以分号为结束，即分号是 C 语句的组成部分。而 PASCAL 语言中分号是分隔符，不属于语句。C 语言本身没有输入输出语句，输入输出操作通过调用库函数来实现。因此上例中由函数调用 printf(…)；或 scanf(…)；形成了语句，但 printf、scanf 不属于关键字。

§ 1.3　C 程序的编译和运行

C 语言与其它高级语言程序的开发过程相似，本质上都要经历以下阶段：编辑、编译、连接、运行可执行的目标程序。但在不同的 C 语言开发运行环境下，操作步骤有区别。许多开发环境不仅提供了各自的全屏幕编辑程序，有的还提供了批命令，使编译、连接甚至运行可以连续进行，方便了用户。近几年 Borland 公司、Microsoft 公司等不仅将 C 语言移植到微机 DOS 操作系统上，并提供了完整的开发环境，将编辑、编译、运行、调试、文件管理等都集成在一个环境下。它们给用户提供了全菜单的操作界面，并以屏幕窗口方式工作，使用

极为方便。本书简要地介绍在几种典型运行环境中的操作步骤。如要详细了解某个C编译器及运行环境，请参考专门手册。

1.3.1 在UNIX操作系统上开发与运行C程序的步骤

(1) 用UNIX的编辑程序(如屏幕编辑程序vi或行编辑程序edlin)输入源程序，并存到磁盘上。源文件名宜用.c结尾。如例1-1用wel.c为名。

(2) 调用编译程序cc对源文件进行编译，对例1-1，打命令：

```
cc wel.c
```

如源文件无错，cc不仅会产生相应的目标文件(浮动目标码)wel.o，并且自动完成连接，装配成可执行文件a.out。系统产生的可执行文件名总是a.out。如果你想把可执行文件保留在磁盘上，并不被再次使用命令cc形成的a.out所更新，那就应该在编译时用选择项-o指定可执行文件名：

```
cc -o wel wel.c
```

这样产生的可执行文件名将是wel。

(3) 运行可执行程序，打命令：

```
a.out      (当用户未曾指定可执行文件名时)
```

或 wel (当用户曾指定wel为可执行文件名时)

于是你就可看到运行结果。

当源文件分布在几个文件中时(如例1-2的两个函数分别在文件wem.c和wels.c中)则打命令

```
cc welm.c wels.c
```

会产生浮动目标代码文件welm.o和wels.o，且自动连接装配为可执行文件a.out。如果你要修改其中之一文件welm.c，而wels.c不变，只要用命令

```
cc welm.c wels.o
```

则对命令行中.c结尾的文件重新编译，形成新的welm.o，并与已有的wels.o连接装配成a.out。另外一种操作方案是先单独编译需要修改的文件welm.c，用任选项-c表明仅编译，不连接，如：

```
cc -c welm.c
```

则形成welm.o。

然后用以下命令形成可执行文件welm：

```
cc -o welm welm.o wels.o
```

大型程序通常都分几个文件设计，并充分利用已有的模块。C编译器允许分别编译，使得修改其中部分模块时不必重新编译全部程序，从而节省了时间。

1.3.2 Turbo C开发环境

美国Borland公司开发的Turbo C是在DOS操作系统上的C开发环境。由于使用极为方便，目前广为流行。Turbo C提供用户两种形式的版本：命令行版本TCC以及集成环境版本TC。前者与UNIX操作系统上的C程序使用方法类似，后者提供的菜单界面，其操作与该公司的Turbo系列软件(如Turbo PASCAL等)相近，读者不难掌握它。现作简要介绍。

一、命令行版本 TCC 的使用方法

在 DOS 提示符下打入 tcc 与需编译和连接的文件名即可。对于例 1－1，打命令

tcc wel.c （对 wel.c 编译、连接，产生可执行文件 wel.exe）

然后，在 DOS 下运行该程序——在 DOS 提示符下打可执行文件名wel(不必打扩展名.exe)。在 tcc 后可加上用户需要的选择项，以便进行编译的选择、连接选择或环境选择。例如，对于例 1－2，命令行可为：

tcc －c wels.c（对 wels.c 编译，形成 wels.obj，但不连接）

tcc －e wel welm.c wels.o（对命令中的.c 文件进行编译，并与.o 文件连接，形成可执行文件 wel.exe）

这是对两个文件分别编译。显然也可以一次编译并连接成可执行文件，即有以下一般形式：

tcc －e file f1.c f2.c …… fn.c

-e 选择用来指定可执行文件名，否则将以第一个出现的文件名(去掉后缀，即扩展名)作为可执行文件名。

总之，命令行的一般格式为：

tcc［选择项　选择项　选择项……］文件名　文件名……

如仅打命令 tcc 则显示 tcc 具有的选择项的形式与意义。

由于 Turbo C 的集成开发环境 TC 更为方便，因而很多人在 DOS 下较少用 TCC；然而，在某些情况下，如在 Turbo C 程序中插入汇编代码时，就必须用 TCC，而不是 TC。

二、在集成开发环境 TC 下开发与运行 C 程序

1. 如何进入 TC

假设 Turbo C 已经安装在磁盘上的某个子目录下，当你在该目录下打入

tc

就进入 TC 环境。屏幕上出现了 TC 软件的封面图，即不仅有 TC 主屏信息，且屏的中央有含产品版本号的框。当击任一键后，版本信息消失，主屏仍保持原状(见图1.2)。编辑、编译和运行等各项工作都在此环境下进行。

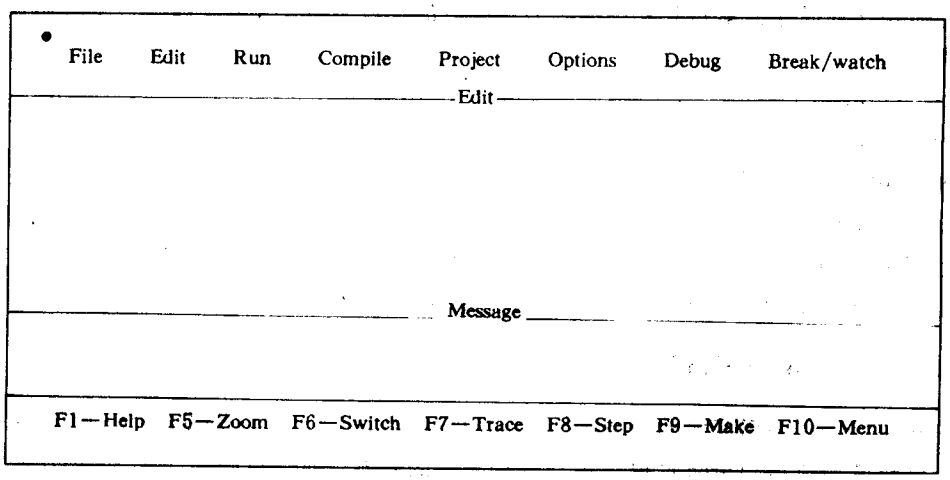


图 1.2　TC 主屏

不难看出，主屏由四个窗口组成：

(1) 主菜单窗口。它在最顶部一行，用→、←键可使亮条沿这一行的8 个菜单项移动。按换行就表示确认(执行)亮条所在项的命令，其中 File 命令为处理文件的各种操作，如装入、存盘、列目录、调用 DOS 及退出程序等(由下拉子菜单列出各项)；Edit 命令用于编辑源程序；Compile 和 Run 分别为编译、连接与控制运行的命令。当主菜单窗口被激活时，也可用命令的首字符来选择，如打 F 则选中 File。

(2) 编辑窗口。从屏幕第三行开始，约占16行。

(3) 信息窗口。屏幕下部含 Message 的框，用来显示编译、连接的结果，如出错信息或结果文件名等。

(4) 功能热键指示窗口。它位于屏幕最底部。在任何状态下打入热键都立即执行其相应功能。例如按 F1就调出当前所需的 Help，按 F10便切换到主菜单。

2. 在 TC 环境下编辑、编译与运行程序的步骤

(1) 文件的建立或加载。选主菜单的 File 命令，按换行键，File 下出现一个窗口，即它的子菜单，如图1.3所示。用↓键移动亮条，移到 Load 或 New 处，按换行键，表示要输入或装载源程序。屏上又出现一个窗口，要求你指定文件名。该窗的标题栏上注明 Load File Name。

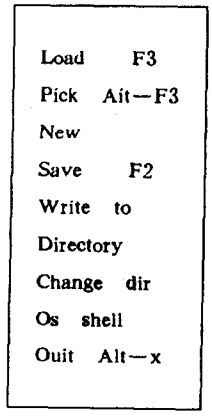


图 1.3　File 的子菜单

对例 1－1，你可输入

wel. c

在按换行键后，File 的各个子菜单立即消失，进入全屏幕编辑状态。编辑窗口右上角自动标示文件名为 wel. c。

(2) 编辑。这时已进入插入状态，可输入或修改源程序。用四个箭头键移动光标，选择编辑的位置。编辑命令与 Word Star 的内部命令相似。也可按 F1，获得联机帮助。你欲结束编辑，按 F10就回到主菜单。

(3) 编译与连接。用 C 键选择主菜单的 Compile 项。出现下拉的子菜单，如图1.4所示。可选择 Compile to OBJ 进行编译。屏上中部会弹出一个标题为 Compiling 的窗口。当源程序有错误时，它告诉你：

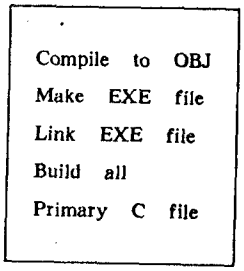


图 1.4　Compile 的子菜单

errors : press any key

当按住一键后，信息窗口中列出各条错误。你据此信息到 Edit 下修改。再次到 Compile 下编译；反复进行，直至编译成功，即编译窗提示

success : press any key

你按任一键，又到了编辑窗。按 F10，仍用 C 键选择 Compile；再用移亮条与回车键选它的子命令 Make EXE file，就可连接装配成可执行文件 f1. exe 。

其实，编译连接的操作方法有多种。对于单个源文件来说，最简单的方法是编辑完毕立即按功能键 F9，即 Make。当无错时，它自动完成编译、连接，形成. exe 文件；当有错时，则报错，你同样到 Edit 下修改，再次按 F9……。

(4) 运行程序。按 F10回到主菜单，用 R 键选择 Run 命令；回车后，出现子菜单。你选