高等学校计算机科学与技术教材

C语言程序设计

高福成 李军 尚丽娜 王瑞文 编著

清华大学出版社
http://www.tup.tsinghua.edu.cn
北方交通大学出版社
http://press.njtu.edu.cn

高等学校计算机科学与技术教材

C语言程序设计

高福成　李军　尚丽娜　王瑞文 编著

清华大学出版社
北方交通大学出版社
·北京·

内 容 简 介

本书以程序设计为主线，以程序设计的需要带动语言知识的学习，系统介绍了C语言及其程序设计技术。全书共10章，包括C程序设计的初步知识、数据类型、数据运算、程序流程控制、数组和字符串、指针、函数、复合数据类型、文件和编译预处理及分割编译，并通过丰富的程序设计实例，详尽介绍了相应的算法知识。各章都编排了大量的练习题，以帮助读者在初步掌握语言的基础上，着重培养程序设计的能力。

图书在版编目（CIP）数据

C语言程序设计／高福成等编著 .—北京：清华大学出版社；北方交通大学出版社，2004.1
（高等学校计算机科学与技术教材）
ISBN 7-81082-214-4

Ⅰ.C… Ⅱ.高… 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（2003）第086307号

责任编辑：谭文芳　　特邀编辑：林　欣
印 刷 者：北京市黄坎印刷厂
出版发行：清华大学出版社　　邮编：100084
　　北方交通大学出版社　　邮编：100044　　电话：010-51686045，62237564
经　　销：各地新华书店
开　　本：787×1092　1/16　　印张：19.25　　字数：496千字
版　　次：2004年1月第1版　　2004年1月第1次印刷
印　　数：1~5 000册　　定价：26.00元

前　言

C语言是计算机程序设计语言中的小字辈，但其卓越的性能使它风靡世界，成为最受欢迎的语言之一。当今流行的面向对象的语言C++及Internet上的通用语言Java就来源于C语言。

在计算机日益普及、计算机技术日新月异、新软件大量涌现的今天，程序设计仍然是当代大学生的基本功，是计算机素质教育的重要环节。尤其对那些希望用计算机解决本专业领域实际问题的有识之士，更需要加强程序设计能力的培养和训练。

程序设计课程包含两个任务：一个是学会一种语言，另一个是具有一定的设计程序能力。相对来说，学习语言比较容易，因为它是一种工具，是一种技能，通过边学边用达到熟能生巧并不困难。而程序设计能力的培养需要花费较大的力气，因为它需要较强的逻辑思维能力，要理解和掌握一批常用的算法，这对初学者来说比较陌生和抽象。算法的严密性需要周密思考，稍有不慎就会出错，如果对算法不能了如指掌，查错也很困难；算法的多样性和多变性则使人眼花缭乱，如果对它望而生畏、望而却步，就不能达到锻炼自己、提高自己的目的。由于程序就是算法的一种描述形式，因此要把学习的重点放在算法上。从读懂已有的程序来理解算法，从自行编制程序来掌握算法。如果能掌握一批基本的程序设计模块，那么，你就具备了一定的程序设计能力。

本书以程序设计为主线，以程序设计的需要带动语言知识的学习。书中内容按两条线索安排：一条是数据类型，从基本数据类型（第2章、第3章、第5章）到指针类型（第6章），再到复合数据类型（第8章）；另一条是程序控制结构，从最简单的顺序结构（第1章）到选择结构、循环结构（第4章），再到模块化结构（第7章），而简单的常用算法则以例题的形式贯穿其中，由易渐难，逐步推进。每章都明确指出本章的知识点、重点和难点，章末附有大量的练习题，其中，选择题和填空题主要用来帮助读者学习和掌握语言知识，编程题则用来指导读者能自行编制程序，达到掌握基本算法的目的。

参与本书编写工作的有：王瑞文（1～3章），高福成（4～6章），李军（7～8章），尚丽娜（9～10章）。

由于作者水平有限，疏漏和错误之处难以避免，恳请使用本书的老师和同学提出宝贵意见。

作　者

2004年1月

目　录

第 1 章　C 程序设计的初步知识 …… (1)
1.1　C 程序的基本结构 …… (1)
1.2　C 程序的书写风格 …… (3)
1.3　简单的 C 程序设计 …… (4)
1.3.1　赋值语句的简单使用 …… (4)
1.3.2　格式输入输出函数的简单使用 …… (4)
1.3.3　库函数和标题文件 …… (5)
1.3.4　简单程序设计举例 …… (6)
1.4　C 程序的开发过程 …… (7)
1.5　Turbo C 2.0 集成环境的使用 …… (8)
习题 …… (12)

第 2 章　基本数据类型 …… (14)
2.1　基本数据类型 …… (14)
2.2　常量及其类型 …… (16)
2.3　变量及其类型 …… (20)
2.4　符号常数 …… (23)
2.5　不同类型数据的输入输出 …… (25)
2.5.1　printf()函数 …… (25)
2.5.2　scanf()函数 …… (28)
2.5.3　单字符输入输出函数(getchar()、putchar()) …… (30)
2.5.4　单字符输入函数(getche()和 getch()) …… (31)
习题 …… (32)

第 3 章　数据运算 …… (37)
3.1　算术运算 …… (37)
3.2　赋值运算 …… (42)
3.3　逗号运算（顺序运算） …… (44)
3.4　关系运算和逻辑运算 …… (45)
3.5　测试数据长度运算符 sizeof …… (48)
3.6　位操作 …… (49)
3.7　数学函数 …… (54)
习题 …… (56)

第 4 章　程序流程控制 …… (60)
4.1　结构化程序设计的概念 …… (60)
4.2　选择结构 …… (61)

4.2.1 用 if-else 语句构成二分支选择结构 …… (61)
4.2.2 用 if-else 语句构成多分支选择结构 …… (66)
4.2.3 用条件表达式实现的选择结构 …… (68)
4.2.4 用 switch 语句构成多分支选择结构 …… (70)
4.2.5 用 switch 和 break 联合构成多分支选择结构 …… (72)
4.3 循环结构 …… (73)
4.3.1 for 循环 …… (73)
4.3.2 while 循环 …… (77)
4.3.3 do-while 循环 …… (79)
4.3.4 循环的嵌套 …… (81)
4.4 转移控制语句 …… (83)
4.4.1 break 语句 …… (83)
4.4.2 continue 语句在循环结构中的作用 …… (84)
4.4.3 goto 语句和标号 …… (86)
习题 …… (87)

第 5 章 数组和字符串 …… (98)
5.1 数组的定义和初始化 …… (98)
5.1.1 数组的定义 …… (98)
5.1.2 数组的初始化 …… (100)
5.1.3 通过初始化定义隐含尺寸数组 …… (101)
5.2 数组的基本操作 …… (103)
5.3 数值数组的应用 …… (107)
5.4 字符串处理函数和字符串的应用 …… (113)
习题 …… (119)

第 6 章 指针 …… (127)
6.1 地址、指针和指针变量的概念 …… (127)
6.2 指针的定义和用指针访问变量 …… (131)
6.3 指针的运算 …… (133)
6.4 用指针访问一维数组 …… (136)
6.5 用指针访问二维数组 …… (137)
6.6 用指针处理字符串 …… (141)
6.7 二级指针 …… (145)
6.8 用指针进行内存动态分配 …… (147)
习题 …… (150)

第 7 章 程序的模块结构和 C 函数 …… (158)
7.1 C 程序的模块结构 …… (158)
7.2 C 函数的定义和调用 …… (159)
7.3 调用函数和被调用函数之间的数据传递 …… (164)

7.3.1 虚实结合方式 …… (164)
7.3.2 函数返回值方式 …… (174)
7.3.3 全局变量方式 …… (175)
7.4 存储类型对函数调用的影响 …… (175)
7.5 函数的递归调用 …… (178)
7.6 main()函数的参数和返回值 …… (180)
习题 …… (182)

第8章 复合数据类型 …… (193)
8.1 结构类型 …… (193)
8.1.1 结构类型的定义和存储模式 …… (193)
8.1.2 访问结构变量和结构数组的成员 …… (196)
8.1.3 结构变量、结构数组和结构指针的初始化和赋值 …… (198)
8.1.4 结构类型的数据在函数间的传递 …… (200)
8.1.5 用递归结构处理链表 …… (205)
8.2 联合类型 …… (210)
8.3 位段结构类型 …… (215)
8.4 枚举类型 …… (218)
8.5 类型定义(typedef) …… (219)
习题 …… (221)

第9章 文件 …… (231)
9.1 文件概述 …… (231)
9.2 文件的打开与关闭 …… (232)
9.3 文件的读写操作 …… (234)
9.3.1 文本文件读写函数 …… (234)
9.3.2 二进制文件读写函数 …… (239)
9.4 文件检测函数 …… (241)
9.5 文件的顺序存取和随机存取 …… (243)
习题 …… (251)

第10章 编译预处理和分割编译 …… (261)
10.1 宏定义 …… (261)
10.2 文件包含 …… (265)
10.3 条件编译 …… (267)
10.4 分割编译 …… (269)
习题 …… (271)

附录 …… (276)
附录A C语言运算符集 …… (276)
附录B ASCII代码表 …… (276)

附录 C　Turbo C 2.0 常用库函数及其标题文件 …… (277)
附录 D　Turbo C 2.0 编译错误信息 …… (280)
附录 E　习题参考答案 …… (281)

参考文献 …… (299)

第 1 章　C 程序设计的初步知识

本章主要介绍 C 程序的基本结构和书写规则，用赋值语句和格式输入/输出函数编写简单的 C 程序，并在 Turbo C 集成环境下实现 C 程序完整的开发过程。

1.1　C 程序的基本结构

1. 一个简单的 C 程序

【例 1-1】给定圆的半径 r，计算圆的周长 c 和面积 a。

```
#define PI 3.14159
main()                  /* 函数名 */
 {                      /* 函数体开始 */
   float r,a,c          /* r:半径; a:面积; c:周长 */
   r=2.5;               /* 给定 r 的值 */
   a=PI*r*r;            /* 计算 a */
   c=2*PI*r             /* 计算 c */
   printf("r=%f,a=%f,c=%f\n",r,a,c); /* 输出 r,a,c */
 }                      /* 函数体结束 */
```

程序的第 1 行用来定义一个符号常数 PI，它代表圆周率；第 2 行是函数名，它的固定名称是 main()；第 3 行是函数体开始的标记，函数体是程序的定义和执行部分；第 4 行定义了 3 个浮点型（即实数）变量，分别代表半径、面积和周长；第 5 行将实数 2.5 赋给变量 r，第 6、7 行通过公式计算面积 a 和周长 c；第 8 行通过 printf()函数显示 r、a、c 的值；第 9 行是函数体结束的标记。程序运行结果如下：

```
r=2.500000, a=19.634956, c=15.707965
```

2. C 程序的基本结构

C 程序的基本结构是函数，一个或多个 C 函数组成一个 C 程序，若干 C 语句构成一个 C 函数，若干基本单词形成一个 C 语句。

（1）C 函数

C 函数是完成某个整体功能的最小单位，它是相对独立的程序段、过程或模块。一个 C 程序由一个主函数和任意多个其他函数组成，所有函数都具有相同的结构。

① 函数名。主函数有固定的名称 main，其他函数则可以根据标识符的命名方法任意取名。主函数通常包括了整个程序的轮廓，由它再调用其他函数。

② 形式参数。在函数名的后面有一对圆括号，其中放置一个或多个形式参数，简称形参、虚参或哑元。一个函数也可以没有形式参数，但圆括号不能省略。

③ 函数体。用花括号包围起来的部分是函数体，即函数的主体。它主要有两大部分：第一部分是本函数内部用到的局部变量类型定义(C 语言中，所有的变量都要先定义，后使用)；

第二部分是语句序列，完成本函数的功能。

图 1-1 描述了 C 程序的一般格式，除了 main()函数外，函数 f1()到 fn()均为用户自行命名的函数。程序执行时，无论各函数的书写位置如何，总是先执行 main()函数，由 main()函数调用其他函数，最后终止于 main()函数(有关函数的详细介绍见第 7 章)。

```
全局变量说明
main()
  { 局部变量说明
    语句序列
  }
f1(形式参数表)
  { 局部变量说明
    语句序列
  }
f2(形式参数表)
  { 局部变量说明
    语句序列
  }
 ……
fn(形式参数表)
  { 局部变量说明
    语句序列
  }
```

图 1-1　C 程序的一般格式

（2）C 语句

C 语句是完成某种程序功能(如赋值、输入、输出等)的最小单位，具有独立的程序功能。所有的 C 语句都以分号结尾。C 语句包括表达式语句、复合语句和空语句。

① 任何 C 表达式末尾加上分号后，就构成一条表达式语句。如“i=0;”，“x=x+1;”等。

② 一组 C 语句用花括号括住，就构成复合语句。如

```
while(i<10)
{  sum=sum+i;
   i++;
}
```

复合语句被视为一个整体，通常用在条件分支或循环语句中。有时为了数据隐藏的目的，用复合语句形成一个代码块，块中定义的局部变量不会对程序的其他部分发生副作用。

③ 只有一个分号的语句称为空语句。例如：

```
for(i=0; i<1000; i++)
;
```

由一条 for 语句(循环语句)和一条空语句组成。空语句用做循环语句的循环体，表示什么也不做。事实上，这个循环的功能是延迟一小段时间。有时，空语句被用做转向点。

④ 注释是以“/*”开始，而以“*/”结束的一串字符。如：

```
/* This is a main function */
```

注释既不被编译也不被执行，使用注释主要是为了增加程序的可读性，因此严格地讲，注释不是 C 语句。

（3）基本单词

基本单词是构成 C 语句的最小单位。C 语言共有 5 种基本单词，即关键字(亦称保留字)、标识符、常数、操作符和分隔符。例如语句“float r,a,c;”中，float 是关键字（代表数据类型

为实数)，r、a 和 c 是标识符(表示变量)；又如语句“a=PI*r*r，c=2*PI*r;”中，“＝”、“*”和“,”是操作符，“2”是常数，PI 是符号常数。

① 关键字是 C 语言中有特定意义和用途、不得作为它用的字符序列，其中 ANSI C 标准规定的关键字有 32 个，Turbo C 扩充的关键字有 11 个，见表 1-1。

表 1-1　Turbo C 关键字一览表

ANSI C 标准关键字					
auto	break	case	char	const	continue
default	do	double	else	enum	extern
float	for	goto	if	int	long
register	return	signed	short	sizeof	static
struct	switch	typedef	union	unsigned	void
volatile	while				
Turbo C 扩充关键字					
asm	interrupt	near	far	_ds	pascal
_ss	cdecl	_cs	huge	_es	

所有的 C 关键字都必须是小写字母。

② 标识符是 C 语言中用来表示变量名、数组名、函数名、指针名、结构名、联合名、枚举常数名、用户定义的数据类型名及语句标号等用途的字符序列，可由 1～32 个字符组成，第一个字符必须是字母或下划线，后面的字符可以是字母、数字或下划线。例如：AB，Ab，aB，ab，A_b，_ab，ab_，s2d，W_length 等都是正确的标识符，而 A+B，A'B，A.B，2abc，α，β，d%等是错误的标识符。

标识符不能与 C 关键字相同，而且区分大小写。例如 AB 和 Ab 是两个不同的标识符，ELSE 可以作为标识符，它不会与 C 关键字 else 混淆。

顺便指出，在 C 编译系统的库函数中，经常使用以下划线“_”打头的函数名或变量名，所以在程序中也应尽量避免使用以“_”打头的标识符，免得与库函数冲突。

③ 常数包括数值常数(如 123、–23.5、1.2E4 等)、字符常数(如' a '、'B'、'c'等)、字符串常数(如"xyz"、"good morning"等)、符号常数及枚举常数(见第 8 章)。

④ 操作符包括各种运算符(如＋、－、*、/)，有特定意义的标点符号(如花括号、方括号、圆括号、逗号)等。

⑤ 分隔符用来分隔相邻的标识符、关键字和常数，最常用的分隔符是空格，此外还可以用制表符、换行符、换页符等作为分隔符。

1.2　C 程序的书写风格

C 程序的书写格式完全自由。

（1）每个函数在整个程序文件中的位置任意。主函数不一定出现在程序的开始处，但不管主函数位于程序中的何处，程序运行时总是从主函数开始。

（2）每个程序行中的语句数量任意。既允许一行内写几条语句，也允许一条语句分几行书写，但每条语句都必须以分号(;)结束。有时还可以在程序的适当地方(如两个函数之间)加进一个或多个空行，使程序结构更加清晰。

（3）注释的位置任意。注释可以出现在程序的任何地方，既可以独占一行或几行，也可以出现在某语句的开头或结尾处。如果注释占有几行，则每一行都要以“/*”开始，以“*/”结束，“*”和“/”之间不能有空格。注释对程序的编译和运行没有影响，但使用注释可以明显增加程序的可读性。

（4）尽管C程序的书写几乎没有限制，但为使程序清晰易读，通常每行写一条语句；不同结构层次的语句从不同的位置开始，即按缩进格式书写成阶梯形状，可以用Tab键或空格键调整各行的起始位置。

1.3 简单的C程序设计

一个程序一般需要具备3个功能，即输入数据、数据运算和输出结果。在C语言中，数据运算主要是由赋值语句完成的，数据的输入、输出则需要调用C编译系统提供的输入/输出函数。本节简要介绍赋值语句和格式输入、输出函数的使用，以便能进行最简单的程序设计。

1.3.1 赋值语句的简单使用

赋值语句用来对表达式进行计算，并把计算结果存储在指定的变量中，其一般格式为：

```
v = e;
```

其中，v是变量名；e是表达式。例如：x=a+b，其功能是先计算表达式a+b的值，并把该值转换成和x相同的数据类型后赋给变量x。

1.3.2 格式输入输出函数的简单使用

输入指的是通过输入设备将原始数据送入计算机，以便计算机对这些数据进行处理；输出指的是将保存在内存中的计算结果送到输出设备上，以便人能阅读。由于C语言不提供输入输出语句，C程序中的输入和输出主要是通过C编译系统提供的输入输出函数实现的。其中用得最多的是格式输出函数printf()和格式输入函数scanf()。本节对它们做一简单介绍，详细介绍见第2章2.5节。

1. 格式输出函数printf()

格式输出函数printf()用来按指定的格式输出数据，是内存与显示器之间进行数据交换的主要手段，也是用户获得程序运行结果的主要途径，其一般形式为：

```
printf("格式控制字符串",输出项目清单)
```

其中，printf是函数名，其后的括号中是该函数的参数：格式控制字符串用双引号括住，用来规定输出格式；输出项目清单中包含0个或多个输出项，它们可以是常数、变量或表达式，当有多个输出项时，相互之间用逗号隔开。例如：

```
printf("%d",125);
```

表示将整常数125按十进制显示在屏幕上，其格式为输出数据右对齐，按输出项的实际位数显示。而：

```
printf("%f,%f",x,y)
```

则表示按小数形式显示浮点型变量x和y的值，其格式为整数部分按实际位数，而小数部分

固定为 6 位小数，不足 6 位小数时，末尾补 0。

2. 格式输入函数 scanf()

格式输入函数 scanf()用来按指定的格式接收输入数据，是内存与键盘之间进行数据交换的主要手段，也是用户为程序运行提供原始数据的主要途径，其一般形式为：

```
scanf("格式控制字符串",输入项目清单)
```

其中，scanf 是函数名，其后的括号中是该函数的参数：格式控制字符串用双引号括住，用来规定输入格式；输出项目清单中包含 1 个或多个输出项，它们必须是变量的地址。变量地址的表示形式是在变量名前面加一个“&”，当有多个输入项时，相互之间用逗号隔开。例如：

```
scanf("%d",&x);
```

要求从键盘输入一个十进制整数，该整数将被存放在变量 x 中。而：

```
scanf("%f, %d",&a,&b);
```

要求从键盘输入一个实数和一个整数，它们被分别存放在变量 a 和 b 中。

1.3.3　库函数和标题文件

上面介绍的 printf()和 scanf()是 Turbo C 编译系统提供的库函数。库函数不是 C 语言本身的组成部分，而是由 C 编译系统提供的一些非常有用的功能函数。例如各种输入输出函数、数学函数、字符串处理函数等，可供用户直接在自己的程序中调用。这些库函数的说明、类型和宏定义都分门别类地保存在相应的标题文件(也称头文件)中，而对应的子程序则存放在运行库(.lib)中，当需要使用系统提供的库函数时，只要在程序开始用：

```
#include  <标题文件>
```

或

```
#include "标题文件"
```

就可以调用其中定义的库函数。printf()和 scanf()是在标题文件 stdio.h 中定义的，因此在调用它们之前，应先用#include <stdio.h>或#include "stdio.h"将它们所在的标题文件包含进来。这里，由#开头的 include 行称为命令行，它是 C 语言的编译预处理命令，不是 C 语句，include 也不是关键字。

Turbo C 提供的常用标题文件如下(标题文件名不区分大小写)：

ALLOC.H	动态分配函数
CONIO.H	屏幕处理函数
CTYPE.H	字符处理函数
GRAPHICS.H	图形函数
MATH.H	数学函数
STDIO.H	标准输入输出及文件操作函数
STDLIB.H	标准实用函数
STRING.H	字符串处理函数

从技术角度讲，任何函数都可以自行编制，完全可以不使用 C 编译系统提供的函数库。但很少这样做，是因为函数库是 C 编译系统提供的一种重要的软件资源，充分利用这一资源，可避免重复自行编制，达到事半功倍的效果。另外，C 语言为了减少对硬件的依赖，不提供任

何执行 I/O 操作的方法，这时，自行编制 I/O 函数也存在很大困难。读者应在学习 C 语言本身的同时，逐步了解和掌握各种库函数的功能和用法，以简化编程。

需要说明的是，不同的 C 编译系统提供的库函数在数量、种类、名称及使用上都有一些差异，甚至连标题文件的名称也不一定相同。

顺便指出，由于各种 C 编译程序都提供 printf 和 scanf 函数，在编译连接时，会自动将用户程序与这两个标准库文件相联。因此，可以不用#include <stdio.h>就能直接调用这两个函数。如果要调用其他库函数，则应包含相应的标题文件，否则编译系统会提示错误。

1.3.4 简单程序设计举例

【例 1-2】为使例 1-1 的程序更具通用性，在每次程序运行中，可由用户从键盘输入半径 r 的值，计算并打印该半径所对应的圆的周长 c、面积 a、球的表面积 s 和球的体积 v，这样，输入不同的半径值，就会得到不同的运行结果。程序如下：

```
#include <stdio.h>
main()
  { float r,c,a,s,v;
    printf("Input value of r:");
    scanf("%f",&r);
    c=2*3.141593*r;
    a=3.141593*r*r;
    s=4*a;
    v=s*r/3;
    printf("r=%f\t",r);
    printf("c=%f\n",c);
    printf("a=%f\t",a);
    printf("s=%f\t",s);
    printf("v=%f\n",v);
  }
```

程序运行到第一个 printf()时，先在屏幕上显示“Input value of r:”，称之为提示信息，目的在于提醒用户从键盘输入数据。当执行到 scanf("%f",&r)时，将暂停下来，等待用户从键盘输入半径 r 的值。一旦用户输入某个实数（如 2.5）并按了回车键后，程序将继续执行后续的语句。当执行到 printf()时，将按照指定的格式输出各变量的值。每次运行时，可以输入不同的半径值，相应输出不同的计算结果。某次程序运行结果如下：

```
Input value of r:2.5↙
r=2.500000      c=15.707965
a=19.634956     s=78.539825     v=65.449852
```

【例 1-3】从键盘输入一个小写字母，打印该字母及其对应的十进制 ASCII 代码值，然后打印该字母对应的大写字母及其对应的十进制 ASCII 代码值。

我们知道，同一个英文字母，其大小写字母的十进制 ASCII 代码值相差 32，于是有了小写字母就容易得到大写字母，反之也一样。程序可编制如下：

```
main()
```

```
{ char ch1,ch2;
  scanf("%c",&ch1);
  printf("%c,%d\n",ch1,ch1);
  ch2=ch1-32;
  printf("%c,%d\n",ch2,ch2);
}
```

程序运行后，要求用户从键盘输入任意一个小写字母，用户输入某个字母并按回车键后，屏幕先显示该小写字母及其对应的十进制 ASCII 代码值，然后显示该大写字母及其对应的十进制 ASCII 代码值。某次运行结果为：

```
d↙
d,100
D,68
```

1.4　C 程序的开发过程

C 语言是一种编译型的程序设计语言，开发一个 C 程序要经过编辑、编译、连接和运行 4 个步骤，才能得到运行结果。

1. 编辑

编辑是指 C 语言源程序的输入和修改，最后以文本文件的形式存放在磁盘上，文件名由用户自己选定，扩展名一般为“.c”，例如：a.c，sample.c，gfc.c 等。

2. 编译

编译是把 C 语言源程序翻译成可重定位的二进制目标程序。编译过程由编译程序完成，编译程序自动对源程序进行句法和语法检查，当发现这类错误时，就将错误的类型和在程序中的位置显示出来，以帮助用户修改源程序中的错误。如果未发现句法和语法错误，就自动形成目标代码并对目标代码进行优化后生成目标文件。目标文件的名称由编译系统自动规定，也可以由用户指定。

3. 连接

连接也称链接或装配，是用连接程序将编译过的目标程序和程序中用到的库函数连接装配在一起，形成可执行的目标程序。可执行目标文件的扩展名由系统自动确定。

4. 运行

运行是将可执行的目标文件投入运行，以获取程序的运行结果。通常，在 DOS 操作系统提示符下直接键入可执行文件名，或在 Windows 操作系统下用鼠标双击可执行文件名即可。

上述 4 个步骤示于图 1-2。图中，带箭头的实线表示操作流程，带箭头的虚线表示操作所需要的条件和产生的结果。例如，在 Turbo 2.0 集成环境中的操作步骤为：在编辑菜单下进行输入和编辑，产生 C 源程序文件(扩展名为.c)；对该文件进行编译，生成可重定位的目标文件(扩展名为.obj)；再将目标文件进行连接装配，生成可执行的目标文件(扩展名为.exe)，也称可

执行文件；运行可执行文件即可获得运行结果。

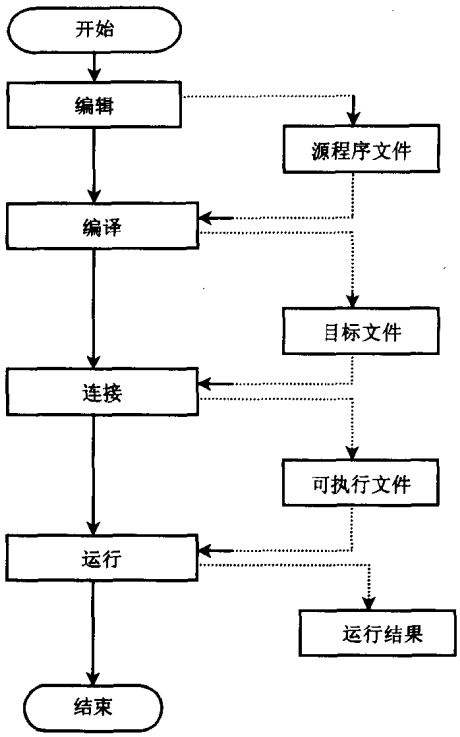

图 1-2　C 程序的开发过程

1.5　Turbo C 2.0 集成环境的使用

Turbo C 2.0 是美国 Borland 公司开发的一个 C 语言集成环境，它集编辑、编译、连接及运行功能于一身，使得 C 程序的编辑、调试和测试非常简捷，编译和连接速度极快，使用也很方便。

1. Turbo C 2.0 的启动

Turbo C 2.0 可以在 DOS 或 Windows 环境下运行。

（1）在 DOS 环境下启动。假设 Turbo C 2.0 安装在 C 盘根目录下的 TC 子目录中，要启动 Turbo C，只要先启动 DOS，如果需要使用汉字，则再启动汉字 DOS(但在大多数汉字 DOS 下，Turbo C 不直接支持使用汉字，在 UCDOS 或天汇 DOS 下，Turbo C 可直接处理汉字，这给用户带来方便)，然后可以按下面两种方式进入 Turbo C 集成环境。

① 直接进入 TC 子目录，启动 Turbo C。即：

```
C>CD C:\TC↙
C:>tc↙
```

② 先进入用户子目录(假设用户子目录为 C:\USER)，然后启动 Turbo C。即：

```
C>CD C:\USER↙
```

```
C:\USER>c:\TC\tc↙
```

使用前一种方式进入 Turbo C，用户的程序文件将存放在 TC 子目录下；使用后一种方式时，用户的程序文件将存放在用户子目录下。

（2）在 Windows 环境下启动。在 Windows 环境下，打开“资源管理器”，找到 Turbo C 所在的文件夹，用鼠标左键双击该文件夹下的 tc.exe 文件。

2. Turbo C 2.0 集成环境窗口

不管在哪种环境下，Turbo C 启动后，屏幕将出现如图 1-3 所示的画面(主屏幕)。

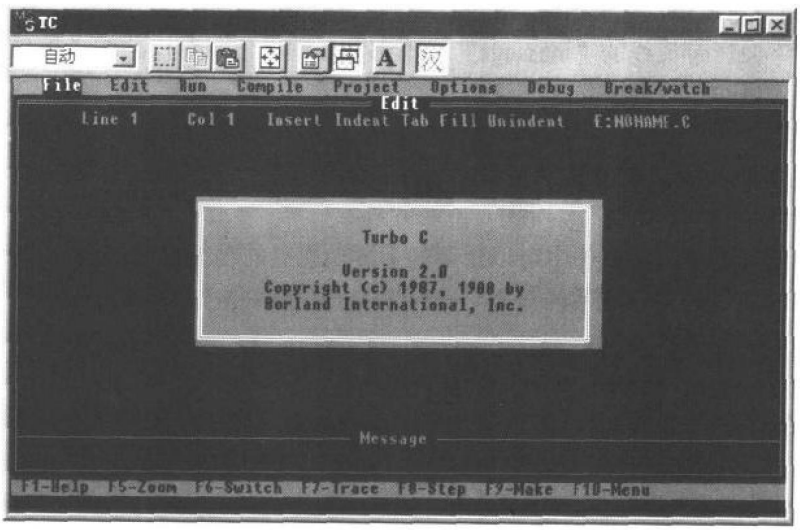

图 1-3　Turbo C 2.0 主屏幕

上述主屏幕自上而下可分为 4 部分：最顶行是主菜单，中间是编辑窗口和编译信息窗口，最底行是功能键提示行。当系统刚启动时，编辑窗口内显示 Turbo C 版本信息，按任意键后，版本信息消失。

（1）主菜单。主菜单提供 Turbo C 环境的功能，共有以下 8 个选择项。

① File——文件管理，用于建立、装入和存储源程序文件，处理目录，切换到 DOS 及退出 Turbo C 环境。

② Edit——Turbo C 编辑器，用于建立和编辑修改源程序。

③ Run——运行，用于编译、连接和运行当前内存中的源程序。

④ Compile——编译器，用于编译当前内存中的源程序。

⑤ Project——项目管理，用于支持大型多程序文件的开发和维护。

⑥ Options——可选项，用于设置 Turbo C 集成环境的各种操作方式。

⑦ Debug——调试，用于设置各种调试选择项，进行调试操作。

⑧ Break/Watch——中断/监视，用于对正在执行的程序指定表达式进行监视，以及设置断点使程序执行时在指定的位置中断。

主菜单的选择方法有两种：

① 用→、↑键将光标亮区移到所需选择项上，然后按回车键；

② 按住 Alt 键，再按所需菜单项的第一个字母，大小写均可。例如，要选择 Edit，可按 Alt+E 键；要进入 File，可按 Alt+F 键。