

GAOZHIGAOZHUAN JISUANJI JIAOCAI

· 高职高专计算机教材

C语言

程序设计案例教程



汪金营 编著



人民邮电出版社
POSTS & TELECOM PRESS

高职高专计算机教材

C 语言程序设计案例教程

汪金营 编著

人民邮电出版社

图书在版编目 (CIP) 数据

C 语言程序设计案例教程/汪金营编著. —北京: 人民邮电出版社, 2004.8
(高职高专计算机教材)

ISBN 7-115-12385-3

I. C... II. 汪... III. C 语言—程序设计—高等学校: 技术学校—教材 IV. TP312
中国版本图书馆 CIP 数据核字 (2004) 第 073401 号

内 容 提 要

本书以 Turbo C 2.0 为学习环境, 涵盖了最新全国计算机等级考试二级考试大纲“C 语言程序设计”的全部内容。本书分为 14 章, 主要内容包括: C 语言程序的构成、数据类型、运算符和表达式、数据的输入与输出、结构化程序设计语句、数组和字符串、指针、函数、变量的存储类型、编译预处理、结构体和共同体、位运算、文件及精选题解, 附录还介绍常见错误分析等。

本书紧扣“语言和程序设计”两个环节, 循序渐进, 由浅入深, 重点难点分析、讲解透彻, 适合作为高职高专 C 语言程序设计课程的教材, 也可作为计算机等级考试的辅导教材。

高职高专计算机教材

C 语言程序设计案例教程

◆ 编 著 汪金营

责任编辑 赵慧君

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号

邮编 100061 电子函件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

读者热线 010-67129259

北京汉魂图文设计有限公司制作

北京隆昌伟业印刷有限公司印刷

新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16

字数: 379 千字

印张: 15.75

2004 年 8 月第 1 版

印数: 1-5 000 册

2004 年 8 月北京第 1 次印刷

ISBN 7-115-12385-3/TP · 4048

定价: 21.00 元

本书如有印装质量问题, 请与本社联系 电话: (010)67129223

编者的话

C 语言是一种优秀的结构化程序设计语言,许多著名的系统软件和应用软件都是用 C 语言编写的,如 UNIX 操作系统等。目前,已经很少有软件工程师单纯用 C 语言编写实用程序,但几乎所有院校的计算机及相关专业都把 Turbo C 2.0 环境下的“C 语言程序设计”作为专业基础课来设置。全国计算机二级考试、全国计算机应用证书考试都规定设置了 Turbo C 2.0 环境下的“C 语言程序设计”考试项目;全国计算机等级考试三级信息技术、三级数据库技术也要求应试者必须掌握 C 语言;有的院校也把“C 语言程序设计”作为计算机及相关专业的专升本必考专业课之一;当今流行的 Java 语言也来源于 C 语言,这些足以说明 C 语言的重要性。

作者教授“C 语言程序设计”课程多年,对这门课程的两个方面:C 语言和程序设计都有很好的把握。学习语言的侧重点就是记忆和理解,本书全部示例的程序行都给出了行号,并逐行进行剖析、讲解,必要时还给出了“行”的图解形式,目的是使读者理解、习惯 C 语言解决问题的“思维”方法,从而达到用 C 语言编写程序来解决实际问题。

根据作者多年的教学经验,本书在内容的编排上遵循从循序渐进的原则,避免过多记忆性的知识和内容“一股脑”摆在读者面前,使读者陷于“茫然”境地。如变量存储类方面的内容安排在第 9 章讨论,之前未曾涉及。本书最后一章集中分析部分精选题及题解,结束本章的学习后,读者就可对用 C 语言编程有一个全面的掌握。

另外,本书未给出各章习题的参考答案,目的是让读者主动地去学习,减少依赖性。各章例题的.c 文件和各章习题的参考答案可进入网址:<http://www.bvca.com.cn> 下载。

由于作者水平有限,书中错误之处在所难免,敬请读者批评指正。

编 者
2004 年 5 月

第 1 章 C 语言概述	1
1.1 C 语言简述	1
1.2 C 语言程序的构成	1
1.2.1 初步认识 C 语言程序	1
1.2.2 C 语言程序的基本结构	2
1.2.3 仅由 main 函数构成的 C 语言程序的一般结构	3
1.2.4 C 语句的构成	4
1.3 C 语言程序的调试与运行	5
1.3.1 开发一个 C 语言程序	5
1.3.2 Turbo C 2.0 的运行环境和启动	6
1.3.3 Turbo C 2.0 集成开发环境的使用	6
1.3.4 编辑、运行 C 语言程序	9
习题	10
第 2 章 C 语言的基本数据类型、常量和变量	13
2.1 C 语言的基本数据类型	13
2.1.1 5 种基本数据类型	13
2.1.2 修饰符	13
2.1.3 存储格式	14
2.2 常量	15
2.2.1 整型常量	15
2.2.2 实型常量	16
2.2.3 字符常量	16
2.2.4 字符串常量	17
2.2.5 符号常量	18
2.3 变量	18
2.3.1 变量的定义	19
2.3.2 变量的初始化	19
习题	19
第 3 章 运算符和表达式	22
3.1 算术运算符和算术表达式	22
3.1.1 算术运算符	22
3.1.2 算术表达式	22

2 目录

3.1.3 算术运算符的优先级和结合性	23
3.1.4 算术表达式的数据类型	23
3.2 赋值运算符、赋值表达式和赋值语句	25
3.2.1 赋值运算符	25
3.2.2 赋值表达式	26
3.2.3 赋值语句	27
3.2.4 赋值表达式的类型	27
3.3 自增、自减运算符	29
3.4 关系运算符和关系表达式	30
3.4.1 关系运算符	30
3.4.2 关系表达式	30
3.5 逻辑运算符和逻辑表达式	31
3.5.1 逻辑运算符	31
3.5.2 逻辑表达式	31
3.6 逗号运算符和逗号表达式	32
3.7 求占字节运算符 (sizeof)	33
习题	34
第 4 章 数据的输入和输出	37
4.1 scanf()函数	37
4.2 printf()函数	39
4.3 scanf()和 printf()应用举例	42
4.4 getchar()函数与 putchar()函数	45
习题	46
第 5 章 结构化程序设计和 C 语言程序控制语句	50
5.1 流程图和结构化程序设计	50
5.1.1 流程图	50
5.1.2 结构化程序设计	51
5.2 选择结构语句	52
5.2.1 if 语句	52
5.2.2 if else 语句	53
5.2.3 if else 语句的嵌套	54
5.2.4 if-else if-else 语句	55
5.2.5 条件运算符及表达式	57
5.3 switch 语句和 break 语句	58
5.3.1 switch 语句	58
5.3.2 break 语句	60
5.4 循环结构	62
5.4.1 for 语句	62
5.4.2 for 语句的其他形式	63

5.4.3	while 语句	64
5.4.4	do-while 语句	65
5.4.5	多重循环结构	66
5.5	其他控制语句	68
5.5.1	break 语句	68
5.5.2	exit()函数	69
5.5.3	continue 语句	69
5.5.4	goto 语句	70
习题		71
第 6 章	数组和字符串	77
6.1	一维数组的定义和元素的引用	77
6.1.1	一维数组的定义	77
6.1.2	一维数组的初始化	78
6.1.3	一维数组的引用	79
6.2	二维数组	80
6.2.1	二维数组的定义	80
6.2.2	二维数组的初始化	81
6.2.3	二维数组元素的引用	81
6.3	字符型数组和字符串	82
6.3.1	字符型数组的定义	82
6.3.2	字符型数组的初始化	83
6.3.3	字符数组的输入和输出	84
6.4	字符串处理函数	86
6.4.1	求字符串长度函数 strlen()	86
6.4.2	字符串拷贝函数 strcpy()	86
6.4.3	字符串比较函数 strcmp()	86
6.4.4	字符串连接函数 strcat()	87
6.5	数组应用举例	88
习题		91
第 7 章	指针	96
7.1	指针的概念和定义	96
7.1.1	指针的概念	96
7.1.2	指针变量的定义	97
7.2	指针和简单变量	97
7.2.1	指针的初始化和赋值	97
7.2.2	空指针和 void 型指针	98
7.2.3	用指针访问变量的值	98
7.3	指针和一维数组	99
7.4	字符型指针和字符型一维数组	102

4 目录

7.4.1	字符指针与字符数组的关系	102
7.4.2	对字符型指针进行字符串的整体赋值	103
7.4.3	用指针处理字符串举例	103
7.5	指针和二维数组	105
7.5.1	对二维数组的再认识	105
7.5.2	把二维数组看作一维数组用指针引用其元素	106
7.5.3	定义行指针引用二维数组	107
7.5.4	指针数组	109
7.6	二级指针	111
7.6.1	二级指针的概念	111
7.6.2	二级指针的定义	111
7.6.3	二级指针应用举例	112
习题		114
第 8 章	函数	120
8.1	函数的构成和定义	121
8.1.1	函数的构成	121
8.1.2	函数的定义	121
8.1.3	对函数定义的说明	122
8.1.4	函数的调用	123
8.1.5	函数参数传递	123
8.1.6	函数的说明	124
8.2	函数参数传递举例	124
8.3	返回值是指针的函数	128
8.4	函数指针与实参是函数名的传递	129
8.4.1	函数指针	129
8.4.2	实参是函数名的传递	131
8.5	函数的递归调用	132
8.5.1	递归调用	132
8.5.2	递归执行过程	133
8.5.3	用递归算法求解问题的条件	134
8.6	main()函数的参数	135
习题		136
第 9 章	变量的存储类型和动态分配内存	142
9.1	变量的存储类型	142
9.2	变量及其作用域和生存期	143
9.2.1	局部变量和全局变量	143
9.2.2	局部变量的作用域和生存期	144
9.2.3	全局变量的作用域和生存期	146
9.3	说明符 extern 和 static	146

9.3.1 全局变量的说明符	146
9.3.2 外部函数、内部函数及说明符	148
9.3.3 说明符 <code>extern</code> 和 <code>static</code> 应用举例	150
9.4 动态分配内存	151
9.4.1 动态分配内存的概念	151
9.4.2 动态分配内存函数	152
习题	154
第 10 章 编译预处理	158
10.1 宏定义和宏替换	158
10.1.1 宏定义	158
10.1.2 带参数的宏定义	159
10.1.3 宏定义的解除	161
10.2 文件包含	161
10.3 条件编译	163
10.3.1 <code>#if</code> 语句	163
10.3.2 <code>#ifdef</code> 语句	164
10.3.3 <code>#ifndef</code> 语句	165
习题	165
第 11 章 结构体和共用体	167
11.1 类型定义 (<code>typedef</code>)	167
11.2 结构体	168
11.2.1 结构体类型的定义	168
11.2.2 结构体变量的定义	170
11.2.3 结构体类型变量的引用	171
11.2.4 结构体数组的定义、初始化和引用	173
11.2.5 结构体指针的定义、初始化和使用	174
11.2.6 函数间结构体的传递	176
11.2.7 结构体类型函数和结构体类型指针函数	179
11.2.8 链表	181
11.3 共用体	184
习题	186
第 12 章 位运算	192
12.1 按位逻辑运算	192
12.2 移位运算	193
12.3 应用举例	195
习题	197
第 13 章 文件	200
13.1 文件概念	200
13.1.1 C 语言的数据文件	200

6 目录

13.1.2 文件的存取方式	200
13.2 文件的打开与关闭	201
13.2.1 文件指针	201
13.2.2 打开文件	201
13.2.3 文件的关闭	202
13.3 文件读写函数	202
13.3.1 字符读写函数 <code>getc()</code> 和 <code>fgetc()</code> 、 <code>putc()</code> 和 <code>fputc()</code>	203
13.3.2 字符串读写函数 <code>fgets()</code> 和 <code>fputs()</code>	204
13.3.3 带格式的读写函数 <code>fscanf()</code> 和 <code>fprintf()</code>	206
13.3.4 数据块读写函数 <code>fread()</code> 和 <code>fwrite()</code>	207
13.4 检测文件末尾函数 <code>feof()</code>	208
13.5 文件的定位与存取	209
13.5.1 文件的定位	209
13.5.2 文件的存取举例	210
习题	211
第 14 章 精选题解	216
附录 1 常见错误分析	233

第 1 章 C 语言概述

1.1 C 语言简述

C 语言的设计始于 1971 年, 后经多次改进, 使其迅速成为最受欢迎的语言之一, 且许多流行一时的著名软件都是用 C 语言编写的。

Turbo C 是美国 Borland 公司的产品。该公司在 1987 年首次推出 Turbo C 1.0 产品, 其中使用了全新的集成开发环境, 即使用了一系列下拉式菜单, 将文本编辑、程序编译、链接以及程序运行一体化, 大大方便了程序的开发。1989 年推出 Turbo C 2.0 版本, 以后又有新产品不断问世。本书仍以 Turbo C 2.0 为开发环境介绍 C 语言程序设计的方法。

1.2 C 语言程序的构成

1.2.1 初步认识 C 语言程序

例 1.1 给出圆的半径 r 是 2.5, 求圆的面积 s 和周长 l 。

程序如下:

```
1. main()
2. {
3.   float r,s,l;
4.   r=2.5;
5.   s=3.14*r*r;
6.   l=2*3.14*r;
7.   printf ("r=%f,s=%f,l=%f\n",r,s,l);
8. }
```

说明: 例 1.1 中, 行 1 定义一个名为 `main` 的主函数, 后面的一对圆括号不能省略。行 2 表示 `main` 函数的开始。行 3 定义 3 个可以是小数的变量 r 、 s 、 l , 分别表示圆的半径、圆的面积和周长。行 4 给半径 r 赋值为 2.5。行 5、行 6 根据 r 的值计算 s 、 l 的值。行 7 将 r 、 s 、 l 的值显示在屏幕上。行 8 表示 `main` 函数的结束。

例 1.2 由键盘输入一个整数 x , 显示 x 的绝对值。

程序如下:

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

2 C语言程序设计案例教程

```
2.  main()
3.  {
4.  int x,abs_1(int);
5.  scanf("%d",&x);
6.  printf("|x|=%d",abs_1(x));
7.  }
8.  int abs_1(x)
9.  {
10. if(x>=0)
11. return x;
12. else
13. return -x;
14. }
```

说明：例 1.2 中，行 2 定义主函数 `main`。行 4 定义整型变量 `x`，并说明程序中还有一个整型函数 `abs_1`。行 5 要求用户从键盘输入一个整数给 `x`。行 6 输出调用 `abs_1` 函数返回的值。行 8 定义一个求 `x` 绝对值的函数，函数名为 `abs_1`。行 9~行 12 判断 `x` 值如果大于等于 0，就返回 `x` 的值；否则就返回 `-x` 的值，以达到求 `x` 绝对值的目的。

上面两例是让读者对 C 语言程序有个初步的了解，下面介绍 C 语言程序的基本结构。

1.2.2 C 语言程序的基本结构

例 1.1、1.2 的程序都可以用文件的形式存在磁盘上，不妨将例 1.1 命名为 `tc1-1.c`，将例 1.2 命名为 `tc1-2.c`。实际上一个 C 语言程序可以包含许多这样的 `.c` 文件和其他文件，但至少得有一个 `.c` 文件，例如 `tc1-1.c` 或 `tc1-2.c`。若由一个以上的 `.c` 文件组成一个 C 语言程序文件，则需使用项目管理功能，这方面的内容仅在第 9 章中作简单介绍，其余各章均以 `.c` 文件为例讨论。

C 语言程序是函数结构。函数是完成某个整体功能或某个模块的最小单位。一个 `.c` 文件由一个以上用户定义的函数和多个库函数组成，每个函数完成一定的功能。如例 1.1 中的程序由用户定义的一个 `main` 函数组成，而例 1.2 中的程序由用户定义的一个 `main` 函数和一个 `abs_1` 函数组成，两个例子中都用到了 C 语言定义的库函数 `printf`。Turbo C2.0 中规定：一个 `.c` 文件可以由一个以上的函数（用户定义的函数）组成，但必须有且仅有一个命名为 `main` 的函数，其余函数名称由用户自定，如图 1.1 所示。

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 语言程序的一般结构

一个 C 语言程序中无论有多少个函数，它总是从 `main` 函数开始执行的。涉及到两个以上函数的程序设计将在第 8 章介绍，之前只介绍由一个函数构成的 C 语言程序设计，即仅由 `main` 函数构成的 C 语言程序。

1.2.3 仅由 `main` 函数构成的 C 语言程序的一般结构

结构如下：

```
#include<标题文件 1>
#include<标题文件 2>
.....
#define 宏定义 1
#define 宏定义 2
.....
main()
{
    语句序列;
}
```

其中，`#include<>` 是 C 语言的命令行，`#include` 是 C 语言语法规定的，一对尖括号也可由一对双引号取代，其不同之处在第 10 章说明。前面讲过，C 语言是由若干个函数组成的，这些函数包括用户定义的函数（如例 1.2 中的 `abs_1`）和库函数（如例 1.1、1.2 中的 `printf`、`scanf` 都是 C 语言的库函数）。当用户在 C 语言程序中使用这些库函数时，必须将包含这些函数的文件用 `#include` 命令包含进去，哪个函数应包含在哪个标题文件中见附录 4，在介绍库函数的用法时也会说明应包含的标题文件，如 `#include<stdio.h>` 或 `#include "string.h"` 等。

`#define` 是宏定义命令，功能是用一个标识符代表一串字符，这也在第 10 章介绍。如宏定义：

```
#define PI 3.1415926
```

函数中用到圆周率时只需书写 `PI` 就可以了，既简洁又可减少书写带来的错误。如例 1.1 可改写为：

```
1. #define PI 3.1415926
2. main()
3. {
4.     float r,s,l;
5.     r=2.5;
6.     s=PI*r*r;
7.     l=2*PI*r;
8.     printf ("r=%f,s=%f,l=%f\n",r, s, l);
9. }
```

说明：行 1 是宏定义，行 5、6 用 `PI` 代表到圆周率 3.1415926，其面积、周长的精确度也高于例 1.1 的计算结果。`main()` 行也称函数的头部。它由函数的名称 `main` 和之后的一对圆括

4 C 语言程序设计案例教程

号组成，程序的执行都是从主函数开始的。紧随圆括号之后的是左大括号，表示 main 函数的开始，最后用右大括号作为 main 函数的结束，左右大括号在 C 语言程序中是成对使用的，缺一不可。位于大括号之间的是一系列的 C 语句，称为函数体。

在 C 语言中，开头的语句一般是对程序中出现的变量进行定义和说明，因为 C 语言遵循对程序中的变量要先定义后引用的规则，如例 1.1 中的 float r, s, l; 例 1.2 中的 int x; 等都是变量说明语句，之后才是实现某些功能的语句。

1.2.4 C 语句的构成

一个 C 语言函数由若干条 C 语言语句组成，C 语言语句是完成某种程序功能（如赋值、输入、输出等）的最小单位。一条语句一般可由下面 6 个基本部分组成。

1. 关键字

关键字在 C 语言中有特定意义和用途，不得作为它用，其中 ANSI C 标准规定的关键字有 32 个，见表 1.1。所有 C 语言关键字都必须小写，这些关键字的用法将在后续内容中介绍。

表 1.1 C 语言关键字

auto	default	float	register	signed	volatile
break	do	for	return	switch	while
case	double	goto	short	typedef	
char	else	if	sizeof	union	
const	enum	int	static	unsigned	
continue	extern	long	struct	void	

2. 标识符

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 1~31 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_, 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。

标识符区分大小写字母，也不能与关键字相同。例如 AB 和 Ab 是两个不同的标识符，ELSE 可以作为标识符，它不会与关键字 else 混淆。

3. 常数

常数包括数值常数（如 123、-23.5、1.2E4）、字符常数（如'a'、'B'、'c'）、字符串常数（如"xyz"、"good morning"）等。

4. 操作符

操作符包括各种运算符（如+、-、*、/）及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

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，此外还有制表符、换行符及换页符等。

6. 分号

所有的C语言语句都以一个分号结尾，一个分号表示一条语句。

例如，例1.1中的行3：

```
float r,s,l;
```

是一条C语言语句。其中，float是关键字；float与r之间的空格，r与s之间的逗号都是分隔符；r、s、l是标识符，即变量的名称；结尾的分号说明这是一条语句。

又如例1.1中的行5：

```
s=3.14*r*r;
```

也是一条语句。其中“=”号、“*”号都是操作符；3.14是常数。

另外，C语言语句有3种形式，即简单语句、复合语句和空语句。

- 简单语句。任何C语言表达式加上分号后，就构成一条C语言语句。如：

```
i=0;
```

```
x=x+1;
```

- 复合语句。若干条C语言语句用花括号括住，就构成复合语句，如：

```
{
    sum=sum+1;
    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。

- 空语句。如：

```
;
```

空语句表示什么也不做。

1.3 C语言程序的调试与运行

1.3.1 开发一个C语言程序

开发一个C语言程序一般包括以下4个步骤。

1. 程序设计

程序设计也称程序编辑。设计人员首先依据需要解决的问题编写程序代码，如例1.1中的行1~行7，然后用任一编辑软件（编辑器）将编写好的程序输入计算机，并以文本文件的形式保存在计算机的磁盘上。编辑的结果是建立源程序文件（.c文件）。

2. 程序编译

编译是指将编辑好的源文件翻译成二进制目标代码的过程。编译过程是使用C语言提供的编译程序（编译器）完成的。不同操作系统下的各种编译器的使用命令不完全相同，使用时应注意计算机环境。编译时，编译器首先要对源程序中的每一个语句检查语法错误，当发现错误时，就在屏幕上显示错误的位置和错误类型的信息。此时，要再次调用编辑器进行查错修改，然后再进行编译，直至排除所有语法和语义错误。正确的源程序文件经过编译后在磁盘上生成目标文件（.obj文件）。

6 C语言程序设计案例教程

3. 链接程序

编译后产生的目标文件不能直接运行。链接就是把目标文件和其他分别进行编译生成的目标程序模块（如果有的话）及系统提供的标准库函数链接在一起，生成可以运行的可执行文件（.exe 文件）的过程。链接过程使用 C 语言提供的链接程序（链接器）完成，生成的可执行文件存在磁盘中。

4. 程序运行

生成可执行文件后，就可以在操作系统控制下运行。若执行程序后达到预期目的，则 C 语言程序的开发工作到此完成。否则，要进一步检查修改源程序，重复编辑——编译——链接——运行的过程，直到取得预期结果为止。

1.3.2 Turbo C 2.0 的运行环境和启动

Turbo C 2.0 系统可在 DOS 3.0 以上版本及 Windows 环境中的 DOS 命令状态下运行，一般安装在 C 盘根目录的某一目录下，如 C:\TC，该目录下还建立了两个子目录 LIB 和 INCLUDE，LIB 子目录中存放.lib 库文件，INCLUDE 子目录中存放所有.h 头文件，其余文件安装在 TC 子目录下。运行 Turbo C 2.0 时，只要在 TC 子目录下键入 TC 并回车即可进入 Turbo C 2.0 集成开发环境，如图 1.2 所示。

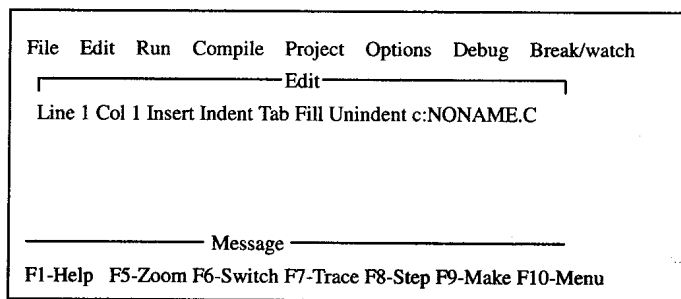


图 1.2

1.3.3 Turbo C 2.0 集成开发环境的使用

在图 1.2 中，第 1 行为 Turbo C 2.0 的主菜单，中间窗口为编辑区，接下来是信息窗口，最底下一行为参考行。这 4 个窗口构成了 Turbo C 2.0 的主屏幕，以后的编辑、编译、调试以及运行都将在这个主屏幕中进行。

1. 主菜单

主菜单中有下列内容：File、Edit、Run、Compile、Project、Options、Debug 及 Break/watch，除 Edit 外，其他各项均有子菜单，只要用 Alt 加上某项中的第一个字母，就可进入该项的子菜单。

(1) File（文件）菜单

按 Alt+F 键可进入 File 菜单，该菜单包括以下内容。

Load（调入）：用来装入一个已经存在的文件，屏幕上默认显示当前目录下的.c 文件。

用户只须使用光标移动键选择所需调入的文件,按回车即可。也可用DOS的通配符(如*或?)来进行列表选择,也可装入其他扩展名的文件,或直接输入要调入的文件名按回车装入一个已经存在的文件。该项的热键为F3,即只要在主菜单中按F3即可进入该项,而不需要先进入File菜单再选择此项。

Pick (选择): 将最近装入编辑窗口的8个文件列成一个表让用户选择,选择后将该程序装入编辑区,并将光标置在上次修改过的地方。其热键为Alt+F3。

New (新文件): 说明文件是新的,编辑一个新的C语言程序可使用New项,缺省文件名为NONAME.C,存盘时可改名。

Save (存盘): 将编辑区中的文件存盘,若文件名是NONAME.C时,将询问是否更改文件名,其热键为F2。

Write to (存盘): 可按用户给出的文件名将编辑区中的文件存盘,若该文件已经存在,则询问要不要覆盖。

Directory (目录): 显示目录及目录中的文件,并由用户选择目录。

Change dir (改变目录): 显示当前目录,用户可以改变显示的目录。

Os shell (暂时退出): 暂时退出Turbo C 2.0到DOS提示符下,此时可以运行DOS命令,若想回到Turbo C 2.0中,只要在DOS状态下键入EXIT即可。

Quit (退出): 退出Turbo C 2.0,返回DOS操作系统,其热键为Alt+X。

说明:以上各项可用光标移动键进行选择,回车则执行。也可用每一项的第一个大写字母直接选择。若要退到主菜单或从它的下一级菜单列表框返回均可用Esc键,Turbo C 2.0所有菜单均采用这种方法进行操作,以下不再说明。

(2) Edit (编辑) 菜单

按Alt+E键可进入编辑菜单,若按回车键,则光标出现在编辑窗口中,此时用户可以进行文本编辑。

(3) Run (运行) 菜单

按Alt+R键可进入Run菜单,该菜单有以下各项。

Run (运行程序): 运行当前编辑区的文件或由Project/Project name项指定的文件名。如果对上次编译后的源代码未做过修改,则直接运行,否则先进行编译、链接,然后才运行,其热键为Ctrl+F9。

Program reset (程序重启): 中止当前的调试,释放分给程序的空间,其热键为Ctrl+F2。

Go to cursor (运行到光标处): 调试程序时使用,选择该项可使程序运行到光标所在行。光标所在行必须为一条可执行语句,否则提示错误,其热键为F4。

Trace into (跟踪进入): 在执行一条调用其他用户定义的子函数时,若用Trace into项,则执行长条将跟踪到该子函数内部去执行,其热键为F7。

Step over (单步执行): 执行当前函数的下一条语句,即使用户函数调用,执行长条也不会跟踪进函数内部,其热键为F8。

User screen (用户屏幕): 在屏幕上显示程序运行的结果,其热键为Alt+F5。

(4) Compile (编译) 菜单

按Alt+C键可进入Compile菜单,该菜单有以下几个内容。

Compile to OBJ (编译生成目标码): 将一个C语言源文件编译生成.obj目标文件,同时