



高职高专“十二五”规划教材——计算机系列



C语言程序设计

C YUYANCHENGXUSHEJI

付倩儒 廖银花 主编



西苑出版社

高职高专“十二五”规划教材——计算机系列

C 语言程序设计

主 编 付倩儒 廖银花

副主编 王用鑫 陈小华

张小红 王 飞

内 容 简 介

我们针对高职高专学生自身的特点与培养目标,从高职高专学生的思维方式、理解能力以及在后续课程中的应用等诸方面的因素出发,针对高职高专学生的认知特点,由具有多年 C 语言程序设计教学经验的一线教师根据实践教学经验和体会编写而成的。

图书在版编目 (C I P) 数据

C 语言程序设计/付倩儒,廖银花主编. — 北京:西苑出版社,2011.7

ISBN 978-7-80210-371-9

I. ①C… II. ①付… ②廖… III. ①c 语言—高等学校:技术学校—教材 IV. ①X110

中国版本图书馆 CIP 数据核字(2011)第 772019 号

C 语言程序设计

主 编	付倩儒 廖银花	
出版发行	西苑出版社	
通讯地址	北京市海淀区阜石路 15 号	邮政编码: 100143
	电 话: 010-88624971	传 真: 010-88637120
网 址	www.xycbs.com	E-mail: xycbs8@126.com
印 刷	北京紫瑞利印刷有限公司	
经 销	全国新华书店	
开 本	787mm×1092mm	1/16
字 数	420 千字	
印 张	18	
版 次	2011 年 7 月第 1 版	
印 次	2011 年 7 月第 1 次印刷	
书 号	ISBN 978-7-80210-371-9	
定 价	35.00 元	

(凡西苑版图书如有缺漏页、残破等质量问题,本社邮购部负责调换)

版权所有 翻印必究

前 言

由于 C 语言的重要性,当前高职院校都把 C 语言程序设计作为入门程序设计课程来对待。《C 语言程序设计》作为一门专业基础课,必须理论与实践并重,作为高职高专院校的基础课,必须充分考虑高等职业院校学生的学习基础、学习习惯与培养目标,在教材中体现职业性特色。我们针对高职高专学生自身的特点与培养目标,从高职高专学生的思维方式、理解能力以及在后续课程中的应用等诸方面的因素出发,针对高职高专学生的认知特点,由具有多年 C 语言程序设计教学经验的一线教师根据实践教学经验和体会编写而成的。

我们的目标是编写一本具有新体系、新结构的通俗易懂、结构新颖的 C 语言实用教材,训练学生掌握计算机程序设计的基本操作技能;使学生具有程序设计的能力,并掌握用计算机解决和处理问题的思维方法,为进一步学习和应用计算机其他专业知识打下良好基础。同时为进行嵌入式编程的开发与应用打下基础。

本教材取材新颖、概念清楚、语言简洁流畅、结构合理、通俗易懂、适用性强,便于教师指导教学和学生自学,可作为应用性、技能型人才培养的各类教育《C 语言程序设计》课程的教学用书,也可供各类培训、计算机从业人员和爱好者参考使用。C 语言是目前世界上应用范围最广,使用最多的高级程序设计语言。国内外各高等职业院校、本科院校中计算机、电子等相关专业均开设了 C 语言程序设计课程。

本书由付倩儒、廖银花担任主编,由王用鑫、陈小华、张小红、王飞担任副主编。全书由付倩儒统稿。

在编写过程中,作者翻阅了大量的资料,参考了许多专家及学者的研究成果,由于时间关系来不及与作者联系,在此向各位表示感谢!

由于编者水平有限,加之时间仓促,书中难免有不足与疏漏之处,敬请各位专家、学者及使用本书的师生批评指正。如在使用本书的过程中有其他意见或建议,恳请向我们(bjlichunfeng@163.com)提出宝贵意见。

编 者

2011 年 7 月

目 录

第 1 章 概述	1	3.1.3 赋值语句	44
1.1 C 语言概述	1	3.2 数据的输入和输出	45
1.1.1 C 语言简介	1	3.2.1 数据输入输出的概念	45
1.1.2 C 语言的特点	1	3.2.2 字符数据的输入输出	45
1.2 简单的 C 程序介绍	2	3.2.3 格式输入输出	47
1.3 C 程序的上机步骤	5	3.3 顺序结构的程序设计举例	52
1.3.1 Turbo C(TC)集成开发环境	6	习题	53
1.3.2 C 语言程序的上机步骤	8	上机实验	56
习题	10		
上机实验	11	第 4 章 选择结构程序设计	58
第 2 章 数据类型、运算符和表达式	13	4.1 if 语句	58
2.1 数据类型	13	4.1.1 if 语句的 3 种形式	58
2.2 C 语言的基本符号	13	4.1.2 if 语句的嵌套	62
2.2.1 基本符号集	13	4.2 switch 语句	65
2.2.2 标识符	14	4.3 选择结构的程序设计举例	71
2.3 常量与变量	15	习题	73
2.3.1 常量	15	上机实验	77
2.3.2 变量	20		
2.4 运算符与表达式	23	第 5 章 循环结构程序设计	78
2.4.1 运算符的优先级和结合性	23	5.1 while 语句和 do ~ while 语句	78
2.4.2 算术运算符及其表达式	24	5.1.1 while 语句	78
2.4.3 关系运算符及其表达式	27	5.1.2 do ~ while 语句	80
2.4.4 逻辑运算符及其表达式	28	5.1.3 while 和 do ~ while 循环的比较	81
2.4.5 赋值运算符及其表达式	29	5.2 for 语句	82
2.4.6 逗号运算符及其表达式	30	5.3 循环的嵌套	84
2.4.7 条件运算符及其表达式	31	5.4 其他控制语句	86
2.4.8 位运算符和位运算	32	5.4.1 break 语句	87
2.4.9 表达式中数据类型的转换	34	5.4.2 continue 语句	87
习题	36	5.4.3 goto 语句	89
上机实验	38	5.5 循环结构的程序设计举例	90
		习题	93
第 3 章 顺序程序设计	40	上机实验	96
3.1 结构化程序设计的基本概念	40		
3.1.1 C 语言的语句	40	第 6 章 数组	97
3.1.2 程序的 3 种基本结构	42	6.1 一维数组	97

6.1.1 一维数组的定义	97	第 8 章 编译预处理	146
6.1.2 一维数组的初始化	98	8.1 宏定义	146
6.1.3 一维数组元素的引用	98	8.2 文件包含	149
6.1.4 一维数组应用举例	99	8.3 条件编译	150
6.2 二维数组	100	习题	152
6.2.1 二维数组的定义	101	上机实验	155
6.2.2 二维数组元素的引用	101		
6.2.3 二维数组的初始化	102	第 9 章 指针	157
6.2.4 二维数组应用举例	102	9.1 指针和指针变量	157
6.3 字符数组	104	9.1.1 指针的基本概念	157
6.3.1 字符数组的定义	104	9.1.2 指针变量	158
6.3.2 字符数组的初始化	105	9.1.3 指针变量的运算	162
6.3.3 字符数组的引用	106	9.2 指针与数组	164
6.3.4 字符串的输入与输出	106	9.2.1 指向数组元素的指针	164
6.3.5 字符数组应用举例	107	9.2.2 通过指针引用数组元素	165
6.4 字符数组应用举例	111	9.2.3 指向一维数组的指针变量	166
习题	113	9.2.4 指向二维数组的指针变量	167
上机实验	115	9.3 字符指针与字符串	171
第 7 章 函数	117	9.3.1 指向字符串常量的指针变量	171
7.1 模块化程序设计概述	117	9.3.2 字符指针的引用	171
7.2 函数的定义	117	9.3.3 指向字符数组的指针变量	172
7.2.1 概述	117	9.3.4 字符串指针作函数参数	173
7.2.2 函数的定义	119	9.4 指针与函数	174
7.2.3 函数的参数和函数的返回值	120	9.4.1 指针变量作函数参数	174
7.3 函数的调用	122	9.4.2 函数的返回值为指针变量	176
7.3.1 函数的调用	122	9.4.3 指向函数的指针变量	177
7.3.2 函数的嵌套调用	125	9.5 指针数组	178
7.3.3 函数的递归调用	126	9.5.1 指针数组的定义与初始化	178
7.4 数组作为函数的参数	129	9.5.2 指针数组元素的引用	180
7.4.1 数组元素作为函数实参	129	9.5.3 指针数组作主函数的形参	180
7.4.2 数组作为函数参数	130	9.6 多级指针	182
7.4.3 用多维数组名作函数参数	133	习题	184
7.5 变量的作用域和存储类型	134	上机实验	186
7.5.1 变量的作用域	134		
7.5.2 局部变量和全局变量	136	第 10 章 结构体与共用体	188
7.5.3 变量的存储类型	139	10.1 结构体类型和结构体变量	188
7.6 内部函数和外部函数	144	10.1.1 结构体型的定义	188
7.6.1 内部函数	144	10.1.2 结构体变量的定义	189
7.6.2 外部函数	144	10.1.3 结构体变量的引用	191
		10.1.4 结构体变量的初始化	192
		10.2 结构体数组	192
		10.2.1 结构体数组的定义	192

10.2.2 结构体数组的初始化	193	11.1.4 流	216
10.3 结构体指针和结构体嵌套	194	11.2 文件的打开与关闭	217
10.3.1 结构体指针的定义与引用	194	11.2.1 文件的打开	217
10.3.2 指向结构体数组的指针	196	11.2.2 文件的关闭	218
10.3.3 用结构体变量和指向结构 体的指针作函数参数	197	11.3 文件的读写	219
10.3.4 结构体嵌套	199	11.3.1 字符读/写函数	219
10.4 共用体	201	11.3.2 字符串读/写函数	222
10.4.1 共用体类型的定义	201	11.3.3 数据块读/写函数	223
10.4.2 共用体变量的定义	202	11.3.4 格式化读/写函数	225
10.4.3 共用体变量的引用	203	11.4 文件的定位与出错检测	226
10.4.4 共用体类型数据的特点	204	11.4.1 文件头定位函数 rewind	227
10.5 枚举类型	204	11.4.2 文件随机定位函数 fseek	227
10.5.1 枚举型的定义	204	11.4.3 文件当前读/写位置函数 ftell	229
10.5.2 枚举型变量的定义	205	11.4.4 检测函数	229
10.5.3 枚举型变量的引用	206	习题	230
10.6 用户自定义类型	208	上机实验	234
10.6.1 基本类型自定义	208	第 12 章 C 语言项目实训	235
10.6.2 数组类型自定义	208	12.1 系统需求分析	235
10.6.3 结构型自定义	209	12.2 系统设计	235
10.6.4 指针型自定义	209	12.3 设计代码	238
习题	210	上机实验	251
上机实验	214	附录 1 标准 ASCII 码字符编码表	253
第 11 章 文件	215	附录 2 Turbo C 2.0 常用库函数	254
11.1 文件概述	215	附录 3 运算符及其结合性	261
11.1.1 文件的分类	215		
11.1.2 文件类型指针	215		
11.1.3 设备文件	216		

第 1 章 概 述

1.1 C 语言概述

1.1.1 C 语言简介

C 语言是一种通用的、结构化的程序设计语言。无论是系统软件还是应用软件都可以很方便地使用 C 语言。

C 语言是由美国贝尔实验室的 D. M. Ritchie 于 1972 年在 B 语言的基础上设计出来的。它既保持了 B 语言精练和接近硬件的特点,又克服了 B 语言过于简单和数据无类型等不足。不过最初的 C 语言只是为描述和实现 UNIX 操作系统而提供的一种工具语言,并主要在贝尔实验室内部使用。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》。C 语言因此具有的良好可移植性,使 UNIX 操作系统迅速在各种机器上运行,并与此相辅相成,也使 C 语言得到了迅速推广和广泛应用。1978 年以后,C 语言已完全独立于 UNIX,先后移植到大、中、小和微型计算机上,成为计算机程序设计语言的主流语种。

随着微型计算机的日益普及,大量的 C 语言版本相继出现。尽管这些版本之间几乎可以高度兼容,但却没有统一的标准。为了改变这种状况,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为 C 语言制定了 ANSI 标准,称为 ANSI C。1987 年,该标准又进行了进一步的改善,公布了新标准——87 ANSI C。目前流行的各种 C 编译系统如 Microsoft C、Quick C 和 Turbo C 都是以 87 ANSI C 为基础的。

1990 年,国际标准化组织(ISO)接受 87 ANSI C 为 ISO C 的标准。考虑到 C 语言版本众多,本书阐述的内容将主要以 87 ANSI C 为基础,试验设计环境主要基于 Turbo C 2.0 集成开发工具。对于在不同微机上使用的 Microsoft C、Turbo C 等不同版本的 C 语言编译系统,虽然有所差异,但基本部分是相同的。

1.1.2 C 语言的特点

C 语言已成为国内外广泛使用的一种计算机语言,它有很多突出的优点。C 语言的主要特点有以下几个方面。

1. C 语言简洁、紧凑

C 语言一共只有 32 个关键字,9 种控制语句,程序书写形式自由,主要用小写字母表示,压缩了一切不必要的成分。

2. C 语言是一种结构化程序设计语言

C 语言的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 和 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. 具有丰富的数据类型

C 语言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)和字符型(char)。在此基础上按层次可产生各种构造类型,如数组、指针、结构体和共用体等。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、链表和树等。

4. 提供了丰富的运算符和运算能力

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。

5. 可以直接对硬件进行操作

C 语言因为能进行位(bit)操作,所以能实现汇编语言的大部分功能,可以直接对硬件进行操作。因此 C 语言既具有高级语言的功能,又具有低级语言的许多功能,可用来写系统软件。

6. 移植性好

用 C 语言编写的程序可移植性好,基本上不做修改就能用于各种型号的计算机和各种操作系统。

C 语言的优点很多,但也有一些不足。如语法限制不太严格、类型检验太弱和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。

1.2 简单的 C 程序介绍

用 C 语言编写的源程序,简称 C 程序。C 程序是一种函数结构,一般由一个或若干个函数组成,其中必有一个名为 main()的主函数,程序的执行就是从这里开始的。

例 1-1 编写一求任意两个整数的和的 C 程序。

```
#include "stdio. h"

main()                                /* 主函数首部 */
{ int integer1, integer2, sum;        /* 定义 3 个整型变量 */
  scanf("%d, %d", &integer1, &integer2); /* 读入两个整数 */
  sum = integer1 + integer2;          /* 求和 */
  printf("Sum is %d\n", sum);        /* 输出所求的和 */
}                                     /* 函数体结束 */
```

下面来分析本程序的结构:

① # include 是编译预处理命令。由双引号括起来的 stdio. h 称做为“头文件”,在“stdio. h”文件中定义了 I/O 库所用到的某些宏和变量,其作用是将由双引号(或尖括号)括起来的文件中的内容,读入到此命令的位置处。在使用 C 语言的输入/输出库函数时,一般需要使用 # include 命令将“stdio. h”包含到源文件中。

②main()为主函数名,花括号{ }部分构成主函数的函数体。

③语句 int integer1, integer2, sum; 是变量的定义部分,说明变量 integer1, integer2, sum 为 3 个整型变量,可以用来存放整型数据。

④scanf 功能是调用系统提供的标准库函数读入变量 integer1 和 integer2 的值。

⑤printf 功能是调用系统提供的标准库函数输出计算结果 sum 的值。

例 1-2 阅读并分析下面的程序。

```
main( )                                /* 主函数 */
{ int a,b,c;                           /* 声明部分,定义变量 */
  scanf("%d,%d",&a,&b);                 /* 输入变量 a 和 b 的值 */
  c=max(a,b);                           /* 调用 max 函数,将得到的值赋给 c */
  printf("max=%d",c);                   /* 输出 c 的值 */
}

int max(int x,int y)                   /* 定义 max 函数,函数值为整型,形式参数 x,y 为整型 */
{ int z;                               /* max 函数中的声明部分,定义本函数中用到的变量 z
                                     为整型 */
  if(x>y)z=x;
  else z=y;
  return(z);                           /* 将 z 的值返回,通过 max 带回调用处 */
}
```

下面来分析一下本程序的程序结构:

①本程序包括两个函数:主函数 main 和被调用的函数 max。max 函数的作用是将 x 和 y 中较大者的值赋给变量 z。return 语句将 z 的值返回给主调函数 main。返回值是通过函数名 max 带回到 main 函数的调用处。main 函数中的 scanf 是“输入函数”的名字(scanf 和 printf 都是 C 系统提供的标准输入输出函数)。程序中 scanf 函数的作用是输入 a 和 b 的值。&a 和 &b 中的“&”的含义是“取地址”,此 scanf 函数的作用是将两个数值分别输入到变量 a 和 b 的地址所标志的单元中,也就是输入给变量 a 和 b。这种形式是与其他语言不同的。它相当于 BASIC 语言中的 INPUT a,b 或 PASCAL 语言中的 Read(a,b)。&a 和 &b 前面的“%d, %d”的含义与前相同,只是现在用于“输入”。它指定输入的两个数据按十进制整数形式输入。

②main 函数中第 4 行为调用 max 函数,在调用时将实际参数 a 和 b 的值分别传送给 max 函数中的形式参数 x 和 y。经过执行 max 函数得到一个返回值(即 max 函数中变量 z 的值),把这个值赋给变量 c,然后输出 c 的值。printf 函数中双引号内的“max=%d”,在输出时,其中“%d”将由 c 的值取代之,“max=”原样输出。程序运行情况如下:

```
8,5✓      (输入 8 和 5 给 a 和 b)
max=8      (输出 c 的值)
```

通过以上两个例子,可以看到 C 程序结构具有以下特点:

1. C 程序是由函数构成的

一个 C 源程序至少包含一个 main 函数,也可以包含一个 main 函数和若干个其他函数。因此,函数是 C 程序的基本单位。被调用的函数可以是系统提供的库函数(例如 printf 和

scanf 函数),也可以是用户根据需要自己编制设计的函数(例如,例 1-2 中的 max 函数)。C 的函数相当于其他语言中的子程序,用函数来实现特定的功能。程序中的全部工作都是由各个函数分别完成的。编写 C 程序就是编写一个个函数。C 的函数库十分丰富,ANSI C 建议的标准库函数中包括 100 多个函数,Turbo C 和 MS C 4.0 提供 300 多个库函数(参见附录 2)。

C 的这种特点使得实现程序的模块化很容易。

2. 一个函数由两部分组成

(1) 函数的首部

函数的首部,即函数的第 1 行。包括函数名、函数类型、函数属性、函数参数(形参)名和参数类型。

例如,例 1-2 中的 max 函数的首部为

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

一个函数名后面必须跟一对圆括弧,函数参数可以没有,如 main()。

(2) 函数体

函数体,即函数首部下面的大括弧{……}内的部分。若一个函数内有多个大括弧,则最外层的一对{ }为函数体的范围。注意:{ }必须配对使用。

函数体一般包含变量定义和执行语句两部分。

3. main() 函数

C 程序必须有 main() 函数,习惯上称其为主函数。C 语言程序运行时,总是从 main() 函数开始,并且在 main() 函数中结束。main() 函数可以放在整个程序的任意位置,通常我们总是把 main() 函数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“\”作续行符。

5. C 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;分号必不可少,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() 等函数来完成的。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,能帮助我们阅读和理解程序。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。注释应括在/*……*/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,但不允许嵌套。

1.3 C 程序的上机步骤

前面介绍了两例用 C 语言编写的程序。所谓程序,就是一组计算机能识别和执行的指令。每一条指令使计算机执行特定的操作。程序可以用高级语言(例如 QBASIC、FORTRAN、PASCAL 和 C 等)编写。用高级语言编写的程序称为“源程序”。从根本上说,计算机只能识别和执行由 0 和 1 组成的二进制的指令,而不能识别和执行用高级语言写的指令。为了使计算机能执行高级语言源程序,必须先用一种称为“编译程序”的软件,把源程序翻译成二进制形式的“目标程序”,再将该目标程序与系统的函数库和其他目标程序连接起来,形成可执行的目标程序。

在编好一个 C 源程序后,该怎样上机运行呢?在纸上写好一个程序后,要经过以下几个步骤:

- ①上机输入与编辑源程序。
- ②对源程序进行编译。
- ③与库函数连接。
- ④运行目标程序。

以上过程如图 1-1 所示。其中实线表示操作流程,虚线表示文件的输入输出。例如,编辑后得到一个源程序文件 f. c,然后在进行编译时再将源程序文件 f. c 输入,经过编译得到目标程序文件 f. obj,再将目标程序文件 f. obj 输入内存,与系统提供的库函数等连接,得到可执行的目标程序文件 f. exe,最后把 f. exe 调入内存并使其运行。

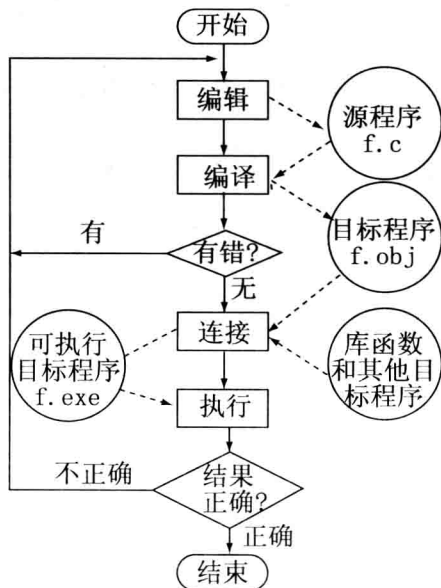


图 1-1 上机运行的步骤

在了解了 C 语言的初步知识后,大家最好上机运行一个 C 程序,以建立对 C 程序的初步认识。

下面就 Turbo C 环境下如何运行一个 C 的源程序作具体介绍。

1.3.1 Turbo C (TC)集成开发环境

Turbo C 由美国 Borland 公司推出,是在微机上广泛使用的编译程序。它集源程序编辑、编译、调试和运行于一体,具有使用简单、方便、运行速度快、效率高和功能强等特点。

在 Windows 操作系统下启动 Turbo C 软件后,屏幕上出现的是一个对话框,如图 1-2 所示。利用 Enter 键+空格键可以将 Turbo C 全屏显示,屏幕上分别是主菜单、状态行、编辑窗口和功能键提示行,如图 1-3 所示。

1. Turbo C 主菜单的介绍

Turbo C 的主菜单共有 8 个菜单项:

(1)File(文件)

主要用于对文件的操作。

(2>Edit(编辑)

建立、编辑源文件。

进入 Turbo C 全屏编辑状态。在程序中按 F10 键或按 Alt 键+主菜单项的首字母可返回主菜单;在主菜单中按 Esc 键或 E 键,即可返回编辑状态。

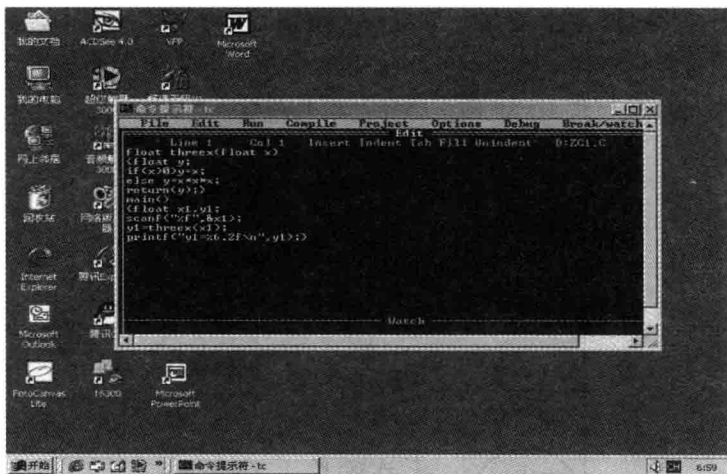


图 1-2 Turbo C 对话框

(3)Run(运行)

控制运行程序。

(4)Compile(编译)

编译、生成目标及可执行文件。

(5)Project(项目)

可选择编辑器任选项及定义宏,也可以记录 Include 及 Library 文件目录,保存编译任选项和从配置文件加载任选。

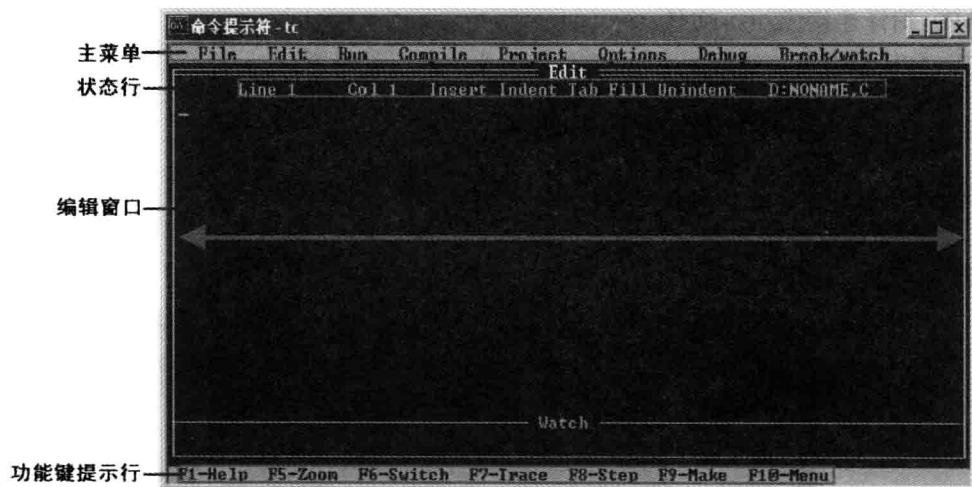


图 1-3 Turbo C 窗口

(6) Options(选择项)

该菜单含有控制集成工作环境的设置。

(7) Debug(调试)

检查、改变变量的值,查找函数,程序运行时查看调用栈。

(8) Break/Watch(控制断点和监视表达式)

增加、删除、编辑监视表达式及设置、清除、执行至断点。

2. 编辑窗口的介绍

为了进入编辑窗口,用 F10 键激活主菜单,然后将光标移到 Edit 菜单,再回车;或在主菜单里的任意位置直接按 E 键即可。

(1) 状态行

编辑窗口的顶端为状态行,其中:

- ① Line n: 光标在文件的第 n 行。
- ② Col n: 光标在文件的第 n 列。
- ③ Insert: 表示编辑状态处于“插入”状态。用键盘上的 Insert 键可以实现“插入”与“改写”状态的转换。

④ C:\NONAME.C: 显示当前正在编辑的源程序名。对于源程序没作保存的都显示为“NONAME.C”,若用 F2 键或用 File/Save 命令则可以将源程序保存命名。

(2) 功能键提示行

Turbo C 屏幕底端所显示的为功能键提示行,其中:

- ① F1: 打开一个帮助窗口,给出有关 Turbo C 编辑命令的信息。
- ② F5: 扩大编辑窗口到整个屏幕。再按一次 F5 键又回到原状态。
- ③ F6: 从一个活动窗口切换到另一活动窗口。
- ④ F7: 在调试模式下一次执行程序的一行,跟踪进函数内部。
- ⑤ F8: 在调试模式下一次执行程序的一行,不跟踪进函数内部。

⑥F9:编译连接成. exe 文件。

⑦F10:从编辑窗口激活主菜单,或从任何菜单激活编辑窗口。

1.3.2 C 语言程序的上机步骤

其具体步骤如下:

1. 建立 C 源程序

激活主菜单,选择并执行 File/New 命令,如图 1-4 所示。

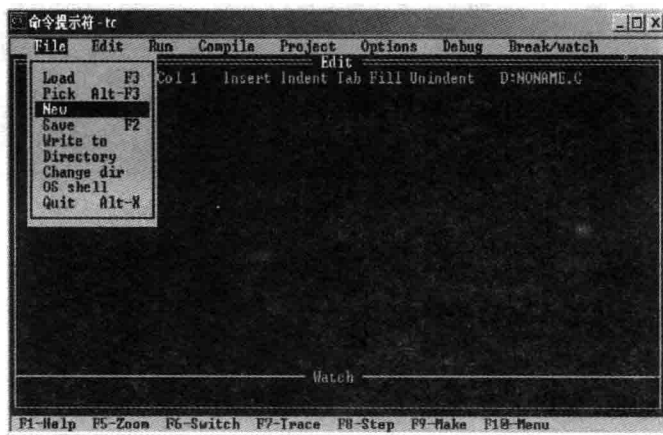


图 1-4 建立 C 源程序

2. 保存 C 源程序

激活主菜单,选择并执行 File/Save 命令或直接键入 F2 键,如图 1-5 所示。将“NONAME.C”修改为用户所建的源程序文件名。



图 1-5 保存 C 源程序

3. 编译、连接源程序

激活主菜单,选择并执行 Compile/Make EXE File 命令或直接键入 F9 键,则 Turbo C 将自动完成对当前正在编辑的源程序进行编译、连接并生成可执行文件(. EXE)。

如果源程序有语法错误,此时按任意键激活屏幕下端的消息窗口(Message),显示出错信

息,亮块显示在第1条消息上。这时编辑窗口也有一条亮块,表明错误在此位置或就在邻近位置。按回车键后,即可修改相应的错误。但有时即使源程序没有一处语法错误,可编译、连接却不能通过,而是出现“Unable to open the file COS.OBJ”的错误提示。这主要是因为包含文件目录“Include directories”和库文件目录“Library directories”的设置有问题引起的。其解决办法是按实际安装 Turbo C 的路径,重新设置。具体操作如下:

①在 Turbo C 环境下按 F10 键,激活主菜单,选取 Options/Directories 命令,按回车键,打开相应的窗口,如图 1-6 所示。

②将黑块移到“Include directories:”选项上,按回车键,系统弹出包含当前文件路径的编辑窗口,重新输入包含文件的实际所在的盘符和路径,然后按回车键即可。

③将黑块移到“Library directories:”选项上,按回车键,系统弹出当前库文件所在目录的编辑窗口。用修改“Include directories:”项相同的方法,将其改为正确的盘符和路径。

④将黑块移到“Output directories:”选项上,按回车键,将其改正。

⑤再移动黑块到“Turbo C directories:”选项上,将 Turbo C 文件所在的目录和路径修改正确。

⑥按 Esc 键返回 Options 菜单,再执行 Save options 命令,将所设置的内容保存。

返回子菜单,再按 Esc 键,则返回原编辑窗口。

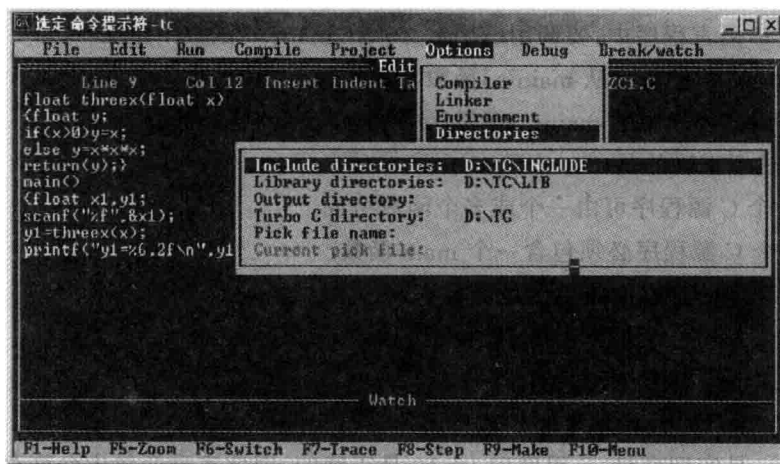


图 1-6 “Options/Directories”命令

4. 执行程序

利用快捷键 Ctrl+F9 或选择 Run/Run 命令,按回车键,即可运行程序。若程序没有输入函数语句,则立即返回编辑窗口;若程序有输入函数语句,则输入相关数据按回车键返回。

5. 查看结果

利用快捷键 Alt+F5 或选择 Run/User screen 命令,即可显示程序的结果。查看结果后,可按任意键返回编辑窗口。

若发现有逻辑错误,则需要重新编辑修改;然后重新按上述步骤执行,直至运行结果无误。

以上步骤是运行一个完整 C 语言源程序的过程,当需要重新编辑下一个源程序时,只需选择 File/New 命令即可。

习 题

一、选择题

1. C 语言程序的基本单位是()。
A. 程序行
B. 语句
C. 函数
D. 字符
2. 一个 C 语言程序是由()。
A. 一个主程序和若干子程序组成
B. 函数组成
C. 若干过程组成
D. 若干子程序组成
3. C 语言的程序一行写不下时,可以()。
A. 用逗号换行
B. 用分号换行
C. 在任意一空格处换行
D. 用回车符换行
4. 以下说法中正确的是()。
A. C 语言程序总是从第 1 个定义的函数开始执行
B. 在 C 语言程序中,要调用的函数必须在 main()函数中定义
C. C 语言程序总是从 main()函数开始执行
D. C 语言程序中的 main()函数必须放在程序的开始部分
5. 以下叙述不正确的是()。
A. 一个 C 源程序可由一个或多个函数组成
B. 一个 C 源程序必须包含一个 main 函数
C. C 程序的基本组成单位是函数
D. 在 C 程序中,注释说明只能位于一条语句的后面
6. 以下叙述不正确的是()。
A. 在 C 程序中,语句之间必须要用分号“;”分隔
B. 若 a 是实型变量,C 程序中 a=10 是正确的;因此实型变量中允许存放整型数
C. 在 C 程序中,无论是整数还是实数,都能正确无误地表示
D. 在 C 程序中,%是只能用于整数运算的运算符
7. 一个 C 程序的执行是从()。
A. 本程序的 main 函数开始,到 main 函数结束
B. 本程序文件的第 1 个函数开始,到本程序文件的最后一个函数结束
C. 本程序的 main 函数开始,到本程序文件的最后一个函数结束
D. 本程序文件的第 1 个函数开始,到本程序 main 函数结束
8. 分析以下程序的结构,它定义了()变量。
#include <stdio.h>
main()