

福建省高校计算机系列教材

C 语言程序设计 与应用教程

主 编 严桂兰

编写者 (以姓氏笔画为序)

黄思彪

彭 洪

厦门大学出版社

图书在版编目(CIP)数据

C 语言程序设计与应用教程/严桂兰主编. —厦门:厦门大学出版社,2001.8
ISBN 7-5615-1779-3

I. C… II. 严… III. C 语言-程序设计-水平考试-教材 IV. TP312

中国版本图书馆 CIP 数据核字(2001)第 034265 号

厦门大学出版社出版发行
(地址:厦门大学 邮编:361005)

<http://www.xmupress.com>

xmup@public.xm.fj.cn

福建沙县方圆印刷有限公司印刷

2001 年 8 月第 1 版 2003 年 3 月第 3 次印刷

开本:787×1092 1/16 印张:27.5

字数:700 千字 印数:6 001—10 000 册

定价:38.00 元

本书如有印装质量问题请直接寄承印厂调换

内容简介

福建省一批资深的教师,他们长期从事着 C 语言教学与科研,又多年参加福建省 C 语言二级等级考试命题,对 C 语言的内涵、规律有着独到的见解,他们根据自己的经验,以研讨学术的态度编写了本书。

在书的内容组织上,除按常规讲授 C 语言的基本、一般的内容外,还依逻辑思维方式将内容归类,如在数组、指针、函数的基本内容之后,开设一章来描述三者的简单应用;为了拓宽知识面,本书讲授了图形、调用中断方面的基本应用;为了上机需要,本书在有关章节安排了上机步骤、程序调试与出错信息;最后,还对 VC++ 作了简单介绍,它将 C 与 VC++ 连贯一气,顺理成章。本书在同类书中具有全面、应用性强、概念清晰等诸多特点。

本书可作为大专院校计算机与其他各类专业的教材,同时,也可供各行各业从事计算机工作的人员使用。

前 言

我们受福建省计算机等级考试指导委员会的委托,编写一本结合我省具体情况的C语言教材。虽然我们长期从事着C语言的教学与科研工作,又多年参加福建省C语言二级等级考试的命题,对C语言的内涵、规律的掌握有着一定优势,但要做到“统观全局,展望未来”、“深入浅出,突出实用”却不是那么轻松的。我们根据自己的经验,以研讨学术的态度,认真吸取各家的长处,利用集体的智慧,尽量以最好的质量来回报福建省各高校师生及广大读者。

首先,我们全面展开C语言的各知识点,但侧重不一。核心部分仍为“数组、函数、指针”,它们既是本书的重点,也是本书的难点。我们本着复杂问题简单化、简单问题实用化的思路,在除讲授数组、指针、函数的基本内容之外,还专门开设第七章来描述三者的简单应用,以让读者进一步掌握。

其次,我们以层层深化的写法,将各章节的本质与内涵呈现在读者的面前。因此,本书的另一特点就是层次分明,描述细化。

C语言是一门实践性很强的、且十分灵活的高级程序设计语言,其上机环节十分重要,为此,我们对上机步骤、程序调试等方面的内容作了具体章节安排,以突出其作用。

为了拓宽知识面,本书讲授了图形、调用中断并对C++进行了简单的介绍,这些内容并不是教纲所要求的,只作为课外参考内容(以*标示)。在时间安排上,我们建议课堂讲授、上机与自学并进的方式。虽然本书篇幅稍多,但很多部分是留给读者自学的,如附录、部分例子、以*标示的章节等。建议课堂讲授时间至少为54学时,上机为0.8~1倍的课堂讲授时间,自学为1~1.5倍的课堂讲授时间。

本书由华侨大学严桂兰教授主编,其中第1、2、4、5、11章由福建农林大学黄思先老师编写,第3、6、8、9章由厦门大学彭洪老师编写,第7、10章由华侨大学严桂兰教授编写,最后,由福州大学刘传才博士审阅。编写期间,多次得到福建省教育厅高教处、福建省计算机等级考试指导委员会、厦门大学出版社的大力支持与帮助;福建农林大学吴锤红教授、华侨大学陈启泉教授等提出了中肯而宝贵的意见。在此,一一致以衷心的感谢。

由于时间仓促,书中不妥之处在所难免,敬请多多批评指正。

编 者

2001年5月

目 录

前言

第一章 C 语言特点与上机操作	(1)
1.1 C 语言特点	(1)
1.1.1 C 语言的产生与发展	(1)
1.1.2 计算机语言与程序设计	(1)
1.1.3 C 语言的特点	(2)
1.2 C 语言程序基本组成	(2)
1.3 Turbo C 2.0 上机步骤	(5)
1.3.1 编辑、编译、连接、执行及调试程序的概念	(5)
1.3.2 Turbo C 的上机步骤	(6)
习题	(7)
第二章 C 语言的语法基础	(8)
2.1 基本数据类型	(8)
2.1.1 标识符与基本数据类型	(8)
2.1.2 常量与变量	(10)
2.1.3 内存的概念	(15)
2.2 基本输入、输出函数	(16)
2.2.1 格式输入函数和格式输出函数	(16)
2.2.2 非格式化输入、输出函数	(26)
2.3 运算符与表达式	(28)
2.3.1 算术运算	(29)
2.3.2 增 1 与减 1 运算	(30)
2.3.3 关系、逻辑及条件运算	(31)
2.3.4 位运算	(33)
2.3.5 赋值运算	(35)
2.3.6 类型转换	(39)
2.3.7 逗号运算	(39)
2.3.8 长度运算符	(39)
2.4 小结	(40)
习题	(40)

第三章 程序控制结构	(45)
3.1 C 语言的语句	(45)
3.2 顺序结构	(46)
3.3 分支结构	(47)
3.3.1 if 结构	(47)
3.3.2 switch 结构	(51)
3.4 循环结构	(54)
3.4.1 当型循环(前判定循环)	(55)
3.4.2 直到型循环(后判定循环)	(58)
3.4.3 break 语句与 continue 语句	(61)
3.5 goto 语句与标号	(66)
习题	(67)
第四章 构造型数据类型	(71)
4.1 数组	(71)
4.1.1 一维数组	(71)
4.1.2 字符数组	(78)
4.1.3 二维数组	(81)
4.2 结构体	(86)
4.2.1 结构体的概念	(86)
4.2.2 结构体类型及结构体变量	(87)
4.2.3 结构体变量的使用	(90)
4.2.4 结构体变量、结构体数组的初始化	(94)
4.2.5 位段	(95)
4.3 共用体	(96)
4.3.1 共用体的概念、类型说明和变量定义	(96)
4.3.2 共用体变量的使用	(99)
4.4 枚举型	(101)
4.5 typedef 的用途	(102)
4.6 小结	(103)
习题	(104)
第五章 指针	(112)
5.1 指针与指针变量	(112)
5.1.1 指针的基本概念	(112)
5.1.2 指针变量的定义	(113)
5.1.3 指针变量的赋值	(114)
5.2 指针运算符	(116)
5.2.1 指针运算符	(116)

5.2.2 无类型指针·····	(118)
5.3 指针与数组·····	(118)
5.3.1 指针与数组·····	(118)
5.3.2 移动指针及两指针相减运算·····	(120)
5.3.3 指针比较·····	(121)
5.3.4 字符串·····	(123)
5.3.5 指针与二维数组·····	(127)
5.4 指向指针的指针·····	(136)
5.4.1 指向指针的指针·····	(136)
5.4.2 定义指向指针的指针变量·····	(137)
5.4.3 指向指针的指针变量的应用·····	(137)
5.5 指针与结构·····	(138)
5.5.1 指向结构体变量的指针变量·····	(138)
5.5.2 指向结构体数组的指针变量·····	(139)
5.5.3 通过指针变量存取位段数据·····	(141)
5.6 指向共用体和枚举型的指针·····	(142)
5.6.1 指向共用体变量的指针变量·····	(142)
5.6.2 指向枚举型的指针变量·····	(143)
5.7 指针小结·····	(143)
5.7.1 指针概念综述·····	(144)
5.7.2 指针运算小结·····	(144)
5.7.3 等价表达式·····	(146)
习题·····	(146)
第六章 函数 ·····	(154)
6.1 常见的系统库函数·····	(154)
6.1.1 字符与字符串函数·····	(155)
6.1.2 简单数学函数·····	(158)
6.1.3 类型转换函数·····	(160)
6.2 用户自定义函数·····	(161)
6.2.1 函数定义、调用和说明·····	(161)
6.2.2 函数返回值·····	(164)
6.2.3 函数参数·····	(165)
6.3 函数的嵌套调用及递归调用·····	(166)
6.3.1 函数的嵌套调用·····	(166)
6.3.2 函数的递归调用·····	(167)
6.4 局部变量与全局变量·····	(174)
6.5 变量的存储类型与变量的初始化·····	(176)
6.6 外部函数与内部函数·····	(180)
6.7 编译预处理·····	(185)

6.7.1 文件包含	(185)
6.7.2 宏定义	(185)
6.7.3 条件编译	(187)
习题	(188)
第七章 数组、指针、函数的应用	(191)
7.1 概述	(191)
7.2 函数之间的数据传递	(193)
7.2.1 函数数据按数值传递	(193)
7.2.2 函数数据按地址传递	(194)
7.2.3 利用函数返回值和外部变量进行函数数据传递	(203)
7.2.4 结构作为函数参数传递	(204)
7.3 函数指针与指针函数	(207)
7.3.1 函数指针	(207)
7.3.2 指针函数	(209)
7.4 数组指针、指针数组与带参的 main 函数	(211)
7.4.1 数组指针	(211)
7.4.2 指针数组	(211)
7.4.3 带参的 main 函数	(212)
7.5 单向链表	(214)
7.5.1 单向链表的概念	(214)
7.5.2 链表的建立	(214)
7.5.3 链表结点的删除	(217)
7.5.4 链表结点的插入	(218)
7.6 小结	(220)
习题	(221)
第八章 文件	(229)
8.1 文件、流和文件系统	(229)
8.2 缓冲文件系统	(230)
8.2.1 文件的打开、关闭和文件结束测试	(231)
8.2.2 文件的读写	(232)
8.2.3 文件的定位	(237)
8.2.4 出错的处理	(240)
8.3 非缓冲文件系统	(240)
习题	(243)
* 第九章 实用程序设计初步	(246)
9.1 图形处理	(246)
9.1.1 图形模式的初始化	(246)

9.1.2	独立图形运行程序的建立	(249)
9.1.3	屏幕颜色的设置和清屏函数	(249)
9.1.4	基本图形函数	(251)
9.1.5	填充	(254)
9.1.6	图形窗口和图形屏幕操作函数	(257)
9.1.7	图形模式下的文本输出	(259)
9.2	中断处理	(263)
9.2.1	中断的允许和禁止	(264)
9.2.2	DOS 与 BIOS 功能调用	(264)
9.2.3	中断服务程序	(269)
* 第 10 章	C++ 简介	(271)
10.1	C++ 的新特征	(271)
10.1.1	C++ 的输入/输出	(271)
10.1.2	C++ 的函数原型	(273)
10.1.3	C++ 函数的缺省参数	(273)
10.1.4	C++ 的 new 与 delete	(274)
10.1.5	C++ 的内联函数	(275)
10.1.6	C++ 的引用	(276)
10.1.7	C++ 面向对象编程基础	(277)
10.2	C++ 编程的核心技术	(279)
10.2.1	类的定义与使用	(279)
10.2.2	数据的封装	(281)
10.2.3	函数的重载	(283)
10.2.4	对象的初始化	(283)
10.2.5	缺省构造函数、拷贝构造函数与析构函数	(285)
10.3	类成员与对象的构建	(287)
10.3.1	使用 this 指针	(287)
10.3.2	使用静态成员	(289)
10.3.3	使用友员	(293)
10.3.4	使用对象成员	(294)
10.3.5	使用对象数组	(296)
10.3.6	使用指向对象的指针	(298)
10.3.7	类型的转换	(299)
10.4	派生类的构造	(300)
10.4.1	派生类的定义	(300)
10.4.2	类的保护成员	(302)
10.4.3	访问权限的设置	(303)
10.4.4	派生类的构造函数与析构函数	(305)
10.4.5	多重继承	(307)

10.4.6	在派生类中改写基类的成员函数	(308)
10.4.7	虚拟函数	(310)
10.4.8	纯虚拟函数与抽象类	(311)
10.5	运算符重载	(314)
10.5.1	运算符重载的作用与形式	(314)
10.5.2	类运算符与友元运算符	(315)
10.5.3	++与--运算符的重载	(317)
10.5.4	重载 new 和 delete	(320)
10.6	输入/输出流的使用	(320)
10.6.1	标准的屏幕输出	(321)
10.6.2	标准的键盘输入	(327)
10.6.3	用户自定义类的输入/输出	(329)
10.6.4	格式化字符串流类的使用	(331)
10.6.5	磁盘文件的输入/输出	(333)
10.6.6	打印机的使用	(342)
10.7	模板	(343)
10.7.1	模板的基本概念	(343)
10.7.2	函数模板的定义与使用	(344)
10.7.3	类模板的定义与使用	(346)
习题		(348)
第十一章	Turbo C 集成开发环境中调试程序	(353)
11.1	Turbo C 集成开发环境调试程序基本要领	(353)
11.1.1	纠正编译错误	(353)
11.1.2	纠正连接错误	(357)
11.1.3	纠正逻辑错误	(358)
11.2	调试程序实例	(362)
11.3	调试程序命令和热键小结	(367)
11.4	Turbo C 程序的常见错误	(368)
11.4.1	使用变量容易出现的错误	(368)
11.4.2	编写表达式容易出现的错误	(368)
11.4.3	使用语句容易出现的错误	(370)
11.4.4	使用数组容易出现的错误	(372)
11.4.5	使用库函数容易出现的错误	(373)
11.4.6	使用自定义函数容易出现的错误	(374)
11.4.7	使用指针变量容易出现的错误	(377)
11.4.8	其他常见错误	(378)
11.5	小结	(379)
习题		(379)

附录 A	C 语法摘要	(380)
附录 B	数值系统	(385)
附录 C	Turbo C 2.0 集成开发环境的使用	(390)
附录 D	ASCII 字符集	(402)
附录 E	运算符的优先级与结合性	(404)
附录 F	Turbo C 的部分标准函数	(405)
附录 G	编译错误信息	(409)
附录 H	习题参考答案	(417)

第 1 章

C 语言特点与上机操作

学习计算机程序设计语言是提高人们计算机知识水平的重要步骤。C 语言作为当今最为流行的程序设计语言之一,不但成为计算机专业的必修课程,而且也越来越多地成为非计算机专业的学习课程。本章介绍 C 语言的发展与特点,叙述 C 语言程序的组成与结构,阐明 C 语言的上机步骤和方法。

建议本章授课 2 学时,上机 2 学时,自学 3 学时。

1.1 C 语言特点

1.1.1 C 语言的产生与发展

C 语言是 1971 年由美国贝尔实验室的 D. M. Ritchie 用了一年的时间设计发明的,1972 年投入使用。1973 年 K. Thompson 和 D. M. Ritchie 用 C 语言重写 UNIX 操作系统获得了巨大成功。随着微型计算机的日益普及,出现了许多 C 语言版本。1983 年美国国家标准化协会(ANSI)为 C 语言制定了一套 ANSI 标准,1987 年 ANSI 公布的 87 ANSI 标准成为现行的 C 语言标准。20 世纪 90 年代至今,美国 Borland 公司陆续推出了 Turbo C、Turbo C++、Borland C++以及 C++ Builder 等系列产品,Microsoft 公司也推出了 Microsoft C、Visual C 等产品。目前这些产品均提供了面向对象的可视化开发环境,用户可以快速、方便地建立 DOS/Windows 应用程序。C 语言已成为程序员使用最多的编程语言之一。无论是面向硬件编程,还是面向大型数据库编程,无论是编写应用软件,还是编写操作系统,C 语言都是首选编程语言。

本书内容以 Turbo C 2.0 为标准。

1.1.2 计算机语言与程序设计

计算机要完成某一特定的任务,必须执行一系列计算机指令。程序就是由这样的一系列计算机指令组成的。程序设计就是针对某一要处理的问题,设计出解决该问题的计算机指令序列。因此程序设计是一项创造性的工作。进行程序设计必须借助语言来描述,这些用来描述的语言就是程序设计语言。只有严格按照程序设计语言的语法规则来书写程序,才能让计算机正确执行指令序列,完成指定的任务。程序设计语言分为低级语言和高级语言两大类。低级语言直接面向机器,如机器语言和汇编语言;高级语言独立于机器,用高级语言编写的程序在不同的机器上必须使用不同的翻译程序。C 语言程序是一种高级语言程序,它必须被翻译成计算机

能识别的语言,即机器语言,才能在计算机上运行。

1.1.3 C 语言的特点

C 语言之所以能迅速崛起,并成为最受欢迎的程序设计语言之一,是因为它有许多优于其他语言的特点。C 语言具有下列特点:

1. C 语言功能齐全

C 语言的数据类型有:整型、实型、字符型、无符号整型、数组类型、指针类型、结构体类型、共用体类型、枚举型等。C 语言运算符丰富,表达式类型有:赋值表达式、算术表达式、关系表达式、逻辑表达式、条件表达式、逗号表达式以及位运算等。

2. C 语言简洁、紧凑,使用方便、灵活

C 语言的一个语句可完成多项操作,一个表达式也可以完成多项操作。书写简练,源程序短,因而输入程序的工作量小。

3. C 是面向结构化程序设计的语言

结构化语言的显著特点是代码、数据的模块化,C 程序是以函数形式提供给用户的,这些函数调用方便。C 语言具有多种条件语句、循环控制程序流向语句(如 if /else 语句、switch 语句、while 语句、do/while 语句、for 语句、break 语句、continue 语句等),从而使程序完全结构化。

4. C 是中级语言

C 语言把高级语言的基本结构和语句与低级语言的实用性结合起来。C 语言可以像汇编语言一样对位、字节和地址进行操作,实现汇编语言的大部分功能,可直接对硬件进行编程。用 C 语言加上一些汇编语言子程序编程,更能显示 C 语言的优势。C 语言源程序编译后代码短,执行效率高。

5. C 语言适用范围大

C 语言还有一个突出的优点就是适合于多种操作系统,如 DOS、Windows、UNIX,也适用于多种机型。源程序代码可移植性好。

1.2 C 语言程序基本组成

下面是两个 C 语言程序的例子:

[例 1.1] 由键盘输入三角形的三个边长,计算出该三角形的面积。

```
#include <stdio.h>    /* 文件包含,输入、输出函数 */
#include <math.h>      /* 文件包含,数学函数 */
main()                /* 主函数 */
{
    float a, b, c, l, area; /* 定义局部变量 */
    printf("请输入三角形三条边的边长:");
    scanf("%f%f%f", &a, &b, &c); /* 由键盘输入三角形三边的边长 */
    l = (a+b+c)/2.0;
    area = sqrt(l * (l-a) * (l-b) * (l-c)); /* 函数 sqrt(x)是求 x 的平方根 */
}
```

```
printf("该三角形的面积是:%6.2f\n", area);
}
```

运行结果:

请输入三角形三条边的边长:3.4 5.6 7.4

该三角形的面积是: 9.05

[例 1.2] 与[例 1.1]一样,由键盘输入三角形的三个边长,计算出该三角形的面积。但使用函数来求面积。

```
#include <stdio.h>
#include <math.h>
float triangle __area(float a, float b, float c) /* 定义用户函数 */
{
    float l;
    l=(a+b+c)/2.0;
    return sqrt(l*(l-a)*(l-b)*(l-c));
}
main() /* 主函数定义 */
{
    float a, b, c, area;
    printf("请输入三角形三条边的边长:");
    scanf("%f%f%f", &a, &b, &c);
    area=triangle __area(a,b,c);
    printf("该三角形的面积是:%6.2f\n", area);
}
```

运行结果与[例 1.1]相同。

从上面例子可以看出,C程序的组成主要有以下几个特点:

1. 一个C源程序由函数构成,其中必须有且只能有一个主函数(main函数),还可以有0至多个其他函数。C程序由main函数的首句开始执行,由main函数的最后一句结束,函数中可调用其他函数。在C语言中函数分为两种,用户可以自己定义函数(如[例 1.2]中的triangle __area函数),也可以使用C语言系统提供的库函数(如printf函数和scanf函数)。Turbo C提供了300多个库函数,要调用C的库函数,必须在源程序首部加上相应的库文件包含(如上述例子中的#include <stdio.h>)。

2. C程序一般用小写字母书写,大、小写字母是有区别的,如area与Area代表不同的变量。C程序书写格式自由,一行内可写多条语句,若一条语句较长,可分写在多行上。一般情况下语句中的空格和回车符可忽略不计。语句用分号“;”结尾,分号“;”是C语句的一部分。可以在{ }内写若干条语句,构成复合语句,也称语句块。用C语言编程时,建议一行写一条语句,遇到复合语句向右缩进,必要时对程序加上注释行。这样写出的源程序结构清楚,易于阅读、调试、维护和修改。

3. C语言的变量在使用之前必须先定义其数据类型,未经定义的变量不能使用。一般应在可执行语句前面定义变量类型。

4. 函数由函数头与函数体两部分组成。第一部分为函数头(函数说明部分),包括函数返

回值类型、函数名、函数参数及参数的数据类型。第二部分为函数体部分,它是函数功能的实现部分,包括变量定义与执行语句。

5. 一个较完整的程序通常包括:文件包含(一组 #include 语句)、用户函数说明部分、全局变量定义、主函数和若干用户函数。在主函数和用户函数中又包括局部变量定义、若干个 C 库函数调用语句、控制流程语句、用户函数的调用语句等。C 源程序的一般形式为:

```
文件包含
用户函数说明
全局变量定义
main()
{
    局部变量定义语句
    语句
}
fun1()
{
    局部变量定义语句
    语句
}
fun2()
{
    局部变量定义语句
    语句
}
    ⋮
funN()
{
    局部变量定义语句
    语句
}
```

其中 fun1(), fun2(), ..., funN() 代表用户定义的函数,语句指赋值语句、控制流程语句、C 提供的任何库函数调用语句或用户函数调用语句等。C 语言函数内部不能定义函数,函数之间是平等的。主函数 main 可以放在某一用户函数之前,也可以放在某一用户函数之后,但被调用的函数应在主调函数之前定义或说明。在编写较大型的 C 程序时,常把源程序分成多个文件编写,采用文件包含或工程文件(即项目文件 *.PRJ)的方法连接成可执行程序(请参阅附录 C“Turbo C 2.0 集成开发环境的使用”)。

6. 用户为了提高源程序的可读性,可在 C 程序中加上注释部分,编译时注释部分被滤掉。C 程序的注释部分包含在“/*”和“*/”之间, / 和 * 之间不允许有空格。注释部分允许出现在程序中的任何位置。

1.3 Turbo C 2.0 的上机步骤

1.3.1 编辑、编译、连接、执行及调试程序的概念

1. 编辑

程序员用 C 语言编写的程序称为 C 的源程序(一般为 *.C 文件)。编辑就是编写源程序的过程,它包括新建一个源程序文件或修改已有的源程序文件,它的操作有插入、删除、修改源程序。除了 Turbo C 2.0 集成开发环境能够编辑源程序外,还可使用 DOS 环境中的 EDIT、CCED、WPS 或 Windows 环境中的 WORD、记事本、写字板等常用的编辑软件来编辑 C 的源程序,存盘时应采用纯文本方式保存文件。

2. 编译

源程序是以纯文本形式存储的,必须翻译成机器语言才能被计算机识别。完成这一翻译工作的就是所谓的编译程序。源程序经过编译程序翻译成等价的机器语言程序——目标程序(一般为 *.OBJ 文件),这一翻译过程称为编译。Turbo C 2.0 集成开发环境带有编译程序。

3. 连接

如果编译成功,还应将目标程序和 C 的库函数连接成可执行程序(一般为 *.EXE 文件),并存储在计算机的存储设备(外存)中,以便执行。负责目标程序和库函数连接工作的程序称为连接程序。Turbo C 2.0 集成开发环境带有连接程序。

4. 执行

源程序经过编译、连接成为可执行文件(扩展名为 .exe 或 .com)后,一般存于计算机的外存中。所谓执行程序就是把可执行文件从外存调入计算机内存,并由计算机完成该程序预定的功能,如完成输入数据,处理数据及输出结果等任务。执行程序又称为运行程序。

5. 调试

源程序中难免会存在错误,错误一般可分为四类:

(1)编译错误:程序不符合 C 语言语法规则,在编译时将出错,编译错误包括语法错误(error)和警告错误(warning)。例如某一变量未定义先使用,则会出现语法错误。又如某变量未赋初值就用来求和,则会出现警告错误。

(2)逻辑错误:一个程序在编译时没有出现错误,执行后仍然得不到正确结果,这是由于在算法的设计过程或程序的表达式中存在错误,如表达式书写错误、程序控制流程错误等。

(3)运行错误:程序执行时在某些特殊情况发生的错误,如变量越界、除零错误等。

(4)连接错误:把目标程序连接成可执行程序时出现错误,如找不到库文件错误等。

程序调试是指对程序进行查错和排错。最常见的错误是编译错误和逻辑错误,有关程序的调试我们将在第十一章中详细叙述。

上述几个步骤在 Turbo C 2.0 集成开发环境中可以很方便地实现。上机操作的整个过程如图 1.1 所示:

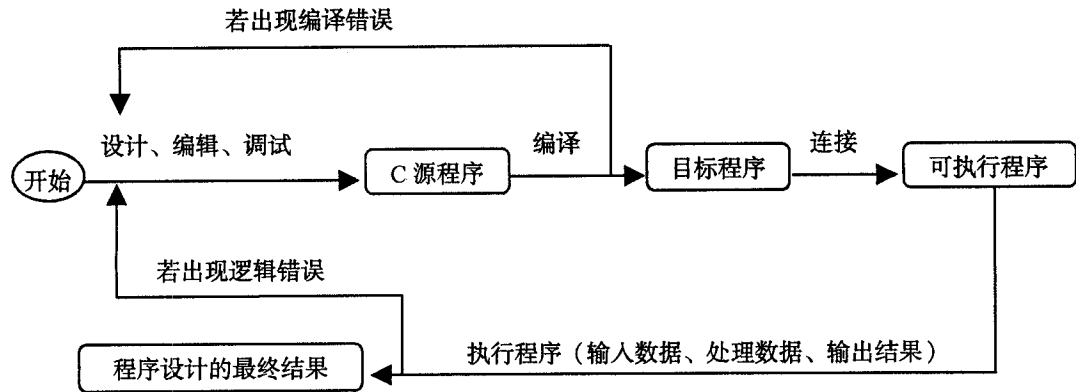


图 1.1 C 语言上机操作过程

1.3.2 Turbo C 的上机步骤

利用 Turbo C 2.0 集成开发环境可以非常方便地完成程序的编辑、调试、编译、连接和运行。以下通过一个简单的例子来说明 Turbo C 2.0 集成开发环境的上机步骤：

1. 在 DOS 状态下直接键入 tc 调用 Turbo C 程序。此时屏幕显示如图 1.2 所示的 Turbo C 主屏幕，按 **Esc** 键光标进入编辑窗口，这样就可以编辑源程序了。

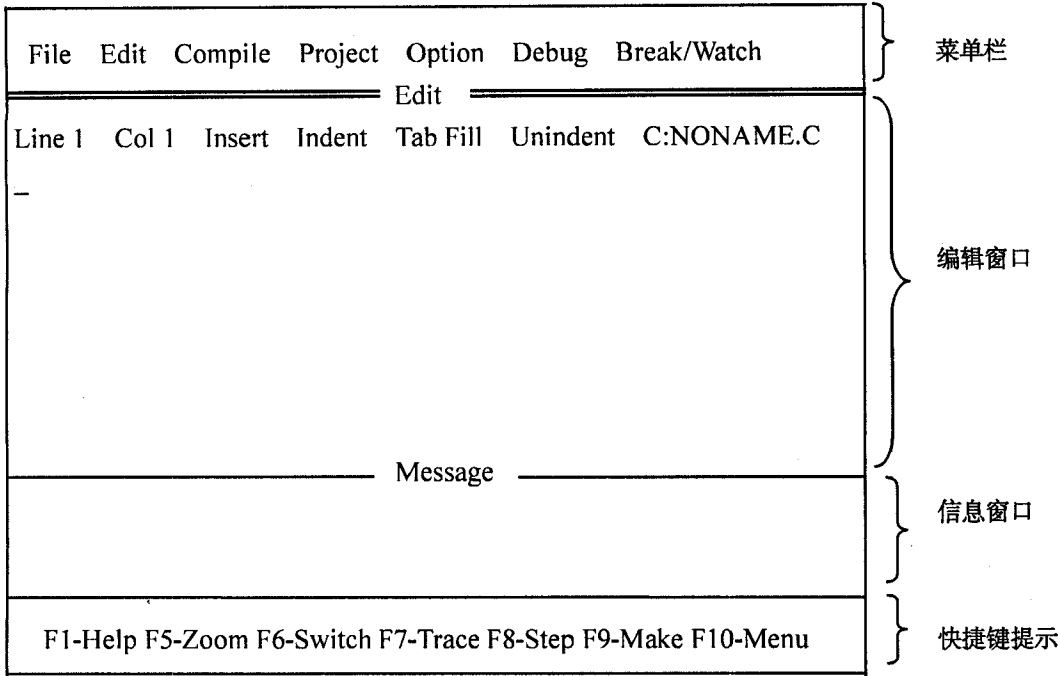


图 1.2 Turbo C 2.0 集成开发环境的主屏幕

2. [例 1.3] 编辑以下源程序：