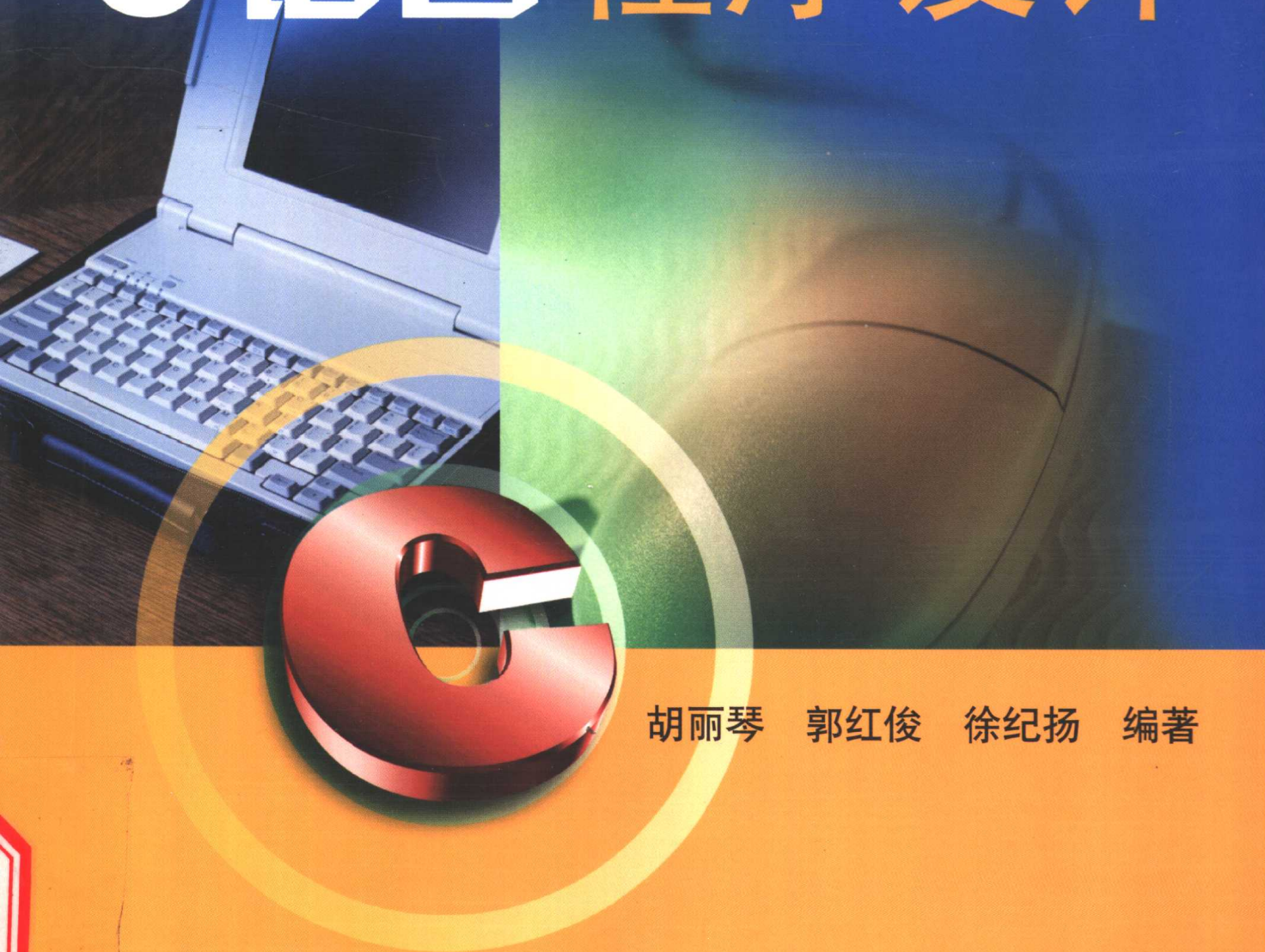


全国高职高专教育计算机系列教材

C语言程序设计



胡丽琴 郭红俊 徐纪扬 编著

北京大学出版社
<http://cbs.pku.edu.cn>

全国高职高专教育计算机系列教材

C 语言程序设计

胡丽琴 郭红俊 徐纪扬 编著

北京大学出版社
北 京

图书在版编目(CIP)数据

C 语言程序设计/胡丽琴,郭红俊,徐纪扬编著. —北京:北京大学出版社,2003.3

(全国高职高专教育计算机系列教材)

ISBN 7-301-06168-4

I. C… II. ①胡… ②郭… ③徐… 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(2003)第 010546 号

书 名: C 语言程序设计

著作责任者: 胡丽琴 郭红俊 徐纪扬 编著

责任编辑: 王 艳

标准书号: ISBN 7-301-06168-4/TP·0707

出版者: 北京大学出版社

地 址: 北京市海淀区中关村北京大学校内 100871

网 址: <http://cbs.pku.edu.cn>

电 话: 邮购部 62752015 发行部 62750672 编辑部 62752021

电子信箱: zpup@pup.pku.edu.cn

排 印 者: 北京大学印刷厂

发 行 者: 北京大学出版社

经 销 者: 新华书店

787 毫米×1092 毫米 16 开本 16.625 印张 414 千字

2003 年 3 月第 1 版 2003 年 3 月第 1 次印刷

定 价: 24.00 元

前 言

人类已经步入信息化的 21 世纪,为了适应社会的需要,使我国计算机的开发和应用进一步向深度和广度发展,在我国的各类高等教育领域中广泛地开设 C 语言程序设计课程是必要的。同时,学习 C 语言也为进一步学习面向对象程序设计语言和可视化编程打下良好基础。

本书编者都是计算机专业教师,都曾参与程序开发,并多次讲授 C 语言程序设计语言课程,了解学生及计算机应用开发者的需求,为此将各自的一些实践、体会、心得及参考到的新内容写到此书中,以期望对学习者的能有所帮助。

由于 C 语言涉及的概念多,规则复杂,书写灵活,容易出错,初学者往往感到不易掌握。针对 C 语言比较难学的特点,为满足高职高专教学改革的需要,本书按照面向应用,重视实践,便于自学的原则编写的;从 C 语言的最基本概念入手,由浅入深,循序渐进的学习规律进行编排;综合大量的编程实例,用通俗易懂的语言和丰富的习题引导初学者从入门到掌握 C 语言。本书的目的就是为初学者提供一本全面了解 C 语言基础及应用开发方面的教材。

本书的内容有如下特色:

(1) 本书所涉及的程序实例都按照标准格式编写,并从程序实例、程序构思、程序代码、运行结果、程序说明以及程序思考等几大部分将每一个程序进行详细解释,便于初学者的学习并养成编程的好思维、好习惯。

(2) 本书通俗易懂,对读者没有特殊要求,使初学者易于学习和掌握本书的基本内容。

(3) 提供了大量的习题,各个部分是按由浅入深、循序渐进的学习规律进行编排的,便于教师和学生配合教学进程,有目的地进行选用。本习题中的内容和题型,与教育部考试中心制定的《全国计算机等级考试二级大纲》中有关 C 程序设计的要求基本吻合,因此对于准备参加全国计算机等级考试二级的考生来说非常适合。

(4) 本书介绍的 C 语言及其程序例题具有通用性,基本上适合任何计算机系统和 C 语言的版本。

本书适用于各种类型的计算机教学的 C 语言程序设计课程,尤其是高职高专学生学习 C 语言程序设计的理想教材,对于已经具备计算机初步知识并且想学习 C 语言的读者也是理想的自学参考书。

本书由胡丽琴、郭红俊、徐纪扬编写。本书在编写的过程中得到海淀走读大学各级领导的支持,在此表示感谢,同时还要感谢信息学院领导及同事。

由于作者水平有限,书中的错误和不足在所难免,恳请广大读者批评指正。

编 者
2003 年 2 月

目 录

第 1 章 绪论	(1)
1.1 C 语言出现的历史背景	(1)
1.2 C 语言的特点	(2)
1.3 C 语言的程序编辑环境	(3)
1.4 基本的程序结构	(8)
习题	(11)
第 2 章 初步掌握 C 语言编程	(12)
2.1 浅谈数据类型	(12)
2.2 常量	(12)
2.3 变量	(17)
2.4 数据类型转换	(20)
2.5 运算符和表达式	(21)
习题	(32)
第 3 章 C 语言程序的控制结构	(35)
3.1 算法简介	(35)
3.2 顺序结构	(37)
3.3 分支结构	(45)
3.4 循环结构	(55)
习题	(67)
第 4 章 数组	(77)
4.1 数组概述	(77)
4.2 数组定义	(77)
4.3 一维数组	(78)
4.4 二维数组	(83)
4.5 数组的初始化	(87)
4.6 字符数组	(90)
4.7 数组应用举例	(99)
习题	(106)
第 5 章 函数	(110)
5.1 系统函数	(110)
5.2 自定义函数及其调用	(114)
5.3 形式参数为数组的函数	(119)

5.4	变量的存储类型	(124)
5.5	函数的嵌套调用	(130)
5.6	递归函数	(132)
	习题	(137)
第 6 章	指针	(144)
6.1	指针的基本概念	(144)
6.2	指针和数组的关系	(149)
6.3	用指针处理字符串	(153)
6.4	用指针变量作形式参数实现函数间的数据传递	(155)
6.5	指向数组的指针	(157)
6.6	指针数组的概念和应用	(159)
6.7	带参数的主函数	(163)
6.8	指针函数	(165)
6.9	函数指针	(166)
6.10	二级指针	(170)
	习题	(171)
第 7 章	结构体、共用体和枚举	(175)
7.1	结构体的定义	(175)
7.2	结构体成员的访问	(178)
7.3	结构体数组	(181)
7.4	结构体指针	(185)
7.5	结构体在函数间的传递	(188)
7.6	共用体和枚举	(190)
7.7	自定义类型	(191)
	习题	(192)
第 8 章	文件	(196)
8.1	文件概述	(196)
8.2	文件的打开和关闭	(198)
8.3	文件的读写	(200)
8.4	文件定位	(217)
8.5	输入与输出转向	(219)
8.6	文件出错的检测	(220)
	习题	(221)
第 9 章	编译预处理	(224)
9.1	宏代换(宏定义)	(224)
9.2	文件包含	(226)
9.3	条件编译	(227)
9.4	其他编译预处理命令	(229)
	习题	(229)

附录 1	C 语言中的常用关键字	(233)
附录 2	运算符的优先级与结合性	(234)
附录 3	ASCII 字符编码表	(235)
附录 4	常用库函数	(236)
附录 5	编译错误信息	(241)
附录 6	全国计算机等级考试 C 语言程序设计(二级)考试大纲	(254)

第 1 章 绪 论

本章主要内容:

- C 语言出现的历史背景
- C 语言的特点
- C 语言的程序编辑环境
- 基本的程序结构

本章主要讲述了 C 语言的发展过程、C 语言的特点及其程序编辑环境。虽然,这一章主要是知识性的内容,很少涉及到具体的编程技术,但是,本章这些内容能够使你对 C 语言有一个大概的认识,了解 C 语言是如何发展起来的,C 语言有什么与众不同的特点,C 语言的产生对软件开发有什么影响,等等。

1.1 C 语言出现的历史背景

1.1.1 C 语言出现的原因

C 语言是国际上广泛流行的、很有发展前途的计算机高级语言。它不仅可用来写系统软件,也可用来写应用软件。

系统软件主要指操作系统软件,如 DOS,Windows,以及潜力极大的 UNIX 和 Linux 等。应用软件则指具有某方面强大功能的软件(为多种应用目的而编制的程序),如字处理方面:Word,Excel 等软件;图像处理方面:Photoshop,Coreldraw 等软件;网页制作方面:Frontpage,Dreamweaver,Flash,Fireworks 等软件。

以前的操作系统大多是用汇编语言编写的。由于汇编语言依赖于计算机硬件,程序的可读性和可移植性都比较差,若要提高程序的可读性和可移植性,则最好使用高级语言。但一般高级语言难以实现汇编语言的某些功能(汇编语言可以直接对硬件进行操作,如对内存地址的操作、位操作等),所以人们设想找到一种既具有一般高级语言的特性,又具有低级语言特性的语言,集它们的优点于一身,于是 C 语言就在这种情况下应运而生了。

1.1.2 C 语言的发展史

C 语言的根源可追溯到 ALGOL58。1958 年出现的 ALGOL 语言是 C 语言的雏形,1960 年出现的 ALGOL60 是一种面向问题的高级语言,它离硬件比较远,不适于写系统程序。1963 年英国剑桥大学推出了 CPL(combined programming language)语言,CPL 语言在 ALGOL60 的基础上更接近硬件一些,但规模比较大,难以实现。1967 年英国剑桥大学的 Martin Richards 对 CPL 语言做了简化,推出了 BCPL(basic combined programming language)语言。1970 年美

国贝尔实验室的 K. Thompson 以 BCPL 为基础,又进一步简化,设计出了很简单的且很接近硬件的 B 语言(取 BCPL 的首字母)。同时 K. Thompson 用 B 语言写了第一个 UNIX 操作系统,在 PDP-7 上实现。但 B 语言过于简单,功能有限。1972 年至 1973 年间,贝尔实验室的 D. M. Ritchie 在 B 语言的基础上设计出了 C 语言(取 BCPL 的第二个字母)。C 语言既保持了 BCPL 和 B 语言的优点(精练、接近硬件),又克服了它们的缺点(过于简单、数据无类型等)。1973 年, K. Thompson 和 D. M. Ritchie 两人合作把 UNIX 的 90% 以上用 C 语言改写,即 UNIX 的第 5 版。C 语言的发展过程见图 1.1。

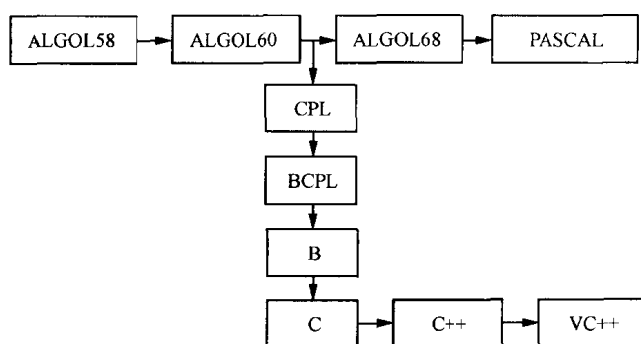


图 1.1 C 语言的发展过程

C 语言在以后做了多次改进,但主要还是在贝尔实验室内部使用。直到 1975 年 UNIX 第 6 版公布后,C 语言的突出优点才引起普遍注意。1977 年出现了不依赖于机器的 C 语言编译文本《可移植 C 语言编译程序》,使 C 语言移植到其他机器时所需做的工作大大简化,这也推动了 UNIX 操作系统迅速地在各种机器上实现。随着 UNIX 的日益广泛使用,C 语言也迅速得到推广。C 语言与 UNIX 操作系统互相依存,二者相互促进得以发展。UNIX 第 7 版在 1978 年研制成功,以其中的 C 语言编译系统为基础,美国国家标准化协会于 1983 年制定了一个 C 语言标准草案,即 83 ANSI C,1987 年又公布了 87 ANSI C。1990 年,国际标准化组织 ISO 接受 87 ANSI C 为 ISO C 的标准。目前流行的 C 语言编译系统都是以它为基础的,本书基本上以 ANSI C 为基础。

目前 C 语言编译系统有多种版本,在微机上常用的有 Microsoft C, Turbo C, Quick C, Borland C 等。虽然它们的基本部分是相同的,但也有一些差异,故用户应了解所使用的计算机系统所配置的 C 语言编译系统的特点和规范。

1.2 C 语言的特点

C 语言是一种高级语言,同时它又提供了类似于汇编语言的低级语言的功能,如它可以直接访问物理地址并能进行位操作,为编写系统程序提供了方便。C 语言的主要特点如下:

(1) 语言简捷、紧凑,使用方便、灵活。C 语言常用关键字 39 个(见附录 1),9 种控制语句,程序书写形式自由灵活。

(2) 运算符丰富。C 语言的运算符包含的范围很广泛,共有 43 种运算符(见附录 2)。C 语言把括号、赋值、强制类型转换等都作为运算符处理,从而使 C 语言的运算类型极其丰富,表达式类型多样化。

(3) 数据类型丰富,具有现代语言的各种数据类型。C 语言的数据类型有整型、实型、字符型、数组类型、指针类型、结构体类型、共用体类型等,能用来实现各种复杂的数据结构(如链表、树、栈)的运算。

(4) 具有结构化的控制语句。C 语言包含 if...else 语句、while 语句、do...while 语句、switch 语句、for 语句。用函数作为程序的模块单位,便于实现程序的模块化。C 语言是理想的结构化语言,符合现代编程风格的要求。

(5) 语法限制不太严格,程序设计自由度大。变量的类型使用比较灵活,如整型变量与字符型数据以及逻辑型数据可以通用。一般的高级语言语法检查比较严,能检查出几乎所有的语法错误。而 C 语言允许程序编写者有较大的自由度,因此放宽了语法检查。

1.3 C 语言的程序编辑环境

“工欲善其事,必先利其器”,程序语言的编写需要一个编辑环境,就像文字处理的工作必须要通过像 Microsoft Word 或 WordPad 之类的软件来完成一样,市面上有相当多的 C 语言编辑工具,比如 Borland 公司的 C++ Builder,Microsoft 公司的 Visual C++,甚至 DOS 时代的 Turbo C 等,都是不错的 C 程序编辑器,我们还可以使用“记事本”、Word、WPS 等来完成 C 程序的编写。

完成编写的程序可以通过编译器来进行查错转换,形成计算机能够识别并可执行的代码,一般的程序编辑工具都有编译的功能。下面分别就两种典型的环境下运行 C 程序作一简单介绍。本书中的范例程序都通过了这两种环境下的编译运行。

1.3.1 用 Turbo C 运行 C 程序的步骤

Turbo C 是在微机上广泛使用的 C 语言编译器。它具有方便、直观、易用的界面和丰富的库函数。它向用户提供一个集成环境,把程序的编辑、编译、链接和运行等操作全部集中在一个界面上进行,使用十分方便。

为了能使用 Turbo C,必须先将 Turbo C 编译程序装入磁盘的某一目录下,例如,放在 C 盘根目录下一级 TC 子目录下。编写调试的步骤如下:

(1) 调用 Turbo C 程序。如果用户的当前目录是 Turbo C 编译程序所在的子目录(例如 TC 子目录),只需从键盘键入“tc”命令(C:\TC>tc),然后按回车键,屏幕上出现 Turbo C 集成环境,见图 1.2 所示。

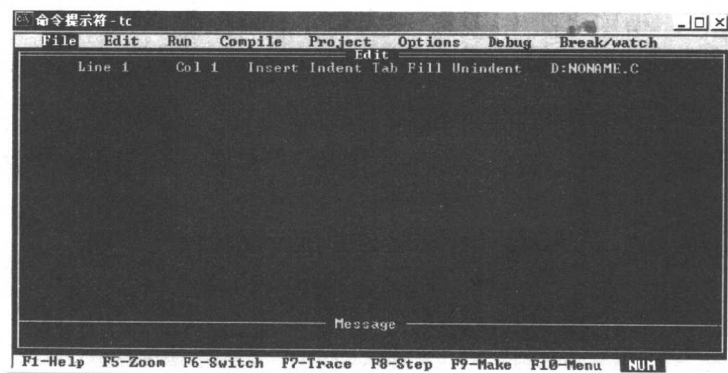


图 1.2 Turbo C 2.0 界面示意图

从图 1.2 可以看到,在其上部有一行“主菜单”,包括以下 8 个菜单项: File、Edit、Run、Complie、Project、Options、Debug、Break/watch。用户可以通过以上菜单项来选择使用 Turbo C 集成环境所提供的 Turbo C 的各项主要功能。以上 8 个菜单项分别代表文件操作、编辑、运行、编译、项目文件、选项、调试、中断/观察等功能。

用键盘上的“←”和“→”键可以选择菜单栏中所需要的菜单项,被选中的项以“反相”形式显示(例如主菜单中的各项原来以白底黑字显示,被选中时改为以黑底白字显示)。此时若按回车键,就会出现一个下拉菜单。例如在选中“File”菜单并按回车键后,屏幕上“File”下面出现下拉菜单,见图 1.3 所示。它是一个子菜单,提供多项选择。可以用键盘上的“↓”键选择所需要的项。例如选择“New”处,并按回车键,表示要建立一个新的 C 源程序。如果选择“Load”,并按回车键,表示要调入一个已有的源文件,此时屏幕上出现一个对话框(见图 1.4),用户可输入该文件名,例如 d:\code\exam.C。如果此文件存在,则系统会将此文件调入内存并显示在屏幕上。此时自动转为编辑(Edit)状态。如果原来不存在此文件名,则系统会建立一个以指定的名字命名的新文件。

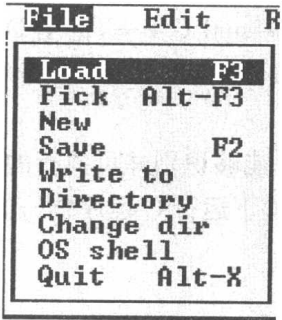


图 1.3 File 菜单的下拉菜单

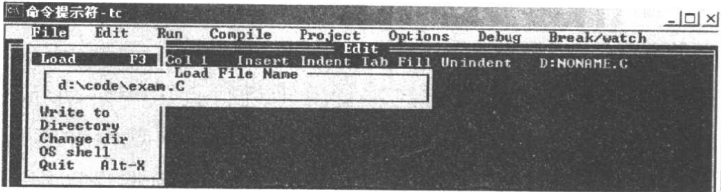


图 1.4 Load 命令装入已存在的文件

(2) 编辑源文件。在编辑(Edit)状态下可以根据需要输入或修改源程序。

(3) 编译源程序。选择“Compile”菜单并在其下拉菜单中选择“Compile to OBJ”,则进行编译,得到一个后缀为.obj的目标程序。然后再选菜单“Compile→Link EXE file”,进行链接操作,可得到一个后缀为.exe的可执行文件。

也可以将编译和链接合为一个步骤进行。选菜单“Compile→Make EXE file”或按“F9”键,即可一次完成编译和链接。在屏幕上会显示编译或链接时有无错误和有几个错误,见图 1.5 所

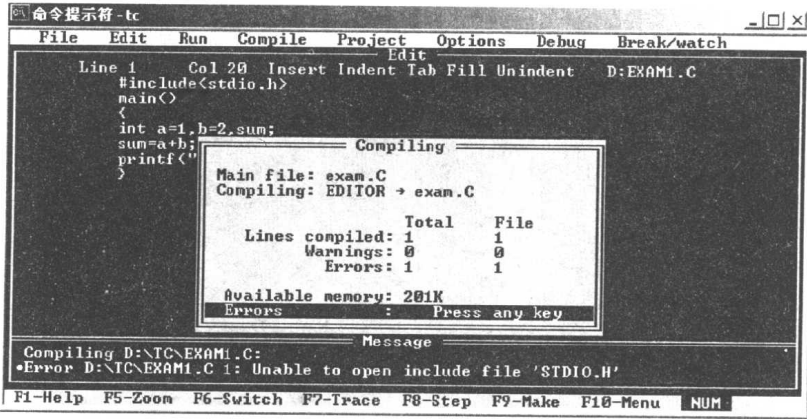


图 1.5 编译信息界面

示。此时按任何一个键,图 1.5 所显示的编译信息框会消失,屏幕上会恢复显示源程序,光标停留在出错之处。在屏幕的下半部分显示出有错误的行和错误的原因。根据此信息修改源程序。修改完毕认为无错后,再按“F9”,再次进行编译和链接,如此反复进行修改、编辑,直到不显示出错为止。

(4) 执行程序。按“F10”键,窗口上部的主菜单中“File”反相显示,见图 1.4 最上面一行。

用链盘上的“→”键将亮块移到“Run”,按回车键,在其下拉菜单中选择“Run”项,或直接按“Ctrl+F9”键,系统就会执行已编译好的目标文件。此时,TC 集成环境窗口消失,屏幕上显示出程序运行时输出的结果。如果程序需要输入数据,则应在此时从键盘输入所需数据,然后程序会接着执行,输出结果。

如果发现运行结果不对,则要重新对源程序进行修改,此时可以再按“F10”键,并用“←”使亮块移到“Edit”处,按回车键,即进入编辑状态,重复步骤(2),(3),(4),直到得到正确结果为止。

(5) 可以用“Alt+X”键(同时按此两键),脱离 Turbo C,回到 DOS 命令状态。此时,可以用 DOS 命令显示源程序和运行程序:

d:\>TYPE exam.C (列出程序源码)

d:\>exam (执行目标程序 exam.exe)

如果想再修改源程序,可以重新执行步骤(1),并输入源程序文件名即可。

关于 Turbo C 环境和使用方法的详细介绍,请参阅相关用户手册。

1.3.2 用 Visual C++ 运行 C 程序的步骤

本书介绍的是 Microsoft 公司出版的 Visual C++(创天中文版),Visual C++是一套可视化的 C 程序编辑软件,非常容易使用。

Visual C++ 的编辑环境与一般的文本编辑软件相似,它利用了简明易懂的可视化分割方式,将环境分成数个局部,包含一个简单的编辑区、一个功能齐全的工具栏,还有一个隐含许多编辑功能的菜单,让程序设计人员可以轻易地在这个环境下编写程序,并对程序进行编辑以及编译。

(1) 调用 Visual C++ 程序。Developer Studio 是一个集成式的程序开发环境(integrated development environment,简称 IDE)。所谓集成式,就是指所有的程序功能一应俱全,包括程序编辑器、编译器、链接器以及链接库等,不仅提供了可视化的 C 程序编辑环境,还可以让程序的编写、编译、链接工作在同一个空间内完成,让工作更有效率。

Visual C++ 的开发环境含有可以输入源程序代码(source code)的程序编辑器(Editor),除了建立新的源程序代码程序外,我们还可以利用菜单或工具栏的选项,包括剪切、复制、粘贴、查找与替换、改变文字颜色或自动校对程序语法等功能,对程序进行编辑工作,如图 1.6 所示。

(2) 建立新工程。执行“文件→新建”,打开 Visual Studio 工程向导,并按照图 1.7 中的 i, ii, iii 顺序执行,其中第 i 步表示要建立的 C 程序的文件类型(一般选 Win32 Console Application);第 ii 步要求选择存放工程文件的文件夹或文件路径,一般已存在,如此例为 F:\hlq;第 iii 步定义该工程文件名称,如此例定为 prog1,当此名称键入后,第 ii 步中的文件夹后会自动出现该文件名,如图 1.7 所示。在图 1.7 中单击“确定”后,出现图 1.8。单击“完成”,以

及下一窗口的“确定”，将出现图 1.9 所示的窗口。

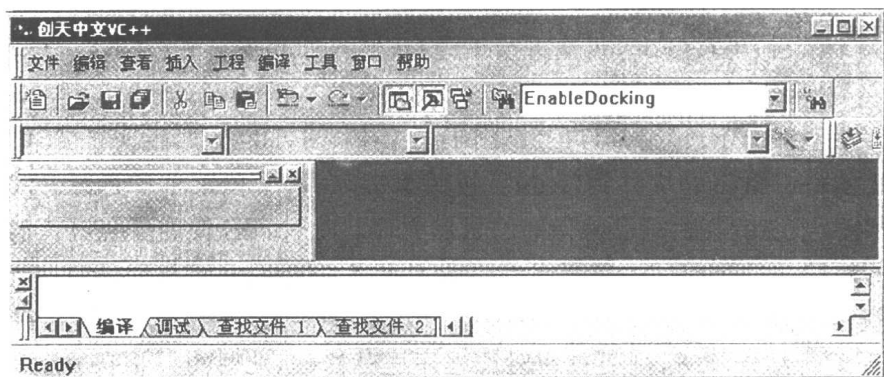


图 1.6 Visual C++ 界面

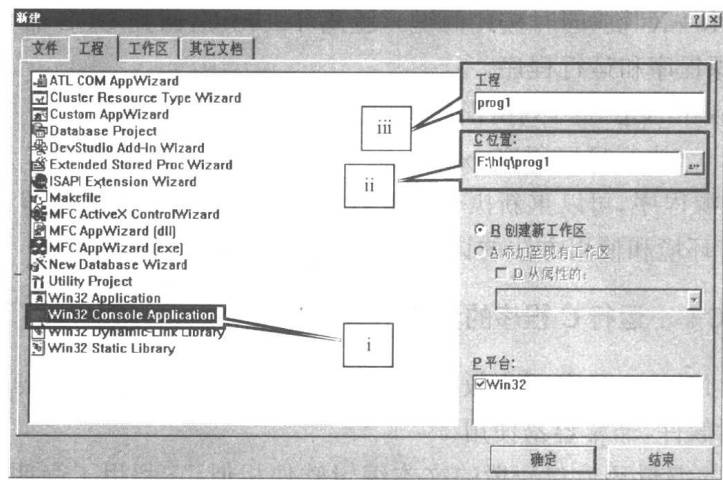


图 1.7

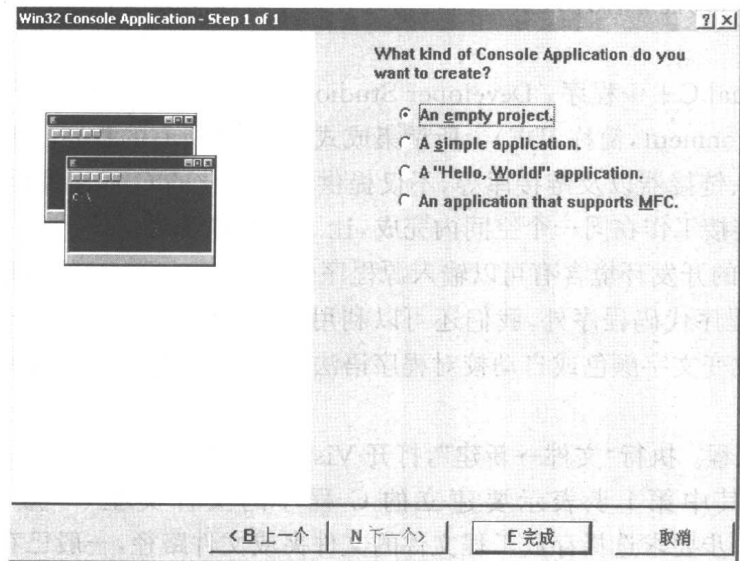


图 1.8

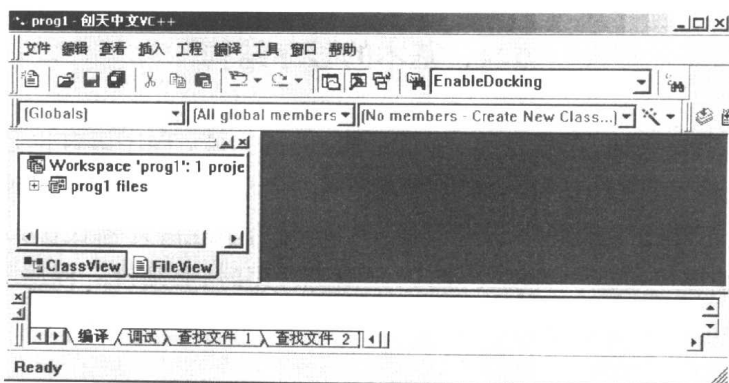


图 1.9

(3) 建立新文件。执行“文件→新建”，出现图 1.10 所示窗口，并按照其中的 i,ii 的顺序操作。

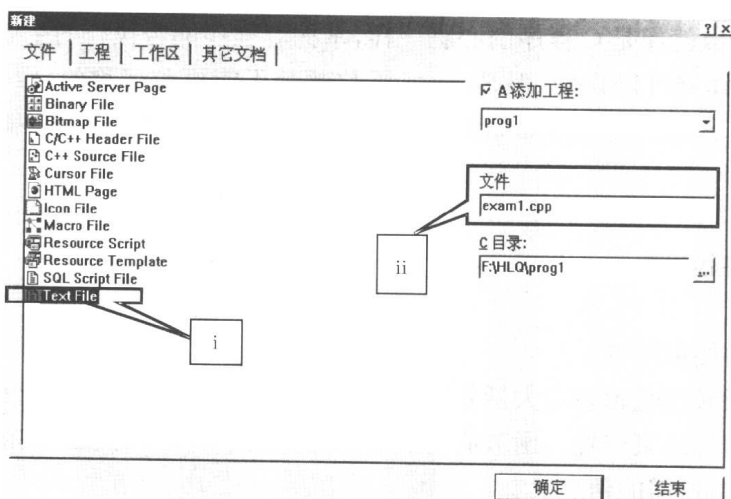


图 1.10

之后,单击“确定”,即可在新工作窗口的右侧白色空白区中编辑程序代码(如图 1.11),而后再单击红色的感叹号进行编译。

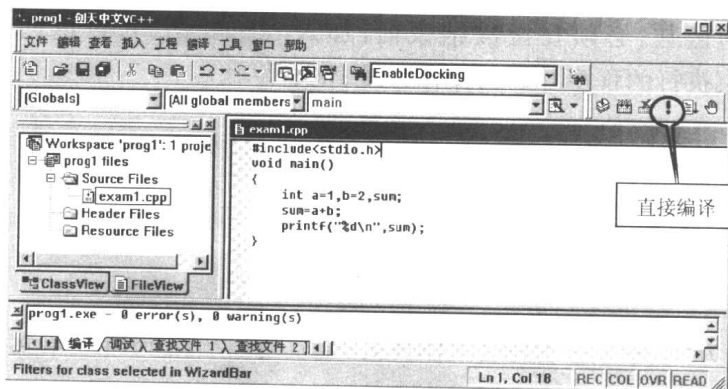


图 1.11

1.4 基本的程序结构

一栋屋子有其组成结构,哪里是梁、哪里是墙,一清二楚。程序语言也是如此。每个建筑师都有各自的设计风格,程序语言的结构也会随着不同的语法而有所差异,例如 Visual C++ 的程序结构就有别于汇编语言的程序结构。程序结构是为了让编译器(Compiler)能够正确地识别程序源代码,并编译成可以执行的文件,因此编写程序时,必须遵守程序语言的结构与规则,才能让设计的程序正常地被编译、执行。

1.4.1 函数

函数(function)是 C 程序最基本的程序架构,可以说 C 语言是由许许多多的函数所组成,即使是最简单的程序至少也会包含一个函数——main 函数(称为主函数)。每个 C 程序的主要程序代码都必须放在 main 函数里,如果程序中没有这个主函数,即使有再多的点子,程序也无法执行。main 函数就像是 C 程序的“心脏”一样,掌握着程序能否执行的生杀大权。函数必须有自己的名称,让编译器可以识别,但是 main 函数却是不能更改名称的,因为编译器在编译程序时,会先从 main 函数的位置开始编译,如果没有这个函数,就无法完成编译的工作。main 函数的结构如下:

```
void main( )
{ /* 函数的起始位置 */
  程序语句;
} /* 函数的结束位置 */
```

函数的语句区必须包括在左大括号“{”和右大括号“}”中,左大括号表示函数的起始位置,右大括号表示函数的结束位置。函数的写法有几种,包括有“返回类型”的函数,有“参数”的函数等,为调用函数时所需的信息。

“返回类型”表示函数返回值的数据类型,也就是函数返回给操作系统的值的类型,例如:

```
int main( );
```

上述式子表示调用 main 函数后,main 函数必须返回一个整型数值。如果返回类型为“void”,则表示该函数不返回任何值。

有返回类型的函数,必须在函数结束前,加入 return 语句,以返回该数据类型的值,“return 0”表示结束执行的程序并返回操作系统,例如:

```
int main( )
{
  :
  return 0;
}
```

如果返回类型为“void”,则不需返回任何值,也就可以省略 return 语句,例如:

```
void main( )
{
  : /* 省略 return 的语句 */
}
```

```
}
```

除了返回类型外,调用函数时还可传递参数,“参数”表示传递给函数的值,可以传递的值包括变量、指针、数组等参数。若参数为“void”或空白,则表示调用该函数时不需要传递任何数值,如:

```
void main( );
```

或

```
void main(void);
```

对于详细的函数定义和用法,我们将在后面章节中配合章节的主题来加以阐述。

1.4.2 输出与输入

C 程序从函数开始,我们在函数的语句区即可输入程序的内容。不论是哪一种类型的程序,都希望通过某种方式与用户产生交互,而用户与程序之间最直接的沟通方法就是利用简单的输入、输出指令,来传达彼此间需要的数据。

C 语言的输入函数具代表性的是“scanf()”,scanf()就是读取用户由键盘输入的数据,函数的语法如下:

```
scanf("%d%d", &变量 1, &变量 2);
```

scanf()函数将会读取键盘输入的数据,其中“%d”表示按某种格式(此例为整型格式)输入数据移入对应的变量中。用户在输入数据时,第一个输入的数据将会传给变量 1,按下回车键或空格键后再输入的数据会传给变量 2。具体的输入细节将在后面章节中谈到。

C 语言的输出函数具代表性的有“printf()”,printf()就是将结果显示在屏幕上,函数的语法如下:

```
printf("%d%d",变量 1,变量 2);
```

表示按某种格式(此例为整型格式)将变量 1、变量 2 的值依次显示在屏幕上。除了变量外,还可以直接输出字符串,也可以通过转义字符“\n”来控制跳行的操作。例如:

```
printf("My age is: ");
```

```
printf("%d\n",age);
```

第一行的语句即是直接在屏幕上输出字符串“My age is:”;第二行的语句则紧接在字符串后输出变量“age”的值,并且由于“\n”光标会显示在下一行。

1.4.3 预处理程序

所谓的预处理程序(Preprocessor),就是在编译程序之前,对程序所做的预处理。例如,我们在编写程序时,会在 main 函数之前,利用“#include”语句引入标题文件到程序中。这些标题文件包含了许多函数的定义,例如之前提到的 scanf()或 printf()函数的定义,就包含在“stdio.h”这个标题文件中,所以,在使用 C 语言的输入函数 scanf()或输出函数 printf()之前,必须先引入“stdio.h”文件(由于 scanf()和 printf()两大函数在 TC 环境中默认,故在程序中可省略“#include <stdio.h>”或“#include “stdio.h””)。

可是这个“#include”语句并不是一个可以执行的程序语句,所以在程序编译之前,必须通过一个中介介质,将这一类编译器无法直接编译的语句,转换成编译器可以编译执行的内容,这个介质就是预处理程序。

预处理程序遇到“#include”语句,会找到所引入的标题文件,并将该文件的内容直接引入到程序内,如果遇到“#define”语句,则会产生一个常量定义。

1.4.4 程序的注释

程序的注释在程序的编写过程中是非常重要的工作。一个没有注释的程序就如同一个没有使用说明书的软件。没有使用说明书,则软件难以让人了解它的功能和使用原则;如果程序少了注释,即使程序设计师本人,日后也难以在密密麻麻的程序代码中理解程序的执行流程和结果。

C 语言的程序注释方式如下:

```
/* 注释内容 */
```

在“/*”和“*/”之间的文字就是程序的注释。但是在“/*”之前和“*/”之后的文字则会被编译器当成程序代码来执行。例如:

```
scanf("%d",&age); /* 输入年龄 */  
printf("%d",age);
```

一般的程序注释都会出现在函数之前或语句结束之后,这样的注释方式让源程序代码看起来更为简捷,自然也较易理解:

```
/* 这是计算标准体重的函数 */  
int is_fat(int height,int weight)  
{  
    return answer; /* 返回一个整型的变量值 */  
}
```

编写程序时应该养成随时加上注释的好习惯,除了可以在程序中快速找到相关的函数,还可以帮助日后追踪错误。但凡事适可而止,程序注释应该“行于所当行,止于所当止”,不适合大篇幅的注释内容。

1.4.5 编写第一个程序

所谓万丈高楼平地起,做大事之前,必先成就小事,打好相关基础才行。看完了之前窗口环境的介绍,以及 C 的程序流程与结构,相信一定有许多人正在摩拳擦掌地准备写出自己的第一个程序,所以咱们来小试身手一番,本书第一个问世的程序则是程序语言中最经典的例子——Hello 程序。

```
/* The first programming: HELLO WORLD */  
#include<stdio.h>  
main()  
{  
    printf("Hello World! \n");  
}
```

在 VC++ 环境下编译运行后,其结果如图 1.12 所示。

注意:由本例可见,C 程序的每行前没有行号,但本书所有程序实例都加了行号,这是 C 程序不允许的。我们加上行号只是为了方便说明程序。