

数据结构与C语言 高级程序设计

赵逢禹 罗道昆 路 玲 杜光耀 编著

北京航空航天大学出版社

数据结构与C语言高级程序设计

北京航空航天大学出版社

TP312

TP312

乙32

数据结构与 C 语言高级程序设计

赵逢禹 罗道昆

编著

路 玲 杜光耀

北京航空航天大学出版社

内 容 简 介

本书系统地阐述了数据的逻辑关系以及它们的存储表示和运算操作,包括线性表、栈、队列、树、图及分类等内容,并用C语言详细地给出了各种算法。为使没有学习过C语言的读者能顺利地学习数据结构及其高级程序设计方法,本书用了一定的篇幅详细介绍了与数据结构实现及算法描述有关的C语言基础知识及编程技术。把C语言、数据结构及其高级编程技术有机地结合在一起,既注重理论介绍,又兼顾实际应用能力的培养。本书可作为高等院校非计算机专业的高年级学生在学习完一门高级语言、且具有一定计算机操作能力之后,学习数据结构及高级程序设计的教材。上机实习可安排24学时至30学时,讲授学时为60~70。

本书同时可作为从事企业管理及其他有关人员自学C语言和数据结构的参考书。

图书在版编目(CIP)数据

数据结构与C语言高级程序设计/赵逢禹等编著. —北京:
北京航空航天大学出版社,1998.4

ISBN 7-81012-745-4

I. 数… II. 赵… III. ①数据结构②C语言-程序设计
IV. TP312

中国版本图书馆CIP数据核字(97)第26252号

数据结构与C语言高级程序设计

赵逢禹 罗道昆 编著

路 玲 杜光耀

责任编辑 娄铁军 齐桂森

责任校对 张韵秋

北京航空航天大学出版社出版发行

北京市学院路37号(100083) 发行部电话 62015720

各地书店经销

铁十八局印刷厂印装

开本:787×1092 1/16 印张:19 字数:486千字

1998年4月第1版 1999年2月第2次印刷 印数5001~7000册

ISBN 7-81012-745-4/TP·264 定价:21.90元

前言

随着计算机在社会活动各个领域的广泛应用,对计算机工作者以及各行各业的管理人员操作与使用计算机的能力提出了更高的要求。

为了提高高校非计算机专业学生的计算机知识水平与操作能力,以适应不断深入与普及的计算机应用,许多高校的教学计划中除开设计算机基础、高级语言和数据库系统之外,还开设了数据结构课程。但由于教学计划要求及学时限制,该类课程的内容大都是从适合于计算机本科的课程中抽取出部分内容。因而,课程系统性较差,内容分散,难以达到预期效果。

传统数据结构教材围绕线性表、树、图的逻辑结构,物理存贮和算法设计展开论述,而高级语言的教材则大多数侧重于语言本身的介绍。因而学生学习完程序设计语言与数据结构课程之后,在遇到较复杂的程序设计问题时,仍无从下手。造成这种现象的原因之一就是高级语言与数据结构的的教学各自独立,对学生应用数据结构的知识进行较复杂程序设计的能力缺乏培养。本教材试图把C语言与数据结构的知识有机结合起来,通过典型的例子,说明如何用C语言进行数据存贮设计、算法设计,从而培养学生利用所学的数据结构的知识解决实际问题的能力。

本教材是在原讲义的基础上经认真讨论与修改而成的。为了使没有学习过C语言的读者也能顺利阅读本书,前六章介绍了C语言的基础知识,在内容上,既保持C语言的完整性,又不拘泥于C语言的所有细节,对与学习数据结构及复杂程序设计紧密相关的内容进行详细的介绍。从第七章到第十二章,系统介绍了数据结构的逻辑表示、C语言实现,包括线性表、栈、队、树和图的逻辑表示及C语言实现的常用算法,并通过精心设计的实例,说明他们如何用于C语言高级程序设计。在第十三章,用两个典型例子分别介绍了链表技术和图的邻接表、队、栈等技术在编程中的应用。第十四章介绍了Turbo C的集成开发环境。在附表中,给出常用的C语言库函数。

本教材的第二章、第四章、第九章和第十章由罗道昆编写;路玲编写了第六章、第十一章和第十二章;杜光耀编写了第三章、第八章、第十四章及附表。赵逢禹负责编写了其余部分,并组织了对全书的修改与统校。

编者要感谢郑州航院计算机教研室对该书的出版提供的支持,感谢李印清、李文华二位教师对该书的修改提出了宝贵的意见,在本书即将出版之际,作者借此机会向本书中所借鉴、引用的参考文献的所有作者和帮助完成本教材的所有同志深深致谢。

由于作者水平所限,书中可能存在错误和结构与内容上的不当之处,恳切希望得到本书读者特别是讲授此书的教师们的批评指正。

编 者

1997年7月

目 录

第一章 C语言概述	1
§ 1.1 C语言的起源与特点	1
§ 1.1.1 C语言的起源	1
§ 1.1.2 C语言的特点	1
§ 1.2 C语言程序的格式	3
§ 1.3 Turbo C2.0 上机初步	5
§ 1.3.1 准备程序	5
§ 1.3.2 启动 Turbo C 进入集成开发环境	6
§ 1.3.3 程序运行	10
第二章 C语言表达式	12
§ 2.1 数据类型	12
§ 2.2 常 量	13
§ 2.3 变 量	14
§ 2.3.1 整型变量	15
§ 2.3.2 实型变量	16
§ 2.3.3 字符变量	16
§ 2.4 运算符和表达式	16
§ 2.4.1 算术运算符和算术表达式	17
§ 2.4.2 关系运算符及关系表达式	19
§ 2.4.3 逻辑运算符及逻辑表达式	20
§ 2.4.4 赋值运算符	22
习 题	23
第三章 程序控制语句	24
§ 3.1 if 语句	24
§ 3.1.1 简单的 if 语句	24
§ 3.1.2 if-else 语句	25
§ 3.1.3 else if 语句	26
§ 3.2 switch 语句	28
§ 3.3 for 循环语句	32

§ 3.4 while 语句	34
§ 3.5 do while 语句	35
习 题	39
第四章 函 数	41
§ 4.1 函数定义的一般形式	41
§ 4.2 函数参数与函数的返回值	43
§ 4.2.1 形式参数和实际参数	43
§ 4.2.2 函数的返回值	44
§ 4.3 函数调用的一般形式	44
§ 4.3.1 函数调用的一般形式	44
§ 4.3.2 函数调用的方式	44
§ 4.3.3 库函数的调用	47
§ 4.4 函数的嵌套调用	50
§ 4.5 局部变量与全局变量	51
§ 4.5.1 局部变量	51
§ 4.5.2 全局变量	52
习 题	54
第五章 指针和数组	57
§ 5.1 指针变量与指针表达式	57
§ 5.2 数 组	59
§ 5.2.1 一维数组	59
§ 5.2.2 指向数组的指针	60
§ 5.2.3 二维数组	61
§ 5.3 字符串与字符串函数	62
§ 5.3.1 字符串输入	62
§ 5.3.2 字符串函数	63
§ 5.3.3 字符串数组	64
§ 5.4 数组初始化	66
§ 5.4.1 数组初始化的一般形式	66
§ 5.4.2 不定大小数组初始化	66
习 题	68
第六章 结构与动态内存分配	70
§ 6.1 结构类型和结构变量	70
§ 6.1.1 结构类型的定义	70

§ 6.1.2 结构变量的定义	71
§ 6.1.3 结构变量的初始化	72
§ 6.2 结构变量成员的引用	74
§ 6.3 结构数组	76
§ 6.3.1 结构数组的定义	76
§ 6.3.2 结构数组的初始化	77
§ 6.3.3 举 例	77
§ 6.4 结构指针	79
§ 6.4.1 结构指针的定义	79
§ 6.4.2 结构指针的使用	80
§ 6.4.3 举 例	81
§ 6.5 动态内存分配	83
§ 6.5.1 C 语言程序的存储空间的分配	83
§ 6.5.2 动态内存分配	84
§ 6.5.3 动态内存分配函数的使用	84
习 题	85
第七章 数据结构概述	87
§ 7.1 数据结构课程的形成与发展	88
§ 7.2 数据结构	88
§ 7.3 基本术语	90
习 题	92
第八章 线性表	93
§ 8.1 线性表的逻辑结构	93
§ 8.2 线性表的顺序存储结构	94
§ 8.3 线性表的链式存储结构	97
§ 8.3.1 线性链表	97
§ 8.3.2 循环链表	104
§ 8.3.3 双向链表	105
习 题	113
第九章 栈和队列	116
§ 9.1 栈	116
§ 9.1.1 栈的定义和基本操作	116
§ 9.1.2 栈的表示和实现	116
§ 9.1.3 链 栈	120

§ 9.1.4 栈的应用	122
§ 9.2 栈与递归	123
§ 9.3 队 列	126
§ 9.3.1 队列的定义和基本运算	126
§ 9.3.2 队列的顺序存储结构	126
§ 9.3.3 链队列	129
习 题	132
第十章 树	133
§ 10.1 树的基本概念和术语	133
§ 10.2 树的存储结构	134
§ 10.3 二叉树	136
§ 10.3.1 二叉树的定义和性质	136
§ 10.3.2 二叉树的存储结构	139
§ 10.3.3 二叉树的建立	140
§ 10.4 遍历二叉树	142
§ 10.4.1 遍历二叉树的定义及递归算法	142
§ 10.4.2 遍历二叉树的非递归算法	143
§ 10.5 二叉树的具体操作	146
§ 10.5.1 排序二叉树	146
§ 10.5.2 二叉树的插入操作	148
§ 10.5.3 二叉树中元素的查找	150
§ 10.5.4 从二叉树中删除一个成员	150
§ 10.5.5 遍历一棵二叉树	153
§ 10.6 哈夫曼树及哈夫曼编码	155
§ 10.6.1 最优二叉树(哈夫曼树)	155
§ 10.6.2 哈夫曼编码	159
§ 10.7 二叉树的应用实例	161
习 题	168
第十一章 图	169
§ 11.1 图的概念	169
§ 11.2 图的存储表示	171
§ 11.2.1 图的邻接矩阵表示	171
§ 11.2.2 图的邻接表表示	174
§ 11.3 图的遍历	177
§ 11.3.1 图的深度优先遍历	177

§ 11.3.2 图的宽度优先遍历	179
§ 11.4 生成树与最小生成树.....	181
§ 11.4.1 基本概念	181
§ 11.4.2 prim 算法的基本思想	181
§ 11.4.3 prim 算法的 C 语言实现	184
§ 11.5 关键路径和最短路径.....	186
§ 11.5.1 拓扑排序	186
§ 11.5.2 关键路径	192
§ 11.5.3 最短路径	194
习 题.....	197
第十二章 分类与检索.....	199
§ 12.1 基本概念.....	199
§ 12.2 选择排序.....	200
§ 12.3 插入排序.....	202
§ 12.3.1 直接插入排序	202
§ 12.3.2 二分法插入排序	203
§ 12.4 快速排序.....	204
§ 12.5 检 索.....	207
习 题.....	209
第十三章 数据结构应用举例.....	210
§ 13.1 计划网络.....	210
§ 13.1.1 网络计划的方法	210
§ 13.1.2 关键路径及时间参数计算	212
§ 13.1.3 计划网络程序的主要数据结构	213
§ 13.1.4 计划网络程序的主要函数实现	216
§ 13.1.5 计划网络源程序清单	220
§ 13.2 用链表技术实现下拉式菜单.....	237
§ 13.2.1 问题提出	237
§ 13.2.2 软件中使用的标准函数	240
§ 13.2.3 下拉式菜单软件包函数	244
§ 13.2.4 下拉式菜单源程序	247
§ 13.2.5 下拉式菜单程序的总结	258
第十四章 Turbo C 2.0 集成开发环境	261
§ 14.1 Turbo C 2.0 的安装	261

§ 14.1.1 用安装工具 INSTALL 在硬盘安装 TC	261
§ 14.1.2 手工安装 TC	262
§ 14.2 Turbo C 2.0 集成开发环境简介	263
§ 14.3 C 程序的编辑和运行	264
§ 14.3.1 源程序的建立和编辑	264
§ 14.3.2 C 程序的编译和运行	268
§ 14.4 C 程序的调试	273
§ 14.5 设置 TC 工作环境	274
§ 14.6 运行环境	279
§ 14.7 调试技巧	280

附 录

Turbo C 2.0 主要库函数的格式和功能	284
-------------------------------	-----

第一章 C 语言概述

本章首先对 C 语言的产生与发展过程作一个简单介绍,讨论了 C 语言的特点及使用 C 语言时应注意的问题,最后介绍了适应于微机的 Turbo C 2.0 集成开发环境。

§ 1.1 C 语言的起源与特点

§ 1.1.1 C 语言的起源

C 语言是由 Dennis Ritchie 设计并在 DEC PDP-11 上的 UNIX 操作系统环境下实现的。实际上 C 语言编译程序、UNIX 操作系统及 UNIX 应用程序均由 C 语言书写。C 语言是在 1970 年由 Ken Thompson 设计的 B 语言发展而来。

C 语言是一种通用的程序设计语言。很多年来,C 语言的标准一直由 UNIX 操作系统的第五版所支持。然而 C 语言并没有被束缚在某种机器或某种操作系统上。随着微机普及与大众化,微机上使用的各种 C 语言的实用版本纷纷出现。在不同的版本上,C 语言源程序保持了奇迹般的兼容性。另一方面,由于没有一个标准。它们彼此之间也有一些差异。为了改变这些情况,美国国家标准局 ANSI 成立了一个委员会,在 1983 年夏初建立了一个 C 语言的标准 ANSI C。目前微机上使用的大部分 C 语言的编译器都是基于 ANSI C 标准设计的。

目前微机上常用的 C 语言有多种:Turbo C、Microsoft C、C++。Turbo C 与 Microsoft C 都是基于 ANSI C 标准的编译程序。C 语言的所有特点都是相同的。Turbo C 带有集成开发环境及丰富的图形功能,使得编程既容易又方便;Microsoft C 虽然使用起来不如 Turbo C 方便,但通常产生的代码比相等量的 Turbo C 代码运行的快一些。C++语言是 ANSI C 的超集,因而,C 语言的知识仍然适用。该语言是由贝尔实验室的 Bjarne Stroustrup 于 1983 年设计的,目的是为了帮助管理、编码和维护大型软件系统。C++对于 C 语言在三个方面进行了扩充,分别是类别、函数复用和操作符复用,同时增加了面向对象的程序设计特征。因而 C++是面向对象的程序设计的主流语言。

§ 1.1.2 C 语言的特点

一、C 语言是一种中级程序设计语言

把 C 语言称作中级程序设计语言,并不指它的能力弱与难于使用,也不是说它比 BASIC 或 PASCAL 等高级语言功能差。它不过是把高级语言的特征同汇编语言的能力结合起来形成所谓的中级语言。表 1.1 列出了有关语言的分类以及 C 在这些语言中的位置。

所有的高级语言都支持数据类型这个概念,所谓数据类型就是一组值和定义在其上的一组操作。普通的数据类型包括整型、字符型和实型。尽管 C 语言支持五种基本类型,但它不像 PASCAL 或 FORTRAN 中对类型要求那么严格。C 语言几乎允许所有类型互相转换。C 语言不提供程序运行时数组越界或变量类型不匹配等错误检查。这些检查是程序员设计程序时的任务之一。

表 1.1 C 语言在计算机语言中的位置

高级语言	中级语言	低级语言
Ada	C	Assemble
Modula-2	FORTH	
PASCAL		
COBOL		
FORTRAN		
BASIC		

C 语言允许对位、字节和指针直接操作,这一特性很适合于系统程序设计。C 语言的另一个特征是仅有 32 个关键字(ANSI 规定),而微机上 BASIC 语言中有 159 个关键字。

二、C 语言是面向程序员的语言

并不是所有的计算机语言都是面向程序员的。COBOL 与 BASIC 就是面向非程序员的。COBOL 语言的设计目标是为了改善程序的可读性,对程序的执行速度、编程效率不关心。BASIC 语言最初的目的是为初学者编写简单程序而设计的。

C 语言完全是为程序员建立和使用的语言。C 语言提供了程序员想要的一切功能:丰富的函数、很少的关键字,程序的执行效率可与汇编语言程序相媲美,同时还保留了 Modula-2 等语言的结构化特性,因而,C 语言是一种很通用的程序设计语言。

最初的 C 语言是为编写系统程序设计的。所谓系统程序是指操作系统及其支持程序,主要包括:操作系统、解释系统、编译程序、汇编程序、DBMS 等。由于 C 语言允许每个程序员建立与维护自己的库函数,以适应自己的编程风格并应用于不同的程序设计中,使软件设计具有较好继承性,从而提高编程效率。C 语言的编译程序可以产生非常紧凑的目标代码,该目标代码可达到汇编程序执行效率,而且 C 语言有汇编语言无法相比的结构性与可移植性,又没有 PASCAL 和 Modula-2 中的限制,因而受到程序员的喜爱。

三、C 语言是一种结构化语言

结构化语言一个显著的性质是代码和数据分离。这种语言允许为完成特定任务的数据和指令段同程序的其他部分分隔开。所采用的技术就是用局部变量和子程序,编写对程序其他部分没有副作用的子程序。

在结构化语言中,goto 语句的使用是被禁止或不提倡的,因为 goto 语句会破坏程序的结构性,使程序难以阅读与维护。结构化语言支持多种循环结构,如:while、do ... while 和 for 语句。另外,结构化语言允许采用缩进语句书写方法,没有严格的语句位置要求,而 FORTRAN 与 COBOL 则要求严格的语句位置。

结构化语言比非结构化语言更易于编写程序,程序的结构清晰,使程序阅读与维护方便。下面给出了几个结构化与非结构化语言:

非结构化语言	结构化语言
FORTRAN	PASCAL
COBOL	C
BASIC	Ada
	True BASIC

C 语言的主要结构成分就是函数。在 C 语言中,函数被描述成一个功能块,功能块内所有

程序动作的作用域均限制在其内部。该语言允许把程序的每个任务的定义与编程独立实现,使程序易于模块化。一旦函数建立,就可以在程序的任何地方使用它,只需知道该函数干什么就行了,不需要知道它怎么做,更不必担心会对程序的其他部分产生副作用。

C 语言中另一个结构是使用代码块。一个代码块是指一组程序语句在逻辑上可以当作一个单位来使用。C 语言中代码块是用 {} 把一组语句括起来构成的。例如:

```
for (i=0;i<0;i++)
{  scanf("%d",&x);
   printf("%d",x * x);
}
```

for 语句后面的 {} 内由两条语句构成了一个代码块,它们是一个逻辑单位,必须依次执行。C 语言中使用代码块可使算法描述更清晰。

§ 1.2 C 语言程序的格式

C 语言程序是由一个或多个函数组成的,每个函数都定义了实际完成的功能。C 语言中的函数类似于 PASCAL 中的过程或者 True BASIC 中的外部自定义函数和外部子程序。函数的名字是由编程人员定的,但 main() 是一个特殊的必不可少的函数,它在程序开始执行时第一个调用。实际上,main() 函数中包含了程序的轮廓。这种轮廓由一系列函数调用构成。不能把 main 当作变量名使用。

函数间的数据通讯方法是参数传递,函数名后面的括号把参数表括起来,{} 把构成函数的语句括起来,在调用函数时,只需写上函数名字与实际参数,实际参数与形式参数要一一对应,而且要用圆括号括起来。即使调用的函数是无参函数,也必须带圆括号。函数调用时没有 True BASIC 中那种 CALL 语句。

ANSI C 标准中定义了 32 个关键字,它们列在表 1.2 中,这些关键字不能用作一般标识符。前面标记有 (AN) 的是 ANSI C 对《Kernighan and Ritchie》C 的扩充。

表 1.2 ANSI C 与 K&R C 的关键字集

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

不同的 C 语言编译系统,为了更好地利用特定的硬件环境,支持混合语言程序设计与中断处理,关键字在 ANSI C 的基础上被扩充。Borland 公司的 Turbo C 是一种可在微机上运行的 C 语言编译系统,为了更好地利用微机的硬件环境,Turbo C 又扩充了一些关键字,表 1.3 中列出了其中一部分常用关键字。

表 1.3 Turbo C 2.0 部分扩充关键字

asm	cdecl	far	huge
interrupt	near	pascal	_cs
_ds	_es	_ss	

在 C 语言中小写字母与大写字母是不同的,所有关键字都是小写字母。例如,小写的 while 是关键字,大写的 WHILE 则不是。关键字在 C 语言程序中不能用作其他目的,不能用作变量或函数的名字。

C 语言程序的一般形式如表 1.4 所示,在这里 f1()到 fn()代表用户定义的函数。

表 1.4 C 语言程序的一般形式

```

globe declarations
return _type f1(parameter list)
{
    语句序列 1;
}
return _type f2(parameter list)
{
    语句序列 2;
}
.
.
.
return _type fn(parameter list)
{
    语句序列 n;
}
return _type main(parameter list)
{
    语句序列;
}
.
.
.
    
```

下面给出了一个计算 $S = \sum_{i=1}^n i$ 的 C 语言程序。在该程序中,n 的值是 main 函数执行到 scanf()时,从键盘输入的,然后调用 sum()函数计算。程序的开头 #include<stdio.h>表示编译时首先把 stdio.h 调入并当作程序的一部分,stdio.h 中包含了标准输入输出函数的函数原型(关于函数原型在第四章详细介绍)及常数定义。

```

#include<stdio.h>
int sum(int x)
{ int i,y;
    
```

```

    y=0;
    for (i=1;i<=x;i++)
        y=y+i;
    return(y);
}

main()
{
    int n;
    int s;
    scanf("%d",&n);
    s=sum(n);
    printf("%s %d","sum of 1+2+...+n is ",s);
}

```

§ 1.3 Turbo C 2.0 上机初步

Turbo C 2.0 是美国 Borland 公司的产品,是在其 V1.5 版本上发展而产生的较新版本。它为程序员提供了功能很强且相当方便的程序设计与调试环境,因而,得到广泛地使用。

Turbo C 2.0 可在 IBM PC 机及其兼容机上运行,DOS 版本要求在 2.0 以上,至少需要 448 KB 内存。

Turbo C 2.0 系统文件可以放在两张 1.2 MB 的高密软磁盘上或者 6 张 360 KB 的低密软盘上。表 1.5 给出了 6 张软盘上的 Turbo C 2.0 版系统文件的主要划分。

Turbo C 2.0 有两种独立的操作方式。第一种是集成开发环境,简称 IDE。利用集成开发环境,可以很容易地对程序进行编辑、编译和运行;第二种是传统的命令行方式,首先使用文本编辑器(如 WS、PE 等)编辑源程序文件,然后进行编译、连接、运行。本节只介绍 Turbo C 2.0 集成开发环境的简单使用方法,使初学者能够利用它调试简单程序,有关集成开发环境在第十四章详细介绍。

表 1.5 Turbo C 2.0 文件分类

盘 号	内 容
1#	安装与帮助文件
2#	集成开发环境文件
3#	实用工具,命令行编译、连接器
4#	库文件
5#	库文件、头文件
6#	例子程序、BGI 图形库

本节假定 Turbo C 2.0 已安装在计算机的 C 盘中,如果尚未安装,请按照第十四章的介绍安装 Turbo C 2.0。

§ 1.3.1 准备程序

在上机之前,首先要编制好源程序,准备好输入数据。例如,从键盘输入一整数 n,计算

$\sum_{i=1}^n i$, 该程序为:

```
#include<stdio.h>
main()
{ int i,n,s;
  s=0;
  scanf("%d",&n);
  for (i=1;i<=n;i++)
    s=s+i;
  printf("The sum is %d",s);
}
```

在这里 scanf 就是一个输入函数,执行该函数时,等待用户从键盘上输入一个整数值,存放在变量 n 所对应的单元中。而 printf 是一个输出函数,它把计算出来的值在屏幕上显示出来。程序中其他语句在后续章节中有详细介绍,在这里就不一一说明。

§ 1.3.2 启动 Turbo C 进入集成开发环境

启动 Turbo C 时,要先把当前目录设置到 tc.exe 文件所在的子目录下,然后在 DOS 提示符下键入:

C> tc (回车)

就能启动 Turbo C 进入集成开发环境。这时,屏幕上出现如图 1.1 所示的画面。屏幕画面由四个部分组成,从上至下依次为:

- 主菜单;
- 编辑窗口;
- 信息窗口;
- 状态行;

下面对以上四个部分作简单介绍。

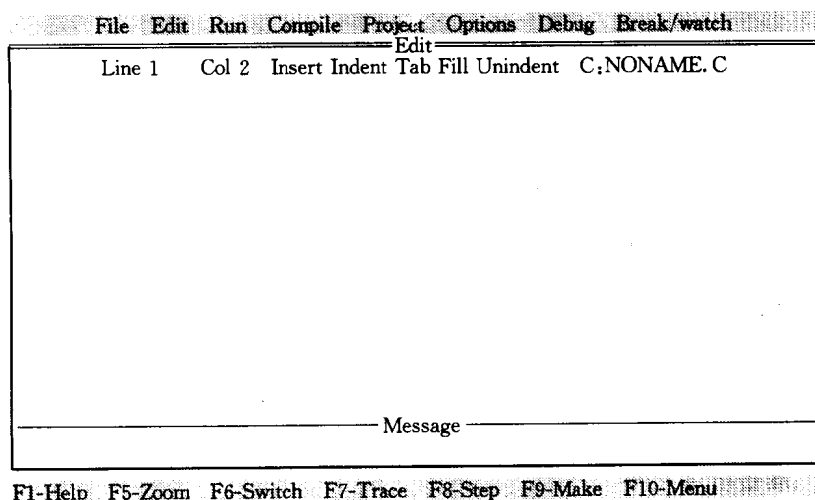


图 1.1 Turbo C 2.0 主菜单

一、主菜单

当进入 Turbo C 集成开发环境后,在屏幕的最顶行显示出一个菜单,该菜单显示了 IDE 的主要功能,表 1.6 给出了菜单选项的说明。

表 1.6 主菜单选择项

选 项	功 能
File	文件管理、目录操作、DOS 调用和退出 TC
Edit	建立和编辑源文件
Run	按各种方式运行程序
Compile	编译、连接生成目标文件和可执行文件
Project	多文件项目的程序管理
Options	TC 及其工具的工作环境设置
Debug	在线调试,监视或跟踪程序的运行过程
Break/Watch	设置、消除断点/在线监视和跟踪表达式

更为详细的介绍见第十四章。

主菜单中共有 8 个选择项。如果当前的状态是在执行 IDE 的某个处理(如正在编辑源程序),按 F10 键即可激活主菜单。当主菜单激活后,主菜单的某个选项以高亮度的光条显示。选择菜单中其他项有两种方法:第一种方法是使用光标移动键→,←左右移动光条到所要选择的项上,选定后按 ENTER 键;第二种方法可以直接按下 Alt+要选择的菜单项的第一个字母。例如,当要选择 Edit 项时,只要按下 Alt+E 键,就进入编辑状态。

当对主菜单的某个项进行了选择,屏幕上可能会弹出一个下拉菜单,它包含一个进一步的选择表。同样,既可以用光标移动键移动光条,也可以直接按选择项的某个字母来进行选择。

例如,在主菜单下把光标移到 File 选择项,然后按 ENTER 键,屏幕会弹出如图 1.2 的下拉菜单。

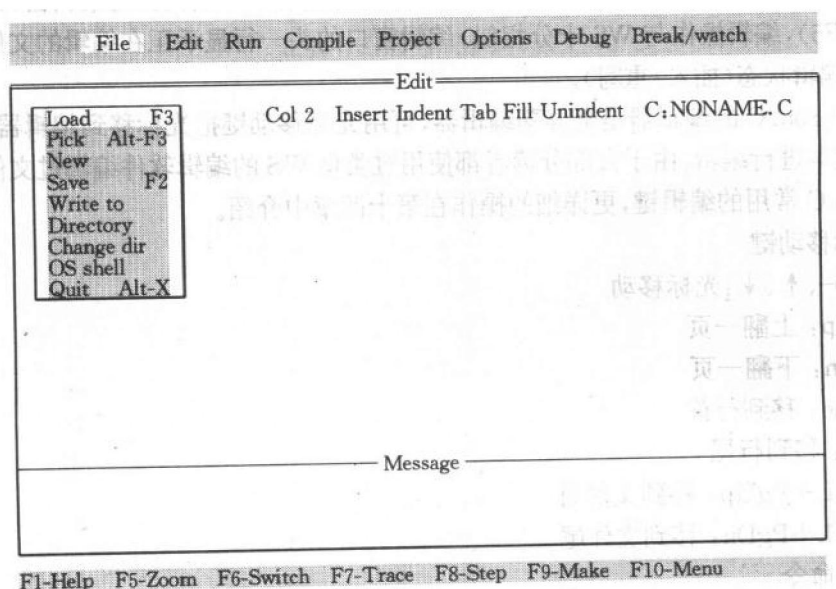


图 1.2 File 的下拉菜单