

高等学校适用教材



实用C程序设计教程

陈学德 陈玲 赵珠海 编著



机械工业出版社

高等学校适用教材

实用 C 程序设计教程

陈学德 陈 玲 赵珠海 编著



机械工业出版社

(京)新登字 054 号

本书在系统地讲述 ANSI 标准 C 的同时,溶进了作者多年使用 C 语言编程的经验和技巧。内容包括:C 语言的数据描述与基本操作;流程控制语句与结构程序设计;函数与模块化程序设计;数组;指针;结构、位段与共用体;位操作;文件系统;编程与调试技巧;基于 Turbo C 的高级应用等共十章。每章均含有一定量的习题和上机实验题目。全书重点突出,简明实用。

本书可供从事软件开发的工程技术人员、大专院校师生、培训班学员及业余爱好者学习使用。

实用 C 程序设计教程

陈学德 陈 玲 赵珠海 编著

*

责任编辑:王琳 版式设计:范兴国

封面设计:姚毅 责任校对:丁丽丽

*

机械工业出版社出版(北京阜成门外百万庄南街一号)

邮政编码:100037

(北京市书刊出版业营业许可证出字第 117 号)

北京市京建照排厂排版

北京市密云县印刷厂印刷

新华书店北京发行所发行·新华书店经售

*

开本 787×1092 1/16·印张 11·字数 275 千字

1994 年 9 月北京第 1 版·1994 年 9 月北京第 1 次印刷

印数 00 001-4 900·定价 13.50 元

*

ISBN 7-111-04345-6/TP·242(G)

前 言

在众多的程序设计语言中，C语言具有无可否认的优势，适用于各种各样的领域，已成为众多程序员的主要编程工具。掌握了C语言及其程序设计技术，不仅可直接用它来开发软件；而且，以此为基础，可以很快地掌握C++，使用面向对象的方法学。

目前，在高等学校的很多专业（不仅仅是计算机专业）都已开设了C语言或C程序设计之类的课程，许多由社会力量举办的培训班也讲授有关C语言的内容。在多年的C语言的教学实践中，我们感到需要编写一本适合的教材，并动手尝试这一工作。

我们编写此教材的指导思想是：利用我们多年从事C语言教学和利用C语言开发项目的经验和体会，简明系统而又重点突出地介绍标准C语言的内容，同时注意培养学员利用C语言进行程序设计的能力。

我们提倡讲授要重点突出，注重编程实践。为此，本书适当地结合了比较流行的C编译系统——Turbo C的使用；本书还设计一个“上机实验工程”，设计并实现一个“档案管理系统”。该工程使学员掌握自顶向下逐步求精的模块化程序设计方法，培养学员利用C语言开发具有一定规模程序的实践能力。因此，我们请学员从第一章开始即实施该工程，循序渐近，学完本书时，工程即随之完成。

编者感谢周良柱、彭照、胥雪村等同学对本教材手稿录入所做的工作，感谢沈阳工业大学计算机学院有关领导、软件教研室的老师和实验室的同志们对编写本书的支持，感谢为本书出版给予方便和支持的所有人。

殷切希望读者对本教材不足之处给予指正。

编 者

1994年2月

目 录

第1章 引 论

1.1 计算机语言与程序设计的概念	1
1.2 C语言概述	1
1.2.1 C语言的起源与发展	1
1.2.2 C语言是“中级”语言	2
1.2.3 C语言是结构化语言	2
1.2.4 C语言是程序员的语言	2
1.3 C程序的基本结构	2
1.3.1 关键字	2
1.3.2 C程序的一般形式	3
1.3.3 简单C程序举例	3
1.4 使用C语言上机的一般步骤	4
1.5 Turbo C的使用	5
1.5.1 准备Turbo C系统	5
1.5.2 Turbo C的启动	6
1.5.3 Turbo C集成开发环境选择项的设置	7
1.5.4 利用集成开发环境编辑、编译、链接和运行C程序	8
1.5.5 管理多源文件工程	10
1.5.6 利用集成开发环境调试C程序	10
1.5.7 Turbo C集成开发环境的常用热键	11
习题一	11
上机实验工程一	11
第2章 数据描述与基本操作	12
2.1 数据类型的概念	12
2.2 常量、变量及标识符	12
2.2.1 常量	12
2.2.2 变量	13
2.2.3 C语言的标识符	13
2.3 基本数据类型	13
2.3.1 整型	14
2.3.2 字符型	14
2.3.3 浮点型	15
2.3.4 无值类型及枚举类型	16
2.3.5 类型修饰符及基本数据类型的长度与值域	16
2.4 变量的初始化	17
2.5 赋值语句和表达式	17
2.6 运算符	17

2.6.1 算术运算符	17
2.6.2 关系和逻辑运算符	19
2.6.3 位操作符	19
2.6.4 特殊运算符	21
2.6.5 C语言运算符及其优先级总汇	23
2.7 表达式中的数据类型转换	23
2.7.1 自动类型转换	23
2.7.2 强制类型转换	24
2.8 数据的输入/输出	24
2.8.1 getch和 putchar 函数	25
2.8.2 scanf 和 printf 函数	26
习题二	29
上机实验工程二	30
第3章 流程控制语句	31
3.1 结构化程序设计	31
3.2 C语句概述	32
3.2.1 基本语句	32
3.2.2 复合语句	33
3.3 选择结构语句	34
3.3.1 if 语句	34
3.3.2 switch 语句	38
3.4 循环结构语句	40
3.4.1 while 语句	40
3.4.2 do-while 语句	42
3.4.3 for 语句	43
3.4.4 循环语句的嵌套	44
3.4.5 三种循环语句的比较	45
3.4.6 循环语句的变体	46
3.5 流程转向语句	47
3.5.1 break 语句	47
3.5.2 continue 语句	48
3.5.3 goto 语句	49
3.5.4 exit()函数	51
习题三	52
上机实验工程三	53
第4章 函数	54
4.1 模块化软件设计	54
4.2 用函数组装C程序	55
4.3 函数的定义与调用	57
4.3.1 函数的定义及其返回值	57
4.3.2 函数的调用	60
4.4 变量的作用域	62
4.4.1 局部变量	62

4.4.2 形式参数	63
4.4.3 全程变量	63
4.5 变量的存储类别	65
4.5.1 全程变量加 extern 修饰——外部变量	65
4.5.2 全程变量加 static 修饰——静态全程变量	66
4.5.3 局部变量加 auto 修饰——自动变量	66
4.5.4 局部变量加 static 修饰——静态局部变量	66
4.5.5 局部变量加 register 修饰——寄存器变量	67
4.6 递归函数	67
4.7 编译预处理	68
4.7.1 宏定义	69
4.7.2 文件包含	71
4.7.3 条件编译	72
习题四	74
上机实验工程四	74
第5章 数组	75
5.1 一维数组	75
5.1.1 一维数组的说明与引用	75
5.1.2 一维数组的初始化	76
5.1.3 一维数组的应用举例	77
5.2 二维与多维数组	79
5.2.1 多维数组的一般说明形式	79
5.2.2 多维数组的初始化	79
5.2.3 二维数组应用举例	80
5.3 字符数组与字符串	82
5.3.1 引言	82
5.3.2 字符串的存储与结束标志	82
5.3.3 字符串操作函数	83
5.3.4 字符数组和字符串的输入输出	85
5.4 数组作为函数参数	86
5.4.1 一维数组元素作为函数实参	86
5.4.2 数组名作为函数参数	87
习题五	88
上机实验工程五	88
第6章 指 针	89
6.1 指针就是地址	89
6.2 指针运算	90
6.2.1 指针赋值	90
6.2.2 指针专用运算符	90
6.2.3 指针的算术运算	91
6.2.4 指针的比较	92
6.3 指针与函数	92
6.3.1 指针作为函数参数	92

6.3.2 函数返回指针值	93
6.3.3 Void 型指针与动态内存分配函数	93
6.3.4 函数指针	95
6.4 指向指针的指针	97
6.5 指针与数组	98
6.5.1 指针与数组的紧密联系	98
6.5.2 通过指针引用数组元素	99
6.5.3 数组名作为函数参数	100
6.5.4 字符串与指针	102
6.5.5 指针数组	105
6.5.6 多维数组的指针	107
6.6 main 函数带参数——命令行变元	109
习题六	110
上机实验工程六	111
第7章 结构体	112
7.1 结构体类型的定义	112
7.2 结构体变量的说明和引用	113
7.2.1 结构体变量的说明	113
7.2.2 结构体变量的引用	114
7.2.3 结构体变量的初始化	117
7.3 结构体的指针	118
7.4 结构体数组	119
7.5 动态数据结构及其处理	120
7.5.1 链表	120
7.5.2 二叉树	124
7.6 共用体	127
7.7 位段	131
习题七	132
上机实验工程七	133
第8章 I/O 系统	134
8.1 C 语言 I/O 系统概述	134
8.1.1 流式文件	134
8.1.2 文本流与二进制流	134
8.1.3 缓冲文件系统和非缓冲文件系统	135
8.2 文件结构体与文件指针	135
8.3 文件的打开与关闭	136
8.3.1 文件打开函数——fopen()	136
8.3.2 文件关闭函数——fclose()	137
8.4 文件的定位与操作错误检测	137
8.4.1 ftell 函数	137
8.4.2 rewind 函数	138
8.4.3 fseek 函数	138
8.4.4 feof 函数	139

8.4.5 ferror 函数和 clearerr 函数	139
8.5 文件的读写	140
8.5.1 按字符读写	140
8.5.2 按字符串读写	143
8.5.3 按格式要求读写	144
8.5.4 按记录(块)读写	144
习题八	149
上机实验工程八	149
附录 A C 语言运算符汇总表	150
附录 B C 标准库函数	151
附录 C BIOS 调用说明	159
附录 D DOS 功能调用说明	163
参考文献	167

0000
11515

第1章 引 论

1.1 计算机语言与程序设计的概念

在人类生活和发展过程中,人类语言(如汉语、英语等)起着非常重要的作用,人们靠它来表达意愿,交流思想和抒发情感。也就是说,人类语言是人与人之间交流信息的工具。

计算机正在许多方面给人以很大的帮助。然而,人要想使用计算机,让计算机做某些所期望的工作,人与计算机之间必须交流信息。那么,交流信息的工具又是什么呢?就是计算机语言。

所谓的计算机语言就是为了描述计算机操作步骤而制定的一整套标记符号,表达格式,及使用的语法规则。当人们想利用计算机解决某个问题时,要先用计算机语言安排好处理步骤并存入计算机内,这包括:应先做什么,遇到什么情况应如何处理等等;计算机则按处理步骤不折不扣地完成人们所规定的工作。我们把用计算机语言描述的处理步骤称为计算机程序,简称程序;而将编制处理步骤(程序)的过程称为程序设计;从事程序设计工作的人员即称为程序员。计算机语言就是程序员与计算机之间进行信息交流的工具。

计算机语言通常被粗略地分为低级语言和高级语言两类。

低级语言通常是指机器语言、符号语言和汇编语言,其特点是,面向具体机器类型,给予程序员达到最大灵活性和最高效率的潜力。用低级语言编写程序具有算法刻画细致、程序紧凑、占用内存空间少、执行速度快以及可以直接使用计算机的全部资源等优点,但是存在着通用性差、程序编制与调试困难等缺点。低级语言通常用于开发计算机系统软件、或用于过程控制、实时控制等场合。

高级语言是为了更好地描述解题算法、为了程序设计者的方便而发展起来的通用型语言,如 BASIC, FORTRAN, COBOL, PASCAL 等。用高级语言编写的程序,其优点是容易表达算法,不依赖于具体的机器、通用性强;其缺点是计算机不能直接认识执行,须通过翻译变成机器语言(二进制)程序,且处理过程刻画不具体,运行速度低,对硬件设备进行操作较困难等等。高级语言多用于科学计算和事务处理等方面。

随着计算机及应用技术的发展,人们期望能有一种语言即具有高级语言的通用性、开发程序容易、结构化和模块化等优点,又兼有低级语言的运行效率高、紧凑和对硬件操作能力强等优点,以满足不同的使用要求。C 语言就是这样的高级程序设计语言。

1.2 C 语言概述

1.2.1 C 语言的起源与发展

C 语言是由贝尔实验室的 Dennis Ritchie 发明并首先在配备 UNIX 操作系统的 DEC PDP-11 计算机上实现的。C 语言是一个比较古老的语言 BCPL(目前在欧洲仍有应用)发展过程中的产物。BCPL 是由 Martin Richards 开发的,它影响了由 Ken Thompson 发明的 B 语言,而 B 语言导致了 C 语言在 70 年代的发展。

C 语言一直与 UNIX 操作系统有着某种渊源和联系,这表现在:C 和 UNIX 都是由贝尔实验室发明的;现行 UNIX 的绝大部分代码都是用 C 语言写的;C 促进了 UNIX 的移植实现的同时,本身得以迅速推广应用;在 ANSI 标准公布之前,UNIX 支持的 C 被公认为是 C 的“标准”。

美国国家标准局(ANSI)于 1983 年根据 C 语言问世以来各版本对 C 语言的发展和扩充,制定了一个 C 语言标准草案(83 ANSI C),87 年又公布了更进一步的新标准(87 ANSI 标准 C)。目前广泛流行的各种 C 语言编译系统都已经或正向 87 ANSI 标准 C 靠拢、兼容。

1.2.2 C 语言是“中级”语言

C 语言通常被称为“中级”语言。称 C 语言为中级语言没有丝毫贬义,而是赞誉它把高级语言的基本结构与低级语言的功能结合起来并兼有二者的优点,它是高级语言的佼佼者。

作为中级语言,C 语言支持高级语言的数据类型概念,但它不象 Pascal 和 Ada 语言那样有很强的类型区分,而是允许几乎所有的数据类型转换;C 语言具有高级语言容易移植的特点;C 语言允许对位、字节和地址进行操作,这通常是低级语言的特征;C 语言仅有 32 个关键字,非常简洁,但功能很强。

1.2.3 C 语言是结构化语言

尽管 C 语言不允许函数的嵌套定义,不是严格意义下的模块结构化(block-structured)语言,但是它还是常被称为结构化语言。这是因为 C 语言具有结构化语言的典型特征。

①代码与数据的分离这是结构化语言的显著特征,它使程序的各个部分除了必要的信息交流外彼此互不影响相互隔离。C 语言为了实现这种分离,提供了两种方法。一是复合语句,二是允许定义和调用带有局部变量的子程序(即 C 函数)。

②提供多种程序流程控制语句 C 语言支持多种循环结构,如 while,do-while 和 for;支持灵活的分支语句,如 if,switch,break 和 continue 等;提倡尽量少用 goto 语句。

1.2.4 C 语言是程序员的语言

“难道还有不属于程序员的语言吗?”,答案是完全肯定的。考虑一下两种非专业程序员的语言 COBOL 和 BASIC 就会明白了。COBOL 设计者的本意是打算使非程序员能读程序(这是不大现实的事),结果使得专业程序员也难以改善 COBOL 程序的可靠性、甚至不能提高程序的编写速度;而 BASIC 的主要目的则是允许非专业程序员在计算机上编程解决比较简单的问题,而专业程序员对 BASIC 却很不满意。

相比之下,C 语言可以说是独树一帜,它是由一批富有实际经验的专业程序员设计并在实际使用中不断完善和考验的。最终的结果是 C 语言所提供的恰好是程序员所需要的:很少限制、很少缺陷、模块结构、彼此独立的函数和十分紧凑的关键字集合。用 C 语言编程,程序员可以获得高效的机器代码,其效率几乎接近于汇编语言代码,同时又具有 ALGOL 和 MODULA-2 的结构形式;程序员还用 C 语言来代替汇编语言,在达到高效和灵活的同时,又取得了可移植性。这真令人惊奇!C 语言这样快就成为第一流专业程序员的最常用语言,也就毫不奇怪了。

随着 C 语言的普及,由于它的可移植性和效率,许多程序员开始用它设计各种类型的程序。也许 C 语言被用于所有各类程序设计的主要原因是由于程序员喜欢它。

1.3 C 程序的基本结构

1.3.1 关键字

表 1-1 列举了 C 语言所有的 32 个关键字,它们用来定义变量、类型和功能语句等,并与标

准 C 语法结合,形成程序设计 C 语言。

表 1-1 C 语言关键字表

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

C 语言中区分大写和小写,C 语言的关键字都是小写的。在 C 程序中,关键字不能用于其它目的(即不能作为变量或函数名使用)。

1.3.2 C 程序的一般形式

C 程序都是由一个或多个函数构成的;一个 C 程序中至少存在一个函数,即 main 函数,它是程序运行开始时被调用的第一个函数。

C 程序的一般形式如图 1-1 所示。其中 f1()到 fN()代表用户定义的函数。

```
全局说明
main()
{
    局部说明
    语句序列
}
f1()
{
    局部说明
    语句序列
}
f2()
{
    局部说明
    语句序列
}
fN()
{
    局部说明
    语句序列
}
```

图 1-1 C 语言程序的一般形式

1.3.3 简单 C 程序举例

这里,我们先举两个简单的例子,以便读者对 C 语言编程有一个初步的感性认识。

例 1-1 编一个 C 程序,在 CRT 屏幕上显示一个短句“I learn programming in C.”

```
main()
{
```

```
printf("I learn programming in C. \n");
}
```

这个程序的输出结果是：

```
I learn programming in c.
```

这个程序没有说明部分。`printf` 是 C 系统的一个函数，其作用就是在控制台上（终端上或显示器上）输出信息。其中的 `\n` 是一个转义字符，它告诉计算机系统执行一个特殊功能——换行，而其它被双引号括起来的字符均被原样输出。

例 1-2 编一程序求两个整数的和并输出。

```
main()
{
    int a,b,sum;
    a=45;
    b=55;
    sum=a+b;
    printf("The sum of %d and %d is %d\n",a,b,sum);
}
```

在这个程序当中使用了三个整型变量 `a`、`b` 和 `sum`，在 C 语言中所有变量都必须先进行说明而后才能引用。程序中出现的 `%d` 是一种格式命令，它告诉系统在该位置上要输出一个十进制整数，其值依次取后面给出的变量的值。

C 程序的每条语句均以分号结尾，分号是一条语句必要的组成部分。

关于 `printf` 函数、转义序列以及格式控制命令将在下章详细讨论。

1.4 使用 C 语言上机的一般步骤

前面已经提及高级语言是描述算法的，是面向程序设计人员的，它不能被计算机直接认识，需通过翻译变成二进制数“0”和“1”表示的机器语言程序才能执行。C 语言作为高级语言的佼佼者也不例外。使用 C 语言编写一个程序并使之正确执行，一般要经过编辑、编译、链接和执行四个过程。

“编辑”就是使用某种编辑工具（如 UNIX 系统下的行编辑程序 `ed` 或全屏幕编辑程序 `vi`；MS-DOS 系统下的行编辑程序 `EDLIN` 或全屏幕编辑程序 `wordstar`，以及 Turbo C 集成环境中的内含编辑器等等）将程序员设计的 C 语言源程序输入计算机，经修改认为无误后存入文件系统，形成源程序（源代码）文件 `*.C`。源程序（源代码）文件以 ASCII 码形式存储，不能被计算机直接执行。

“编译”就是使用某种特定的 C 编译系统（如 UNIX 系统支持的 `CC`；MS-DOS 支持的 `MSC` 和 Turbo C 等）将源代码文件 `*.C` 翻译成二进制形式的目标代码文件 `*.O`（UNIX 系统下）或 `*.OBJ`（MS-DOS 系统下）。编译系统所做的工作具体地包括词法分析、语法分析、中间代码生成、优化处理和目标代码生成等等。当一个源程序较大时，可将之分解成几个源代码文件来存储，并分别单独编译这些源代码文件，生成几个目标代码文件，再由链接程序将之链接在一起。

“链接”就是使用链接程序对编译生成的用户目标代码（存在一个或多个目标代码文件中）以及所调用的 C 库函数代码进行连接装配，生成可执行文件 `*.out`（UNIX 系统下）或 `*.exe`

(MS-DOS 系统下)。可执行文件就是能被计算机系统直接执行的文件。

“执行”就是运行得到的可执行文件,以获得执行结果。

以上四个过程的进行中,可能由于某种原因(如用户源程序设计错误等)而导致中止,这时,程序调试人员应分析错误所在及其原因,并编辑修改源代码文件后,重新进行编译、链接、执行过程。

图 1-2 说明了这种程序调试过程。其中实线表示操作流程、虚线表示各操作过程的输入输出。

1.5 Turbo C 的使用

Turbo C 是目前 MS-DOS 系统上最好的 C 编译器之一,深受初学 C 语言者喜爱。它的主要特点是:①拥有完好的集成开发环境,很好地支持程序员调试程序的全过程,易学易用;②编译速度奇快,且允许编译、链接、运行连续进行;③工程项目管理功能较好地支持大程序开发;④Turbo C 支持 ANSI C 标准和传统 C 的功能,并结合 PC 机系统做了许多扩充,是 MS-DOS 环境下的较好的程序开发工具。所以,我们推荐初学者使用 Turbo C。以下简称 Turbo C 2.0 的使用。

1.5.1 准备 Turbo C 系统

若你的计算机系统配有硬盘,可将 Turbo C 系统盘(Turbo C 2.0 一般为六片软盘)的所有内容利用安装程序 INSTALL(在 #1 盘中)全部装入硬盘中。在安装时,INSTALL 将要求用户回答设置相应的子目录,主要包括:头文件目录、库文件目录、输出目录以及 Turbo C 目录,对此,若你的硬盘为 C 盘,一般分别设置成:C:\TC\INCLUDE, C:\TC\LIB, C:\TC 和 C:\TC 即可。

若你的计算机系统无硬盘,但有高密度软盘驱动器,则可进行简化安装形成一张软盘的 Turbo C 系统,方法是:把 #1 盘中的 TC.EXE 和 TCHELP.TCH(若不需要帮助信息,可不拷贝该帮助文件)文件拷贝到你的软盘之根目录;然后在根目录下建立一个 INCLUDE 子目录,并将所有头文件 *.h 拷贝到该子目录下;再在根目录下建立一个 LIB 子目录,并将 C0?.OBJ, C?.

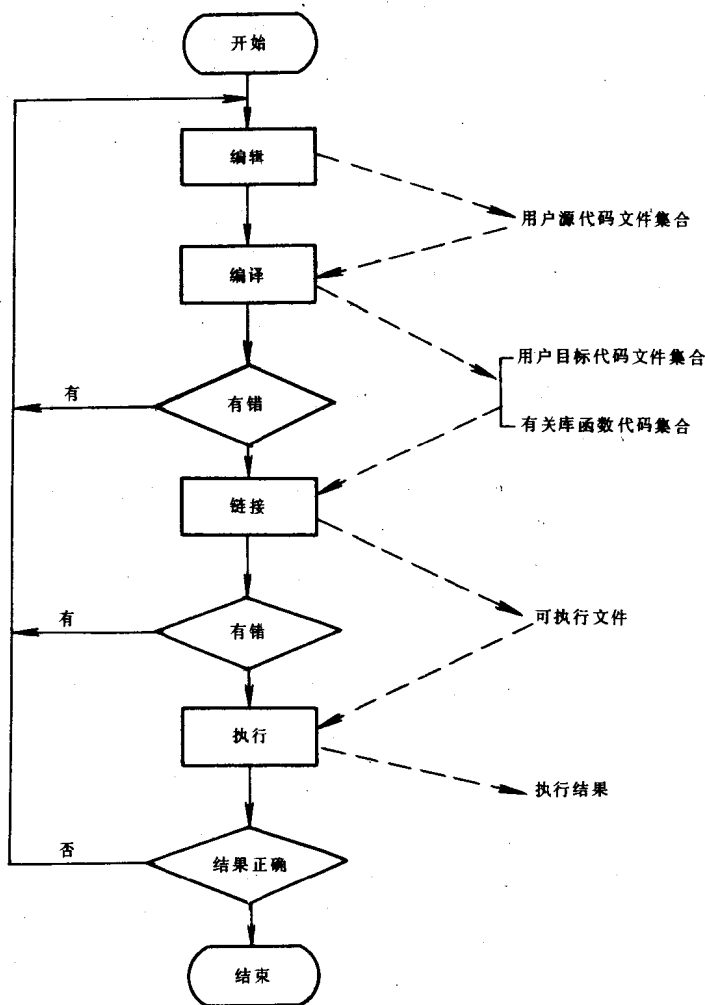


图 1-2 C 程序的上机调试过程

LIB, EMU. LIB, FP87. LIB, MATH?. LIB 拷贝到该子目录中, 其中的? 统一用 T(对应 Tiny 模式), S(对应 Small 模式), M(对应 Medium 模式), C(对应 Compact 模式), L(对应 Large 模式) 或 H(对应 Huge 模式) 之一来代替, 即只拷贝固定选用的一种编译存储模式所需的库文件。

Turbo C 可以按六种不同的内存管理模式(Model), 来编译程序, 程序员可根据需要来选择其一。这六种模式分别称为: Tiny, Small, Medium, Compact, Large 和 Huge, 其差别如下:

Tiny Model(微模式)代码、数据和堆栈都限制在同一个 64K 段内, 所有段地址寄存器均设置为同一个值, 所有的寻址都使用 16 位来进行。因此, 将产生运行速度最快, 所占内存最小的可执行文件。按这种模式编译生成的可执行文件可用 DOS 命令 EXE2BIN 转换为 .COM 文件。

Small Model(小模式)小模式是 Turbo C 的缺省模式, 适用于多种编程任务。它使得程序代码单独占一个段, 数据、堆栈则另占一个段; 即段寄存器 CS 的值与 DS, ES 和 SS 的值是不同的, 但所有寻址仍使用 16 位偏移进行。可见, 小模式编译的程序可占 128 K, 分为代码段和数据段。用小模式编译产生的文件之执行速度接近于微模式。

Medium Model(中模式)代码不再被限制在一个段内, 需要用 20 位寻址; 数据段、堆栈段和附加段被限制在它们各自的段内, 并使用 16 位寻址。这种模式适用于数据量小的大程序, 其数据调用速度未降低, 但代码执行速度要慢得多。

Compact Model(紧凑模式)代码被限制在一个段内, 用 16 位寻址; 但数据段可以占多个段, 用 20 位寻址。因此, 代码执行速度与小模式相同, 但调用数据时较慢, 这适用于代码较短但使用大量数据的程序。

Large Model(大模式)代码段和数据段均可多于一个段, 但其中的静态数据不能超过一个段(即不能多于 64K)。可用于处理大量数据的大程序, 所产生可执行文件的速度大大慢于上述几种模式。

Huge Model(巨模式)除静态数据不被限制在 64K 之内以外, 与大模式基本相同。产生的文件运行速度更慢。

1.5.2 Turbo C 的启动

在你的 Turbo C 系统所在的盘符(如 C:)下, 执行下列命令:

```
C>CD TC<CR>
```

```
C>TC<CR>
```

即可启动 Turbo C 系统, 进入其集成开发环境, 主屏幕显示如图 1-3 所示。

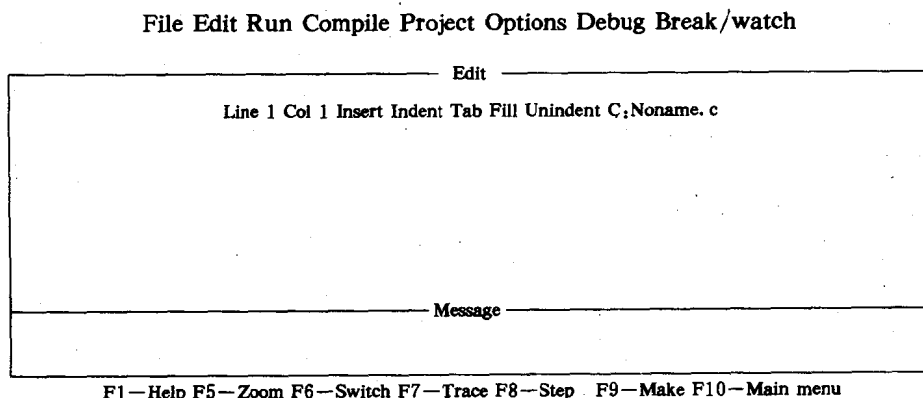


图 1-3 Turbo C 集成环境主屏幕

主屏幕从上到下可分为四部分：①主菜单；②编辑程序的状态行和编辑窗；③编译程序的信息窗；④“热键”快速参考行。

主菜单用来设置环境选择项，或者用来指示 Turbo C 去执行某个任务。若当前正处于主菜单命令选择状态，此时主菜单中某一项上有高亮度光条，则选择执行主菜单中的命令可有以下途径：

- ①使用箭头键将高亮度光条移到你所要选择的主菜单项目上，并按一下 Enter 键；
- ②直接键入你所要的主菜单项目的第一个字母(大小写均可)。

若当前状态不是主菜单选择状态(主菜单上无高亮光条)，则可通过如下方式进行主菜单命令选择：

- ①先按 F10 热键退出编辑状态，进入主菜单选择状态，再按上面两种方式之一来选择主菜单项目；
- ②按“Alt—主菜单项目的第一个字母”，直接进入主菜单项目下的子菜单(“AltE”进入编辑状态，即使光标处于编辑窗)。

主菜单选择项的功能如表 1-1 所示，这里只是其概要，其下一级的常用子项的细节在下面分类介绍。

表 1-1 主菜单选择项功能概要

项 目	功 能 概 要
File	装入或保存文件，管理目录，调入 DOS 和退出 Turbo C
Edit	调用 Turbo C 的编辑程序
Run	编译，链接和运行装入环境下的当前程序
Compile	编译在环境下的当前程序
Project	管理多文件工程
Options	设置编译程序、连接程序和环境的各种选择项
Debug	设置各种调试选择项
Break/watch	设置执行断点和观察变量

1.5.3 Turbo C 集成开发环境选择项的设置

主菜单项目 Options 下子菜单选项及其选项之子项的内容概要如表 1-2 所示。

表 1-2 Options 选项的内容概要

子 项	子子项	概 要
Compi ler	Model	存储模式，共六种选择，缺省选择为 Small
	Defines	用于定义临时预处理符号以支持条件编译
	Code generation	用于设置编译生成的代码与硬件的关系
	Optimization	用于选择编译的优化策略
	Source	用于确定标识符中有效字符的个数，注释是否可嵌套以及与 ANSI 标准的关系
	Errors	让你决定如何显示编译过程中的出错信息
	Names	进行段命名

(续)

子 项	子子项	概 要
Linker	Map file	是否产生映像文件;缺省值为 Off
	Init. segments	是否要链接程序去初始化段;缺省为 Off
	Default libraries	链接由其它 C 编译系统编译生成的模块时才使用;缺省为 Off
	Graphics libraries	是否要求图形库的存在;缺省为 On
	Warn dupl. symbols	是否警告全程标准符的重复;缺省时为 On
	Stack warnings	是否警告“没有指定栈”(在与汇编语言程序链接时可能出现);缺省为 On
	Case—sensitive	大小写敏感连接(与 FORTRAN 模块连接时用到;缺省为 On
Enviro— ment	Message tracking	跟踪错误的范围,缺省为当前文件
	Keep messages	错误信息的保持范围,缺省为不保持
	Config auto save	是否自动存储对 Config 文件的修改; 缺省为 Off
	Edit auto save	是否自动存储对源程序文件的修改; 缺省为 Off
	Backup files	是否创建 *.bak 文件;缺省为 On
	Tab size	定义 Tab 键对应的空格数;缺省为 8
	Zoomed window	使活动窗口扩大到整个屏幕;缺省为 Off
	Screen size	定义一屏幕容纳的行数,缺省为标准 25 行
Direc— tories (此处指 明的目 录应与 实际安 装一致)	Include dir.	指明包含头文件的目录
	Library dir.	指明包含目标文件和库文件的目录
	Output dir.	指明编译系统生成的 .OBJ, .EXE 和 .Map 文件的存贮目录
	Turbo C dir.	指明配置文件,帮助文件和缺省处理文件 所在的目录
	Pick file name	指明要缺省处理的文件名
Argument		精确地给出命令行参数
Save options		在配置文件中存贮所有可选的当前选择
Retrieve options		装入一个由 Save options 命令存贮的配置 文件,重起 Turbo C 系统

1.5.4 利用集成开发环境编辑、编译、链接和运行 C 程序

(1) 编辑源程序

选择主菜单中的 File 选项,即出现一个 File 子菜单.其中常用的子项主要有:

- Load (其功能是装入或建立一个 .C 源程序文件并进入编辑状态)
- Save (其功能是将环境中的当前文件按当前路径名存盘)
- Write to (其功能是将环境中的当前文件按新路径名存盘)
- Os Shell (其功能是调入 DOS,允许程序员执行 DOS 的各种命令;
在 DOS 下键入 EXIT 并回车又可回到 Turbo C 当前状态)
- Quit (其功能是退出 Turbo C 系统)

进入编辑状态(光标出现在编辑窗)后,程序员即可键入或修改源程序,其编辑命令与