第一章　C 语言和 C 程序概述

1.1　C语言由来

C 语言是国际上公认的最重要的少数几种通用程序设计语言之一，在计算机领域中甚至把能否熟练掌握 C 语言程序设计作为区分职业的还是业余的程序员的一个重要标志。

C 语言的开发历史源于高级语言和 UNIX 操作系统。众所周知，早期的系统程序设计使用的是汇编语言。这主要是因为汇编语言最能够体现计算机硬件指令级的特性，其表达能力足以描述系统设计的各个方面的特性，且由汇编语言程序生成的代码有较高的质量，使得大系统运行时的资源开销能够为计算机所承担。但是，汇编语言有它本身不能克服的缺点，即程序的可读性、可移植性以及问题的可描述性远不如高级语言。

能否推出一种语言，它既向下靠拢汇编语言，使用户能够接近硬件，且生成的代码有较高的质量；又向上靠拢高级语言，使得用这种语言编写的程序具有较好的可读性、可移植性以及问题的可描述性呢？世界上许多杰出的计算机科学家对这一问题作了尝试，其中，美国贝尔实验室的里奇（Dennis M. Ritchie）和汤普森（Ken L. Thompson）尤为出色。

汤普森于 1970 年在 PDP-7 机上实现了 B 语言，即为 B 语言编写了解释程序，并用 B语言编写了第一个 UNIX操作系统。1971 年汤普森又在 PDP-11/20 机上实现了 B语言，并用 B 语言编写了 UNIX 操作系统。B 语言的主要思想源于英国剑桥大学理查特（Richards 教授的 BCPL 语言（1967年），而 BCPL 语言又是基于理查特的 CPL 语言（Compined Programming Language）（1963 年）发展而来的，它们都是 Algol 语言（1960 年）的分支，是 Algol语言的改进，使之接近计算机的硬件，并逐步精练而成的新语言。但 B 语言采用字编址，不能适应 PDP-11 机用字节编址进行存取的要求；B语言还缺少具有一定表达能力的数据类型；而且 B编译生成的是解释性的执行代码，执行速度慢。这些都使 B 语言的实际使用有一定困难。

里奇和汤普森一起从事 UNIX 操作系统的开发工作，里奇很欣赏汤普森为 B 语言开发的解释程序。在 B语言的基础上，里奇开发了 C语言。里奇设计 C 语言时追求的目标是：保持 BCPL 语言和 B 语言较精练和接近计算机硬件的优点，恢复这些

语言所失去的通用性，使得 C 语言真正同它的宿主 UNIX 一样，具有简洁、高效、灵活和移植性好的特点。第一个 C 语言编译程序在 1972 年投入使用，并在 1973 年用 C 语言改写了 UNIX 系统，加进了多道程序功能，使 UNIX 发生了本质的变化，特别是把整个系统（包括 C 语言编译本身）都建立在 C 语言的基础上，使 C 语言具有良好的可移植性。从 70 年代中期开始， UNIX 系统以及它所支持的 C 语言在贝尔实验室内部和大学中得到了普遍使用，特别是在 1978 年 Prentice-Hall Inc 出版由 Brian W. Kernighan 与 Dennis M. Ritchie 发表的世界上第一本有关 C 语言的专著‘ The C Programming Language ”后，C 语言的发展进入一个崭新的阶段。

UNIX 的成功和它的威力使 UNIX 成为公认的第一个标准的操作系统，几十万个系统在运行 UNIX，使 C 语言被迅速推广。由于 C 语言本身的特点，使得在各种操作系统下都有 C 语言的编译程序。

基于 DOS 平台的 C 语言编译系统很多，例如：广泛流行的 Borland 公司的 Borland C 和 Turbo C，Microsoft 公司的 Microsoft C 以及 Watcom C 等等，它们的高版本都主要面向 C++。这些编译系统各有特色，有的还有着较强的针对性，如 Watcom C 主要用于保护模式下的编程，想要突破 1M 内存限制而又不想调用 XMS、EMS 功能的程序员可以使用它进行编程。

目前 C 编译器的平台渐渐由 DOS 转向 Win 9x（Win 95、Win 98、Win NT 以及即将推出的 Win 2000）在 Win 9x 平台上,通用的 C 语言编译环境有 Borland 公司的 C++ Builder、Microsoft 公司的 Visual C++ 以及 Watcom C++32 等，由于 Win 9x 的内核和 API 调用遵循 C 语言规范，C（++)语言编程在 Win 9x 下显示出无可比拟的高效性和可由用户设计风格的自主性。

1.2 C 语言的主要特点

如前所述，C 语言的开发源于高级语言和 UNIX 操作系统的发展要求，由于 UNIX 操作系统的广泛使用，使得 C 语言首先被用在安装有 UNIX 操作系统的计算机上开发各种应用程序。但是如果 C 语言本身不具备某些不同一般的优点，C 语言就不能成为当今最流行的程序设计语言。

1.移植性好，多种操作系统都支持 C 语言

C 语言本身不针对具体的机器和操作系统，而且 C 语言相当精练，使得其编译系统可以做得很紧凑，因而无论在价格低廉的个人电脑上的、还是在功能强大的巨型机上的多种操作系统都可以接受、支持 C 语言。也就是说 C 语言具有很好的可移植性，

特别是当 C语言被成功地移植到 DOS 操作系统后，更得到了计算机用户的青睐。

2.应用面广，特别适用于编写大的系统软件

用 C 语言编写大软件有两个原因：

(1)C 语言移植性好，使得用 C 语言编写的应用程序移植性也好；

(2)C 语言提供了指针和地址操作的能力，提供了对位进行操作的能力，具有很好的硬件控制能力，使得 C语言可以代替汇编语言用于编写操作系统一类包含设备驱动程序的大型软件。例如已用 C语言编写 UNIX、CP/M 等操作系统。

上述的 C 语言可以在不同类型的计算机上实现移植和适用于各种应用领域的能力说明 C 语言是一种通用性极为良好的程序设计语言。

3运算符丰富（见附录 2）

C 语言的运算符共有 34 个，除包括一般高级语言中的算术、逻辑、关系运算符外，还有自增和自减运算符、位逻辑运算符，且把各种有意义的操作，例如：括号、赋值、强制类型转换都作为运算符处理。这 34 个运算符还被划分成 15 个运算优先等级并具有左、右结合性，故使得表达式的形式非常灵活，各种运算的实现简捷方便。但要注意，初学者对某些运算符却不容易正确运用。

4.程序设计自由度大

C 语言接近于汇编语言，对变量类型、变量范围和存储空间的存取制约少，因此 C 语言的程序设计能自由地进行，这是好处。但是，对于初学者而言，若使用其他高级语言，只要严格遵守语言的语法规则，就能够编写出正确的程序；而由于 C语言制约少，程序设计的自由度大，更由于 C 语言的运算符丰富和表达式形式非常灵活，编写出正确的程序相对就难，错误也难以检查。这意味着，同其他高级语言相比， C 语言具有明显的专业性特征。

5.表示方式简洁实用

(1)语言本身简洁

C 语言和 Pascal 语言都是由 Algol 语言发展而来的，它们是一对双胞胎，但 C 语言要比 Pascal 简洁。例如：

C 语言	Pascal 语言	
{ }	begin end	复合语句
if(e) s;	if(e) then s	条件语句
int i;	var i:integer	定义整型变量
int f();	function f():integer	说明返回整型值的函数

int a[10];	var a:array[1..10] of integer	定义整型的一维数组
int *p;	var p:^integer	定义指向对象是整型的指针变量

(2) 表达式中能包含赋值语句，例如：

```
while((c=getchar())!=EOF)    putchar(c);
```

这里 getchar() 是取键盘输入字符的函数，c=getchar()是赋值语句，把函数的返回值赋于字符变量 c；而 putchar(c) 是把字符变量的内容送显示器输出。

(3) 变量更新的表示方式简洁，例如 i=i+2 可以表示为 i+=2。

(4) 变量的自增和自减表示方式简洁，若 i=5，则

i++ 使用变量 i 的当前值后，再使变量 i 自增 1；语句 a=i++;执行后，a=5，i=6。

++i 先使变量 i 自增 1，再使用变量 i 的值；语句 a=++i;执行后，a=6，i=6。

i-- 使用变量 i 的当前值后，再使变量 i 自减 1；语句 a=i--;执行后，a=5，i=4。

--i 先使变量 i 自减 1，再使用变量 i 的值；语句 a=--i;执行后，a=4，i=4。

(5) 保留的关键字少（见附录 1）。

各种 C 处理系统一致使用的关键字总共只有 32 个。

6.丰富的数据类型

C 语言提供给用户直接可以使用的数据类型有整型、实型、字符型和枚举型等基本的数据类型，还有数组类型、指针类型、结构体类型和共用体类型等导出性数据类型及空类型。如此丰富的数据类型给 C 程序设计带来很多方便。

7.具有预处理功能

宏定义和宏替换允许程序员以简短的字符串替代较长的字符串，减少了程序的输入量；程序员以说明用途的标识符定义常量，增强了程序的可读性。

文件包含允许程序员在程序的某处插入另一个 C源文件，以达到对某些通用的 C 源文件的共享，提高程序设计的效率。

条件编译允许程序员按操作环境指定编译程序编译源程序中的某些段或跳过源程序中的某些段，以使该源程序可以在不同类型的计算机上实现编译、连接和运行，增强了源程序的可移植性。

8.是结构化的理想语言

C 语言简练，仅包括 32 个关键字，许多广泛使用的基本操作都由函数的调用来实现。每种 C 编译程序都配有包含许多能实现各种功能的函数的函数库，供用户编写应用程序时调用，因此容易实现程序的模块化设计。而 32 个不多的关键字却包括了足以控制程序执行流向的控制语句（if—elseif—else、while、do—while、for、switch—case—default ），使程序的编制可以非常标准化。

9.生成的代码质量高

程序执行的效率高，仅比用汇编语言编写的程序所生成的目标代码效率低 10%～20%.

1.3 C程序概貌

用 C语言所编写的程序称为 C语言源程序,简称 C 程序。C 程序一般由一个或多个函数组成，这些函数既可以集中放在一个文件中，也可以分散放在几个文件中，每个 C 语言源程序的文件都必须以 .C 作为文件的扩展名，以说明该文件是 C 语言编写的源程序文件。

【例 1.1】完成二个整数四则运算的 C 程序。该程序由文件名为 arith1.c 和 arith2.c 的二个文件组成。

文件 arith1.c 的内容如下：

```
/* 主函数 */
main()
{   int a,b                          /* 定义二个名为 a 和 b的整型变量 */
    printf("请键入二个整数\n")       /* 屏幕显示输入二个整数的提示信息 */
    scanf("%d%d",&a,&b)              /* 接收键入的二个整数并存入变量 a 和 b */
    add(a,b)                         /* 调用函数 add 处理变量 a 和 b 中的内容 */
    sub(a,b)                         /* 调用函数 sub处理变量 a和 b 中的内容 */
    mul(a,b)                         /* 调用函数 mul 处理变量 a和 b中的内容 */
    div(a,b)                         /* 调用函数 div 处理变量 a 和 b 中的内容 */
}
/* 对二个整型数做加法并输出显示结果 */
add(int x,int y)                     /* 二个整型的形式参数 x 和 y */
 {  int z;                           /* 定义一个名为 z的整型变量 */
    z=x+y;                           /* 将 x 和 y 的内容相加并把结果存入 z */
    printf("%d+%d=%d\n",x,y,z); /* 输出显示 */
}
/* 对二个整型数做减法并输出显示结果 */
sub(int x,int y)
{   int z;
```

```
    z=x-y;                          /* x 和 y 的内容相减并把结果存入 z */
    printf("%d-%d=%d\n",x,y,z);
}
```

文件 arith2.c 的内容如下：

```
/* 对二个整型数做乘法并输出结果显示 */
mul(int x,int y)
{   int z;
    z=x*y;                          /* x 和 y 的内容相乘并把结果存入 z */
    printf("%d*%d=%d\n",x,y,z);
}
/* 对二个整型数做除法并输出结果显示 */
div(int x,int y)
{   int z;
    z=x/y;                          /* x 和 y 的内容相除并把结果存入 z */
    printf("%d/%d=%d\n",x,y,z);
}
```

说明：

（1）该源程序中包括五个函数，其函数名分别是 main、add、sub、mul 和 div，其中名为 main、add 和 sub 的函数在文件 arith1.c 中，名为 mul 和 div 的函数在文件 arith2.c 中。函数名是标识符（Identifier），标识符是程序员用于区分 C 程序中用到的各个函数、变量、符号常量、语句标号、宏名、用户自定义的数据类型名等等而取的名称，它是由以英文大小写字符或下划线开头后跟若干个英文大小写字符、数字字符和下划线的字符串。为了说明标识符是函数名，必须在此标识符后跟一对圆括号 ()，例如标识符 main 后跟一对圆括号()，因此 main 是函数名；标识符 add 后跟一对圆括号()，因此 add 也是函数名。与一般函数名不同的是，main 是一个特殊的函数名。C 语言规定：一个完整的 C 语言程序中只允许有一个函数的函数名使用 main，且必须有一个名为 main 的函数。main 是主函数的约定词，它告诉 C 编译和连接程序在生成可执行程序后，程序的执行从名为 main 的函数开始，在执行完 main 函数时返回系统。

（2）以/*开头、*/结束之间的内容是注释，本例中使用以/*开头、*/结束的注释说明各个函数能实现的功能，也可对某些语句使用以/*开头、*/结束的注释说明语句实现的功能。注释在编译时被忽略，即不产生任何代码，它仅仅是帮助人们理解程序。

（3）本例中成对出现的花括号{ }是函数的界限符，其中左花括号{ 相当于 pascal 语言中的 BEGIN，标志该函数的语句从此开始，而右花括号 }相当于 pascal 语言中的 END，表示该函数的语句到此结束，{ }之间称为函数体。在以后的程序

中我们会看到这成对出现的花括号{ }还可以用作复合语句的界限符。必须注意，{ 和 }必须成对出现，否则编译时会报告错误。

（4）本例的 main 函数中函数体的第一行是 int a,b; ，它被称之为变量定义语句。在该语句中出现了二类单词符号，属于关键字的 int 和属于标识符的 a 和 b。关键字（ Keyword）是 C 语言中为特定目的而保留的单词，关键字 int 用于指明整型数据。C 语言规定对程序中用到的变量都需要指明变量的数据类型和变量的存储类别，而变量定义语句就是用于指明程序中用到的变量是哪种数据类型以及以哪种存储类别存储。

C 语言中规定变量定义语句的语法是：

[存储类别] 数据类型变量名,变量名,……,变量名；

这里存储类别用一对 [ ]括起来，是指 C 语言中把指定存储类别的几个关键字中最常使用的关键字作为可以缺省的存储类别，因此在变量定义语句中存储类别可以缺省。

按 C 语言的语法规定，语句 int a,b; 是变量定义语句。在此，标识符 a 和 b 是变量名，所定义的二个变量 a 和 b 的数据类型是由数据类型关键字 int 所指明的整型数据，而存储类别是缺省的存储类别。

一般把变量定义语句安排在函数体的开头，我们可以在本例各个函数的函数体开头看到有类似于 int a,b;的变量定义语句。

注意：在同一语句中定义的同类型变量之间用逗号分隔，而语句应该用 ';'结束；在 C 语言中，分号是语句的不可缺少的组成部分。

（5）在 main 函数的函数体中第二行是 printf("请键入二个整数\n");，它被称之为函数调用语句。在该语句中标识符 printf 后跟一对圆括号 (),故此标识符是函数名，是被调用函数的函数名。包括在一对圆括号（）中的"请键入二个整数\n"称为实际参数，程序员要求在本次调用该 printf 函数后能在屏幕上显示：

请键入二个整数

类似于" printf("请键入二个整数\n");", main 函数第三行" scanf("%d%d",&a,&b);"也是函数调用语句。在调用该函数的语句中，包括在一对圆括号（）中的实际参数有三个，彼此之间以逗号间隔，程序员要求在本次调用该 scanf 函数时能接收键盘输入的二个十进制整数，并存放在变量 a 和 b 中。

C 语言中规定函数调用语句的语法是：

函数名(实际参数,实际参数,……,实际参数);

在函数中，常包括一些函数调用语句，被调用的函数或者是同函数调用语句处于同一个源文件中的函数，如本例的 main 函数中语句" add(a,b); "调用名为 add 的函数、语句" sub(a,b); "调用名为 sub 的函数，名为 add 和 sub 的函数同调用它们的语句处于同一个源文件 arith1.c 中,或者处于另一个源文件中,本例的 main 函数中语句" mul(a,b); "

和“ div(a,b); ”调用源文件 arith2.c 中名为 mul 和 div 的函数，被调用的函数同调用它们的语句不是处于同一个源文件中；或者被调用的函数是 C 版本提供的函数库中的函数，还可以是用户的函数库中的函数，在本例的名为 add、sub、mul 和 div 函数中都有一个调用 printf 函数的语句“ printf("%d/%d=%d\n",x,y,z);”，printf 函数就是 C 版本提供给用户使用的库函数中的函数，它是格式输出函数，能够把要求输出的信息送到屏幕上。名为 scanf 的函数也是 C 版本提供给用户使用的库函数中的函数，它是格式输入函数，能够按实际参数指定的方式接收键盘输入的数据并存放到实际参数指定的变量中。

（6）在本例中，函数 main 的定义形式可以归纳为：

```
main()
{ 函 数 体 }
```

类似地，函数 add 的定义形式可以归纳为：

```
add(int x,int y)
{ 函 数 体 }
```

二者的区别是在本程序中调用 main 函数时不需要实际参数，而编制函数 add 的目的是用于实现两个整数的相加，调用 add 函数时要能够让该函数得到是对哪两个整数做加法。本例的 main 函数中语句“ add(a,b); ”要求函数 add 对变量 a 和 b 中的内容（整数值）相加，这两个参与加法操作的值必须在 add 函数执行前传递给 add 函数，在 add函数的定义中应该有接收存放两个参与加法操作的值的变量，这两个变量就是 add 函数定义中跟在函数名后面的一对圆括号（）中的变量 x 和 y。为了同函数调用语句中放在一对圆括号（）中的实际参数相区别，函数定义中跟在函数名后面的一对圆括号（）中的变量被称为形式参数。类似于函数 add 的具有形式参数的函数称为有参函数，反之称为无参函数。

C 语言中规定函数定义的语法是：

```
[存储类别] [数据类型] 函数名（数据类型参数名，……，数据类型参数名）
{   函数体   }
```

在本例中，函数 add 定义时使用缺省的存储类别和数据类型（其意义将在后面介绍),它有两个数据类型由关键字 int 指明的名为 x 和 y 的形式参数。

需要指出的是，即使是无参函数，在函数定义时函数名后的一对圆括号（）也不能少，正是这一对圆括号（）指明其前面的标识符是函数名。

（7）类似于“ z=x+y; ”的语句都是赋值语句，赋值语句使用“ = ”作为运算符，把右边表达式的值赋于左边的变量，在该语句中指明要完成的操作是把变量 x 的内容同变量 y 的内容相加后的结果送入变量 z保存。

（8）该程序运行时屏幕先输出：

```
请键入二个整数
```

若用户键入 653 ↙ -17 ↙，则屏幕继续输出：

```
653
-17
653+-17=636
653--17=670
653*-17=-11101
653/-17=-38
```

其中前两行是在执行函数 scanf 时用户键入数据在屏幕上的回显，后四行是程序分别执行函数 add、sub、mul和 div 的结果输出。

1.4 C程序的建立、编译、连接及执行

C语言是一种计算机程序设计语言，程序员按照 C语言规定的语法和词法用文本编辑程序（编辑器）生成的 C源程序（包括一个或若干个扩展名为 .c 的文件）还不能够直接上机运行，因为计算机不能直接理解这些属于文本文件的.c 文件，更不能执行。因此必须有一个专门的程序，其任务是把计算机不能理解的 .c 文件翻译成计算机能够识别的机器语言程序（称为目标程序），这就是 C 编译程序（编译器）。C 编译程序首先按 C语言规定的语法规则和词法规则检查需要编译的 .c 文件，若没有发现错误，则生成计算机能够识别的目标程序（.obj文件）；否则将把在 .c 文件中发现的错误报告给程序员，以让程序员作修正。目标程序还不是可运行的程序，它必须经过连接程序（连接器）处理。为了加快程序的开发，一个规模较大的程序往往使用多种语言分块编制，然后使用各自的编译程序生成多个目标程序，连接程序将这些目标程序连接成一个可执行文件（.exe文件），若在连接过程中发现错误，则同样把错误报告给程序员。

由源程序到生成一个可执行文件往往要多次经过编辑、编译和连接三个环节才能排除由编译程序和连接程序发现的错误，但是在上机运行可执行文件时还可能发现运行的结果同程序设计预期得到的结果不一致，这时必须对错误结果仔细分析，找出错误原因，以修改源程序，这就是程序调试。

如上所述，开发一个 C程序需要经过用编辑器生成 C源程序、用编译器生成目标程序、用连接器生成可执行文件以及上机调试通过生成的可执行文件四个过程。Borland 公司为 C程序的开发作出了巨大的贡献，其中之一是把编辑器、编译器、连接器和调试器的功能集成在一个名为 TC.EXE 的程序中，为 C程序员提供了一个 Turbo C集成开发环境，为 C程序的开发带来极大的方便。

1.4.1 启动 Turbo C

运行 TC.EXE 程序，屏幕上呈现 Turbo C 的初始窗口如图 1.1 所示。

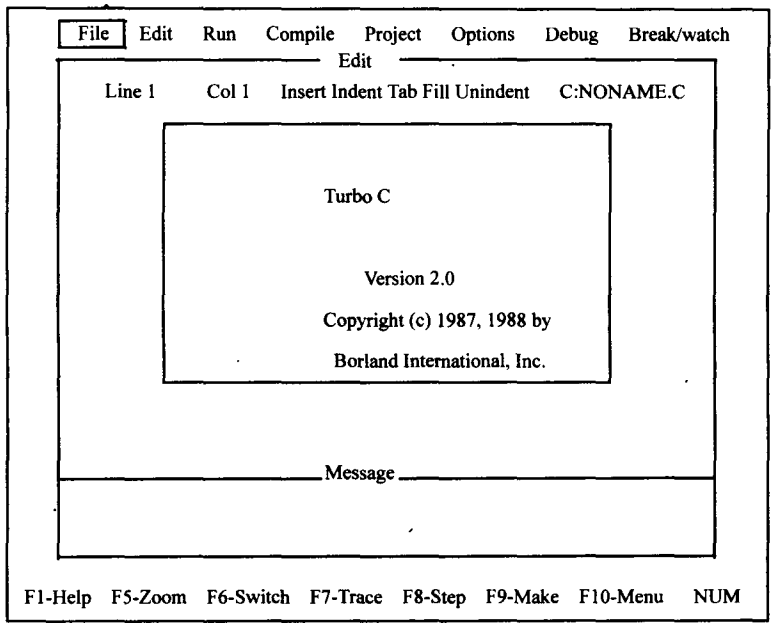

图 1.1 Turbo C初始窗口

初始窗口的最上行为主菜单行，此时选择框在 File。最下行为常用的热键功能提示，其中 F1 是求助热键，在 TC环境下按下 F1 键，TC将反馈一个帮助窗口，程序员可以在该窗口中选择帮助条目，学习 Turbo C 集成软件的使用。热键功能提示行的上方是在 TC 程序执行过程中提供给用户的信息显示区，例如编译中发现的错误会显示在该区域中。初始窗口的中央提示 Turbo C 的版权，按任意键将取消该提示，此时该区域将作为编辑区。选择主菜单某个选项的二种常规方法是：按热键 F10 将使主菜单的某个选项以高亮度显示，再按光标左右移动键，当选择框移到指定操作的选项时，按回车；或者在按热键 F10 后，再按指定操作选项的第一个字母。

1.4.2 C文件的建立

从本质上讲 C 文件与一般文本文件无异，只不过 C 文件以 “c ”为文件后缀，因此 C 文件可以用多种编辑器建立。在 Dos 下可以用 edit 文本编辑程序建立 C文件、甚至可以直接用 Dos 的“ copy con… ”命令建立 C 文件。

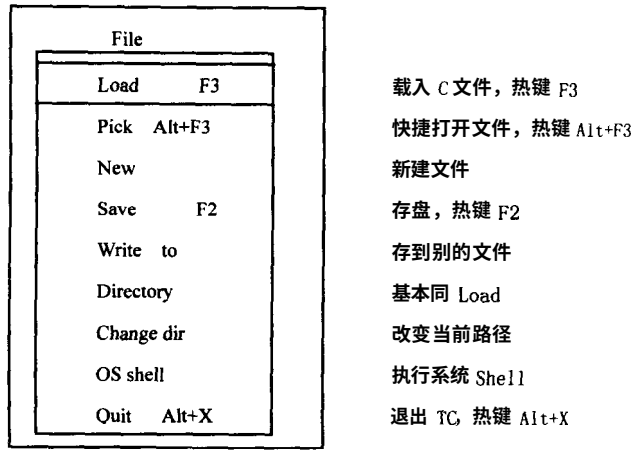

图 1.2　文件子菜单

Turbo C 主菜单的第一个选项是 File，该选项被选中时的下拉子菜单中主要包括从磁盘装入一个已存在的文件、建立一个新文件以及把编辑的文本存盘保存的选项，有些选项在选中时还以对话框方式让用户提供必要的信息。选项 File 的下拉子菜单如图 1.2 所示，使用光标上下移动键选择指定操作的选项。

1. Load

对话框提示用户键入要从磁盘装入文件的名字，扩展名缺省为 .C 文件，若文件存在，则装入文件；否则作为待编辑的新文件名字。 .在这两种情况下都会首先询问用户是否保存当前编辑的文本，假如当前编辑的文本作过修改且尚未存盘的话。键入文件名允许使用 DOS 的通配符 * 和 ?，此时将出现一个列表框，供用户选择列表框中的文件。还允许键入目录名，此时也将出现一个列表框，列出该目录下的文件和子目录名，供用户选择文件或进入选中的下级子目录。

2. Pick

在 TC 程序执行期间从磁盘装入过的文件的名字和新编辑过的文件的名字都被登记，在 Pick 选项被选中时，这些被登记的文件名被放在列表框内供用户选择文件。

3. New

使用缺省文件名 NONAME.C 作为待编辑文本的临时名，在存盘时再确定文件的名字。

4. Save

在编辑文本时可随时按热键 F2 或进入该选项把编辑的文本存盘保存。

5. Write to

对话框提示用户键入要把编辑的文本写入磁盘文件的名字，扩展名缺省为.C 文件，若文件存在，则会提问是否覆盖该文件。

6. Change dir

对话框提示当前目录并等待用户键入目录名作为新当前目录。

7. OS shell

暂时离开 Turbo C, 返回 DOS 命令行提示符，以执行 DOS 命令或其他程序。键入 EXIT 将重返 Turbo C。

8. Quit

在编辑文本时可随时按热键 ALT+X 或进入该选项退出 Turbo C,返回 DOS。若编辑的文本被修改过且尚未存盘，则会提问是否需要保存。

1.4.3 C 文件的编辑

Turbo C 集成开发环境中的编辑器有些像 Dos 下的 edit，只不过 TC 环境下具有更多的快捷热键，使用更方便，而且，由于 TC 是编辑器、C 编译器、连接器和调试器的集成系统，编程人员可以编辑后立即编译，非常快捷和方便。

进入编辑窗口还可以使用热键 ALT+E ，按热键 ALT+E 将无条件进入编辑窗口。Turbo C 的编辑窗口如图 1.3 所示。

编辑窗口的编辑区顶部是编辑状态行，用户可在该行获得当前编辑文本的有关信息，其中：

Line 5—— 当前光标在编辑文本的行位置，如图所示为第 5 行；

Col 1——当前光标在编辑文本的列位置，如图所示为第 1 列；

Insert——指示插入或替换状态，按 Insert 键或 Ctrl+V 将在两种状态间实现切换。当 Insert 隐藏时，键盘输入的文本将替换当前光标处的字符；否则，键盘输入的文本将插在当前光标处。

Indent——指示处于自动缩紧状态，即按回车结束本行时，下一行的起始列位置定位在上一行的第一个非空白字符的列，上下两行左对齐。当 Indent 隐藏时，下一行

的起始列位置定位在第 1 列。键 Ctrl+O+I 或 Ctrl+Q+I 用于实现这两种状态间的切换。

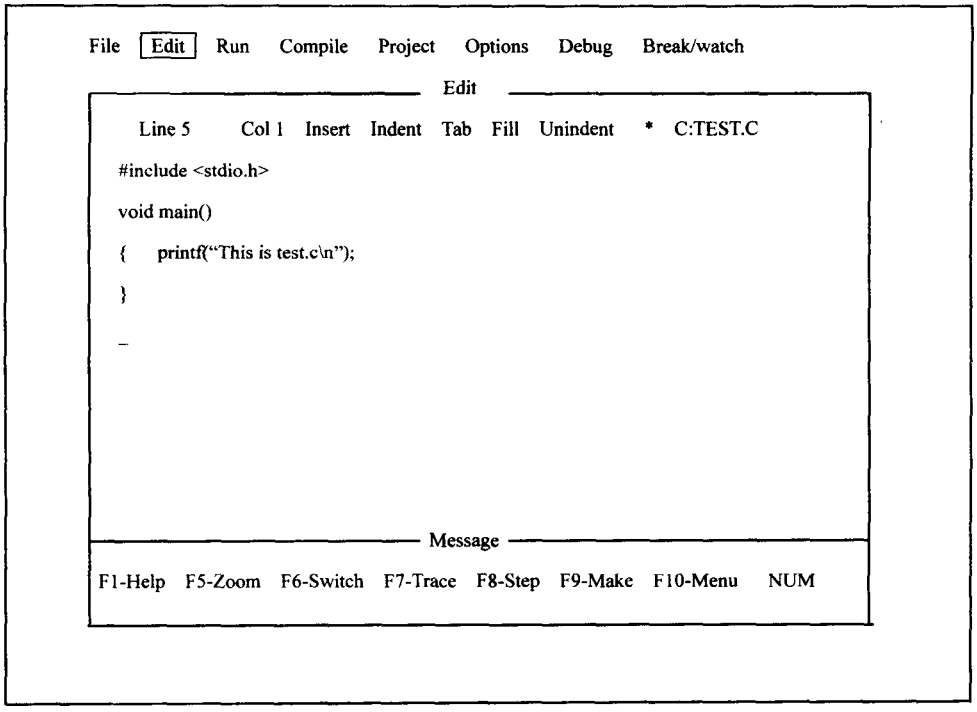

图 1.3 Turbo C 编辑窗口

Tab——指示制表键 Tab 处于有效状态，即每按一次 Tab 键，光标移动到下一显示区的起始列位置，缺省的显示区宽度为 8 列，依此选择菜单 Options/Environment/Tab size 可以设定显示区宽度。当 Tab 隐藏时，键 Tab 无效。键 Ctrl+O+T 或 Ctrl+Q+T 用于实现这两种状态间的切换。

Unindent——一般而言，退格键只清除当前光标处的前一个字符。当光标处于一行第一个非空白字符位置时， Unindent 指示按退格键可清除前导的若干个空白字符；当 Unindent 隐藏时，每按一次退格键只清除前导的一个空白字符。键 Ctrl+O+U 用于实现这两种状态间的切换。

* ——指示正在编辑的文本同保存该文本的磁盘文件内容不一致，即已被修改而未被存盘。

C:TEST.C——正在编辑的文件的文本,如图 1.3 所示,为硬盘 C 中的文件 TEST.C。

在编辑时，有许多提高编辑效率的快捷键可以被使用，这些快捷键的定义，可以在 TC 环境下随时键入 F1，在帮助窗口选择 Editor，通过按 PageUp/PageDown 实时找到解答。随着用户使用 TC 次数的增加，一些常用的热键将不知不觉地被熟练掌握。

1.4.4 C文件的编译

TC环境下的Compile子菜单是编译和连接C源程序的命令集，TC把C源程序编译和连接过程中产生的信息集中在一个缓冲区中,并显示在信息窗口内。用户可以根据TC在信息窗口告知的错误类型和错误所在的行,直接在编辑窗口修改C源程序。图1.4是编译子菜单。

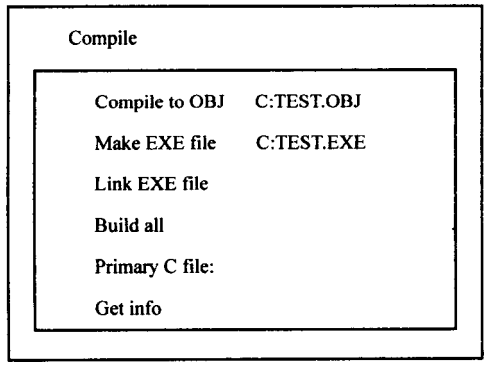

图1.4 编译子菜单

1. Compile to OBJ

选择该选项首先检查是否有与C源文件同名的.OBJ文件，若没有或者保存在.OBJ文件中的C源文件的生成日期和时间同当前C源文件生成日期和时间不一致，则认定该C源文件已经修改，重现编译该C源文件；否则将不执行编译操作。若编译成功，则生成后缀为.OBJ的文件存盘保存，该.OBJ文件的主名取被编译的C源文件的主名，图中C:TEST.OBJ是对C源文件C:TEST.C编译成功后存盘的文件；否则在信息窗口显示在编译过程中发现的错误，并在信息窗口和编辑窗口分别显示一高亮度行，信息窗口的高亮度行定位在TC编译器报告的第一个错误信息或可疑信息行，编辑窗口的高亮度行定位在C源文件对应第一个错误或有疑问的语句行。使用上下光标移动键可以改变信息窗口和编辑窗口中的高亮度行。

2. Make EXE file

该选项的热键是F9，选择该选项将把C源文件或由工程文件（以.PRJ为后缀）说明的文件编译并连接成.EXE文件，若成功，则把编译生成的.OBJ文件存盘保存，并以该C源文件或工程文件的主名把连接生成的.EXE文件存盘保存，图中C:TEST.EXE是对C源文件C:TEST.C编译并连接成功后存盘的文件；否则在信息显示区显示在编译和连接中发现的错误。该选项在工作时也将首先检查.C、.OBJ和.EXE文件的生成日期和时间，以确定是否编译和连接。

3. Link EXE file

选择该选项不检查 .C、.OBJ 和 .EXE 文件的生成日期和时间，而把当前已经存在的 .OBJ 文件连接成 .EXE 文件。

4. Build all

该选项的作用同 Make EXE file ，但不检查 .C、.OBJ 和 .EXE 文件的生成日期和时间，无条件进行编译和连接，若成功，则生成新的 .OBJ 和 .EXE 的文件存盘保存。

5. Primary C file

当该选项被选中时，将有一个等待用户键入当前文件名字的输入框，用户键入的文件名指定当前欲编译或编译并连接的是哪一个 C 源文件，然后可以选择 Compile 子菜单中的编译或编译并连接的选项对该 C 源文件编译或编译连接，当发现 C 源文件有错时，将把该 C 源文件送编辑窗口供用户检查或修改。

6. Get info

该选项的作用是向用户提供一个信息窗口，该窗口的内容包括当前的工作目录、正在编辑的文件名字和文件的字节长度、最新一次编译 C 源文件时保留的信息（C 源文件的行数、警告数、错误数）、程序的退出码以及可用的内存大小等。

.4.5 C 程序的调试

在 TC 环境下编辑好 C 源程序后，按热键 F9 进入源程序的编译连接工作，一旦编译连接成功，用户可以按 Ctrl+F9 运行这个生成的用户程序，再按 Alt+F5 在用户窗口观察程序运行结果，上述工作都可在 TC 环境下完成，相当方便。但是编译即使成功，程序的执行结果却可能与编程人员的预期不同，这就需要耐心的调试，也就是人们常说的“ Debug”。在 TC 主菜单中有三个与程序调试相关的选项 Run、Debug 和 Break/Watch。需要说明的是：如果所编写的程序涉及到系统，如中断矢量的替换、显示模式的切换、扩展内存的使用或大内存空间的分配，为安全着想推荐退出 TC 在系统环境下执行。

1.Run——运行子菜单（如图 1.5 所示）

（1）Run

该选项用于运行用户程序，若程序在编译后作过修改，则首先编译、连接该 C 源程序。若 C 源程序在编辑时设置了断点，则从用户程序的开头运行到第一个断点处停

止，再次选择该选项将继续运行到下一个断点处停止，用户可在用户窗口查看程序运行到每个断点处在屏幕上的输出情况，或在监视窗口观察被监视表达式的当前状况。有关断点和监视表达式的设置、删除等操作在 TC 主菜单中的断点／察看选项 Break/Watch 的子菜单中。对于需要参数的用户程序，在运行用户程序前由 TC 主菜单的选项 Options 的子菜单 Argument 选项设置传递参数。

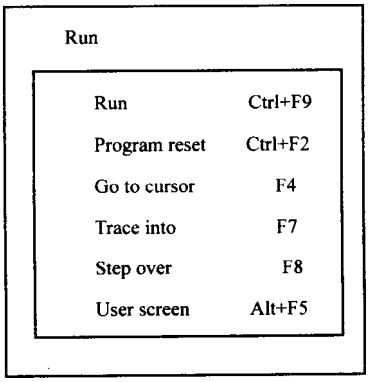

图 1.5　运行子菜单

(2) Program reset

中止调试、释放所占内存并关闭所有打开的文件。

(3) Go to cursor

从亮条开始执行程序，直到编辑窗口中光标所在行停止，除非在此区间有断点，遇到这种情况，可再次按该选项的热键 F4。若光标所在行不含可执行语句，该命令弹出一个 Esc 框,按 Esc 键可返回。

(4) Trace into

该选项用于跟踪程序的一个个语句行的执行，选中该选项或按其热键 F7 执行一高亮度显示的语句行，并使下一个要执行的语句行高亮度显示。不仅如此,若执行一高亮度显示的语句行中有函数调用，且被调用的函数是可访问的函数，则该被调用函数的第一个语句行高亮度显示，即该选项可跟踪进入下一层函数。

(5) Step over

同选项 Trace into 一样，该选项也用于跟踪程序的一个个语句行的执行。但区别在本选项不能跟踪进入下一层函数。

(6) User screen

该选项的热键是 Alt+F5,按热键 Alt+F5 或选中该选项将把 TC 窗口切换到用户窗口，在用运行子菜单的其他选项调试程序的过程中可随时用该选项，以便在用户窗口查看程序运行到现时在屏幕上的输出情况。按任意键可从用户窗口切换到 TC 窗口。

2.Debug——调试子菜单（如图 1.6 所示）

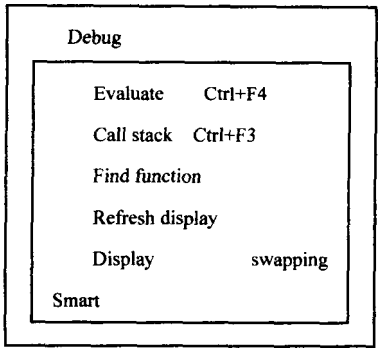

图 1.6 调试子菜单

3.Break/Watch——断点/察看子菜单（如图 1.7 所示）

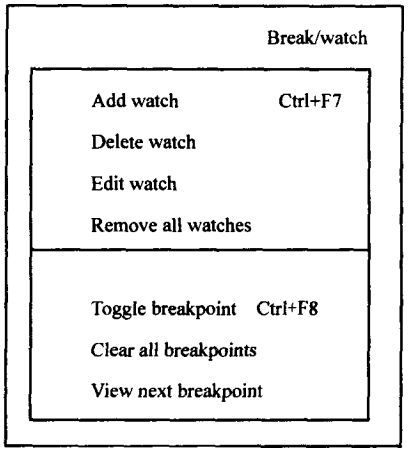

图 1.7 断点、察看子菜单

1.4.6 TC 主菜单的其他选项

1.Project—— 工程子菜单（如图 1.8 所示）

一个语法正确的 C 源文件经编译能生成一个.OBJ 文件，若一个完整的 C 程序仅由该 C 源文件组成，则对该 C 源文件编译连接可生成一个可执行的.EXE 文件。而一