中华人民共和国教育部考试中心
全国计算机应用技术证书考试(NIT)

程序设计(C语言)教程

教育部考试中心　　组编
张基温　　编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书是全国计算机应用技术证书考试(NIT)程序设计模块(C 语言)的指定教材。作者从具体问题出发,重点阐述如何利用程序设计解决这些问题,同时把 C 语言程序设计的方法融入其中。书中有很多例题和程序测试题,可帮助读者尽快地熟悉 C 语言程序设计的原理和应用。

除作为 NIT 指定教材外,本书还非常适合大专院校师生和广大计算机程序设计的初学者使用。

书　名: 程序设计(C 语言)教程

作　者: 张基温　编著

出版者: 清华大学出版社 (北京清华大学学研楼,邮编 100084)

http://www.tup.tsinghua.edu.cn

印刷者: 北京市丰台区丰华印刷厂

发行者: 新华书店总店北京发行所

开　本: 787× 1092 1/16　印张: 13.25　字数: 325 千字

版　次: 2000 年 2 月 第 1 版　2000 年 2 月 第 1 次印刷

书　号: ISBN 7-302-02146-5/TP・2180

印　数: 0001～6000

定　价: 19.00 元

第一届全国计算机应用技术证书考试

委 员 会 名 单

(以姓氏笔画为序)

主 任 委 员: 杨学为　谭浩强

副主任委员: 王建军　刘瑞挺　吴文虎　潘桂明

委　　员: 王成钧　王　耆　王景新　毛汉书　边奠英

刘百惠　刘长占　任威烈　求伯君　吴立德

吴功宜　苏运霖　陈　禹　杨一平　杨明福

杨炳儒　林毓材　周明德　张基温　张　森

孟志华　高　林　徐士良　徐惠民　赵鸿德

侯炳辉　裴纯礼　潘　阳

秘　书　长: 潘　阳

全国计算机应用技术证书考试教材编审

委 员 会 名 单

(以姓氏笔画为序)

主 任 委 员: 杨学为　谭浩强

副主任委员: 王建军　刘瑞挺　吴文虎　潘桂明

委　　员: 王成钧　王　耆　吴功宜　赵鸿德　侯炳辉

姜春红　高　林　徐士良　徐海涛　韩庆久

熊燕清　潘　阳

“全国计算机应用技术证书考试(NIT)”系列教材

序

人类社会已经进入了信息时代。计算机的应用日益成为人类生活、工作、学习所必备的一种基本能力，愈来愈多的人迫切希望掌握计算机的应用技术，以符合信息时代的要求。毫无疑问，中国需要一批人掌握深奥的信息技术理论与复杂的信息技术，但是对于大多数人来说，只需要掌握实用技术就足够了。在几年前我们就注意到这种趋势，并开始了这种变革。在借鉴英国剑桥大学考试委员会举办的剑桥信息技术(CIT)的成功经验的基础上，实行以实践为主的操作培训和技能考试，这就是全国计算机应用技术证书考试(NIT)。它在系统设计上采取了一种全新的思路，首次将考试分为过程式考核、作业设计及上机考试三个阶段，以实际应用为目的，培养和测试考生在计算机应用领域的独立操作能力和应用技能。根据计算机技术发展的特点和学习者在应用领域中的需要，它采用模块化结构，在培训内容设置上紧跟计算机技术的发展，在教学过程中充分体现考生的个性，侧重于考生应用技能的培养；采用指导评估的方式进行能力考核，对考生的独立操作能力和独立解决问题能力进行综合测试。

为了规范培训和考试，我们决定选择最新和最流行的计算机应用软件，编写系列丛书，作为全国计算机应用技术证书考试的指定教材。为了体现 NIT 侧重培养和测试考生在计算机应用领域的独立操作能力的特点，我们改变了以往同类教材的传统写法，采用以任务驱动的方式，引导读者在完成每个任务的过程中学会相应的操作，并希望通过培训来帮助大多数人掌握计算机的应用技能。这套丛书图文并茂、循序渐进、易学易懂，有的还配有多媒体教学光盘，以帮助读者的学习。

我们邀请国内一些著名的专家编写这套丛书，他们夜以继日地紧张工作，圆满完成了任务，在此谨向他们致以衷心感谢。

由于我们缺乏经验，书中不足之处在所难免，敬请各位读者及关心我们的同志批评指正。

教育部考试中心　主任

1999 年 3 月

前　　言

程序设计应当是信息时代的必修课,学习程序设计不仅能培养使用计算机进行应用开发的基本能力,更重要的是,它能提供人们解决复杂问题的思维训练。然而,如何提高程序设计课程的效率,越来越被人们关注。

过去的程序设计都是面向语法的,所使用的教材可以说是语法手册稍加改造而成的,程序设计语言无非是提供了一个环境。然而,程序设计并非仅仅需要环境,在程序设计语言百花齐放、不断推陈出新的今天,程序设计语言已逾数千种,每种语言又有不同的版本,把学习者的精力过分地集中到语法的学习上,企图在学习者的头脑中先建立一种程序设计语言的完整语法体系,是不现实的,也是不必要的。对初学者来说,首要的是进行思维方式的训练。

世界上的事物是极为复杂的,解不同的问题需要不同的思路。然而,问题又是可以被分类的,每一类问题都有其共性。一些典型问题的解法是具有代表性的,掌握这些解法对进一步提高程序设计的能力具有基础性和启发性。实践证明,只要掌握了为数不多的典型问题的解法,就可以具备基本的程序设计能力。大约十年之前,本人就构思过一种面向问题的程序设计教材,但由于事繁力薄,一时难以落实。自 1997 年起,随着 NIT 的实施进入实质性阶段,教育部考试中心教育测量学术交流中心组织编写配套教材,才被迫把写作提到议事日程上来。

一种新的体系的教材的出世又谈何容易。程序设计课不可能脱离开一种语言环境,但又希望淡化语言语法,因此在结构的设计、素材的组织、内容的取舍等诸方面,常常左右为难,直到出版也难以尽意。好在有 NIT 考试委员会中著名计算机教育家谭浩强教授、吴文虎教授等专家的热情鼓励,有许多热心读者的支持,还有董惠丽、张秋菊等人的协力,使我终于能够完成本书。诚望广大专家赐教,特别寄望于使用本书的同行和读者们来自实践中的宝贵意见,以便逐步将它改得更加完美。

张基温

1999 年 8 月

目　　录

第零单元　引言 …… 1
　引言一　程序与程序设计 …… 1
　引言二　程序开发环境——Turbo C …… 8
　引言三　C 语言入门 …… 11

第一单元　判断与选择 …… 41
　任务一　多重选择 …… 41
　任务二　多情况选择 …… 43
　语法小结(一)——分支结构 …… 47
　程序测试(一)——结构测试法 …… 49
　程序测试(二)——等价分类法 …… 52

第二单元　穷举 …… 55
　任务一　大奖赛评分程序 …… 55
　任务二　求素数 …… 57
　任务三　搬砖问题 …… 59
　程序测试(三)——边值分析法 …… 62

第三单元　递推 …… 64
　任务一　欧几里德算法 …… 64
　任务二　猴子吃桃子问题 …… 66
　语法小结(二)——流程转向控制 …… 67

第四单元　模拟 …… 69
　任务一　时间步长法 …… 69
　任务二　简单概率问题 …… 71
　任务三　事件步长法 …… 76
　语法小结(三)——循环结构 …… 79

第五单元　递归 …… 80
　任务一　计算 n! 的递归函数 …… 81
　任务二　二分查询 …… 83
　任务三　排序 …… 87
　语法小结(四)——变量的作用域和生存期 …… 93
　语法小结(五)——　一维数组 …… 100

第六单元　字符串操作…… 102
任务一　基于一维数组的字符串操作…… 102
任务二　基于指针的字符串操作…… 107
语法小结(六) —— 指针类型…… 111
第七单元　二维数组…… 118
任务一　成绩处理…… 118
任务二　日期转换…… 122
任务三　迷宫问题…… 124
语法小结(七) —— 多维数组…… 128
第八单元　结构体…… 131
任务一　单个学生的学籍管理…… 131
任务二　一组学生的学籍管理…… 134
语法小结(八) —— 结构体…… 137
第九单元　文件…… 141
任务一　写若干行字符串到文本文件…… 141
任务二　文件复制…… 146
语法小结(九) —— 文件…… 150
第十单元　程序文档…… 155
任务一　用户文档和技术文档…… 155
任务二　程序文档示例…… 156
附录…… 164
附录一　ASCII 字符编码一览表…… 164
附录二　C 语言关键字及其用途…… 166
附录三　C 语言运算符的优先级别和结合方向…… 167
附录四　C 语言库函数…… 168
附录五　Turbo C 编译错误信息…… 183
附录六　全国计算机应用技术证书考试(NIT)培训与考试大纲
—— 程序设计模块(C 语言)…… 187
附件一　全国计算机应用技术证书考试(NIT)学员评估记录表
—— 程序设计模块(C 语言)…… 191
附件二　全国计算机应用技术证书考试(NIT)作业设计考核表
—— 程序设计模块(C 语言)…… 192
附件三　全国计算机应用技术证书考试(NIT)作业设计参考示例
—— 程序设计模块(C 语言)…… 194
附件四　全国计算机应用技术证书考试(NIT)上机考试题型举例
—— 程序设计模块(C 语言)…… 200

第零单元 引 言

引言一 程序与程序设计

一、程序与程序设计语言

1.程序的一般概念

通常,完成一项复杂的任务,需要进行一系列的具体工作。这些按一定的顺序安排的工作—操作序列,就称为程序。例如,下面是某一个会议的程序:

宣布会议开始;
奏国歌;
校长讲话;
宣布获奖者名单;
颁奖;
获奖代表讲话;
宣布会议结束。

再如,下面是做某一个菜的程序:

洗菜;
备菜;
坐锅;
锅热后加油;
油热至八成,放入葱、姜、蒜和调料略煸炒后,加入菜煸炒;
加入些许水;
菜八成熟时,加入酱油、味精;
出锅,并装盘。

简单地说,程序主要用于描述完成某项功能所涉及的对象和动作规则。如上述的国歌、首长、名单、代表、话、奖以及菜、锅、油、水、酱油、味精等都是对象,而奏、作、颁、讲、宣布、放入、出锅等都是动作,这些动作的先后顺序以及它们能作用的对象,要遵守一定的规则。如“颁”的作用对象是“奖”而不是“话”;不能先颁奖,后宣布获奖名单。再如,做菜时,若锅热后,先放水、再放菜,就变成了煮菜,而不是炒菜了。

可见,程序的概念是很普遍的。但是,计算机出现后,“程序”几乎成了要计算机完成一项任务的代名词,主要用于描述计算机完成某项功能所涉及的对象和动作规则。

2.计算机程序设计语言

就像同样一个意思,用汉语、英语和日语描述出来的形式各不相同一样,要计算机实现同样一个功能,用不同的计算机程序设计语言描述出来,形式也各不相同。

计算机程序设计语言(简称计算机语言、程序设计语言等)是在计算机技术发展和应用过程中不断发展的。目前已经有上千种计算机程序设计语言出现,通常被分为低级语言和高级语言两大类。低级语言是与机器的结构有关的编程语言,所以也称为面向机器的语言。高级语言,是与机器结构无关并考虑了人的描述和思维习惯的的编程语言。本书介绍的C语言就是一种高级语言。下面是几个C语言程序的片段。

例0.1

```
#include <stdio.h>
main()
{
    printf("This is a C program.");
}
```

说明:

(1) C语言程序的基本形式为

```
main()
{
    <语句序列>
}
```

“main()”是C语言程序的标志,称为主函数。所有的C语言程序都有这样一个标志。

(2) C语言用语句对对象和操作进行描述,语句要写在“main()”后面的一对花括弧中。语句用分号结尾。

(3) 本例中只有一条语句,它使用一个函数printf()将双引号中的一串字符原样输出。printf()的具体操作是预先定义好的。这样的预先定义的函数还有许多,如被系统分门别类存在的称为函数库的地方,也将它们称为库函数。使用库函数,将使程序十分简洁、清晰。stdio.h称为printf()的头文件,它定义了要使用printf()等输入输出函数所要使用的必要信息,程序中要使用函数printf()时,应当用预处理命令#include将其包含到程序中。

例0.2

```
#include <stdio.h>
main()
{
    int a, b, c;
    a=5;
    b=3;
    c=a*b;
```

```
    printf("c = %d", c);
}
```

说明:

(1)“int a, b, c;”称为声明语句,它定义了 a, b, c 三个变量。通常,变量具有如下特点:

占有一片可用于存放数据的内存空间,空间的大小决定于它要存放什么样的数据。本例中,用关键字“int”定义了 a, b, c 是可存放整型数(integer)的变量。

变量的值即它所存放的数据值不是固定不变的,通过赋值操作(如 a = 5; b = 3; c = a * b;)可以用一个新的数据替换原来的数据。

每个变量都要有一个名字(本例中分别为 a, b, c)。给变量命名要按一定的规则进行。一般来说,变量名应以字母打头,后面是数字和字母组成的串。

(2) 在 C 语言中,赋值操作用符号“=”表示。注意,它不是等号,在 C 语言中,等号为“==”。赋值号表示一种操作,把其后面的数据送到其前面的变量中;等号则表示一种关系——其前后两个数据相等。

(3) 字符“*”在 C 语言中表示乘运算。在 C 语言中,四则运算的运算符分别为 +, -, *, / 。

(4) 库函数 pinrtf()称为格式化输出函数,圆括号中双引号中的字母表示格式:“%d”表示输出的数据是一个整型数,其前的“c =”将以原样输出。本例的输出结果为:

```
c = 15
```

例 0.3

```
#include <stdio.h>
main()
{
        int a=5, b=3, c;
        c=a;
        a=b;
        b=c;
        printf("a=%d,b=%d",a,b);
}
```

说明:

(1) 在声明语句“int a = 5, b = 3, c;”中,“a = 5, b = 3”称为对变量 a 和 b 的初始化,即定义变量 a 的初始值为 5,变量 b 的初始值为 3。

(2)“c = a; a = b; b = c;”三个语句的操作结果是交换了变量 a 和 b 的值,变量 c 称为中间变量。具体操作为:先将变量 a 的值赋给变量 c(即 a, c 的值均变为 5),再将变量 b 的值赋给变量 a(即 b, a 的值均变为 3),最后将变量 c 的值赋给变量 b(即 b, c 的值均变为 5)。这一交换过程相当于有 a, b 两盘磁带,各录有不同的内容,要交换 a, b 两盘磁带的内容时,必须借助于第三盘磁带 c。

(3) 语句“printf("a = %d, b = %d", a, b);”的输出结果为:

```
a = 3, b = 5
```

格式参数中的两个“%d”,分别指示要输出的两个数据 a 和 b 的格式,其余字符将依原样输出。

例 0.4

```
#include <stdio.h>
main()
{
    int a=5, b=3,max;
    if(a>b)
      max=a;
    else
      max=b;
    printf("max=%d",max);
}
```

说明:这个程序完成下面的功能:取 a,b 之中的较大者,赋值给变量 max,最后输出 max 的值。具体的操作为:如果 a>b,则将 a 的值赋给 max,否则将 b 的值赋给 max;接着输出 max 的值。

例 0.5

```
#include <stdio.h>
main()
{
        int a, b,max;
        printf("Please input two integers:\n");
        scanf("%d %d",&a,&b);
        if(a>b)
            max=a;
        else
            max=b;
        printf("max=%d",max);
}
```

说明:

(1) 与例 0.4 相比,本例的不同在于 a,b 两个变量的值不是预先在编程序时给定的,而是在程序运行过程中由键盘输入的。

(2) 语句

```
scanf("%d%d", &a, &b);
```

的功能是由键盘上给 a,b 两个变量输入值。注意,这时变量名前一定要写上符号“&”,同时格式参数中的格式符(本例中为“%d”)数要与后面的变量数一致。

(3) 语句

```
printf("Please input two integers:\n");
```

的功能是向使用程序的用户给出一个提示,要他从键盘上输入两个数。否则,系统遇到 scanf()函数,便显示一个空白的屏幕。该语句中的“\n”,是产生一个换行。

3 . 数据结构与算法

前面给出了几个小的 C 程序。一般说来,对不同的问题,有不同的解题思路;对同一个问题,也可能有不同的解题思路;对同一个问题的同一个解题思路,可能有不同的程序描述。

程序设计是一项艰巨的脑力劳动。为了使得问题更容易求解,在程序设计的初期要把精力集中在问题求解的思路上,暂不考虑程序描述的细节。解题思路的考虑分为两个方面:如何组织求解对象和制定求解过程的操作步骤和规则。这两个方面称为数据结构和算法。数据结构描述了问题涉及对象间的联系和组织结构,算法描述了求解问题的步骤或规则。著名的计算机科学家 Niklaus Wirth 有一个简洁的描述:

算法 + 数据结构 = 程序

也就是说,程序设计的关键是构造程序的数据结构并描述问题所需要的、施加在这些数据结构上的算法。

算法可以用自然语言描述,也可以用专用的算法描述语言描述,还可以用程序设计语言描述,用程序设计语言描述的算法就是程序。常用的算法专用描述语言是流程图。图 0 .1 中的(a),(b)分图,分别为例 0 .3,例 0 .5 的流程图。

图 0 .1

图中,矩形框表示操作,菱型框表示选择,带箭头的线表示流程。

二、程序开发过程

程序开发的一般过程为:

1 . 分析

一般说来,一个具体的问题要涉及许许多多的方面,这是问题的复杂性所在。为了便于求解,往往要忽略一些次要方面。这种通过忽略次要方面,而找出解题规律,就称为建立模型。分析问题就是要搞清对问题相关领域的了解和所具备的知识。

2 . 设计

设计分为两步:总体设计和详细设计。

(1) 总体设计

当问题较大时,常常要把问题分为一些子问题;当子问题还大时,还应进一步分割。这样,一个较大的问题就被分成一些容易求解的子问题,每一个子问题将作为程序的一个模块。这是总体设计的一个任务。但是,仅仅分解还是不够的,在总体设计时,还要考虑如何组织程序模块。图 0.2 为一个教学管理程序的结构图。

图 0.2

(2) 详细设计

详细设计就是设计每个模块的数据结构和算法。

3. 程序编码以及编辑、编译和连接

程序编码就是用具体的程序设计语言表现各模块的算法和数据结构。这一阶段的关键是运用某一种具体的计算机语言,来描述前面设计的数据结构及其算法。为此就要掌握一种计算机程序设计语言。

程序编码可以先写在纸上,也可以直接在计算机上书写,形成一个程序文件。在一定的环境下进行程序的书写以及修改的过程称为程序的编辑。

写出一个高级语言程序后,并不是就可以立即拿来执行。要让机器直接执行,还要将它翻译成由机器可以直接辨认并执行的机器语言程序。为区别它们,把用高级语言写的程序(文件)称为源程序(文件),把机器可以直接辨认并执行的程序(文件)称为可执行程序(文件)。这一过程一般分为两步:

第 1 步,在程序编辑过程中输入到计算机内部的是 ASCII 码。ASCII 码是计算机不能直接辨认的,为此,首先要将源程序文件翻译成机器可以辨认的程序文件——目标程序文件。这一过程称为编译。在编译过程中,还要对源程序中的语法和逻辑结构进行检查。编译任务是由称做编译器(compiler)的软件完成的。目标程序文件还不能被执行,它们只是一些可重定位的目标程序模块。

第 2 步,将目标程序文件中可重定位的目标程序模块以及程序所需的系统中固有的目标程序模块(如执行输入输出操作的模块)链接成一个完整的程序。经正确链接所生成的文件才是可执行文件。完成链接过程的软件称为链接器(linker)。

图 0.3 为 C 语言程序的编辑、编译和链接过程示意图。

图 0.3

目前,程序的编辑、编译、连接以及运行,都可以在一些集成环境下进行。后面介绍的Turbo C就是一种集成环境。

4.程序测试

每一个人编写出一个程序后,在正式交付使用前,总要试通一下。“试通”就是试运行程序,也就是对程序进行测试。今天,尽管程序测试作为程序设计的一个重要组成环节已再无人怀疑,但是在下面的两个问题上,并不是所有的人都能有一个正确的认识。

(1) 程序测试的目的

基于不同的立场,存在着两种截然相反的观点:一种人认为,测试的目的是为了证明程序是正确的;另一种认为,测试的目的是为了发现程序中的错误。实际上前一种观点将指导一种自欺欺人的行为,它对于提高程序的质量毫无价值。正确的观点是后者,Glenford J.Myers把它归结为如下三句话:

测试是程序的执行过程,目的在于发现错误;

一个好的测试实例在于能发现至今未发现的错误;

一个成功的测试是发现了至今未发现的错误的测试。

(2) 测试方法和技术

测试是以程序通过编译,没有语法和连接上的错误为前提的。在此基础上,通过让程序试运行一组数据,来检测程序的逻辑以及程序中的各语句有无错误。这一组测试数据应是以“任何程序都是有错误的”为前提精心设计出来的,而不是随心所欲地乱凑成的。它不仅应含有被测程序的输入数据,而且还应包括程序执行它们后预期的结果;每次测试都要把实际的结果与预期的结果相比较,以观察程序是否出错。

著名计算机软件科学家E.W.Dijkstra曾断定:“程序测试只能证明错误的存在,而不能证明错误不存在。”可以证明,除很小的程序外,无论使用任何方法,要想做到彻底的测试,即发现程序中的所有错误,是不现实的。现实是如何提高测试的效率,尽可能多地检测出程序中的错误。这就要求注重测试方法和技术。不同的测试方法所对应的测试用例的设计方法不同。

5.编写程序文档

经过了问题分析、设计、程序编码、测试后,程序开发的工作基本上结束了。但是,这时还不能交付使用。因为还存在两个问题没有解决:一是用户拿到程序后会不会使用它;二是用户在使用过程中,对程序是否满意,永远满意,而不需要做任何修改。

实际上,随着程序规模的增大和日益复杂化,程序的使用和运行也越来越不那么直接,用户要运行程序,还需要知道许多信息,如:

程序的功能;

需要输入的数据类型、格式和取值范围;

需要使用的文件数量、名称、内容以及存放位置等;

程序运行需要的软、硬件环境;

程序的装入、启动方法以及交互方式等。

为此,程序开发者需要向用户提供这些资料——称为程序使用说明书或用户文档。需要说明的是,在许多软件中,这些内容已经部分或全部地以“readme”或“help”的形式提供。

另一方面,一个程序开发好后并不是一劳永逸了,由于下列原因,程序还要进行必要的修改:

(1) 由于在开发过程中,开发者同用户之间的交流不可能非常充分和开发者对用户的要求没有充分理解等原因,程序同用户要求之间还会存在一些差距;

(2) 软件界有一句这样的谚语:“没有不存在错误的软件。”由于开发者的疏忽以及测试的不完全性限制,程序一定会存在这样或那样的错误;

(3) 在程序的使用过程中,由于条件、环境以及应用对象的改变。

在使用过程中进行的这些修改,称为程序的维护。为了便于程序的维护,也需要提供一个专门的文档——程序技术说明书或称程序技术文档,其内容包括:

程序各模块的的描述;

程序使用硬件的有关信息;

主要算法的解释和描述;

各变量的名称、作用;

程序代码清单。

目前,程序文档已经称为软件开发产品的必要部分。文档在程序使用和维护中的重要性也改变了软件的概念,使之由早期的“计算机程序的总称”演化为“计算机的程序连同计算机化的文档的总称”。

引言二　程序开发环境——Turbo C

程序开发环境一般是集编辑、编译、连接、调试为一体的集成环境。这一节介绍一个C语言程序的开发环境——Turbo C。

一、Turbo C 概述

C语言的程序需要具体的C编译器和链接器的支持。不同的C编译器和链接器具有不同的特点和功能,形成了不同的C语言的版本。Turbo C是Borland公司开发的一种C语言程序开发环境,它不仅提供了基于ANSI C的编译器和链接器,还提供了一个供程序员书写和修改程序的方便的编辑器,形成一个集成化的C语言程序开发和运行环境。同时,它还在实践中不断自我完善,产生出一个不断走向高级的Turbo C版本系列。本书所介绍的开发环境是基于Turbo C 2.0的。它对机器的要求较低,可以适合更多的学习者使用。

Turbo C定义了两种屏幕状态:程序开发状态和程序运行状态。两个状态是相互独

立的,通常,Turbo C 位于开发环境状态,只有程序运行时才进入运行状态(也称用户屏幕)。

在安装有 Turbo C 的计算机中,在 DOS 平台上,进入含有 Turbo C 的子目录后,打入命令 TC;或在 Windows 平台上,打开含有 Turbo C 的文件夹后,点击可执行文件 TC,即可进入 Turbo C 2 .0 集成开发环境。这时,图 0 .4 所示的主画面将展现在你的面前。

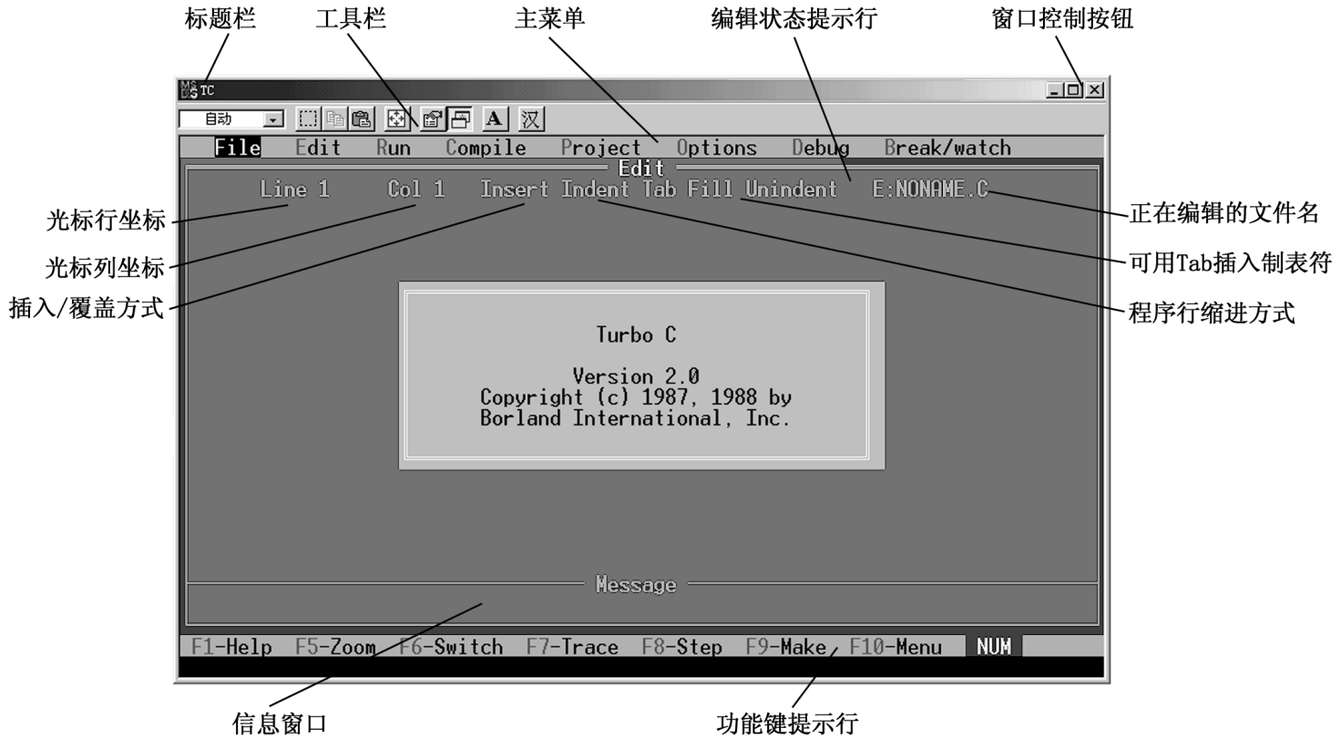

图 0 .4

Turbo C 集成开发环境主画面由如下元素组成:

(1) 标题栏

标题栏用于显示当前程序名“TC”。标题栏左端的图标一方面表明它是一个 Turbo C 集成环境系统,另一方面点击它可以下拉一个控制菜单,用于对 Turbo C 集成环境系统的关闭以及窗口大小的控制。这些控制功能,也可以用标题栏右端的三个按钮简捷地执行。

(2) 主菜单

Turbo C 的开发环境中提供有如下 8 个菜单(括弧中为进入热键):

File(Alt + F)	进行文件和目录操作
Edit(Alt + E)	编辑当前编辑窗口中显示的程序
Run(Alt + R)	控制程序的运行状态
Compile(Alt + F)	进行程序的编译、连接
Project(Alt + P)	对工程文件(由多个 C 文件组成的程序)进行管理
Options(Alt + O)	设置选项
Debug(Alt + D)	调试程序:显示变量值,查找函数,查看调用栈状态
Break/ Watch(Alt + B)	调试程序:设置/ 清除断点,监测变量值变化

(3) 工具栏

对主菜单中常用功能进行简捷操作。