高职高专计算机专业系列教材

C 语 言 程 序 设 计 实 训

刘振安　编著

清 华 大 学 出 版 社

(京)新登字 158 号

内 容 简 介

本书密切结合《C 语言程序设计》一书,重点介绍使用 Borland C++ 3.11 和 Visual C++ 6.0 集成环境编辑、编译、调试和运行 C 语言程序的具体方法,以便读者能够尽快掌握集成环境,通过上机练习帮助学习理解理论知识。本书将结合课本中的实训题,分析解题的思路并给出参考方法。

学习语言是为了使用,在实际的应用中仅以介绍语法结构为目的是不够的,应重点保证程序的结构化设计质量。尤其是大的程序设计,更应如此。本书将单列一章讨论如何防止错误,以便读者能写出可靠性高的程序;本书还结合软件工程的知识介绍如何测试程序以及测试用例设计技术和程序维护,帮助读者进一步提高 C 语言的编程能力。

本书题例典型,结构合理,实用性强,重在培养学生的实际动手能力。不仅可与《C 语言程序设计》教材配套使用,而且可作为培养实验能力的教材单独使用,还可以作为自学教材及工程技术人员的参考书。

图书在版编目(CIP)数据

C 语言程序设计实训/ 刘振安编著 . —北京:清华大学出版社,2002
高职高专计算机专业系列教材
ISBN 7-302-05716-8

Ⅰ.C… Ⅱ.刘… 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(2002)第 056668 号

出 版 者:清华大学出版社(北京清华大学学研大厦,邮编 100084)
http:// www.tup.tsinghua.edu.cn
责任编辑:徐跃进
印 刷 者:北京顺义振华印刷厂
发 行 者:新华书店总店北京发行所
开 本:787×1092 1/16 印张:11.5 字数:263 千字
版 次:2002 年 10 月第 1 版 2002 年 10 月第 1 次印刷
书 号:ISBN 7-302-05716-8/ TP·3373
印 数:0001～6000
定 价:15.00 元

序

1999 年 10 月,教育部高教司主持召开了全国高职高专教材工作会议,会议要求尽快组织规划和编写一批高质量的、具有高职高专特色的基础和专业教材。根据会议精神,在清华大学出版社的支持下,于 2000 年 1 月在上海召开了由来自全国各地的部分高职、高专、成人教育及本科院校的代表参加的“高职高专计算机专业培养目标和课程设置体系研讨会”。与会的专家和教师一致认为,在当前教材建设严重滞后同高职教育迅速发展的矛盾十分突出的情况下,编写一套适应高等职业教育培养技术应用型人才要求的、真正具有高职特色的、体系完整的计算机专业系列教材十分必要,也十分迫切。会议成立了高职高专计算机专业系列教材编审委员会,明确了高职计算机专业的培养目标,即掌握计算机专业有关的基本理论、基本知识和基本技能,尤其要求具有对应用系统的操作使用、维护维修、管理和初步开发的能力。

根据上述目标,编委会拟定了本套教材的编写原则。在教材内容安排上,以培养计算机应用能力为主线,构造该专业的课程设置体系和教学内容体系;从计算机应用需求出发进行理论教学,强调理论教学与实验实训密切结合,尤其突出实践体系与技术应用能力的实训环节的教学;教材编写力求内容新颖、结构合理、概念清楚、实用性强、通俗易懂、前后相关课程有较好的衔接。与本科教材相比,本套教材在培养学生的应用技能上更有特色。

根据目前各高职高专院校计算机专业的课程设置情况,编委会确定了首批出版的十几本教材。这些教材的作者多是在高职高专院校或本科院校的职业技术学院任教的、具有多年教学经验的教师,每本书均由计算机专业的资深教授或专家主审把关。我们还将在此基础上,陆续征集出版第二、三批教材,力争在 3 到 5 年内完成一套完整的高职高专计算机专业教材。

应当说明的是,凡是高等职业教育、高等专科教育和成人高等教育院校的计算机及其相关专业均可使用本套教材。各学校可以根据实际需要,在教学中适当增删一些内容、实训项目和练习题,从而更有针对性地帮助学生掌握计算机专业知识,并形成相关的应用能力。

由于各地区各学校在教学水平、培养目标理解等方面有所不同，加上这套教材编写时间仓促，难免会出现这样或那样的错误，敬请各　校在使用过程中及时将修改意见或好的建议返回给教材编审委员会，以便我们及时修订、改版，使该系列教材日趋完善。

恳切希望高职高专院校任课的专业教师和专家对后续教材的编写提出建设性的意见，并真诚地希望各位教师参与我们的工作。

高职高专计算机专业
系列教材编审委员会
2000 年 5 月

前　言

程序设计是计算机专业领域中最核心的工作。在计算机领域中,任何好的创意和设计最终都需要通过高水平的程序设计实现,才能够真正成为有社会价值、市场价值的产品。但是,目前常规的计算机科学和程序设计课程都不大关注那些程序实践方面的论题;如测试、排错、可移植性、性能、设计选择以及程序设计风格等。近年来程序设计环境确实取得了长足的进步。有人可能会提出疑问,熟悉编程环境之后,许多程序似乎都能在一系列指指点点之中完成,在这种情况下程序设计难道不是非常简单吗?实际情况并不是那么简单,任何一个有点价值的程序都不仅是一个图形界面外壳。虽然在最终的可执行代码里,界面外壳可能占据了相当大的比例,但一个程序的特点恰恰就在那些人们自己动手写出的代码中。而良好的设计、卓越的编码实际上永远都是不可替代的,是不可能自动生成的,正是它们形成了一个软件真正的价值。实际上,在程序设计环境中有价值的东西不过是在一些局部领域中优秀代码的浓缩和沉淀,所以既要熟悉编程环境,也要掌握编程实践环节。学习语言是为了使用,一般教科书仅以介绍语法结构为目的,重点是保证语法正确。但在实际的应用中,则应重点保证程序的结构化设计质量;尤其是大的程序设计,更应如此。

本教材的编写以培养 C 语言编程应用能力为主线,通过实训环节加强实际动手能力的训练。注意结合学科发展方向引入必要的新的基础知识,掌握流行的编程环境,为后续课程的学习打下坚实的基础。力争在体系上有所创新,体现高职高专教育的特点,重点训练基本技能,熟练掌握编译环境,尤其是从事初步开发的能力。本书力求题例典型、实训合理、实用性强。虽然实训密切配合教材,为讲授和学习教材提供了方便,但实训又可以单独使用,为同类教材配套,从而扩大了它的使用范围。

本书具有以下特色。

(1) 本教材内容按照自己的创作体系进行组织,各章均有配合教材的典型题例和实训,并结合本章内容给出参考实训题。本书还结合实训题介绍流行编译环境的使用方法和技巧,以及程序的调试和测试方法、文档的书写格式等。

(2) 对 C 语言这门编程课而言,编写一个语法正确的函数,在实际使

时,可能出现意想不到的错误。语法正确并不等于实用,所以不仅应该强调实际技能的训练,还应该加强在实际工作中所需编程知识的训练。本训练教程同时也从实用的观点出发,专门开设一章讨论如何防止错误,以便写出可靠性高的程序;结合软件工程的知识介绍如何测试程序以及测试用例设计技术和程序维护的基本知识,以便进一步加强学生应用技能的培养,提高 C 语言的编程能力。

(3) 介绍最流行的编译器的特点及使用方法,力图反映当前的发展方向。

(4) 精心设计实训,要求学生掌握下列实际工作的基本技能:

① 对问题进行分析;

② 对算法进行选择和设计并加以细化;

③ 把算法变为程序(编程);

④ 对程序进行调试;

⑤ 对程序进行测试;

⑥ 正确书写文档。

(5) 目前的教材大部分仍然使用 Turbo C 作为实验环境,与目前的实际使用相差甚远,这与学科发展很不适应。本教材将介绍流行的集成环境,让学生先熟悉它们的基本使用方法,为后续学习打下基础。

(6) 本实训教材的最大特点之一是切实可行,步骤明确,可操作性强,适合国情,易于教学。

教育部计算机课程指导委员会副主任委员、中国科学技术大学计算机系高性能计算中心主任陈国良教授及原安徽大学副校长、计算机系程慧霞教授在百忙之中审阅了书稿并提出了许多宝贵意见,特此表示感谢。写作中还参考了大量资料,有的收入参考文献之中,还有些没有收入其中。特此对这些作者表示感谢。

彭程、孙忱、鲁磊、徐菁、葛愿、李祖奎、叶刚、田钢、林育、郭小敏等参加了本书的编写工作,在定稿之后,彭程、孙忱和鲁磊又各自将全部实训重新验证了一遍,以保证实训的正确性。尤其是彭程和孙忱,不仅逐字校对,还反复推敲,提出意见供我修改,为书稿花费了大量心血。

刘振安

2002 年 6 月于中国科学技术大学

目　录

第 1 章　C 语言程序设计基础实训

本章主要要求读者首先熟悉如何编辑、编译、调试和运行一个实际的程序,以便为以后边学边练打下良好的基础。本章将结合一个实例,演示 C 语言编程的全过程。

1.1　典型例题和习题

因为读者可能是第一次接触编程语言的学习,所以首先应树立良好的书写习惯和风格,尤其是变量命名的良好习惯。下面给出一些范例及提示,希望读者能够举一反三,融会贯通。

【例 1.1】　给出下面程序的输出结果。

```
#include <stdio.h>
#include <string.h>
void  main()
{
   int a,b,c,d;
   a = sizeof(char);
   b = sizeof(int);
   c = sizeof(float);
   d = sizeof(double);
   printf("\n%d %d %d %d",a,b,c,d);
}
```

【解答】　“1 2 4 8”,这好像是一个很简单的问题。

由于计算机硬件发展很快,所以 C 编译器也随之更新。这主要是所谓的从 16 位到 32 位操作系统的变化。以 Borland C++ 3.1(本书以后简称 BC)为例,它是为 16 位操作系统配置的,所以即使在 Windows 98 下运行,其运行结果也是“1 2 4 8”。

自从 32 位操作系统盛行以来,编译器也发生了变化。例如,在 Microsoft Visual C++ 6.0(本书以后简称 VC)上,它的运行结果却是“1 4 4 8”,而不是“1 2 4 8”。

这个例子提醒读者注意,对很多 C 语言书上的结果,自己要加以分析。下面再看一个与编译系统有关的例子。

【例 1.2】　给出下面程序的输出结果。

```
#include <stdio.h>
void main()
{
    int inum;
    inum = 8;
    printf("\n%d,%d",inum,inum++);
}
```

【解答】　程序使用 inum 的方式产生歧义。因为在调用 printf 函数时,C 标准并没有规定实参数的求值顺序。inum 和 inum++分别是 printf 函数的两个输出参数。VC 是自左而右,先使用 inum 作为第 1 个和第 2 个参数,然后自增 1,所以输出是"8,8"。BC 是自右而左求值,第 2 个参数为 8,但它使用之后变为 9,将 9 再作为第 1 个参数 inum,所以输出是"9,8"。

在编程时,应该避免使用可能产生歧义的语句,更不要写出别人看不懂,也不知道系统如何执行的程序。尤其是使用"++"和"--"运算符时,更要小心。

【例 1.3】　分析下面程序的输出结果,如果不修改程序,能否通过编译?

```
#include <stdio.h>
main ( )
{
    int a=25, b=5, c=4,d=1;
    a+=b;
    c*=d+5;
    a%=c;
    printf ("%d,%d", a,c);
    printf ("%d", ++a);
    printf ("%d",a++);
    printf ("%d",a);
}
```

【解答】　其实,BC 和 VC 都能通过编译并正确运行,只是 VC 没有警告信息,BC 给出 main 没有返回值的警告(增加 void 即可解决)。printf 语句没有换行符"\n",输出为:6,24778。

1.2　如何编辑、编译、调试和运行一个实际程序

1.2.1　熟悉使用环境的重要性

计算机著名科学家沃思提出"数据结构+算法=程序"的思想,大大促进了程序设计的发展。由于软件产业迅速发展,产生了"软件工程"新学科,人们又认识到"程序设计方法"的重要性。编程是算法的具体实现,所以程序离不开编程。过去的编译环境简单,都

是命令行方式,很容易学习。目前的编译环境都朝集成环境和可视化编程方向发展,而且还实现了一定的自动化功能。由于功能强大,也使其具有一定的复杂性。典型的是 Visual C++ 6.0,提供了 MFC 类库,实现许多 Windows 自动编程功能。因此,如何用好这些语言工具和环境,也是提高编程效率的因素之一。所以目前认为可以用下面的公式表示程序。

数据结构 + 算法 + 程序设计方法 + 编程工具 = 程序

由此可见,一个程序设计人员应该掌握以上 4 个方面的知识。人们一般容易忽视的是第 4 个因素。有些人能写出正确的程序,却无法使用编程工具产生正确的可执行文件。为了消除这种现象,本书将结合具体实训题加强训练,使读者既学会"做什么",也掌握"如何做"的全过程。从而避免可以在纸上编程序,却不知道如何得到程序的可执行文件的现象。

1.2.2 使用 C 程序解题的简单过程

首先要对计算机解决问题的步骤有个初步认识。下面结合一个实际问题,进一步说明使用 C 语言解题的步骤。

(1) 设计一个解题的方法。

(2) 使用一种方式把它描述出来。

(3) 把它们转化成程序形式。

(4) 在计算机上编辑成 C 的源文件。

(5) 编译 C 程序源文件的过程,同时也是查错的过程。如果不能正确编译,进行查错,直到产生正确的 obj 文件为止。

(6) 将它们连接成 exe 文件,如果连接出错,返回步骤(4),再次检查和编译源文件。

(7) 运行 exe 文件,得出初步结果。

(8) 验证结果是否正确。如果结果不正确,就要返回查找原因,直到运行结果正确为止。

表面上看,运行结果不正确要返回(4)检查程序,其实这是对(3)的复查。如果算法设计不对,要转到(1),也即重复(1)～(4)步。如果题目复杂,则要从头开始。因此,一定要重视前 3 个步骤。

下面举一个简单例子,具体说明一下前面的几个过程。

【例 1.4】 编写将输入的两个整数相加,然后输出它们的和的程序。

【解答】 (1) 设计解题方法。显然,这是一个顺序执行的过程,要求按顺序输入两个整数,再把它们相加,最后输出它们的和。

(2) 使用语言方式把它描述出来。例如:

```
程序开始
  要求提示输入 2 个整数
  赋值给变量 number1, number2
  计算 sum = number1 + number2
```

输出 sum
程序结束

如果使用英文及语句说明,可以像下面这样进行说明。

```
BEGIN
  Print  InputMessage
  Input number1, number2
  sum = number1 + number2
  Print sum
END
```

(3) 下面是 C 语言的实现程序。

```
#include  <stdio.h>
void main( )
{
  int  sum, number1, number2;

  printf("Please input two numbers :\n");
  scanf("%d%d",&number1, &number2);
  sum = number1 + number2;
  printf("sum = %d", sum);
}
```

其他几个过程均可以在 C 编译器的集成环境中完成。

(4) 选择一种 C 语言开发环境,建立这个 C 程序的源文件,例如命名为 add.c。在这个文件里输入 C 语言的源程序,这就是源文件的编辑过程。

(5) 使用编译命令对这个源文件进行编译,如果编译正确,产生 add.obj 文件。如果程序有问题,修改后再重复这个过程,直到产生正确的 add.obj 文件。

(6) 使用 Link 命令,将 add.obj 文件连接成可执行的 add.exe 文件。

(7) 运行这个 add.exe 文件,验证结果是否正确。

(8) 如果不正确,需要返回去检查原因,甚至返回到(1)。

由此可见,前 4 步很重要,要提高效率,就要做好这几步的工作。但没有后面的工作,则永远停留在纸上谈兵。一个程序员,必须熟练使用开发环境,才能实现自己的最终任务。

【例 1.5】 将从键盘上输入的一个字符按下面的格式输出,下面是一个运行示范。

```
Input:5
Your input is:5
```

【解答】 下面简述前几步的过程,而且下面给出的 C 语言实现是一个含有错误的程序,留待编译时去查错。

(1) 选择算法

这也是一个顺序执行过程,输入有提示信息,输出增加了说明。

程序开始
  输出提示信息;
  赋值给变量 ch;
  输出组合信息。
程序结束。

(2) 算法描述

```
BEGIN
  Print  InputMessage
  Input ch
  Print OutputMessage
END
```

(3) 编程实现

下面是含有错误的 C 语言程序:

```
main( )
{
  char ch;

  printf("input:");
  scanf("%d", ch);
  printf("Your input is:%d\n", ch);
}
```

1.2.3 BC 上机指南

因为使用 BC 学习 C 编程最方便,所以本节主要介绍使用该集成环境的基本方法,以便读者能顺利解决边学边练的问题。

1.工作界面及基本操作

BC 集成开发环境集内部编辑程序、调试程序及其他实用程序于一体,无需独立地编辑、调试、编译、连接程序,就能建立并运行程序,并且只要通过一个简单清晰的主屏幕,就能使用这些功能。

在使用中常用到双键或三键操作,约定按照按键的先后顺序书写,例如:

Alt + F5　代表先按下 Alt 键,然后按下 F5 键,再同时放开。

Ctrl + KB　代表先按下 Ctrl 键,然后按下 K 键,同时松手之后再按 B 键。

如果 Borland C ++ 是装在 Borlandc 目录下,可在 DOS 提示符下键入 Borlandc\bin\bc 并按 Enter 键,进入 BC 集成开发环境。初次启动屏幕,屏幕除显示主菜单外,还有版本信息。按任意键,版本信息消失并留下如图 1.1 所示的主屏幕。主屏幕由主菜单、编辑窗、信息窗或观察窗及相应的快速参考行等 4 部分组成。

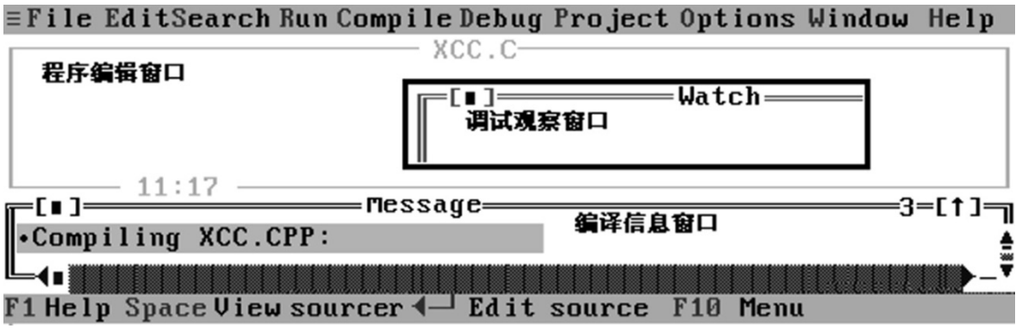

图 1.1　主屏幕信息示意图

快速参考行的内容是变化的,它指示的内容随着所激活的相应窗口而变化,在图 1.1 中是激活信息窗的情况。观察窗与信息窗均处于下部,并且每次只能显示一个,可以使用菜单命令或 F6 键在这些窗口中周游。

(1) 调用帮助系统

① 按 F1 键即可调用该帮助系统,帮助窗口将解释用户当前所处位置的含义,在所有帮助屏幕关键字处按 Enter 键便可得到该选择项更详细的帮助信息。使用 Home 键或 End 键,可分别选择屏幕上第一个或最后一个关键字。按 Alt + F1 键可以回到前次帮助屏幕。在进入帮助系统后,再按 F1 键可以得到帮助索引。当使用 BC 编辑程序时,可能需要了解各种函数信息。将光标置于需要了解的函数名上,按 Ctrl + F1 键,将会得到有关该函数信息的帮助屏幕。要退出帮助系统返回原来的菜单选择,按 Esc 键或使用后面描述的‘热键’即可。

② 按 F5 键可以放大/缩小活动窗口。

③ 按 F6 键可切换活动窗口。

④ 按 F10 键切换菜单和活动窗口。

⑤ 按 Alt 键和主菜单命令的首字母(如 F, E, R,C, W, O 或 D 等)可直接调出该命令。例如按 Alt + F 键可选择文件菜单。

(2) 菜单内的操作

用红色的大写字母选择一个菜单项或使用箭头键把绿色键移动到相应的选择项并按 Enter 键。按 Esc 键可以退出一层菜单。在主菜单或主菜单的某个下拉菜单中,按 Esc 键将直接退回活动窗口(当某窗口被激活时,其顶部为双线,窗口名呈高亮度显示)。按 F10 键从任何菜单层返回先前活动窗口。使用左、右箭头键可从一个下拉菜单进入另一个下拉菜单。

(3) 退出 BC 返回到 DOS

在 File 菜单中选择 Quit(按 Q 键或将光标移至 Quit 再按 Enter 键)将退出 BC 返回到 DOS,但如果在选择 Quit 前未保存当前的工作文件,则系统将询问是否需存盘(也可使用热键 Alt + X 来退出 BC 返回 DOS)。

2.主菜单

(1) 菜单结构及命名约定

在 BC 主菜单及其下拉子菜单中有 3 种类型的菜单选项:命令、开关和设置。

命令：执行一个任务(运行、编译、保存等)。

开关：控制 BC 的某些特性为 On 或 Off。

设置：允许用户为编译程序指定特定的编译和运行时刻所需要的信息，如目录位置、文件名、宏定义等。

本书对所有的菜单项都用一个缩写表示。给定菜单项的缩写是按这种方式形成：按照从主菜单开始，依次进入到该菜单名的首字母序列排列而成。

① 在主菜单下，选择(Options)菜单中编译项(Compile)的代码生成方式(Code Generation)为 O/ C/ C(按 OCC 键)。

② 在主菜单下要选择目录设置。例如：Options/ Directories，只要按 OD 键。

(2) 主菜单的菜单命令

在主菜单中的菜单项都是带有许多选择或子菜单的下拉菜单。主屏幕的顶部是 BC 主菜单窗，如图 1 .1 所示，它的 10 个菜单排列如下所示。

File Edit Search Run Compile Debug Project Options Window Help

以下分别介绍主菜单的 10 个选择项。

File　包括文件的装入、存盘、选取、建立、换名存盘、目录列表和修改，退出程序及调用 DOS 等。

Edit　实现对源文件的编辑功能。

Search　查找及替换功能。

Run　控制运行程序。如用户在 Options/ Debugger/ Source Debugging 下编译和连接程序，且 Source Debugging 开关为 On，则可从此菜单初始化调试过程。

Compile　将源程序编译并生成目标文件或执行文件。

Project　标识组成程序的文件，用以处理工程文件。

Options　选择编译开关(如存储器模式、编译选择项、诊断和连接选择项)；可以定义宏，也可以记录输出文件、库文件和嵌入文件等目录，保存编译选择项或从配置文件装入编译选择项，同时允许用户选择是否将编译时的调试信息也放入执行文件。

Debug　当程序运行时，允许用户检查和修改变量的值、定位任何函数及检查调用栈。

Window　设置窗口的显示方式、隐藏及关闭等状态。

Help　帮助信息。

3 . 快速参考行

无论是处在窗口中或对菜单进行操作时，屏幕的底部都有一个相应的快速参考行。该行是当前可使用功能键的帮助信息或功能提示信息。

4．操作热键

所谓热键就是能够立即完成某一功能的键。如前所述，按 Alt 键和主菜单命令的首字母将直接进入所选的菜单或者完成一个动作。例如 Alt + X 键就是 File/ Quit 的热键。除了 Alt 键加主菜单首字母为热键外，BC 还有一个特殊的用户屏幕热键 Alt + F5，它用来切换 BC 主屏幕和用于输出显示的用户屏幕，它等于 Run 菜单下的 User screen 命令。

使用 BC 时，用户只能看到 BC 主屏幕或用户屏幕之一。BC 主屏幕主要在编辑、编译、连接和调试程序时使用；用户屏幕在执行 BC 程序或用 File/ DOS shell 命令暂时进入 DOS 时使用。当使用集成调试程序时，经常要切换 BC 主屏幕和用户屏幕，BC 也能连续地将后一屏的内容保存于用户屏幕缓冲区中，每当选择运行命令(如 Run，Trace Into 或 Step Over)或 File/ DOS shell 时就更新之。要观察保存的屏幕内容，则在 Run 菜单下选择 User Screen 或按 ALT + F5 键。

下面列出了 BC 中可使用的典型热键，一旦按了这些键就立即执行相应的功能，这里有一例外情况：当有一对话窗口要求按一个特殊键时，直到按下所要求的键后，热键才起作用。

F1	显示当前位置的帮助信息。
F2	保存编辑程序里的当前文件。
F3	装入一个文件。
F4	程序运行到光标所在行。
F5	放大/ 缩小活动窗口。
F6	切换活动窗口。
F7	调试模式下运行程序，跟踪进函数内部。
F8	调试模式下运行程序，跳过函数调用。
F9	执行 make。
F10	切换菜单和活动窗口。
Ctrl + F1	调用与函数相关的帮助信息。
Ctrl + F2	复位运行的程序。
Ctrl + F4	计算表达式。
Ctrl + F7	增加一个监视表达式。
Ctrl + F8	置断点 On 或 Off。
Ctrl + F9	运行程序。
Alt + F1	调出所参考的最后一个帮助屏幕。
Alt + F3	选取文件。
Alt + F5	切换 BC 主屏幕和用户屏幕。
Alt + F7	定位上一个错误。
Alt + F8	定位下一个错误。
Alt + F9	将 BC 编辑程序中的 C ++ 源程序编译成 .OBJ 文件。
Alt + C	进入 Compile 菜单。

Alt + D　　进入 Debug 菜单。
Alt + E　　进入 Edit 窗。
Alt + F　　进入 File 菜单。
Alt + H　　进入 Help 菜单。
Alt + O　　进入 Option 菜单。
Alt + P　　进入 Project 菜单。
Alt + R　　进入 Run 菜单。
Alt + S　　进入 Search 菜单。
Alt + W　　进入 Window 菜单。
Alt + X　　退出 BC 返回到 DOS。

5．文件操作

Borland 公司的 BC 有一个基于 DOS 操作系统的 C 开发环境。对于初学者来说，在 DOS 环境下学习编程，更容易专注于语言本身。

如果没有装入特定文件就进入编辑窗口，BC 运行之后，编辑程序就自动把前一次使用的程序全部装入，若没有使用过文件，就把当前文件命名为 NONAME .CPP。注意将它改为后缀为 .C 的文件名。

(1) 建立新的源文件的两种方法

① 选择 File/ New，建立名为 NONAME .CPP 的文件。将它起个后缀为 .C 的文件名。

② 选择 File/ Open 或 F3 键，出现装入文件名的对话框，这时输入新的源文件名即可。

(2) 装入一个已经存在的源文件

若在主菜单下选 File/ Open，有如下两种方法。

① 输入路径名及文件名，例如，C: \Borlandc\Bin\TESTFILE .C。

② 在装入文件提示框中，键入一通配符（可使用 DOS 中的通配符“ * ”和“ ?”），然后按 Enter 键。键入“ * .* ”，将列出当前目录中的所有文件以及其他目录的目录名，用→，←，↑，↓键移动高亮度条来选择文件名，按回车键后，所选文件就被装入到编辑窗口。

BC 可以同时装入多个文件，每个文件分别占用一个窗口。F6 键是一个开关，可在当前编辑文件与以前装入文件之间进行切换；也可以直接使用鼠标单击所需窗口。

(3) 保存源文件

按热键 F2 或在主菜单下选择 File/ Save，便可把源文件保存到磁盘上。

(4) 换名保存一输出文件

可把当前编辑文件作为一个新文件写盘或覆盖已存在的文件。即写到当前目录（默认时）或指定不同的驱动器和目录。具体做法是在主菜单下选择 File/ Save as，然后在新文件名提示框中，输入新文件名的完整路径名。例如：

C: \DIR\SUBDIR\FILENAME .C

然后按 Enter 键。如果文件已经存在，编辑程序在执行写操作前将询问用户是否真