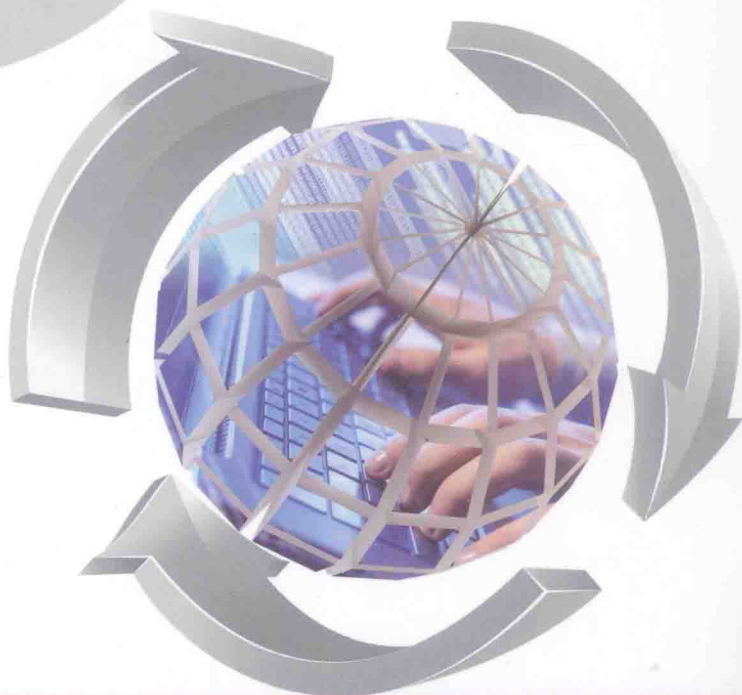




普通高等教育“十二五”规划教材 计算机系列
中国科学院教材建设专家委员会“十二五”规划教材

C语言程序设计

唐国民 王智群 主编



科学出版社

普通高等教育“十二五”规划教材 计算机系列
中国科学院教材建设专家委员会“十二五”规划教材

C 语言程序设计

唐国民 王智群 主编

蒋云良 张建宏 毛德力 副主编

科学出版社

北 京

内 容 简 介

本书根据高等学校 C 语言教学大纲编写,以培养学生基本的程序设计能力为目标。内容包括 C 语言的基础知识和程序设计思想, C 语言的基本数据类型、运算符及表达式、常量、变量、数组、字符串,分支程序、循环程序、函数等基本程序设计技术,指针、结构、联合、位操作及文件 I/O 等高级编程技术和表达技巧。本书将 C 语言程序设计中的内容巧妙进行安排,突出介绍了重点、难点及易错点,并用大量的例题进行分析说明,每章后都附有本章小结和习题。为便于读者上机实践,本书还用专门篇幅介绍了如何用 C-Free 3.5 和 Visual C++ 6.0 集成环境编辑、调试和运行 C 程序。全书内容全面,概念清楚,通俗易懂,循序渐进,重点突出,实用性强。

本书可作为高等学校计算机相关专业的程序设计教材,也可作为高等学校学生和广大计算机爱好者学习 C 语言的自学教材。

图书在版编目(CIP)数据

C 语言程序设计/唐国民,王智群主编. —北京:科学出版社,2014
ISBN 978-7-03-040890-7

I. ①C… II. ①唐… ②王… III. ①C 语言—程序设计—教材
IV. ①TP312

中国版本图书馆 CIP 数据核字(2014)第 119572 号

责任编辑:赵丽欣/责任校对:马英菊
责任印制:吕春珉/封面设计:东方人华平面设计部

科学出版社 出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

铭浩彩色印装有限公司印刷

科学出版社发行 各地新华书店经销

*

2014 年 6 月第 一 版 开本:787×1092 1/16

2014 年 6 月第一次印刷 印张:17

字数:391 000

定价:36.00 元

(如有印装质量问题,我社负责调换〈骏杰〉)

销售部电话 010-62142126 编辑部电话 010-62138978-2009

版权所有,侵权必究

举报电话:010-64030229; 010-64034315; 13501151303

前 言

C 语言作为一门理想的结构化程序设计语言,自 1972 年诞生以来,随着它自身的不断发展,逐渐被人们所认识并广泛地应用于编制各种系统软件和应用软件。由于它具有功能丰富、表达能力强、使用灵活、应用广泛、编写的程序效率高及可移植性好等优点,越来越显示出它的强大生命力。近年来,在 C 语言的基础上开发了面向对象的程序设计语言 Turbo C++和 Borland C++及 Visual C++,又使得 C 语言延伸成为了面向对象的程序设计技术的主要工具。

本书作为计算机程序设计课程的教材,详细地讲述了 C 语言的语法、语义,并以大量的例子来讲解 C 语言的各种成分的用法。由于 C 语言涉及的概念比较丰富,语法规则灵活、难学、难记,会给初学者完整掌握它带来一定的困难。本书围绕 C 语言的核心部分,抓住重点,突破难点,边学边用,并力求在介绍概念时叙述准确、通俗易懂、循序渐进,有助于读者准确理解,全面掌握 C 语言。

全书共分 10 章。第 1 章介绍了 C 语言程序的基本结构及编程环境;第 2 章介绍了基本数据类型及其运算;第 3 章结合实例介绍了顺序、分支和循环三种控制语句的使用;第 4 章介绍了一维数组、二维数组和字符串的应用;第 5 章介绍了指针的基本概念及其应用;第 6 章讲解了函数和程序结构方面一些有一定深度的内容;第 7 章介绍了结构体、联合体及枚举类型在编程中的应用;第 8 章介绍了编译预处理知识及自定义新类型名;第 9 章和第 10 章分别介绍了文件的使用和位操作知识。

本书主要是为大学计算机和相关专业的学生学习计算机程序设计课程而编写的,也可作为计算机水平考试、补习班、各种计算机培训班和自学辅导班的教材。

本书配有《C 语言程序设计实践教程》(张建宏、唐国民主编,科学出版社出版)一书读者可以通过阅读该书中大量的实践题目来领会本书中的知识。《C 语言程序设计实践教程》中还介绍了 Turbo C、C-Free 3.5 和 Visual C++ 6.0 等 C 语言程序编程环境的使用方法。

参加本书编写的有唐国民、王智群、蒋云良、张建宏、李树东、王德力等。

由于作者水平有限,书中难免存在错误和不足之处,敬请读者批评指正。

目 录

第1章 C语言简介	1
1.1 C语言的发展史和特点	1
1.1.1 C语言的发展史	1
1.1.2 C语言的特点	2
1.2 C语言程序的基本结构	2
1.3 C语言程序的编程环境及上机步骤	4
1.3.1 C-Free 3.5 集成开发环境	4
1.3.2 Visual C++集成开发环境	10
本章小结	13
习题	14
第2章 基本数据类型及运算	15
2.1 标识符	16
2.1.1 关键字	16
2.1.2 用户定义的标识符	16
2.2 常量和变量	17
2.2.1 常量	17
2.2.2 变量	22
2.3 运算符和表达式	23
2.3.1 运算符	23
2.3.2 表达式	29
2.4 标准函数和自定义函数	31
2.4.1 标准库函数	31
2.4.2 自定义函数	33
2.4.3 函数调用	33
2.5 格式输入函数和格式输出函数	34
2.5.1 格式输出函数 printf	34
2.5.2 格式输入函数 scanf	39
本章小结	42
习题	43
第3章 语句	45
3.1 基本语句	46
3.1.1 表达式语句	46
3.1.2 空语句	46
3.1.3 复合语句	46
3.2 控制语句	47
3.2.1 if语句	47

3.2.2	switch-case 语句	53
3.3	循环语句	57
3.3.1	while 循环语句	57
3.3.2	do-while 循环语句	60
3.3.3	for 循环语句	63
3.3.4	循环嵌套	66
3.3.5	break 语句和 continue 语句	68
3.3.6	综合应用举例	70
	本章小结	73
	习题	73
第 4 章	数组类型	77
4.1	一维数组	78
4.1.1	一维数组变量的说明	78
4.1.2	一维数组元素的引用和一维数组变量的存储	78
4.1.3	一维数组的应用	79
4.2	字符数组和字符串	82
4.2.1	字符数组的定义	82
4.2.2	字符数组的输入/输出	82
4.2.3	与字符串处理有关的几个函数	85
4.3	二维数组和 multidimensional 数组	88
4.3.1	二维数组变量的说明形式	89
4.3.2	二维数组变量的存储	89
4.3.3	二维数组元素的引用	90
4.3.4	多维数组	90
4.4	字符串数组	93
4.5	数组初始化	94
	本章小结	97
	习题	97
第 5 章	指针类型	100
5.1	指针与指针变量	100
5.2	指针运算	103
5.3	指针与数组	107
5.3.1	一维数组和指针	107
5.3.2	多维数组的地址及其元素的指针表示方法	109
5.3.3	指向一维数组的指针变量	113
5.3.4	指针与字符串	115
5.3.5	指向指针的指针	119
5.3.6	指针初始化	121
	本章小结	124
	习题	125

第 6 章 函数	128
6.1 函数	129
6.1.1 函数的定义形式	129
6.1.2 函数调用及函数的返回值	130
6.1.3 形式参数和实际参数	132
6.1.4 函数声明	136
6.2 函数参数	138
6.2.1 数组作为函数参数	138
6.2.2 字符串作函数参数	143
6.3 指针型函数	145
6.4 函数的嵌套及递归	148
6.4.1 函数的嵌套	148
6.4.2 函数的递归	150
6.5 变量的作用域与存储类型	151
6.5.1 局部变量与全局变量	152
6.5.2 变量的存储类型	154
6.6 内部函数与外部函数	159
6.6.1 内部函数	159
6.6.2 外部函数	159
6.7 动态存储分配	160
本章小结	163
习题	164
第 7 章 结构体、联合体与枚举	167
7.1 结构体类型与结构体类型变量	167
7.2 结构体变量的引用与初始化	171
7.3 结构体数组	173
7.4 结构体与指针	176
7.5 结构体与函数	179
7.6 结构型函数与结构指针型函数	181
7.7 用结构体类型数据处理链表	183
7.7.1 存储空间的动态分配	183
7.7.2 链表	184
7.8 联合体	191
7.9 枚举	194
本章小结	197
习题	198
第 8 章 编译预处理与自定义新类型名	200
8.1 编译预处理	200
8.1.1 “文件包含”指令	201
8.1.2 宏定义指令#define 和#undef	203

8.1.3 宏标识符	207
8.1.4 宏终止指令 #undef	208
8.1.5 条件编译指令	208
8.1.6 #ifdef、#ifndef 指令	211
8.1.7 应用举例	212
8.2 使用 typedef 定义新类型名	215
8.2.1 定义“标准”类型名	215
8.2.2 定义“构造”类型名	216
8.2.3 定义新类型名的一般步骤	217
8.3 如何运行一个多文件的程序	218
本章小结	219
习题	220
第 9 章 输入/输出与文件管理	222
9.1 文件	222
9.1.1 文件、流的概念	222
9.1.2 文件类型与标准流式文件	224
9.1.3 文件的打开/关闭	226
9.1.4 文件 I/O 操作	228
9.1.5 文件的重定位、测试与随机读写	235
9.1.6 检错与处理	238
9.2 应用举例	239
本章小结	244
习题	244
第 10 章 位操作	247
10.1 位逻辑运算	247
10.1.1 按位与运算	247
10.1.2 按位求反运算	249
10.1.3 按位或运算	249
10.1.4 按位异或运算	250
10.2 移位操作	251
10.2.1 左移位操作	251
10.2.2 右移位操作	252
10.3 位操作应用举例	253
10.4 位操作赋值运算	253
10.5 位段	254
本章小结	258
习题	258
附录	260
附录 A 字符与 ASCII 码对照表	260
附录 B 常用 C 库函数	261
主要参考文献	264

本章要点

- C 语言的发展史和特点
- C 语言程序的基本结构
- C 语言的编程环境及上机步骤

本章学习目标

- 了解 C 语言的发展历史和特点
- 熟悉 C 语言程序的基本结构
- 了解并逐步熟悉 C 语言的编程环境
- 掌握 C 语言程序的上机步骤

1.1 C 语言的发展史和特点

1.1.1 C 语言的发展史

C 语言是被广泛应用的计算机高级程序设计语言。使用 C 语言可以编写各种复杂的应用程序和包括操作系统在内的系统程序。

操作系统等系统程序依赖于计算机硬件，以前这类系统程序主要用汇编语言编写，程序的可读性和可移植性都很差，严重妨碍了系统程序的开发和生产。为此，人们便努力寻求一种程序语言，使它既具有高级语言的特性，又具有某些汇编语言的特点，既能描述对硬件的操作，又能编写可读性高便于移植的程序。C 语言就是人们在寻求这样一种集高级语言和汇编语言于一身的高级语言过程中产生的。

C 语言作为 UNIX 操作系统的开发语言与 UNIX 操作系统二者相互依存、相互促进而发展。美国贝尔实验室的 K.Thompson 和 D.M.Ritchie 于 1971 年设计成功了 UNIX 操作系统第 1 版，通过纸带将可执行代码传送到 PDP-7 机上，后又用汇编语言改写，由于汇编语言可读性差且效率低，故 K.Thompson 决定开发一种高级语言来编写 UNIX 操作系统，于是 1972 年 C 语言研制成功并投入使用，1973 年 K.Thompson 和 D.M.Ritchie 把 UNIX 操作系统用 C 语言改写了一遍，增加了多道程序设计功能。到 1975 年，UNIX 第 6 版问世。UNIX 第 7 版在 1978 年研制成功。美国国家标准化协会于 1983 年制定了一个 C 语言标准草案，即

83 ANSI C。随着 UNIX 的巨大成功和被广泛移植到各种机器上, C 语言也被人们所关注。

1989 年, 美国国家标准局 (ANSI) 语言标准化委员会根据 C 语言问世以来的若干补充和修改, 重新制定了 C 语言的标准, 即 89 ANSI C (标准 C), 1994 年制定了 ANSI C++ 标准的草案。它具有高效、灵活、功能丰富、表现力强、移植性好等特点, 既具有高级语言的功能, 还具有机器语言的特征, 因此还可以用于开发计算机系统软件, 成为在计算机程序设计中使用的最广泛的一种语言。

目前 C 语言编译系统有多种版本, 在微机上常用的有 Microsoft C, Turbo C, Quick C 等, 比较流行的版本是 Borland C++ 5.0、C-Free 3.5 和 Visual C++ 6.0。

C++ 在 ANSI C 的基础上, 扩充了类的部分, 因此 ANSI C 是 C++ 的子集。也就是说, 按照 ANSI C 标准编写的程序可以运行在 C++ 的语言环境中。

1.1.2 C 语言的特点

由于 C 语言集中了高级语言和汇编语言的优点, 所以可用它编写不依赖于计算机硬件设施的各种系统程序和应用程序。

C 语言程序可移植性好, 即可从一种类型的计算机系统移植到另一种类型的计算机系统中。

C 语言提供的数据类型丰富, 具有数据类型的构造能力, 特别是指针类型的丰富多样, 非常有助于构造链表、栈、树等。

C 语言包含丰富的运算符, C 语言提供了 40 种运算符, 包含可直接由硬件实现的算术逻辑运算、位运算等, 可以完成其他高级语言难以完成的运算。

C 语言简洁、紧凑, 使用方便灵活。

C 语言生成的目标代码质量高。

C 语言设有预处理功能。

C 语言源程序由一个或若干函数组成。

C 语言虽有诸多优点, 但也有不足。用 C 语言编程自由度大, 语法限制不严谨, 安全性低; 运算符丰富, 运算符的优先级太多, 难记忆。

1.2 C 语言程序的基本结构

本节用三个简单的 C 语言程序说明和分析 C 语言的程序结构及 C 程序的特性。请读者配合实例理解例子后面的几点说明。

【例 1.1】 一个没有任何功能的程序。

程序如下:

```
#include <stdio.h>
main()
{
}
```

运行这个程序, 计算机什么也不做。

【例 1.2】 计算并输出 $x+y$ 的值。

程序如下:

```
#include "stdio.h"
```

```

main()    /*主函数*/
{
    int x,y,c;           /*定义三个整型变量*/
    x=10;                /*给变量 x 赋初值 10*/
    y=20;                /*给变量 y 赋初值 20*/
    c=x+y;               /*计算 x+y 并将它赋给变量 c*/
    printf ("%d+%d=%d\n",x,y,c); /*输出 x+y 的值*/
}

```

运行这个程序，将输出 10+20=30。printf 是 C 函数库中的一个函数，此处是调用函数 printf 来完成输出功能。具体 printf 的使用后续章节将详细介绍。printf 函数的头文件是 stdio.h，因此在使用该函数时，一定要用 #include 命令将 stdio.h 文件包含到源程序文件中，否则编译时将出现错误。

C 语言支持结构化程序设计，结构化程序设计是一种良好的程序设计技术，它由著名计算机科学家 E.W.Dijkstra 于 1969 年提出，它强调程序的模块化和规范化。为了便于理解结构化程序设计思想；下面用结构化程序设计思想来完成上例。

【例 1.3】 结构化编程计算并输出 x+y 的值。

程序如下：

```

#include<stdio.h>
main()
{
    int x,y,c;           /*定义三个整型变量*/
    int add (int a,int b); /*函数说明*/
    x=10;                /*给变量 x 赋初值 10*/
    y=20;                /*给变量 y 赋初值 20*/
    c = add(x,y);         /*调用函数 add() 计算 x+y 的值*/
    printf ("%d+%d=%d\n",x,y,c); /*输出 x+y 的值*/
}
/*以下定义函数 add()*/
int add (int a,int b)    /*用户自定义函数*/
{
    return a+b;          /*返回 a+b 到调用 add() 函数处*/
}

```

本例中源程序包含两个函数：主函数 main 与求和函数 add。在函数 main 中，前两行是说明部分，后面的多行是可执行语句部分。在函数 add 中，圆括号中是形式参数 a、b 及其类型说明，花括号中是函数体部分，可由变量的类型说明和可执行语句部分组成，在本例中只有一行可执行语句。函数 add 的作用是将 a、b 的值相加，然后由返回语句 return 将相加结果返回给主调函数 main 中对 add 的调用处（主函数中的 c = add (x, y)）。

例 1.2 与例 1.3 的重要区别在于：例 1.3 将计算两个数的和进行了很好的封装，实现了两个数相加计算的模块化，在程序中只要涉及两个数相加时，都可通过调用函数 add 实现。例 1.3 只是一个简单的实例，但是对一个复杂的系统，结构化程序思想将体现出它的巨大优越性。

通过以上几个例子，可以看到以下几点。

1) C 语言程序书写自由

C 语言程序在书写时是自由的，几个说明项或几个语句可以书写在一行上；一个语句也可以分写在多行上，但是一个词或一个数不可以分两行写，语句中的空格和回车符忽略

不计。程序中的语句写什么格式主要考虑程序易读和便于维护。

2) C语言程序的总体结构

一个C程序可由多个源程序文件组成,每个源程序文件可由一个或多个函数组成。在程序的所有函数中必须有一个且只有一个名为 `main()` 的函数,称它为主函数。程序的执行总是从 `main()` 开始。`main()` 可以放在程序的任何位置。程序的开始部分可以根据程序的需要写出以“#”为首字符的编译预处理行和全程序使用的变量说明,其后是若干函数段(函数),每个段的最前面是函数名,除了主函数名是 `main` 之外,其他的函数名由程序员自定,每个函数名后一定要有 `()` 作为函数的标志,`()` 内可以放函数的形式参数表,也可以没有参数,没有参数时,`()` 也不能省略。圆括号中每一个形式参数的前面要加上类型说明,最后是用花括号“{}”括起来的函数的主体。

3) 函数的主体

函数的主体也叫函数体。最简单的函数体是空的,即其中一个语句都没有,执行它什么也不做。一般来说,根据程序功能的需要,函数体内可以由两部分组成,即说明部分和可执行部分(语句部分)。说明部分中可以包含多个说明项,用以说明函数中所使用的变量和需调用的其他函数等。语句部分可以包含多个可执行语句。

4) C语言的分隔符

C语言中每个简单语句、说明及变量定义之后都必须以分号“;”结尾,分号是它们必要的组成部分;并列的标识符或项之间用逗号“,”分隔;两个相邻的关键词之间至少用一个空格相间。

5) C语言的程序注释

为了使程序具有易读性,还要在程序中设置注释部分,它可以插在程序的任何地方。注释的内容写在“/*”和“*/”之间,注释不允许嵌套。程序运行时系统忽略其中的内容。注意:在“/”和“*”之间不要有空格。

1.3 C语言程序的编程环境及上机步骤

从编写完整的C程序到在计算机上运行,最后得出结果,大致要经过编辑、编译、连接、运行四个步骤,即首先将源程序输入到计算机中,形成源程序文件;然后由C程序编译系统对源程序进行编译,将源程序编译成机器代码语言的目标程序;再后,由连接系统将目标程序模块(如果使用了库函数等)连接成为可执行程序文件;最后,运行这个可执行程序文件,从而得到运行结果。

其实,在程序的调试过程中,有可能会出现意想不到的错误,需要程序员来排除。图1.1给出了在计算机上调试程序过程的顺序,这种顺序图叫“流程图”。在流程图中使用的各种符号含义如图1.2所示。

目前,支持C语言集成开发环境的工具很多,这里主要介绍两款比较实用的集成开发环境:C-Free和Visual C++。

1.3.1 C-Free 3.5 集成开发环境

C-Free是一款基于Windows的C/C++集成化开发软件(IDE)。利用该软件,使用者可以轻松地编辑、编译、连接、运行、调试C/C++程序。特别地,这款软件对于C/C++的初

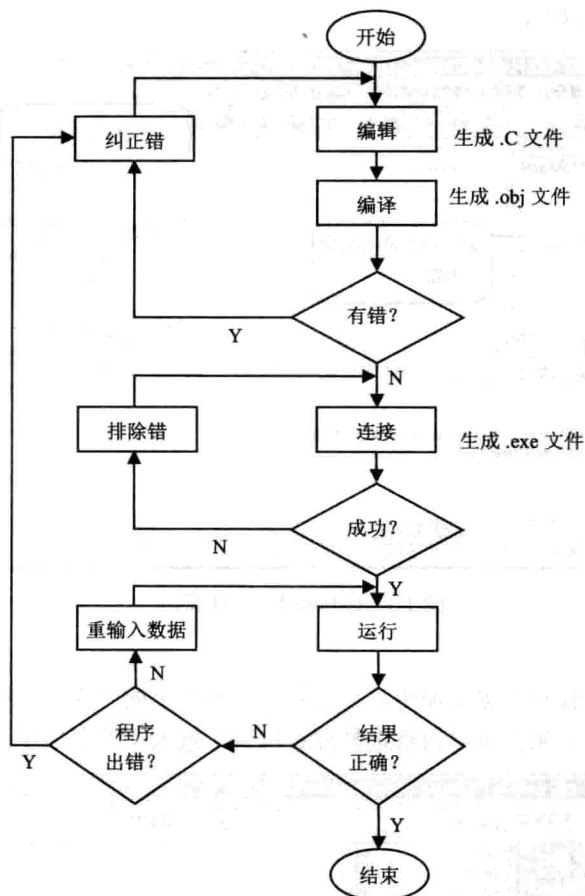


图 1.1 C 程序调试过程流程图

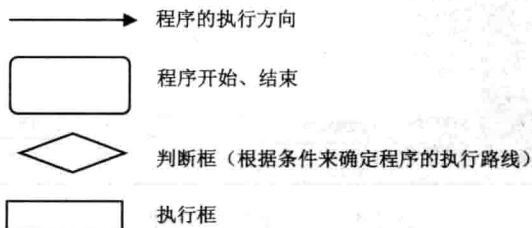


图 1.2 程序流程图中所用符号含义

学者非常容易使用，是迅速提高 C/C++ 水平的好帮手。当然，C/C++ 高手也会在其中找到许多惊喜的功能。

启动进入 C-Free 3.5，打开如图 1.3 所示的工作窗口。

窗口由菜单栏、工具栏、文件列表窗或函数列表窗、程序编辑区、文件浏览窗和编译信息窗等部分组成。各部分功能说明参见 C-Free 3.5 中的“帮助”。

1. C-Free 3.5 的使用

1) 编辑一个新文件

选择“文件”→“新建”菜单命令，或者单击工具栏上的“新建”按钮，可新建一个

空白文件，即可输入源程序。

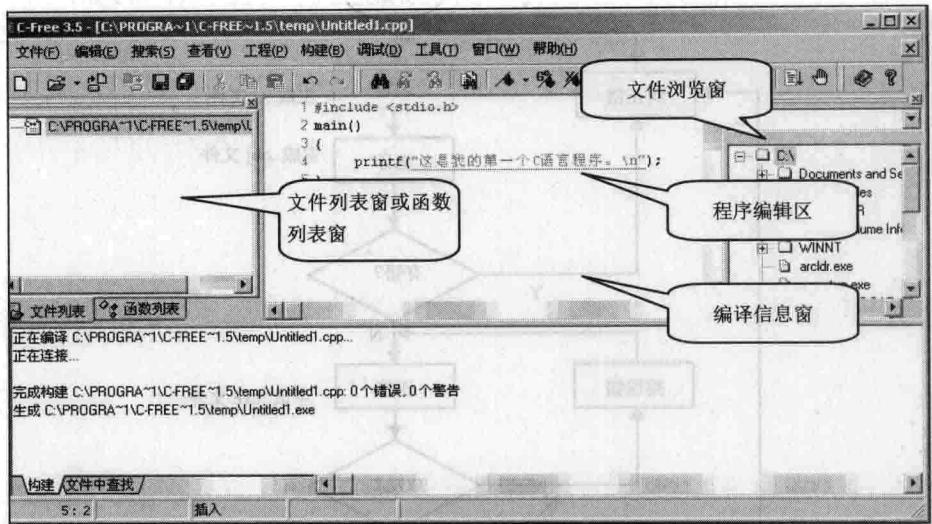


图 1.3 C-Free 3.5 工作窗口

2) 保存文件

选择“文件”→“保存”菜单命令，或者单击工具栏上的“保存”按钮，弹出如图 1.4 所示的“另存为”对话框。用户可以自行选择保存位置，输入文件名后单击“保存”按钮即可。



图 1.4 “另存为”对话框

3) 打开一个已有的文件

选择“文件”→“打开”菜单命令，或者单击工具栏上的“打开”按钮，弹出如图 1.5 所示的“打开”对话框。

在查找范围列表找到文件的保存位置，选择要打开的文件，单击“打开”按钮。

2. C-Free 3.5 的程序调试

1) 快速运行程序

运行一个程序只需按如下步骤操作即可。

- ① 输入源程序并进行静态检查。
- ② 编译源程序，输入完源程序后，选择“构建”→“编译”菜单命令。当程序存在语

法错误时，编译器将提示语法错误，如图 1.6 所示。



图 1.5 “打开”对话框



图 1.6 编译源程序及错误信息

此时要对源程序进行修改，直到编译通过为止。在编译信息窗口显示 0 个错误，0 个警告表示程序已无语法错误。

2) 调试程序

C-Free 3.5 提供了基本的调试功能。下面结合一个简单的例子，详细介绍 C-Free 3.5 的调试功能。

① 设置断点。

假设有如下一段程序：

```
#include<stdio.h>
main()
{
```

```

int i=2,n;
printf("请输入n:\n");
scanf("%d",&n);
if(n<0)
    n=-n;
printf("%d的质因子有:",n);
while(n>=1)
{
    if(n%i==0)
    {
        printf("%d ",i);
        n=n/i;
    }
    else
        i++;
}
}

```

设置断点,最方便快捷的方法是将鼠标移到代码区中某一条代码的左边(灰色区域),光标由“I”字形变成黑色断点形状,然后单击,看到红色断点说明设置完成,如图 1.7 所示。

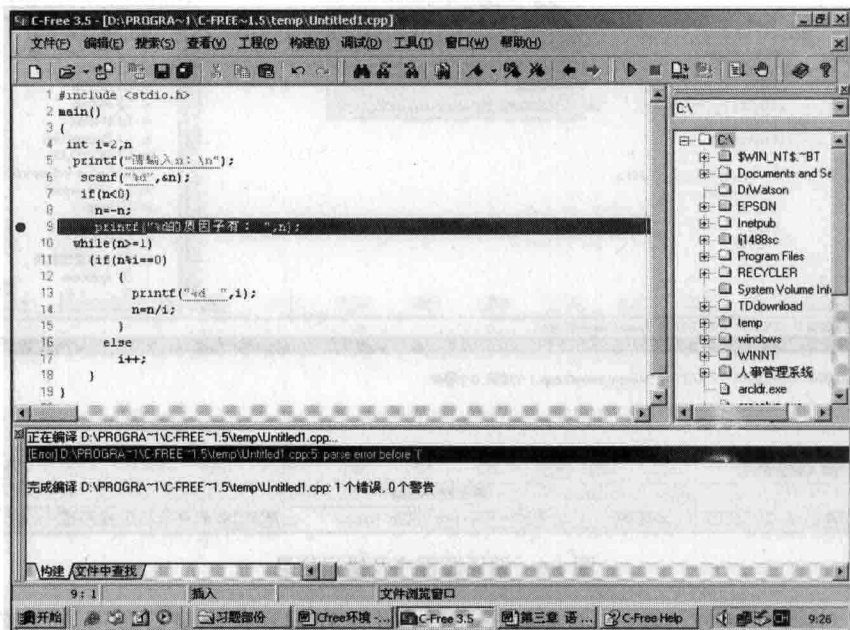


图 1.7 设置断点

② 断点管理。方法是:选择“调试”→“断点管理”菜单命令,弹出如图 1.8 所示的对话框。

③ 开始调试。方法是:单击工具栏上的“调试”按钮,程序开始调试。程序只有以“Debug”方式构建,才能够被调试。

一旦程序开始调试,C-Free会自动显示“Debug”工具栏,如图 1.9 所示。

④ 退出调试。一旦程序处于调试状态,可以单击“Debug/调试”工具栏上的“退出调试”按钮,程序退出调试状态,回到编辑状态。

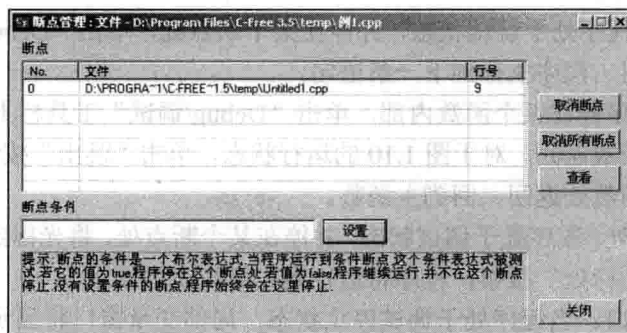


图 1.8 “断点管理”对话框

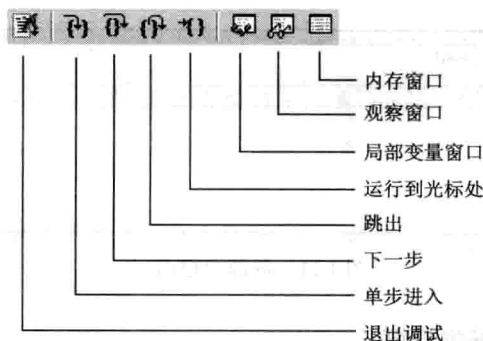


图 1.9 调试工具栏

⑤ 单步进入。当程序处于调试状态，并停在某个断点处，如果该断点处的语句是一条函数调用，单击“Debug/调试”工具栏上的“单步进入”按钮，程序将进入这个函数，可对这个函数进行调试，如图 1.10 所示。

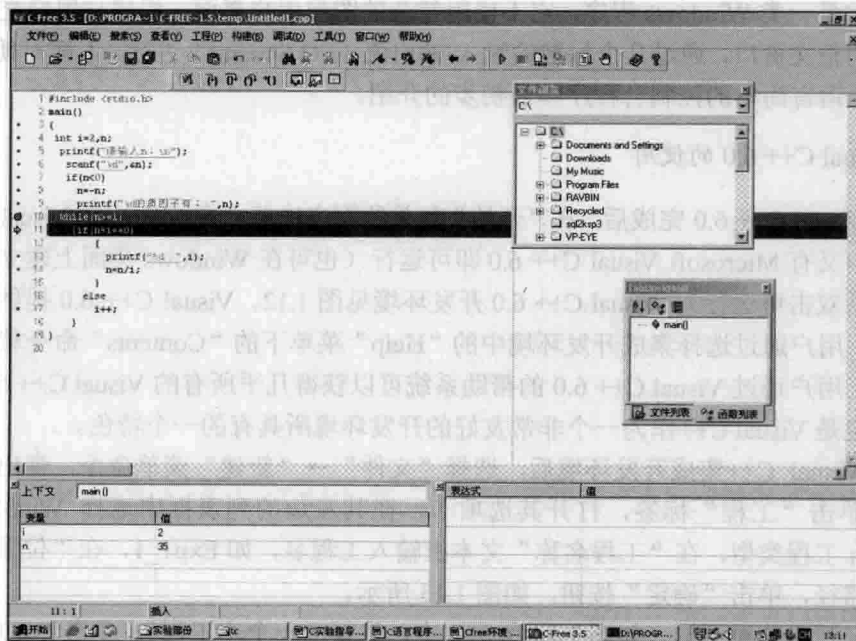


图 1.10 单步进入