21世纪大学计算机系列教材

C/C++ 程序设计教程

（第2版）

孙淑霞　肖阳春　魏　琴　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY　http://www.phei.com.cn

21世纪大学计算机系列教材

C/C++ 程序设计教程

（第2版）

孙淑霞　肖阳春　魏 琴　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书作为C/C++程序设计课程的教科书，共由12章组成。其主要内容包括：C语言简单程序的编写和调试，C语言程序设计基础（其中包括：基本数据类型、基本输入与输出函数以及运算符和表达式），控制结构，数组，指针，函数，编译预处理与变量的存储类型，文件，结构体与共用体，图形程序设计基础，C++程序设计基础，查找与排序。每章后面都附有一定量的编程练习题，书后附有习题参考答案。全书内容安排紧凑，简明扼要，由浅入深，实用性强。该书与《C/C++程序设计实验指导与测试》（第2版）配套使用，将为学生编程能力的提高和课后自学提供更好的帮助。

本书可作为大专院校非计算机专业本科生、研究生的相关课程的教学用书，也可作为计算机专业学生学习C/C++程序设计的教材，同时还可供自学者参考。

图书在版编目（CIP）数据

C/C++程序设计教程/孙淑霞，肖阳春，魏琴编著. —2版. —北京：电子工业出版社，2007.3
（21世纪大学计算机系列教材）
ISBN 978-7-121-03510-4

Ⅰ.C… Ⅱ.①孙…②肖…③魏… Ⅲ.C语言—程序设计—高等学校—教材 Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第017781号

责任编辑：王昌铭
印　　刷：涿州市京南印刷厂
装　　订：涿州市桃园装订有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编　100036
开　　本：787×1092　1/16　印张：23　字数：586千字
印　　次：2007年3月第1次印刷
印　　数：5 000册　　定价：29.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系电话：（010）68279077；邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

第 1 版前言

C 语言是应用很广泛的一种语言，它的结构简单、数据类型丰富、表达能力强、使用灵活方便。C 语言既有高级语言的优点，又具有低级语言的许多特点。用 C 语言编写的程序，具有速度快、效率高、代码紧凑、可移植性好的优点。利用 C 语言，可编制各种系统软件（例如著名的 UNIX 操作系统就是用 C 语言编写的）和应用软件。

C++是一种混合语言，既有面向过程的知识，又有面向对象的理论。经过几年的教学实践，我们认为把面向过程的程序设计作为切入点，由面向过程到面向对象，由浅入深，循序渐进的教学方式比较容易被学生所接受。因此，本书在最后一章介绍了 C++程序设计的基础知识。

本教材由 11 章组成。每一章的基本内容如下：

第 1 章 C 语言简单程序的编写和调试，介绍 C 程序的基本结构。

第 2 章 C 语言程序设计基础，介绍 C 语言的基本数据类型。

第 3 章 控制结构，介绍 C 程序的 3 种控制结构。

第 4 章 数组，介绍一维数组和二维数组的定义和使用。

第 5 章 指针，重点介绍指针变量、指针数组、指向指针的指针等的定义和使用。

第 6 章 函数，讲解函数的定义、函数的调用，函数参数的传递。

第 7 章 编译预处理与变量的存储类型，介绍编译预处理命令和变量的几种存储类型。

第 8 章 文件，介绍文件操作的方法，数据文件的读和写。

第 9 章 结构体与共用体，介绍结构体与共用体的使用，以及它们对内存的占用情况。

第 10 章 图形程序设计基础，介绍编写图形程序的基本步骤，基本图形函数。

第 11 章 C++程序设计基础，介绍 C++对 C 的扩充，以及面向对象的程序设计基础。

由于教材的主要对象是低年级大学生，因此，书中的内容，尤其是举例考虑了低年级学生所具有的数学基础知识，力求使学生学习起来的难度较小。本教材在编写中努力做到概念清楚、实用性强、通俗易懂。在编写中引入了大量的实例来说明相关的知识点，力求让读者尽快上手编写简单程序，引发学习的兴趣。

本书在组织编写上有以下特点：

1．在内容的组织上考虑了 C 语言的特点。例如，在讲解数组后，紧接着就进行指针的讲解，使读者很容易将数组与指针联系起来，更好地理解指针。

2．文件是学生学习的一大难点。本书将文件的使用提前讲解，使读者尽早接触文件，掌握文件的基本操作，给大批量数据的处理带来方便。同时可以较好地解决学生在学习 C 语言时不能熟练地掌握文件的使用方法，而给学习 C 语言留下一大遗憾的问题。

3．本书通过一个关于学生成绩处理的实例，从简单变量到结构体，始终用该实例贯穿各章。使读者通过一个实例循序渐进地、有比较地进行学习。

4．本书提供了习题中的全部参考答案。所有程序均在 Turbo C/Visual C++ 6.0 环境下调试通过。由于篇幅有限，书中的程序只给出了一种参考程序，读者在学习过程中可以举一反三。

与本书一起出版的《C/C++程序设计实验与习题指导》是本书的配套教材，在学习的过程中可通过完成该教材中相应的习题和上机实践加深对所学知识的理解，达到真正掌握C/C++程序设计的目的。

本书第1,4,5,6,8,9章由孙淑霞编写，第2,3章由何建军编写，第7章由肖阳春编写，第10,11章由彭舰编写。丁照宇、魏琴、鲁红英、袁爱新也参加了本书的部分编写工作。由于作者水平有限，书中难免有错误之处，请读者批评指正。

最后要感谢高等学校电子信息类教材编委会在本书的出版过程中给予的指导和帮助，特别要感谢王昌铭老师在该书的出版过程中所做的大量工作，同时还要感谢电子工业出版社给予的大力支持。

编著者

2004年10月

第 2 版前言

本书较为全面地讲述了 C 语言的主要内容，以及 C++语言的部分基本内容，原教材在两年的使用中，得到不少学校的支持，同时提出了一些宝贵意见。为此，我们对原教材进行了修改，使本教材在组织编写上除了具有原教材的特色以外，还具有以下特点：

1．我们坚持把面向过程的程序设计作为切入点，由面向过程到面向对象，由浅入深，循序渐进，使其教学方式容易被学生接受。把 C 和 C++的内容分开，是为了教师更容易选择章节进行教学。

2．为了使初学者更好地理解本书的内容，第 2 版中取消了原教材中较难的例题，增加了一些较简单，且适用的例题。通过学习，使更多的学生能够举一反三。

3．在程序设计中，经常使用查找和排序算法。因此，第 2 版在原教材的基础上，增加了第 12 章，介绍常用的查找与排序算法。为了使学生从一开始就学会在 Visual C++ 6.0 环境下编写和调试程序，在第 1 章中增加了 C++程序的编辑、编译和调试等内容。

4．为了进一步提高学生的编程能力，增加了部分章的编程练习。

5．与该教材配套的《C/C++程序设计实验指导与测试》（第 2 版）为学生的课后练习和测试提供了更好的指导。

要想学好程序设计课程，需要教师和学生的共同努力。对于学习者来说，需要多动手，多实践，多思考。一分耕耘，一分收获，坚持耕耘定会得到意想不到的收获。

本书第 1, 4, 5, 6, 8, 9,12 章由孙淑霞编写，第 2, 3, 7 章由肖阳春编写，第 10,11 章由彭舰编写。魏琴、丁照宇、李思明、刘焕君、陈佩良也参加了本书的部分编写工作。魏琴、刘焕君为本课程制作了美观、符合授课要求的课件。同时还要感谢刘焕君、李思明、丁照宇、陈佩良、鲁红英、安红岩、高嵩在精品课程申报、建设中所做的工作。

由于编著者水平有限，书中难免有错误之处，请读者批评指正。

最后要感谢对本书第 1 版提出宝贵意见的老师和读者以及电子工业出版社在本书的出版过程中给予的大力支持。

该书作为四川省精品课程《C/C++程序设计》使用的教材，进行了配套的资源建设。对于使用本教材的学校，如果需要课件、示例源程序、习题源程序等，可以从该课程的精品课程网站 http://202.115.138.30/ec3.0/c57/zcr-1.htm 上直接下载，也可以直接与我们联系（邮件地址：ssx@cdut.edu.cn）。

编著者

2007 年 1 月

目　录

第 1 章　C/C++语言简单程序的编写和调试 …… (1)
1.1　C 语言的特点 …… (1)
1.2　程序与程序设计 …… (2)
1.3　C 程序的基本结构 …… (2)
1.4　C 程序的调试 …… (6)
1.4.1　编辑 …… (6)
1.4.2　编译 …… (8)
1.4.3　连接 …… (9)
1.4.4　运行 …… (10)
1.4.5　程序的跟踪调试 …… (10)
1.5　C++程序的实现 …… (11)
1.5.1　C++源程序的建立与编辑 …… (11)
1.5.2　单文件程序的编译和运行 …… (12)
1.5.3　多文件程序的编译和运行 …… (13)
1.6　程序举例 …… (15)
习题 …… (16)
第 2 章　C 语言程序设计基础 …… (17)
2.1　基本数据类型 …… (17)
2.1.1　变量 …… (17)
2.1.2　常量 …… (21)
2.2　运算符和表达式 …… (23)
2.2.1　运算符和表达式概述 …… (24)
2.2.2　算术运算符和算术表达式 …… (25)
2.2.3　关系运算符和关系表达式 …… (26)
2.2.4　逻辑运算符和逻辑表达式 …… (27)
2.2.5　赋值运算符和赋值表达式 …… (30)
2.2.6　自增、自减运算符及其表达式 …… (32)
2.2.7　逗号运算符和逗号表达式 …… (34)
2.2.8　位运算符 …… (35)
2.2.9　其他运算符 …… (37)
2.3　基本输入与输出函数 …… (39)
2.3.1　格式输入函数 scanf() …… (39)
2.3.2　格式输出函数 printf() …… (42)
2.3.3　字符输入函数 getchar() …… (46)

2.3.4 字符输出函数 putchar() …… (47)
2.4 程序举例 …… (47)
习题 …… (48)
第3章 控制结构 …… (49)
3.1 C语言程序结构 …… (49)
3.1.1 C语句概述 …… (49)
3.1.2 C程序的3种基本结构 …… (51)
3.2 if语句 …… (51)
3.2.1 简单if语句 …… (51)
3.2.2 if语句的一般格式 …… (52)
3.2.3 if语句的嵌套 …… (53)
3.2.4 if 语句举例 …… (55)
3.3 switch语句 …… (56)
3.3.1 switch语句的一般格式 …… (56)
3.3.2 switch语句的执行过程 …… (57)
3.3.3 switch语句举例 …… (57)
3.4 while语句 …… (60)
3.4.1 while语句的一般格式 …… (60)
3.4.2 while语句举例 …… (61)
3.5 do-while语句 …… (62)
3.5.1 do-while语句的一般格式 …… (62)
3.5.2 while语句的举例 …… (63)
3.6 for语句 …… (65)
3.6.1 for语句的一般格式 …… (65)
3.6.2 for语句的执行过程 …… (65)
3.6.3 for语句的应用 …… (65)
3.7 几种控制语句的比较和结构嵌套 …… (67)
3.7.1 几种控制语句的比较 …… (67)
3.7.2 结构嵌套 …… (68)
3.8 break和continue语句 …… (69)
3.8.1 break语句 …… (69)
3.8.2 continue语句 …… (70)
3.9 goto语句 …… (70)
3.10 程序举例 …… (71)
习题 …… (76)
第4章 数组 …… (77)
4.1 一维数组 …… (77)
4.1.1 一维数组的引入 …… (77)
4.1.2 一维数组的定义和初始化 …… (80)
4.1.3 一维数组元素的引用 …… (81)

4.2 多维数组 …… (84)
4.2.1 二维数组的定义和初始化 …… (85)
4.2.2 二维数组元素的引用 …… (86)
4.3 字符数组 …… (88)
4.3.1 字符串与一维字符数组 …… (88)
4.3.2 二维字符数组 …… (89)
4.3.3 字符数组的输入和输出 …… (90)
4.3.4 字符串处理函数 …… (91)
4.4 程序举例 …… (95)
习题 …… (100)
第 5 章 指针 …… (102)
5.1 指针和地址 …… (102)
5.2 指针变量的定义和引用 …… (104)
5.3 指针运算 …… (106)
5.4 指针与数组 …… (108)
5.4.1 指向一维数组的指针 …… (108)
5.4.2 指向二维数组的指针 …… (111)
5.5 指针与字符串 …… (116)
5.6 指向指针的指针 …… (119)
5.7 用于动态内存分配的函数 …… (121)
5.8 程序举例 …… (123)
习题 …… (128)
第 6 章 函数 …… (129)
6.1 函数的定义和声明 …… (129)
6.1.1 函数的引入 …… (129)
6.1.2 函数的定义 …… (130)
6.2 函数的调用与返回 …… (132)
6.2.1 函数的调用 …… (132)
6.2.2 函数的返回 …… (134)
6.3 函数的参数 …… (136)
6.3.1 传值调用 …… (136)
6.3.2 传址调用 …… (138)
6.4 命令行参数 …… (144)
6.5 递归调用 …… (147)
6.6 程序举例 …… (151)
习题 …… (154)
第 7 章 编译预处理与变量的存储类型 …… (157)
7.1 宏定义 …… (157)
7.1.1 不带参数宏的定义 …… (157)
7.1.2 带参数宏的定义 …… (161)

7.2 文件包含 …… (163)
7.3 变量的存储类型 …… (164)
7.3.1 自动变量 …… (165)
7.3.2 静态变量 …… (165)
7.3.3 寄存器变量 …… (167)
7.3.4 外部变量 …… (168)
7.4 多个源程序文件下的变量使用 …… (170)
7.5 程序举例 …… (171)
习题 …… (173)
第8章 文件 …… (174)
8.1 文件的基本概念 …… (174)
8.1.1 缓冲文件系统 …… (175)
8.1.2 非缓冲文件系统 …… (176)
8.1.3 文件指针和文件位置指针 …… (176)
8.2 文件的打开与关闭 …… (176)
8.2.1 文件的打开函数 fopen() …… (176)
8.2.2 文件的关闭函数 fclose() …… (178)
8.3 文件的输入/输出函数 …… (178)
8.3.1 按字符方式读/写文件的函数 fgetc(), fputc() …… (178)
8.3.2 按行方式读/写文件的函数 fgets(), fputs() …… (180)
8.3.3 按格式读/写文件的函数 fprintf(), fscanf() …… (184)
8.3.4 按块读/写文件的函数 fread(), fwrite() …… (185)
8.3.5 文件定位函数 …… (186)
8.4 错误检测函数 …… (189)
8.5 程序举例 …… (190)
习题 …… (194)
第9章 结构体与共用体 …… (197)
9.1 结构体 …… (197)
9.1.1 结构体类型 …… (197)
9.1.2 结构体变量的定义 …… (198)
9.1.3 结构成员的引用 …… (199)
9.1.4 结构体变量的初始化 …… (201)
9.2 结构数组 …… (202)
9.3 结构指针 …… (206)
9.4 结构与函数 …… (208)
9.4.1 结构体变量作为函数的参数 …… (208)
9.4.2 结构体变量的地址作为函数的参数 …… (209)
9.4.3 结构数组作为函数的参数 …… (211)
9.5 共用体 …… (213)
9.6 枚举 …… (215)

9.7 用 typedef 定义类型 …… (217)
9.8 链表 …… (218)
9.8.1 单向链表 …… (218)
9.8.2 链表的建立 …… (219)
9.8.3 链表的插入和删除 …… (221)
9.9 程序举例 …… (224)
习题 …… (229)
第 10 章 图形程序设计基础 …… (231)
10.1 图形适配器的基本工作方式 …… (231)
10.2 常用图形函数 …… (232)
10.3 图形程序举例 …… (238)
习题 …… (239)
第 11 章 C++程序设计基础 …… (241)
11.1 C++程序结构 …… (241)
11.2 C++的输入/输出流 …… (242)
11.2.1 输出流（cout） …… (242)
11.2.2 输入流（cin） …… (242)
11.3 引用 …… (243)
11.4 函数的重载 …… (245)
11.5 带默认参数的函数 …… (247)
11.6 C++新增运算符 …… (248)
11.6.1 作用域运算符 …… (248)
11.6.2 动态内存分配与撤消运算符 …… (248)
11.7 const 修饰符 …… (249)
11.8 类和对象 …… (250)
11.8.1 类和对象的定义 …… (250)
11.8.2 构造函数和析构函数 …… (255)
11.8.3 类的友元 …… (259)
11.8.4 this 指针 …… (261)
11.9 重载 …… (262)
11.9.1 类成员函数重载 …… (263)
11.9.2 类构造函数重载 …… (264)
11.9.3 运算符重载 …… (264)
11.10 继承 …… (267)
11.10.1 基类与派生类 …… (268)
11.10.2 public 继承 …… (269)
11.10.3 private 继承 …… (273)
11.10.4 protected 继承 …… (274)
11.10.5 多继承 …… (274)
11.10.6 派生类的构造函数和析构函数 …… (277)

11.11 多态性和虚拟函数 …… (282)
11.11.1 多态性 …… (282)
11.11.2 虚拟函数 …… (283)
11.11.3 虚拟析构函数 …… (290)
习题 …… (291)
第 12 章 查找与排序 …… (292)
12.1 查找算法概述 …… (292)
12.2 顺序查找 …… (292)
12.3 二分查找 …… (294)
12.4 排序算法概述 …… (295)
12.5 插入排序 …… (296)
12.5.1 直接插入排序 …… (296)
12.5.2 二分插入排序 …… (298)
12.5.3 希尔（Shell）排序 …… (299)
12.6 交换排序 …… (300)
12.6.1 冒泡排序 …… (300)
12.6.2 快速排序 …… (301)
12.7 选择排序 …… (303)
习题 …… (304)
习题参考答案 …… (305)
第 1 章 C/C++语言简单程序的编写和调试 …… (305)
第 2 章 C 语言程序设计基础 …… (305)
第 3 章 控制结构 …… (307)
第 4 章 数组 …… (312)
第 5 章 指针 …… (319)
第 6 章 函数 …… (324)
第 7 章 编译预处理与变量的存储类型 …… (331)
第 8 章 文件 …… (334)
第 9 章 结构体与共用体 …… (337)
第 10 章 图形程序设计 …… (342)
第 11 章 C++程序设计基础 …… (344)
第 12 章 查找与排序 …… (346)
附录 …… (350)
附录 A 常用字符与代码对照表 …… (350)
附录 B C 语言中的关键字 …… (352)
附录 C 运算符的优先级与结合性 …… (352)
参考文献 …… (354)

第 1 章　C/C++语言简单程序的编写和调试

C 语言是国际上应用最广泛的几种计算机语言之一。它不仅可以用于编写系统软件，如操作系统、编译系统等，还可以用于编写应用软件。最初的 C 语言是为描述和实现 UNIX 操作系统而设计的，随后 C 语言又随 UNIX 的出名而闻名。

随着计算机的发展，出现了不同版本的 C 语言，它们的差异主要体现在标准函数库中函数的种类、格式和功能上。为了有利于计算机应用技术的发展，ANSI 于 1983 年专门成立了定义 C 语言标准的委员会，1989 年制定出 ANSI C 的标准，又称为 C89。1995 年，经过修订的 C，增加了一些库函数，出现了 C++的一些特性，使 C89 成为 C++的子集。1999 年又推出 C99，它在保留 C 语言特性的基础上，增加了面向对象的新特性。

本章简要介绍 C 语言的特点，C 程序的基本结构和 C 程序的调试。

1.1　C 语言的特点

C 语言之所以能够广为流传，是因为它有很多不同于其他程序设计语言的特点。其主要特点有：

① 数据类型丰富。C 语言除了整型、实型、字符型等基本数据类型外，还具有数组、指针、结构、联合等高级数据类型，能够用于描述各种复杂的数据结构（如链表、栈、队列等）。指针数据类型的使用，使 C 程序结构更为简化、程序编写更为灵活、程序运行更为高效。

② 运算符种类丰富。C 语言具有数十种运算符，除了具有一般高级语言具有的运算功能外，还可以实现以二进制位为单位的位运算，直接控制计算机的硬件，还具有自增、自减和各种复合赋值运算符等。C 程序编译后生成的目标代码长度短、运行速度快、效率高。

③ 符合结构化程序设计的要求。C 语言提供的控制结构语句（如 if-else 语句、while 语句、do-while 语句、switch 语句、for 语句）使程序结构清晰，其函数结构使程序模块具有相对独立的功能，便于调试和维护，有利于大型软件的协作开发。

④ 可移植性好。用 C 语言编写的程序几乎不作修改就可用于各种计算机和各种操作系统。

C 语言的这些特点使 C 语言很快应用到了各计算机应用领域中的软件编写，如数据库管理、CAD、科学计算、图形图像处理、实时控制等软件。

然而，C 语言也不是十全十美的，它也有缺点。它的语法限制不太严格，例如，缺乏数据类型的一致性检测和不进行数组下标越界检查。正因为 C 语言允许编程者有较大的自由度，使 C 程序容易通过编译，但却难以查出运行中的错误。初学者一定不要以为编译通过了，程序就一定是正确的，就应该运行出正确结果。要想尽快找到程序中的错误，一定要掌握调试程序的方法和技术，多上机实践。

1.2 程序与程序设计

程序是计算机可以执行的一个指令序列，是为解决特定问题用某种计算机语言编写的语句（指令）序列。计算机科学家沃思（Nikiklaus Wirth）把程序描述为：

程序＝数据结构＋算法

这说明了数据结构和算法对程序的重要性。设计一个合理的数据结构可以简化算法，而好的算法又可以提高程序的执行效率。

计算机通过执行程序完成其工作。计算机可以直接执行的程序称为可执行程序（其扩展名一般为 EXE、COM），它包含的主要部分是二进制编码的机器指令和数据。机器指令直接控制计算机的每一个部件的基本动作，机器指令的表达方式称为“机器语言”。可执行程序通常以文件方式存放在磁盘上，当需要执行某程序时，必须把该程序装入内存。

程序设计是根据计算机要完成的任务进行数据结构和算法的设计，并且编写其程序代码，然后进行调试，直到得出正确结果，其基本过程如下：

① 分析问题，明确要解决的问题和要实现的功能。

② 将具体问题抽象为数学问题，建立数学模型，确定合适的解决方案。

③ 确定数据结构，并根据数据结构设计相应的算法，写出算法描述。

④ 编写程序。

⑤ 调试并运行程序，直到得到正确结果。

程序设计方法经历了由传统的结构化程序设计（面向过程）到面向对象的设计。结构化程序设计采用模块分解与功能抽象和自顶向下、分而治之的方法，有效地将一个较复杂的程序设计任务分解成许多易于控制和处理的子程序（模块）。各模块之间尽量相对独立，便于开发和维护。结构化程序设计在整个 20 世纪 70 年代的软件开发中占绝对统治地位。

20 世纪 70 年代末期，随着计算机科学的发展和应用领域的不断扩大，对计算机技术的要求越来越高。结构化程序设计语言和结构化分析与设计已无法满足用户需求的变化，于是出现了面向对象的程序设计技术。面向对象的程序设计方法不仅吸收了结构化程序设计的思想，而且克服了结构化程序设计中数据与程序分离的缺点，模拟自然界认识和处理事物的方法，将数据和对数据的操作方法放在一起，形成一个对象，使对象成为程序系统的基本单位。面向对象的程序设计技术更加有利于程序的调试和维护，大大提高了程序的可重用性和修改、扩充程序的效率。

1.3 C 程序的基本结构

首先让我们来看两个例子。

【例1.1】在屏幕上输出一串字符。

源程序 1-1.C

```
void main()                                    /* main()表示主函数, void 表示 main()不返回值*/
{                                              /*  函数体开始  */
   printf("Let us studing C program together. \n");   /*  屏幕上输出一字符串  */
}                                              /*  函数体结束  */
```

执行该程序将在屏幕上显示如下信息：

```
Let us studing C program together.
```

该程序由一个主函数 main()组成。函数体内（由大括号{ }括起来的部分）只有一条语句。该语句是调用 C 语言提供的库函数 printf()（格式输出函数）输出双引号中的字符串，其中的“\n”是转义字符（参见表 2.3），表示换行；即当在屏幕上输出到转义字符“\n”时，光标移到下一行的起始位置，以后再有输出就从该位置开始显示。

程序中每一行后面都是注释。注释可以是由“/*”和“*/”括起来的任何文字，它可以出现在程序的任何地方，用来说明程序段的功能、语句行的作用、变量的作用等内容，主要用于向程序阅读者进行说明和交流，使读者能读懂程序，便于程序的调试。每一个程序编写者都应该养成在必要的位置上加写注释的良好习惯。

C 语言的每一条语句都以分号“;”结束。为了清晰地显示程序的结构，程序的书写应该采用缩进格式，一行只书写一条语句。

【例1.2】 求半径为 r 的圆面积。

源程序 1-2.C

```
#define PI 3.1415926                    /* 宏定义 */
void main()                             /* 求半径为 r 的圆面积 */
{                                       /* 函数体开始 */          } 声明部分
  float area;                           /* 定义实型变量 */
  int r;                                /* 定义整型变量 */

  printf("请输入圆的半径：");           /* 屏幕上显示 "请输入圆的半径:" */
  scanf("%d",&r);                       /* 输入圆的半径  */
  area=PI*r*r;                          /* 计算圆面积 */          } 可执行部分
  printf("\n area=%f\n",area);          /* 输出圆面积 */
}                                       /* 函数体结束 */
```

程序运行情况如下：

```
请输入圆的半径：5
area=78.539815
```

C 语言的函数体主要由两部分组成：声明部分和可执行部分。声明部分用于定义和说明变量、数组等；可执行部分由可执行语句和函数调用等语句行组成。

程序的第 1 行，#define 是宏命令，其作用是在编译预处理阶段系统将程序中的 PI 用 3.1415926 替换。该程序中的每一行都加有注释，说明每一语句行的作用。其中函数体中的第 2 行和第 3 行是对函数体中要使用的变量进行定义，称为声明部分；第 4 行是在屏幕上显示“请输入圆的半径：”，这种方法常用于输入前对用户的提示；第 5 行调用输入函数（参见 2.3.1：格式输入函数 scanf()）输入圆的半径；第 6 行是计算半径为 r 的圆面积，并将计算结果赋给变量 area；第 7 行是输出变量 area 的值，即圆的面积。如果输入的半径 r=15，则屏幕上显示的是：

```
area=706.859985
```

由此可见，上面两个 C 程序的执行都是从 main()函数开始，依次执行函数体中各条语句，直到结束。

C 程序的组成特点如下：

① 一个 C 源程序由函数构成，其中有且仅有一个主函数 main()。

② C 程序总是由 main()函数开始执行，且结束于主函数。

③ 分号“;”是 C 语句的一部分，每一条语句均以分号结束。

④ C 程序书写格式自由，一行内可写多条语句。

⑤ 程序的注释部分应括在/*…*/之间，/和*之间不允许留有空格；允许注释部分出现在程序的任何位置上。

在上述例子中还出现了 C 语言程序设计中的标识符、关键字、运算符、常量和变量、函数调用等基本概念。

1. 标识符

例 1.2 中的 area 和 r，在 C 语言中称为标识符。

C 语言的标识符用于给程序中的常量、变量、函数、文件指针和数据类型等命名。用户可以根据需要进行命名，形成用户标识符。标识符的构成规则如下：

① 标识符由英文字母、数字、下划线组成；且第 1 个字符不能是数字，必须是字母或下划线。

② 标识符中的大、小写英文字母的含义不同，通常 C 程序中的变量用小写字母，符号常量用大写字母。

③ 不同的 C 编译系统对标识符所用的字符个数有不同的规定，ANSI C 可以识别标识符的前 31 个字符，但有的 C 编译系统只识别前 8 个字符。

④ 用户取名时，应当尽量遵循“见名知意” 和“简洁明了”的原则。

2. 关键字

例 1.2 中的 float 和 int，在 C 语言中称为关键字。

关键字又称保留字。它是 C 语言中已预先定义，且具有特定含义的标识符。C 语言共有 32 个关键字，如表 1.1 所示。所有关键字都用小写英文字母表示，且这些关键字不允许用作用户标识符。

表 1.1 关 键 字

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

3. 运算符

例 1.2 中的=、*、&是 C 语言中的运算符。

运算符是用来表示某种运算的符号，其中有的由一个字符组成，例如 +、−、*、/ 等；有的由多个字符组成，例如<=、<<、&&、!= 等。C 语言的运算符主要有以下几类：

① 算术运算符　　（+　−　*　/　%）

② 关系运算符　　（>　<　==　>=　<=　!=）

③ 逻辑运算符　　（!　&&　||）

④ 位运算符　　（<< >> ～ | ^ &）
⑤ 赋值运算符　　（= 及其扩展赋值运算符）
⑥ 条件运算符　　（? :）
⑦ 逗号运算符　　（,）
⑧ 指针运算符　　（* 和 &）
⑨ 求字节数运算符　　（sizeof）
⑩ 强制类型转换运算符　　（(类型)）
⑪ 分量运算符　　（. ->）
⑫ 下标运算符　　（[ ]）

有些运算符具有双重含义，例如“%”可以作为求余运算符；但是，当它出现在输入、输出函数中时，就是“格式控制符”了。

4. 常量和变量

例 1.2 程序中的 3.1416 在程序运行过程中是不会发生变化的，C 语言将其称为常量。C 语言的常量分为整型常量、实型常量和字符型常量。如 5，10，–34 是整型常量，2.3，9.8，–15.0 是实型常量，'a'，'b' 为字符型常量。

变量是指在程序运行过程中其值可以改变的量。例如，例 1.2 程序中的 area 和 r 就称为变量。每一个变量都有一个名字，根据变量的不同类型，系统将为每一个变量分配相应的内存单元。例如，系统为整型（int 型）变量分配 2 个字节的内存单元，为实型（float 型）变量分配 4 个字节的内存单元，为字符型（char 型）的变量分配 1 个字节的内存单元。内存单元中存放的是变量的值。程序执行过程中对数据的读、写是通过变量名找到相应的内存单元来实现的。

说明：

C 语言规定：程序中所用到的所有变量，都必须“先定义（说明变量的名字和数据类型），后使用”。任何一个未经定义就使用的变量都会被 C 语言的编译程序认为是非法变量，引起编译出错：

Undefined symbol 'xxxxxx'

5. 函数调用

例 1.2 程序中调用的输出函数 printf()和输入函数 scanf()是 C 语言的库函数中为用户提供的。C 语言中的函数分为：系统提供的库函数和用户自定义函数。对于库函数，用户可以直接调用；用户自定义函数是用户用以解决专门问题所定义的函数（参见第 6 章）。

【例1.3】 从键盘上输入两个整数，求其中的较大数。

源程序 1-3.C

```
#include<stdio.h>
void main()
{
  int x,y,z;

  printf("输入两个整数: ");      /* 提示输入 */
  scanf("%d%d",&x,&y);          /* 输入两个整数 */
  z=max(x,y);                   /* 调用求最大值函数 */
  printf("max=%d",z);           /* 输出最大值 */
```