

崔武子 付 钊 鞠慧敏 编著

C语言程序设计 实践教程

清华大学出版社



21世纪
计算机
科学与
技术
实践
型
教程

丛书主编
陈明

崔武子 付 钊 鞠慧敏 编著

C语言程序设计 实践教程

清华大学出版社
北京

内 容 简 介

本书是以C语言的入门级读者作为主要对象,根据C语言程序设计课程的特点,用编排独特、形式新颖、通俗易懂的方法编写的一本教材。全书包括3大部分。第1部分(第1章和第2章)介绍了C语言的基本框架、C语言的基本概念、函数的定义和使用、常量与变量的概念、C语言数据类型、运算符和表达式等。第2部分(第3章至第5章)介绍了结构化程序设计的3种基本结构。第3部分(第6章至第9章)介绍了指针的概念、数组的应用、字符串的处理方法、结构体的概念、链表的基本操作、文件的读写操作等。全书自始至终遵循开发较大任务时将一个大任务分成若干小任务后,编写函数实现该功能的开发原则。每一章增设上机指导和小结,提供贯穿于整个教学过程的一个应用实例。习题以提高编程能力为目标而设计,单号习题提供参考答案。附录中还提供快速、方便查找各知识内容的方法。书中标有“*”号的部分可以根据实际情况选讲。

本书提供动画丰富、可直接使用的电子教案,该教案允许教师随意修改。源代码包括全书所有实例和贯穿整个教学过程的应用实例的.c文件,它们均在Turbo C 2.0中运行通过。本书既可作为应用型或教学型院校应用类教材,也可用作传统教学模式的课程实践教材或自学者的指导书。

版权所有,翻印必究。举报电话:010-62782989 13501256678 13801310933

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

本书防伪标签采用特殊防伪技术,用户可通过在图案表面涂抹清水,图案消失,水干后图案复现;或将表面膜揭下,放在白纸上用彩笔涂抹,图案在白纸上再现的方法识别真伪。

图书在版编目(CIP)数据

C语言程序设计实践教程/崔武子,付杭,鞠慧敏编著. —北京:清华大学出版社,2005.8

(21世纪计算机科学与技术实践型教程)

ISBN 7-302-11349-1

I. C… II. ①崔… ②付… ③鞠… 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(2005)第077525号

出 版 者: 清华大学出版社 地 址: 北京清华大学学研大厦
http://www.tup.com.cn 邮 编: 100084
社 总 机: 010-62770175 客户服务: 010-62776969

责任编辑: 谢 琛

印 刷 者: 北京国马印刷厂

装 订 者: 三河市新茂装订有限公司

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印张: 18.5 字数: 423千字

版 次: 2005年8月第1版 2005年8月第1次印刷

书 号: ISBN 7-302-11349-1/TP·7469

印 数: 1~5000

定 价: 25.00元

《21 世纪计算机科学与技术实践型教程》

编辑委员会

主 任：

陈 明

中国石油大学教授

委 员(按姓氏笔画排序)：

毛国君

北京工业大学教授

叶新铭

内蒙古大学教授

刘淑芬

吉林大学教授

刘书家

北京工商大学教授

白中英

北京邮电大学教授

汤 庸

中山大学教授

何炎祥

武汉大学教授

陈永义

北京气象学院教授

罗四维

北京交通大学教授

段友祥

中国石油大学教授

高维东

南开大学教授

郭 禾

大连理工大学副教授

姚 琳

北京科技大学副教授

崔武子

北京联合大学副教授

谢树煜

清华大学教授

焦金生

清华大学教授

曹元大

北京理工大学教授

韩江洪

合肥工业大学教授

策划编辑：谢 琛

《21 世纪计算机科学与技术实践型教程》

序

21 世纪影响世界的三大关键技术是：以计算机和网络为代表的信息技术；以基因工程为代表的生命科学和生物技术；以纳米技术为代表的新型材料技术。信息技术居三大关键技术之首。我国已确定，国民经济的发展采取信息化带动现代化的方针，要求在所有领域中迅速推广信息技术，导致需要大量的计算机科学与技术领域的优秀人才。

计算机科学与技术的广泛应用是计算机学科发展的原动力，计算机科学是一门应用科学。因此，计算机学科的优秀人才不仅应具有坚实的科学理论基础，而且更重要的是能将理论与实践相结合，并具有解决实际问题的能力。培养计算机科学与技术的优秀人才是社会的需要、国民经济发展的需要。

制定科学的教学计划对于培养计算机科学与技术人才十分重要，而教材的选择是实施教学计划的一个重要组成部分，《21 世纪计算机科学与技术实践型教程》主要考虑了下述两方面。

一方面，高等学校的计算机科学与技术专业的学生，在学习了基本的必修课和部分选修课程之后，立刻进行计算机应用系统的软件和硬件开发与应用尚存在一些困难，而《21 世纪计算机科学与技术实践型教程》就是为了填补这部分鸿沟。将理论与实际联系起来，结合起来，使学生不仅学会了计算机科学理论，而且也学会应用这些理论解决实际问题。

另一方面，计算机科学与技术专业的课程内容需要经过实践练习，才能深刻理解和掌握。因此，本套教材增强了实践性、应用性和可理解性，并在体例上做了改进——使用案例说明。

实践型教学占有重要的位置，不仅体现了理论和实践紧密结合的学科特征，而且对于提高学生的综合素质，培养学生的创新精神与实践能力有特殊的作用。因此，研究和撰写实践型教材是必须的，也是十分重要的任务。优秀的教材是保证高水平教学的重要因素，选择水平高、内容新、实践性强的教材可以促进课堂教学质量的快速提升。在教学中，应用实践型教材可以增强学生的认知能力、创新能力、实践能力以及团队协作和交流表达能力。

实践型教材应由教学经验丰富、实际应用经验丰富的教师撰写。此系列教材的作者不但从事多年的计算机教学，而且参加并完成了多项计算机类的科研项目，把他们积累的经验、知识、智慧、素质融合于教材中，奉献给计算机科学与技术的教学。

我们在组织本系列教材过程中，虽然经过了详细地思考和讨论，但毕竟是初步的尝试，不完善甚至缺陷不可避免，敬请读者指正。

本系列教材主编 陈明

2005 年 1 月于北京

前 言

目前 C 语言程序设计课程是高等院校的重点课程之一,但由于 C 语言功能较多,使用灵活,初学者普遍感到难学。根据 C 语言程序设计课程的特点,作者多年来以“学生为中心,教师为主导,因材施教、按需施教,理论教学融入实践训练,强调应用能力的培养”为原则,尝试了实践教学的研究方案,并用编排独特、形式新颖、通俗易懂的方法编写了本书。本书既可作为应用型或教学型院校的应用类教材,也可用作传统教学模式的课程实践教材或自学者的指导书。本书的主要对象是 C 语言的入门读者。

本书特点:

(1) 从第 1 章开始学习和使用函数。用 C 语言开发应用程序时,先将一个较大任务分成若干小任务后,编写相应的函数实现每一个小任务。本书自始至终遵循这一开发原则。

(2) 所有教学内容用实例组织。本书重点强调读者在实践的过程中学习 C 语言,领会程序设计方法,所以根据所要介绍的内容都精心设计了相应的实例。读者在实现实例功能的过程中一举三得:学习语法、了解概念、掌握算法。

(3) 每一实例编程思路清晰,知识要点明确。由于本书采用实例教学方法,书中有丰富的实例。针对学生“读程序容易,编程序难”的情况,对于每一个实例,都按问题提出、编程点拨、程序代码、运行结果、程序说明、知识要点等步骤进行介绍。

(4) 分步介绍程序调试方法。为了培养学生调试程序、排除错误的能力,书中通过具体实例分阶段介绍了调试程序的方法。

(5) 增设了贯穿整个教学过程的实例。从第 3 章起增设了贯穿于整个教学过程的一个应用实例,并随着讲授内容的增多,分 7 个阶段逐步补充和完善其程序的功能。

(6) 每一章增设上机指导和小结。上机指导一节中的每一个实验均由实验题目、实验要求、实验目标以及上机指导等 4 项内容组成。小结部分是本章中的重点、难点的概括。

(7) 提供快速、方便查找各知识内容的方法。每章前列出了该章各实例和上机实验的知识要点,附录中提供独特的索引,该索引分别按关键字、运算符、库函数、算法与技巧分类。

(8) 习题形式新颖,提供单号习题答案。习题以提高编程能力为目标而设计,题型有程序改错题、程序填空题、补充完整程序题和编程题。书中单号习题提供参考答案,以方便学生自测和教师布置作业。

(9) 提供了内容生动的电子教案和源代码。为了减轻教师备课的负担,精心制作了可直接使用的电子教案,该教案允许教师随意修改。源代码包括全书所有实例和贯穿整个教学过程的应用实例的.c 文件,它们均在 Turbo C 2.0 中运行通过。

使用本书建议:

(1) 根据知识要点选择教学和实验。由于各学校的课时安排和学生程度的不同,可根据各章前面提供的实例和实验知识要点选择合适的实例和实验进行课堂教学。标有“*”的章节、实例、实验和习题有一定的难度,根据情况选讲。

(2) 采用教学与自学相结合的教学方式。将典型、重点、难点的实例在课堂上介绍,将简单、易掌握的实例留作学生自习,并将自学的实例也作为本课程的考核内容。由于每一个实例均用通俗易懂的方式详细介绍,适当要求学生自习,可解决课时紧张的问题,更重要的是可使学生掌握学习方法,提高独立解决问题的能力。

(3) 讲授和上机最好在机房进行。理论讲授和上机课最好均在机房进行,但应为计算机配置多媒体设备和能够监控学生使用计算机情况的控制台,使教师在教学过程中及时掌握和调节课堂节奏,保障有效地完成教学任务。

(4) 单、双号习题成对做。单号习题提供参考答案,双号习题则在类型上与前一单号习题相同,侧重点也接近(标有“*”的习题有一定难度)。

本书由崔武子主编、统稿,崔武子、付杭、鞠慧敏共同编写。

在编写本书的过程中,李青和李红豫参与了部分代码的编写和调试,赵重敏老师提出了许多宝贵意见,在此一并表示衷心的感谢。

由于作者水平有限,书中难免有错误和疏漏之处,恳请读者批评和指正。

作 者

2005 年 2 月于北京

目 录

第 1 章 C 语言的基本框架	1
1.1 C 语言的基本概念	1
1.2 几个简单的 C 程序	3
1.3 函数的定义和使用	6
1.4 上机指导	11
1.5 小结	16
习题 1	17
第 2 章 C 语言的数据运算	20
2.1 常量与变量	20
2.2 C 语言数据类型	22
2.2.1 整型数据类型	22
2.2.2 实型数据类型	24
2.2.3 字符型数据类型	24
2.3 运算符和表达式	26
2.3.1 算术运算符和表达式	26
2.3.2 赋值运算符和表达式	29
2.3.3 逗号运算符和表达式	30
2.3.4 关系运算符和表达式	31
2.3.5 逻辑运算符和表达式	32
2.3.6 其他运算符	33
2.4 上机指导	36
2.5 小结	39
习题 2	40
第 3 章 顺序结构及其应用	43
3.1 算法与结构化程序设计	43
3.2 数据的输入输出控制	44

3.2.1 格式输入输出的函数	45
3.2.2 字符输入输出的函数	48
3.3 顺序结构应用实例	49
* 3.4 贯穿整个教学过程的实例部分程序	57
3.5 上机指导	60
3.6 小结	64
习题 3	65
第 4 章 分支结构及其应用	69
4.1 双分支结构	70
4.2 多分支结构	75
4.3 分支结构应用实例	81
* 4.4 用条件运算符实现分支	89
* 4.5 贯穿整个教学过程的实例部分程序	90
4.6 上机指导	94
4.7 小结	99
习题 4	99
第 5 章 循环结构及其应用	103
5.1 常用的循环控制语句	104
5.1.1 while 语句	104
5.1.2 do-while 语句	111
5.1.3 for 语句	112
5.2 循环的嵌套	115
5.3 break 语句与 continue 语句的使用	118
5.4 循环结构应用实例	121
* 5.5 贯穿整个教学过程的实例部分程序	130
5.6 上机指导	134
5.7 小结	139
习题 5	139
第 6 章 指针及其使用	143
6.1 指针的概念、定义和使用	143
6.1.1 指针的概念	143
6.1.2 指针的定义和使用	144
6.2 指针的应用实例	146
* 6.3 贯穿整个教学过程的实例部分程序	149
6.4 上机指导	154

6.5 小结	156
习题 6	156
第 7 章 数组及其应用	159
7.1 一维数组及其应用	160
7.1.1 一维数组的定义和使用	160
7.1.2 一维数组与指针	162
7.1.3 一维数组的应用举例	165
7.2 二维数组及其应用	173
7.3 字符串及其处理	178
* 7.4 贯穿整个教学过程的实例部分程序	186
7.5 上机指导	192
7.6 小结	197
习题 7	197
第 8 章 结构体及其应用	199
8.1 结构体类型的使用	199
8.2 结构体类型数组的定义	204
* 8.3 链表	211
8.3.1 动态指针与动态内存分配	211
8.3.2 链表的概念	212
8.3.3 链表的基本操作	212
* 8.4 贯穿整个教学过程的实例部分程序	220
8.5 上机指导	226
8.6 小结	230
习题 8	230
第 9 章 文件及其应用	232
9.1 文件的基本操作	232
9.1.1 文件中的几个概念	232
9.1.2 文件的打开与关闭	233
9.1.3 文件的读写	234
* 9.2 贯穿整个教学过程的实例部分程序	244
9.3 上机指导	252
9.4 小结	254
习题 9	255

附录.....	256
附录 A C 语言关键字.....	256
附录 B 常用字符与 ASCII 码对照表	256
附录 C 运算符的优先级和结合方向	257
附录 D 常用 C 库函数	258
附录 E 常见错误、警告信息表	262
附录 F 关键字、运算符、库函数和算法索引	263
附录 G 单号题参考答案.....	267
参考文献.....	284

第 1 章 C 语言的基本框架

本章实例知识要点

- 实例 1.1 主函数、函数体的概念;使用 printf() 函数输出信息的方法。
- 实例 1.2 使用 scanf() 函数输入数据;变量的定义;注释的用途;数学库函数的使用。
- 实例 1.3 C 程序的基本框架;调用系统提供的函数方法与自己定义的函数方法的比较。
- 实例 1.4 函数定义位置;函数原型说明;函数调用过程;返回值;void 类型。
- 实例 1.5 使用函数的 3 个步骤;“自顶向下,逐步细化”的基本程序设计方法。

本章实验知识要点

- 实验 1.1 Turbo C 2.0 集成环境;C 语言程序的编辑、编译、连接和运行过程。
- 实验 1.2 数学函数的使用;加 printf() 观察结果;处理复杂表达式;合理选择测试数据。
- 实验 1.3 单步执行;设置监视项;参数传递过程和函数调用过程;交换数据的方法。
- 实验 1.4 定义函数;函数的使用方法;F7、F8 的区别;选择数据类型的原则。

1.1 C 语言的基本概念

计算机语言有机器语言、汇编语言、高级语言之分。其中高级语言独立于机器,编码相对短,可读性强。C 语言是高级语言,它具有运算符丰富、使用方便、功能强、程序执行效率高、可移植性好等特点,因此它是开发系统软件和应用软件的有力工具。C 语言是目前国际上广泛流行的一种结构化的程序设计语言。C 语言由附录 A 中列出的 32 个关键字再加上语法规则构成。

1. 程序

程序(program)是为了使计算机完成指定的任务而编写的具有特定语法规则的指令序列。编写程序需要使用计算机语言,用高级语言描述的程序叫做源程序(source program)。

在程序的编写过程中,通常要考虑诸如数据结构和算法等方面的问题。其中数据结构是指程序中的每一数据对象的性质、构造方法及它们间的相互关系;算法(algorithm)则是指对这些数据对象进行处理时的求解方法。著名的计算机科学家 Nikiklaus Wirth 用如下关系描述程序:

程序 = 数据结构 + 算法

2. 程序设计

程序设计(programming)的步骤大体上分为:问题定义、算法设计、流程图设计、编写程序代码、测试与调试、整理文档和系统维护。

(1) 问题定义

问题定义就是要明确所需完成的任务,在认真分析的基础上给出清晰、准确的问题描述和功能要求。同时,在解决问题的过程中,还应给出数据输入(input)和结果输出(output)的方式和方法。

(2) 算法(algorithm)设计

设计算法就是建立计算机求解问题的步骤。

在设计一个较大任务的算法时,通常采用“自顶向下,逐步细化”的模块化程序设计方法,即将较大的任务按照一定的原则拆分为一个个较小的任务,一个程序模块完成一个小任务,需要注意的是这些程序模块在功能上既要相互关联又应相对独立,而且容易理解,在 C 语言中称各程序模块为函数(function)。

设计出的算法应包括有限的操作步骤,而且每一步操作均有明确的含义,一般情况下解决同一个问题的算法并不惟一,但是算法的好坏直接影响程序的执行效率。

(3) 流程图设计

每个小模块均需要设计完成该任务的具体算法,算法的表示方法有多种,其中使用较普遍的是流程图(flowchart),流程图中所使用的常用符号如图 1.1 所示。



图 1.1 流程图中的常用符号

(4) 编写程序代码

算法设计好后必须通过计算机语言编写程序代码(code)实现其功能,编写的代码除具有可靠性外还应具有良好的设计风格。

(5) 测试与调试

在程序设计过程中,需要将编写的程序认真细致地进行测试(test)与调试(debug)。

图 1.2 所示是从编写源程序到结束调试过程中所涉及的操作流程。

上述操作流程中涉及程序的编辑(editing)、编译(compiling)、连接(linking)和运行(running)等操作。其中编辑是指在计算机上进行输入或修改源程序的过程;编译是将源程序翻译成机器能识别的目标程序;连接则是把编写的程序和系统提供的库函数等连接起来产生可执行文件;执行此可执行文件就能看到程序的运行结果。

程序的编辑、编译、连接、运行操作可以在不同的集成环境中进行,本书选用的是 Turbo C 2.0,这是一个将以上四步合为一体的集成环境,其操作简单易行,在微机应用广泛。

(6) 整理文档和系统维护

整理文档就是编写程序的使用说明书和维护说明书;系统维护则是针对该软件在使用过程中出现的错误或根据环境提出的新需求,对程序所做的改进等。

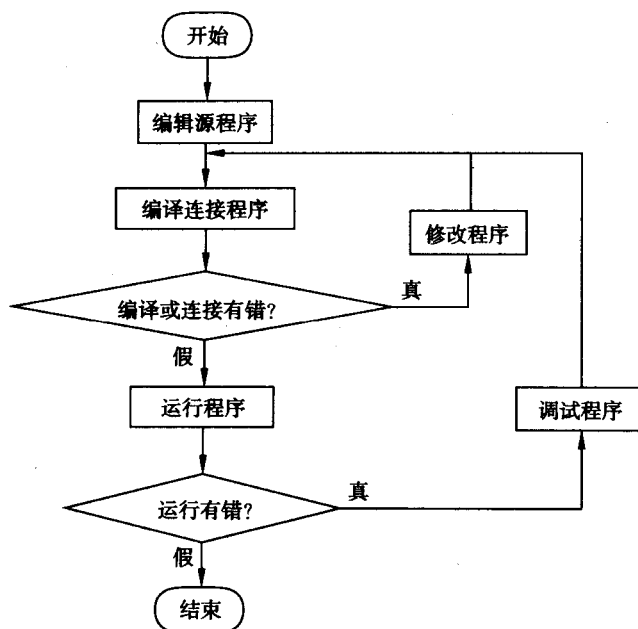


图 1.2 从编写程序到结束调试的操作流程

1.2 几个简单的 C 程序

为了使读者对 C 语言程序有感性的认识,下面先给出几个简单的 C 程序实例。需要说明的是,通常情况下用英文小写字母编写 C 程序代码。

【实例 1.1】 编写程序,在屏幕上显示“How are you?”和“Fine. Thank you.”。

编程点拨:

本题需要输出两个字符串(string),在 C 语言中没有专门的输出语句,所以其操作需调用系统提供的标准库函数 printf()实现。

程序代码:

```
#include <stdio.h>
main()
{
    printf("How are you? \n");
    printf("Fine. Thank you. \n");
}
```

运行结果:

```
How are you?
Fine. Thank you.
```

程序说明:

(1) 本程序是一个仅由 main() 函数(也称主函数)构成的完整 C 程序。在这里用大括号“{ }”括起来的部分称做函数体。

(2) 函数体中的 printf() 是系统提供的标准库函数,其功能是用于输出。例如,语句“printf(“How are you? \n”);”的作用是先在屏幕上输出字符串“How are you?”,然后将光标移到下一行的第 1 列,其中\n 是换行符。printf() 后的“;”是语句的结束符,不能缺少。C 语言程序中的每一条语句均以“;”结束。有关 printf() 函数的具体使用方法将在 3.2.1 节介绍。

(3) 在 C 语言中没有专门的数据输入输出语句,所以输入输出操作均需调用系统提供的标准库函数实现。为了成功地调用输入输出函数,需要将头文件“stdio. h”中的内容添加到程序中,因此需要在程序开始处使用一个命令行“#include <stdio. h>”,该命令行称为预处理命令行。

【实例 1.2】 编写程序,程序运行后通过键盘输入一个数,先计算其绝对值,然后在该值基础上计算平方根。

编程点拨:

在实际应用中经常需要在程序运行过程中通过键盘输入数据,在 C 语言中,需要调用标准库函数 scanf() 完成数据的输入操作。本题计算绝对值和平方根的操作也可直接调用标准库函数 fabs() 和 sqrt() 完成。

本题需要将输入的数据和计算的结果保留并输出,在程序设计中,通常将这些数据存放在变量(variable)中,因此本题需要定义 3 个变量。

程序代码:

```
/* 程序功能: 计算某数的绝对值和平方根 */
/* 设计人员: 张山 */
/* 设计时间: 2004-11-26 */
#include <stdio. h>          /* 调用输入输出函数时添加此行 */
#include <math. h>           /* 调用数学函数时添加此行 */
main()                      /* 程序从这里开始运行 */
{ double x,y,z;             /* 定义 3 个变量 x、y 和 z */
  printf("Input x:");       /* 显示提示信息 */
  scanf("%lf",&x);          /* 从键盘输入 x 的值 */
  y=fabs(x);                 /* 调用 fabs() 标准库函数计算 x 的绝对值 */
  z=sqrt(y);                 /* 调用 sqrt() 标准库函数计算 y 的平方根 */
  printf("x=%lf,y=%lf,z=%lf\n",x,y,z); /* 输出 x、y 和 z 的值 */
}
```

运行结果:

Input x: -4<回车> (本书中用户输入的部分均画下划线)
x=-4.000000,y=4.000000,z=2.000000

程序说明:

(1) 语句“scanf("%lf",&x);”的功能是将从键盘输入的数据存放在变量 x 中。%lf 是在 printf() 和 scanf() 函数中使用的格式说明符。语句“printf("x=%lf,y=%lf,z=%lf\n",x,y,z);”的作用是在输出双引号中的内容时,用 x、y、z 的值从左到右依次代替 3 个 %lf(%lf 控制小数点后输出 6 位)。printf() 和 scanf() 的具体使用方法参见 3.2.1 节。

(2) 语句“y=fabs(x);”的功能是先调用 fabs() 标准库函数计算变量 x 的绝对值,再将计算结果赋给变量 y;语句“z=sqrt(y);”的功能是先调用 sqrt() 函数计算变量 y 的平方根,再将计算结果赋给变量 z。

在程序中需要调用数学函数时应添加头文件“math.h”。C 语言系统提供丰富的标准库函数,为了便于管理和使用,系统将标准库函数分成若干类,并将其函数的说明分别放在不同的头文件中,因此在调用一个标准库函数时只需要添加包含该函数说明的头文件即可。这样可以提高程序的执行效率。

(3) “double x,y,z;”的作用是定义 3 个实型(双精度型)变量 x、y 和 z。在 C 语言中变量必须先定义后使用。有关变量的知识将在 2.1 节介绍。

(4) 书写在符号“/* */”中的内容是注释,这部分对程序的运行没有任何作用,因此可在程序的任意位置添加。

为了提高程序的可读性,通常在程序的开头添加注释,说明本程序的功能、开发时间、修改时间和设计者;在变量的定义后面添加注释,解释该变量的用途等。建议读者在编写程序时尽量多加注释。本书由于篇幅有限,将不对每个实例给出程序注释。

【实例 1.3】 编写程序,通过键盘输入两个数 x、y,计算 x^y 的值以及这两个数的平均值。

编程点拨:

计算 x^y 的值,可以直接调用数学库函数 pow(),但由于系统未提供计算两个数的平均值的函数,所以我们必须自己先编写然后再调用。

程序代码:

```
#include <stdio.h>
#include <math.h>

double myave(double a,double b)    /* 自己定义 myave() 函数计算两个数的平均值 */
{
    double c;
    c=(a+b)/2;                      /* C 语言将圆括弧作为运算符 */
    return c;                       /* 返回计算结果 */
}

main()
{
    double x,y,z,ave;
    printf("Input x,y:");
    scanf("%lf%lf",&x,&y);

    z=pow(x,y);                     /* 调用标准库函数 pow() 计算  $x^y$  */
    ave=myave(x,y);                 /* 调用自己定义的函数 myave() 计算 x、y 的平均值 */
    printf("pow(%lf,%lf)=%lf\n",x,y,z);
```

```
printf("The average is %lf\n",ave);
}
```

运行结果:

Input x,y: 4 3<回车>

pow(4.000000,3.000000)=64.000000

The average is 3.500000

程序说明:

(1) myave()是我们自己编写的函数,其功能是计算两个数的平均值。程序先定义了myave()函数,有了这一定义,就可以像调用标准库函数一样调用它们。函数的定义和使用方法以及C程序的执行过程将在1.3节中介绍。

(2) 通过本程序可以了解C语言程序的基本框架。C语言程序通常由若干函数构成,每个函数完成确定的功能。如果一个项目实现的功能多,那么要编写的函数就多,C语言程序的基本框架如下:

预处理命令行

定义函数 1

定义函数 2

⋮

定义函数 n

主函数

在以上3个实例中,我们不仅学习了如何调用标准库函数,而且还调用了自己定义的函数。虽然C语言已经提供了丰富的标准库函数,但在实际应用中,不可能满足所有的需求,因此程序设计者必须自己定义大量的函数,可以说掌握好函数是学好本课程的关键之一。

1.3 函数的定义和使用

任何一个C语言程序都有一个主函数,而且其名称必须是main。在大多数情况下,C程序除了主函数外还包括编程者自己定义的一些函数。程序中不管定义多少个函数,以及各函数的定义位置如何,运行程序时总是从主函数开始执行,并在主函数中调用其他函数实现相应的功能。函数的一般定义形式如下:

```
[函数值类型名]函数名([类型名 形式参数 1,类型名 形式参数 2,...])
{ [定义部分]
  语句部分
}
```

其中,用方括号括起来的部分根据实际情况可以省略。

定义一个函数其实质就是要指定该函数的名称,确定其功能,设定所需的参数以及明确函数值的类型。下面以实例1.3为例说明如何定义函数,该实例中定义函数的部分是: