



浙江省高等教育重点建设教材

C 程序设计教程

左伍衡 胡同森 / 主 编



ZHEJIANG UNIVERSITY PRESS
浙江大学出版社



浙江省高等教育重点建设教材

C 程序设计教程

左伍衡 胡同森 / 主 编



ZHEJIANG UNIVERSITY PRESS

浙江大学出版社

图书在版编目 (CIP) 数据

C 程序设计教程 / 左伍衡, 胡同森主编. — 杭州: 浙江大学出版社, 2007. 8

ISBN 978-7-308-05409-6

I. C… II. ①左… ②胡… 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 109339 号

C 程序设计教程

左伍衡 胡同森 主编

责任编辑 周卫群
封面设计 刘依群
出版发行 浙江大学出版社
(杭州天目山路 148 号 邮政编码 310028)
(E-mail: zupress@mail. hz. zj. cn)
(网址: <http://www.zjupress.com>)
经 销 浙江省新华书店
排 版 浙江大学出版社电脑排版中心
印 刷 浙江中恒世纪印务有限公司
开 本 787mm×1092mm 1/16
印 张 13.25
字 数 340 千
版 次 2007 年 8 月第 1 版 2007 年 8 月第 1 次印刷
印 数 0001—4000
书 号 ISBN 978-7-308-05409-6
定 价 22.00 元

版权所有 翻印必究 印装差错 负责调换

浙江大学出版社发行部邮购电话 (0571)88072522

总 序

姒健敏

随着高等教育大众化,成人高等教育面临的形势和任务也发生了很大的变化,成人高等教育日益呈现出新发展趋势:教育对象大众化;教育内容职业化;教育体系一体化;教育方式多样化;办学主体社会化。这些发展趋势,对成人高等教育的教学内容、教学方法、管理体制提出一系列改革与创新的要求,而教材改革则是其中最为迫切的改革要求。

经过细致的调查和分析,我们认为成人高等教育的教材,既要保证成人高等教育的规格,又要体现社会对成人高校的要求,才能适应已经变化和飞速发展的成人教育实际:首先,教材内容上,必须以实用性、应用性为主,以区别于普通高校教材侧重理论的特点。成人高等教育的培养目标已逐步转为广大在职者的学历补偿教育,职前职后的培训教育,因此,成人高等教育教材的理论深度,内容的难易程度要适中,同时偏向应用性、实用性。其次,在教材内容安排上,要分单元、分层次,充分考虑成人的学习特点,使教材内容前后连贯,以便从教材内容上充分体现对学习过程的层次控制。第三,在学习效果的检测上,以学生自己测试为主,将传统一次性考试变为对单元,学习时段的多层次考试。因此,材料要根据不同课时和单元配备习题或测试,以突出学生自己检测学习效果的特点。

按照上述思想,我们聘请一线专家和经验丰富的学科带头人精心编写了这套成人高等教育教材。相信这套精心策划、认真编写出版的系列教材会得到广大院校的认可。

2006年12月

前 言

C 语言是一种功能强大、编程方式灵活、特色鲜明、深受广大科技人员和专业编程者喜爱的程序设计语言,也是国内外广泛使用的计算机程序设计语言之一。

根据编者多年教学经验和计算机专业工作经历,认为高校将 ANSI C(标准 C)列为程序设计的首选语种之一,有利于提高学生分析归纳问题能力和专业素质。对于机电、信息类学生而言,可以为今后深入学习 C++ 或汇编语言打下一个良好的基础。

本书编写充分考虑到成人教育的特殊性。在教材体系上重视理论结合实际,以便于读者低起点、高效率地掌握 C 语言;在内容组织上突出重点而不过度纠缠一些晦涩的语法现象;在编排上尽量体现各章节的关联和系统性;在文字叙述上力求条理清晰,概念准确。

为便于学生更好地学习和理解 C 语言,在一些较难理解的部分尽量使用图解的方法,在例题中加入了部分代码注释。书中例题都是精选的,不以单纯解释 C 的语法为目的。习题难易适中、覆盖面广,对较难的题提供提示以帮助学生解答。例题和习题强调了编程能力训练的重要性,淡化了部分语法细节。所有教学内容围绕 ANSI C 展开,考虑到软件系统的流行配置,重点介绍了在 Visual C++ 环境中运行调试 ANSI C 程序的方法。

本书共分 10 章,主要内容包括:

第 1 章:C 程序设计基础知识。介绍程序设计语言及其分类,算法的概念、分类及特点,C 语言的特点与 C 程序简介,重点介绍 C 程序的上机环境与操作步骤。

第 2 章:基本数据类型与常用库函数。

第 3 章:表达式。介绍 C 的算术、关系、逻辑运算符,位运算符,表达式。

第 4 章:基本控制结构。介绍分支结构中的 if、switch 语句,循环结构中的 while、do-while、for 语句。

第 5 章:函数。着重介绍自定义函数的编写与调用,强调传值调用是函数间参数传递的唯一方式,介绍递归函数,变量的作用域、可见性以及存储类型。

第 6 章:编译预处理。介绍编译预处理的概念和常用的编译预处理命令。

第 7 章:数组。介绍了一维数组和二维数组的概念、定义、初始化、数组元素的引用和数组作为函数参数以及字符串、字符串数组的使用。

第8章:指针。介绍指针的概念,用较大篇幅从实用角度对指针变量的应用分类、整理,使之更易于接受。

第9章:结构体。介绍定义结构体的格式与使用方法,链表操作的典型示例。

第10章:文件。介绍文件的概念、文件指针、文件处理的基本过程和相关函数。

本书由浙江工业大学胡同森教授、左伍衡老师主编,浙江工商大学许芸副教授、湖州师范学院唐国民教授、杭州职业技术学院姜盟老师、浙江工业大学的陈珊珊和袁鹤以及金献珍老师参加编写。在本书编写过程中,得到了有关领导和浙江大学出版社的大力支持,在此表示衷心感谢。

因编者水平所限,书中难免存在疏漏之处,恳请广大读者提出宝贵意见。读者使用本书时遇到问题,可以发邮件到 whywhyzuo@yahoo.com.cn 和我们联系。

编 者

2007年5月

目 录

第 1 章 C 程序设计基础知识	1
1.1 计算机语言及其发展	1
1.1.1 机器语言	1
1.1.2 汇编语言	2
1.1.3 高级语言	2
1.2 算法的概念、分类及特点	3
1.2.1 算法的概念	3
1.2.2 算法的分类	3
1.2.3 算法的特点	3
1.2.4 算法的描述工具	3
1.3 C 语言的特点与 C 程序简介	4
1.3.1 简单 C 程序示例	5
1.3.2 C 程序结构	7
1.4 C 程序的上机步骤及环境	8
1.4.1 C 程序的上机步骤	8
1.4.2 Turbo C 2.0 环境下调式、运行 C 程序	9
1.4.3 Visual C++6.0 环境下调式、运行 C 程序	11
1.5 小结	14
第 2 章 基本数据类型与常用库函数	17
2.1 数据类型概述	17
2.1.1 数据类型的含义及作用	17
2.1.2 C 语言的数据类型	17
2.1.3 数据的存储格式	18
2.2 字符集、标识符与关键字	19
2.2.1 字符集	19
2.2.2 标识符与关键字	20
2.3 常量与变量	20
2.3.1 常量	20
2.3.2 变量	23

2.4 常用标准库函数	24
2.4.1 常用数学函数	24
2.4.2 常用字符函数	25
2.4.3 常用输入/输出函数	25
2.5 小结	29
第3章 表达式	33
3.1 表达式简介	33
3.2 算术表达式	33
3.2.1 算术运算符	33
3.2.2 自增、自减运算	35
3.3 位运算	35
3.3.1 位运算符	35
3.3.2 位运算符的运算优先级	40
3.4 逻辑表达式	40
3.4.1 关系表达式	40
3.4.2 逻辑运算符与逻辑表达式	41
3.5 条件表达式、赋值表达式与逗号表达式	42
3.5.1 条件表达式	42
3.5.2 赋值表达式	43
3.5.3 逗号表达式	44
3.6 小结	44
第4章 基本控制结构	49
4.1 结构化程序设计	49
4.1.1 结构化程序设计的基本思想	49
4.1.2 结构化程序的三种基本结构	49
4.2 选择结构	50
4.2.1 if 语句	50
4.2.2 switch 语句	54
4.3 循环语句	56
4.3.1 while 语句	56
4.3.2 do-while 语句	57
4.3.3 for 语句	58
4.3.4 break 与 continue 语句	61
4.3.5 循环的控制方法	63
4.4 多重循环	64
4.5 小结	68

第 5 章 函数	73
5.1 函数概述	73
5.1.1 标准库函数与自定义函数	73
5.1.2 C 程序结构	74
5.1.3 定义函数和函数声明	75
5.1.4 函数调用	77
5.1.5 函数间参数的值传递	78
5.2 函数嵌套调用	81
5.3 递归函数	82
5.4 变量的作用域与可见性	84
5.4.1 变量的作用域	84
5.4.2 变量的可见性	85
5.5 变量的存储类型	87
5.5.1 auto 自动型	87
5.5.2 register 寄存器型	87
5.5.3 static 静态型	88
5.6 小结	88
第 6 章 编译预处理	93
6.1 编译预处理概述	93
6.2 文件包含	93
6.3 宏定义	95
6.3.1 不带参数的宏	95
6.3.2 带参数的宏	96
6.4 小结	98
第 7 章 数组	101
7.1 一维数组	101
7.1.1 一维数组的声明	101
7.1.2 一维数组元素的引用	102
7.2 二维数组	107
7.2.1 二维数组的声明	107
7.2.2 二维数组应用示例	108
7.3 字符串	111
7.3.1 字符数组与字符串	111
7.3.2 常用字符串运算函数	114
7.3.3 字符串应用示例	116
7.4 字符串数组	118

7.4.1 二维字符数组与字符串数组	118
7.4.2 字符串数组应用示例	119
7.5 小结	120
第 8 章 指针	127
8.1 指针的基本概念	127
8.1.1 指针常量	127
8.1.2 指针变量	128
8.1.3 指针运算	130
8.2 指针数组	132
8.2.1 指针数组的声明与初始化	132
8.2.2 指针数组应用示例	132
8.3 指针变量的应用	134
8.3.1 数组存储空间的动态分配	134
8.3.2 间接访问主调函数中的数据	135
8.3.3 指向函数的指针	140
8.3.4 返回指针值的函数	142
8.5 小结	143
第 9 章 结构体	149
9.1 结构体类型数据的声明与引用	149
9.1.1 结构体类型数据的声明与初始化	149
9.1.2 结构体类型数据的引用	151
9.2 结构体类型数据与函数	153
9.2.1 结构体类型变量作函数形参	154
9.2.2 指向结构体类型数据的指针变量作函数形参	155
9.2.3 返回结构体的函数	157
9.2.4 返回指向结构体数据指针值的函数	158
9.3 链表	158
9.3.1 链表的数据结构形式和节点类型声明	159
9.3.2 链表的基本操作	160
9.4 小结	169
第 10 章 文件	173
10.1 文件概述	173
10.1.1 文件的概念	173
10.1.2 文件结构体	174
10.2 文件的打开与关闭	176
10.2.1 打开文件	176

10.2.2 关闭文件.....	177
10.3 文件的顺序读写.....	177
10.3.1 文本文件的顺序读写.....	177
10.3.2 二进制文件的顺序读写.....	184
10.4 文件的定位与随机读写简介.....	187
10.4.1 文件定位函数.....	188
10.4.2 文件随机读写示例.....	188
10.5 小结.....	190
附录 1 字符与 ASCII 码对照表	195
附录 2 运算符优先级	196
附录 3 常用 C 库函数	197

第1章

C 程序设计基础知识

本章首先简单介绍计算机语言的发展过程,然后介绍 C 语言。通过简单的 C 程序实例使读者对 C 语言、C 语言程序设计以及 C 语言程序的调试运行环境有一个初步的了解。

1.1 计算机语言及其发展

什么是程序?程序就是为解决某一问题而设计的一系列指令。

“早晨起床后先刷牙、后洗脸”是一段简单的程序,文艺演出的流程按照事先计划好的节目单进行,节目单也是程序。安排人们日常生活、工作的程序,可以用汉语、英语、日语等不同语言描述。为使计算机能按照人们的意志工作的程序,则要用计算机能够理解和识别的语言,即计算机语言来描述。计算机语言的发展,经历了机器语言、汇编语言、高级语言这几个阶段。

1.1.1 机器语言

早期的计算机只能执行用机器指令编写的程序,一台计算机能够执行的所有机器指令的集合,称为其指令系统(通常包含数百条指令)。指令系统是计算机唯一能够直接识别和执行的语言,因此又称作机器语言。

不同类型的计算机,其指令系统不尽相同,但都是用二进制编码表示的。每条指令的前若干个二进位表示该指令执行何种操作(操作码),其余二进位表示若干个操作数的地址(地址码)。指令按照其长度区分,有 16、32、64 位指令;按照指令中包含地址码的个数区分,有 1 地址、2 地址、3 地址指令;等等。

譬如,某型号计算机的 32 位 2 地址指令格式为 $c_1 c_2 \dots c_{n_1} d_{1_1} d_{1_2} \dots d_{1_{n_2}} d_{2_1} d_{2_2} \dots d_{2_{n_2}}$,其中 $d_{1_1} d_{1_2} \dots d_{1_{n_2}}$ 、 $d_{2_1} d_{2_2} \dots d_{2_{n_2}}$ 分别表示第 1、第 2 操作数的地址。1 条指令可表示将 $d_{1_1} d_{1_2} \dots d_{1_{n_2}}$ 与 $d_{2_1} d_{2_2} \dots d_{2_{n_2}}$ 中的数相加的结果存储到 $d_{1_1} d_{1_2} \dots d_{1_{n_2}}$,也可以表示将 $d_{1_1} d_{1_2} \dots d_{1_{n_2}}$ 中的数送入 $d_{2_1} d_{2_2} \dots d_{2_{n_2}}$,等等。指令中 $c_1 c_2 \dots c_{n_1}$ 由 n_1 个 0、1 字符的不同组合表示不同的操作,而地址码表示操作数的不同地址。

机器语言程序就是由许多行这样的指令所组成,将它们事先存储在内存中,从第 1 条指令开始,计算机就会按照程序事先安排好的顺序自动执行这些指令。

机器语言是计算机唯一能直接识别的一种语言,因此机器语言程序的执行效率高,但不能被广泛应用,尤其不适合用于开发应用程序,主要原因如下:

- 由于指令系统是面向机器的,不同种类的 CPU 对应的指令系统之间往往相差很大,因此,机器语言程序的可移植性差。

- 用机器语言编程极其困难,阅读更加不易,出错率高且难以维护。

用机器语言编程在早期是经过严格训练的专业技术人员的工作,普通程序员一般难以胜任。现在,几乎没有人用机器语言编写应用程序了。

1.1.2 汇编语言

虽然对用机器语言编程有很高的要求和许多不便,但程序的执行效率高,CPU 严格按照程序中的指令序列去执行,没有多余的操作。在保留“程序执行效率高”的前提下,人们开始着手研究一种能大大改善程序可读性的编程工具:选用了一些能反映机器指令功能的单词或词组来代表该机器指令,把 CPU 内部的各种资源符号化,用符号名引用相应资源。

这样,令人难懂的二进制机器指令就可以用通俗易懂的符号指令来表示了,这些具有一定含义的符号称为助记符,用指令助记符、符号地址等组成的符号指令称为汇编指令。

用汇编指令编写的程序称为汇编语言程序,其可读性大大提高,但失去了 CPU 能直接识别的特性。例如指令 MOV AX, BX 若表示将 BX 中的数据赋值到 AX 中,而 CPU 却不能直接识别并执行。为解决这个问题就需要执行一个翻译程序,它能把汇编语言编写的源程序自动翻译成 CPU 能识别的机器指令序列,如图 1-1 所示,该翻译程序被称为汇编程序。

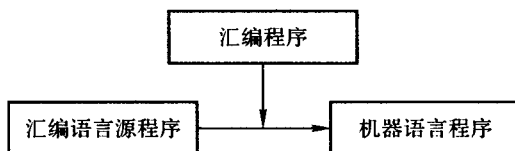


图 1-1 用汇编程序处理汇编语言源程序,生成机器语言程序

尽管采用指令符号化和伪地址(用符号表示存储单元地址)的方法,汇编语言程序的可读性提高,编程难度有所降低,但是从本质上讲,它还是面向机器的。

1.1.3 高级语言

在长期遭受用低级语言(机器语言、汇编语言)低效率编程的困扰中,人们意识到应设计一种这样的语言,它接近于数学语言或人的自然语言、但又不依赖于计算机硬件,编出的程序能在所有机器上通用。第一个完全脱离机器硬件的高级语言—FORTRAN 语言出现于 1954 年,此后又有数百种高级语言出现,其中有重要意义的有几十种。

60 年代中后期,随着计算机应用的普及,所开发的应用程序越来越多、规模越来越大,由于缺乏科学规范的系统规划与测试、评估标准,其恶果是许多耗巨资开发的程序由于含有错误而无法使用,甚至带来巨大损失,给人的感觉是越来越不可靠。这一“软件危机”使人们认识到,大型程序的编制是一项新的技术,应像处理工程一样处理软件研制的全过程,程序的设计应易于保证其正确性、便于验证其正确性。于是,1969 年提出了结构化程序设计方法,1970 年第一个结构化程序设计语言—Pascal 语言出现,标志着结构化程序设计时期的开始。

80年代初开始,在软件设计思想上,又产生了一次革命,其成果就是面向对象的程序设计。正如汇编程序支持用汇编指令编程一样,FORTRAN和C语言的较低版本的系统支持结构化程序设计。在支持面向对象的语言开发工具中,Visual Basic以及Visual C++是广为人知的较成功的代表作。

高级语言的下一个发展目标是面向应用,也就是说:在该语言环境中,只要正确表达你需要做什么,就能自动生成算法以及目标程序,这就是非过程化的程序语言。

1.2 算法的概念、分类及特点

1.2.1 算法的概念

著名计算机科学家Wirth提出了一个著名的公式“数据结构+算法=程序”,是说一个程序应包括以下两方面内容:

- 对数据的描述。在程序中要指定数据的类型和数据的组织形式,即数据结构;
- 对操作的描述。即算法,亦即程序描述了解决一个问题而采取的方法和步骤。

1.2.2 算法的分类

算法一般分为两大类:数值算法和非数值算法。数值算法主要用于求数值解的问题,例如用牛顿迭代法解方程的根,用辗转相除法求两个数的最大公约数等。非数值算法主要用于解决需要用分析推理、逻辑推理才能解决的问题,例如人工智能中的许多问题,以及查找、分类等问题。

1.2.3 算法的特点

- 有穷性。一个算法应包含有限的操作步骤而不能是无限的。
- 确定性。算法中每一个步骤应当是确定的,而不能是含糊的、模棱两可的。
- 有零个或多个输入。
- 有一个或多个输出。
- 有效性。算法中每一个步骤应当能有效地执行,并得到确定的结果。

1.2.4 算法的描述工具

在设计程序时,通常使用专门的算法表达工具对算法进行描述。如自然语言、流程图、N-S图、伪码,等等。下面通过简单例子重点介绍前两种。

1. 用自然语言表示算法,通俗易懂、适合求解简单问题

例 1-1 设有两个整数A和B,它们的值分别为3和5,要求将它们互换。

第一步:定义一个新的变量 C,将 A 的值赋给 C。

第二步:将 B 的值赋给 A。

第三步:将 C 的值赋给 B。

以上程序可以简写为: S1: $C \leftarrow A$ S2: $A \leftarrow B$ S3: $B \leftarrow C$

例 1-2 输入一个数,求其绝对值并输出。

S1: 输入 x

S2: 判断 x 的值,若 x 小于零,则将 x 取反(即: $x \leftarrow -x$)

S3: 输出 x

例 1-3 求 1 到 100 之间所有整数的和

S1: $\text{sum} \leftarrow 0, t \leftarrow 1$

S2: $\text{sum} \leftarrow \text{sum} + t$

S3: $t \leftarrow t + 1$

S4: 若 $t \leq 100$ 则转到 S2, 否则转到 S5

S5: 输出 sum, 结束。

2. 用流程图表示算法,直观、易懂

流程图有如下基本符号,如图 1-2 所示。



图 1-2 流程图的基本符号及含义

例 1-1、1-2、1-3 用流程图表示,则如图 1-3(a)、1-3(b)、1-3(c) 所示。

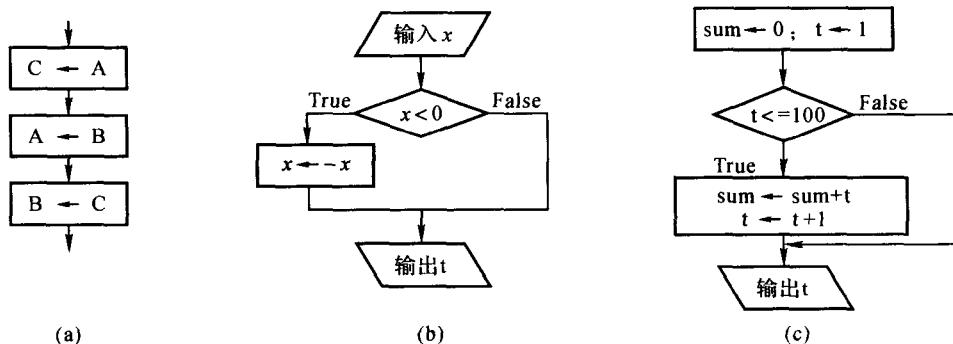


图 1-3 例 1-1、1-2、1-3 的流程图

1.3 C 语言的特点与 C 程序简介

C 语言于上世纪 70 年代诞生于美国的贝尔实验室。此前人们编写系统软件和应用程序主要使用汇编语言,由于依赖于计算机硬件,其可读性和可移植性都比较差。高级语言的可读

性和可移植性虽然较汇编语言好,但低级语言能直接控制和操作硬件,并且执行速度快,这是一般高级语言不具备的。在这种情况下,人们迫切需要一种既具有高级语言一般特性,又具有低级语言某些特点的语言,于是C语言应运而生。由于C语言集中了高级语言、低级语言长处(因此也被称为中级语言),已迅速普及并成为当今最有发展前途的计算机高级语言之一。

C语言的主要特点如下:

- 语言简洁、紧凑,使用方便、灵活。
- 运算符丰富。
- 数据结构丰富。
- 用函数作为程序的模块单位,实现程序的模块化,是良好的结构化语言,符合现代编程风格的要求。
- 能进行位运算,能实现汇编语言的大部分功能,可以直接对硬件进行操作。
- 生成目标代码质量高,程序执行效率高、可移植性好。

尽管C语言具有许多优点,但我们还得提醒大家:C语言语法限制不太严格,程序设计自由度大,导致一些副作用。如对数组下标越界不做检查,须由编程者自己保证程序的正确性,而不像其他高级语言一样对源程序的语法检查比较严、能检查出几乎所有的语法错误。

因此,对用C语言编写的源程序应当仔细检查,而不要过分依赖C编译程序去查错。“限制”与“灵活”是一对矛盾。限制严格就失去灵活性,而强调灵活就必然放松限制。一个不熟练的编程人员,编一个正确的C程序可能会比编一个其他高级语言程序难一些。也就是说,要求用C语言的人对程序设计更熟练一些。

C语言的以上特点,初学者现在也许还不能深刻理解,待学完C以后再回顾一下,就会有比较深的体会。

1.3.1 简单C程序示例

对于C程序设计的初学者来说,没有比自己编制程序、并在计算机上运算出正确结果更使人愉悦的事了。为使大家有一个感性的认识,我们先看几个简单的C程序。

例 1-4 编程,两个整数A和B的值分别为3和5,将它们互换。

```
/* 此程序功能是将两个整数互换 */
#include <stdio.h>
void main()
{ int i=3,j=5;          /* 定义两个整型变量并分别赋初值3和5 */
  int k;                /* 定义一个整型变量k */
  k=i; i=j; j=k;        /* 将i,j互换 */
  printf("i=%d j=%d\n",i,j); /* 输出i和j交换后的结果 */
}
```

程序运行结果为:i=5 j=3

例 1-5 编程,输入一个数,求其绝对值并输出。

```
/* 此程序功能是求一个实数的绝对值 */
#include <stdio.h>
```

```
void main()  
{ float x;                /* 定义一个实型变量 x */  
  scanf("%f",&x);          /* 从键盘输入一个数给变量 x */  
  if(x<0) x=-x            /* 判断 x 的值是否小于零,若是则取其负数 */  
  printf("x=%f\n",x);      /* 输出 x 的值 */  
}
```

运行时,假设从键盘输入 3.14(以回车键结束输入),运行结果为: $x=3.140000$

若输入-3.14,输出结果为: $x=3.140000$

例 1-6 求 1 到 100 之间所有整数的和。

/* 此程序功能是求 $1+2+\cdots+100$ 的和 */

#include <stdio.h>

```
void main()  
{ int sum=0;              /* 定义整型变量 sum,初值为 0。用它存放求和结果 */  
  int t=1;                /* 定义整型变量 t,初值为 1。用它存放循环次数 */  
  while(t<=100) {        /* 循环条件:t 的值不超过 100 */  
    sum=sum+t;            /* 循环体语句 1:累加 */  
    t=t+1;               /* 循环体语句 2:修改循环次数,每次增加 1 */  
  }  
  printf("1+2+...+100=%d\n",sum); /* 输出 sum 的值 */  
}
```

程序运行结果为: $1+2+\cdots+100=5050$

例 1-7 编程,输入两个整数,输出其中较小者。

#include <stdio.h>

```
int min(int x,int y)      /* 用户自定义 min 函数 */  
{ int z;                 /* 定义本函数中用到的变量 z 为整型 */  
  if(x<y) z=x;           /* 比较 x 和 y 的大小 */  
  else z=y;  
  return z;              /* 将 z 的值返回调用处 */  
}  
void main( )              /* 主函数 */  
{ int a,b,c;             /* 声明部分,定义变量 */  
  scanf("%d%d",&a,&b);    /* 输入变量 a 和 b 的值 */  
  c=min(a,b);            /* 调用 min 函数,将返回值赋值给 c */  
  printf("min=%d\n",c);  /* 输出 c 的值 */  
}
```

说明:

• 本程序包括两个函数:主函数 main 和被调用函数 min。任何 C 程序中都必须有且只能有一个主函数,主函数名必须为 main(专用名)。程序从 main 开始执行。

• 自定义函数 min 的作用是将 x 和 y 中较小者的值赋给变量 z,函数中的 return 语句将 z