



北京市高等教育精品教材立项项目

计算机基础系列教材

# C语言程序设计教程

(第二版)

主编 李凤霞  
编著 李凤霞 刘桂山  
陈朔鹰 薛庆  
主审 李书涛


 北京理工大学出版社  
BEIJING INSTITUTE OF TECHNOLOGY PRESS

北京市高等教育精品教材立项项目  
计算机基础系列教材

# C 语言程序设计教程

(第二版)

主编 李凤霞  
编著 李凤霞 刘桂山  
陈朔鹰 薛庆  
主审 李书涛

 北京理工大学出版社

BEIJING INSTITUTE OF TECHNOLOGY PRESS

## 内 容 简 介

本书是为高等院校第一门程序设计课程而编写的教材。全书分为十一章,主要包括:程序设计基础知识、C语言概述、数据类型与运算规则、顺序结构的程序设计、选择结构的程序设计、循环结构的程序设计、数组和字符数据处理、函数与程序结构、指针、结构类型及其他构造类型和文件。本教材在结构上突出了以程序设计为中心,以语言知识为工具的思想,对C语言的语法规则进行了整理和提炼,深入浅出地介绍了它们在程序设计中的应用;在内容上注重知识的完整性,适合初学者的需求;在写法上追求循序渐进,通俗易懂。本教材配有教学参考书《C语言程序设计教程习题与上机指导》,以方便读者复习和上机操作。

本书既可以作为高等院校非计算机专业学生的计算机语言教材,也可以作为高等院校计算机专业本科、专科低年级学生学习计算机语言的入门教材。

版权专有 侵权必究

---

### 图书在版编目(CIP)数据

C语言程序设计教程/李凤霞等编著. —2版. —北京:北京理工大学出版社,2004.6

ISBN 7-81045-785-3

I. C… II. 李… 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(2001)第05734号

---

出版发行 / 北京理工大学出版社

社 址 / 北京市海淀区中关村南大街5号

邮 编 / 100081

电 话 / (010)68914775(办公室) 68912824(发行部)

网 址 / <http://www.bitpress.com.cn>

电子邮箱 / [chiefedit@bitpress.com.cn](mailto:chiefedit@bitpress.com.cn)

经 销 / 全国各地新华书店

印 刷 / 北京圣瑞伦印刷厂

开 本 / 787毫米×1092毫米 1/16

印 张 / 25

字 数 / 486千字

版 次 / 2004年6月第2版 2004年6月第5次印刷

印 数 / 22001~32000册

定 价 / 29.00元

责任校对 / 陈玉梅

责任印制 / 刘京凤

---

图书出现印装质量问题,本社负责调换

## 第二版前言

《C 语言程序设计教程》自 2001 年出版以来,受到了广大读者的关注,在多所高校本科教学中使用受到广泛好评,本教材入选为首批“北京市高等教育精品教材立项项目”。在此,谨对广大读者的支持和鼓励表示最诚挚的谢意。

通过几年来的教学实践,我们收集了大量的反馈信息和修改意见,为进一步提高教材的质量,适应不断发展的计算机本科教学的要求,我们在保留第一版特点的基础上对全书进行了统一校正、充实和调整。首先对第一版中的错误和不妥之处进行了订正,并修改了部分例题,其次对部分章节进行了较大幅度的调整和改写。将原来的第 7 章数组和第 8 章字符数据处理合并为一章,将原来的第 9 章函数进行了重新组织,将编译预处理和 Turbo C 2.0 放入附录中,在原书第 11 章中增加了关于链表的内容,考虑到篇幅限制删掉了原来第 13 章对于 C++ 的介绍。

《C 语言程序设计教程》(第二版)由李凤霞主编,第 1、2、3 章和附录 D 由李凤霞编写,第 4、5、6、7 章由刘桂山编写,第 8、9、10 章和附录的其余部分由陈朔鹰编写,第 11 章由薛庆编写。北京理工大学李书涛教授认真审阅了全书,提出了许多宝贵意见和修改建议。在本书的修订过程中,一直得到了北京理工大学教务处和北京理工大学出版社的大力支持和帮助,在此一并表示衷心感谢。

由于作者水平有限,书中一定还存在不少错误和不妥之处,敬请读者批评指正。

编著者

2004 年 3 月

# 第一版前言

随着计算机技术的发展与普及,计算机已经成为各行各业最基本的工具之一,而且正迅速地进入人类生活的各个领域。C语言作为国际上广泛流行的通用程序设计语言,在计算机的研究和应用中已展现出强大的生命力。C语言兼顾了诸多高级语言的特点,是一种典型的结构化程序设计语言,它处理能力强,使用灵活方便,应用面广,具有良好的可移植性,既适合于计算机专业人员编写系统软件,又适合于应用开发人员编写应用软件。所以长久以来,广泛流行,经久不衰。

但是,由于C语言比其他计算机语言的学习难度要大,所以,在绝大部分高校中,以前总是先开设其他计算机语言课,然后再学习C语言。自从90年代后期,随着计算机新技术的不断涌现,高校计算机基础教学内容一再增加,非计算机专业本科生的计算机语言只能开设一门课程,C语言程序设计就被选为第一门本科生的程序设计必修课。作为程序设计的入门教材,应该介绍程序设计的基本概念和基本方法,在此前提下,再由浅入深地介绍C语言的基本内容。本书是针对计算机程序设计的初级读者,编写的一套《C语言程序设计教程》和《C语言程序设计教程习题与上机指导》。

本书是为大学本科和专科学生第一门程序设计课而编写的C语言程序设计教材。全书共分为十三章,主要内容包括:程序设计基础知识;C语言概述;数据类型与运算规则;顺序结构的程序设计;选择结构的程序设计;循环结构的程序设计;数组;字符数据处理;指针;函数与程序结构;结构类型及其他构造类型;数据文件和面向对象程序设计与C++简介等。全书以美国国家标准化协会颁布的C语言的新版本ANSI C为基础,以C语言程序设计为主线,介绍了程序设计的基本概念、C语言的语法规则和实用的C程序设计技术。书中结合应用实例,强调“好的”C程序编写方式,力图展示给初学者一个好的程序设计入门向导。

本教材在结构上突出了以程序设计为中心,以语言知识为工具的思想,对C语言的语法规则进行了整理和提炼,深入浅出地介绍了它们在程序设计中的应用;在内容上注重知识的完整性,以适合初学者的需求。例如安排了基础知识一章,用于介绍程序设计的基本概念、算法、流程图等内容。在第9章安排了工程文件一节,在第13章简要介绍了面向对象程序设计的概念与C++、Visual C++,使读者对实际应用有更广泛的了解;在写法上追求循序渐进,通俗易懂,旨在引导初学者入门。

本书以实际应用为目的,侧重于C语言基本知识,着重讲解概念,讲解用计算机解决问题的方法。删除了其他教科书中对C语言中过于“技巧性”问题的讨论与介绍。

对于初学者来说,掌握程序设计的技术和方法,在最初往往是枯燥乏味的。作为一种尝

试,我们在教学中曾经将各种程序设计的技术和方法融于趣味问题之中,通过对一些饶有趣味的问题进行讨论和求解,使读者在轻松、愉快的气氛中理解和探索程序设计的奥妙,从而达到事半功倍的学习效果。基于这样指导思想,本书在加强基础训练、介绍基本算法的同时,也选用了一些具有趣味性的例题。其目的就是要加强本书的可读性,使读者在轻松自然的学习过程中,掌握程序设计的方法。

本书中介绍的 C 语言,覆盖了国家教育委员会考试中心编写的《全国计算机等级考试考试大纲》中的二级考试大纲“C 语言程序设计考试要求”。同时,参考了部分省市计算机应用知识和应用能力水平考试大纲对 C 语言部分的要求。

全书各章配有习题,并有配套的习题及上机指导参考书。全部的例题和习题均在 Turbo C 环境下调试、运行。本书既可以作为高等院校非计算机专业学生的计算机语言教材,也可以作为高等院校计算机专业本科、专科低年级学生学习计算机语言的入门教材。本书还可以作为科技人员自学 C 语言的自学参考书。

参加本书编写的人员有:李凤霞、刘桂山、陈朔鹰和薛庆等。北京理工大学张鸿业教授审阅了全书,并提出了宝贵的意见。在本书的策划和出版过程中,一直得到学校各级领导的大力支持,许多从事教学工作的同仁也给予了关心和帮助,他们对本书提出了很好的建议。北京理工大学出版社,对本书的出版倾注了极大的热情和支持,在此一并表示衷心的感谢。

鉴于作者水平有限,加之时间仓促,书中一定存在不少错误和不妥之处,敬请读者批评指正。

编著者

2001 年 1 月

# 目 录

<b>第 1 章 程序设计基础知识</b> .....	( 1 )
1.1 程序与程序语言 .....	( 1 )
1.1.1 程序与程序语言 .....	( 1 )
1.1.2 程序设计 .....	( 2 )
1.2 算法和算法的表示 .....	( 4 )
1.2.1 算法的概念 .....	( 4 )
1.2.2 算法的基本特征 .....	( 6 )
1.2.3 算法的表示 .....	( 7 )
1.2.4 几种常用算法介绍 .....	( 10 )
1.3 结构化程序设计方法 .....	( 14 )
1.3.1 程序的三种基本结构 .....	( 15 )
1.3.2 N-S 流程图 .....	( 17 )
1.3.3 结构化程序设计方法 .....	( 19 )
小结 .....	( 22 )
习题 .....	( 22 )
<b>第 2 章 C 语言概述</b> .....	( 25 )
2.1 C 语言概况 .....	( 25 )
2.1.1 C 语言的发展 .....	( 25 )
2.1.2 C 语言的特点 .....	( 26 )
2.2 C 程序的基本结构 .....	( 27 )
2.3 C 语言的基本组成 .....	( 30 )
2.3.1 基本字符集 .....	( 30 )
2.3.2 标识符 .....	( 30 )
2.3.3 关键字 .....	( 31 )
2.3.4 语句 .....	( 31 )
2.3.5 标准库函数 .....	( 32 )
2.4 C 程序的上机执行过程 .....	( 32 )
小结 .....	( 34 )
习题 .....	( 34 )

第3章 数据类型与运算规则 .....	( 37 )
3.1 数据与数据类型 .....	( 37 )
3.1.1 什么是数据和数据类型 .....	( 37 )
3.1.2 C语言中的数据类型 .....	( 37 )
3.2 C语言的基本数据类型及其表示 .....	( 38 )
3.2.1 常量与变量 .....	( 38 )
3.2.2 整型数据及其表示 .....	( 41 )
3.2.3 实型数据及其表示 .....	( 46 )
3.2.4 字符型数据及其表示 .....	( 48 )
3.2.5 指针型数据及其表示 .....	( 51 )
3.2.6 变量的初始化 .....	( 53 )
3.3 算术运算与赋值运算 .....	( 53 )
3.3.1 C语言中的运算规则 .....	( 53 )
3.3.2 算术运算符与算术表达式 .....	( 55 )
3.3.3 自增、自减运算 .....	( 57 )
3.3.4 赋值运算符和赋值表达式 .....	( 58 )
3.3.5 组合赋值运算符和组合赋值表达式 .....	( 60 )
3.4 关系运算与逻辑运算 .....	( 61 )
3.4.1 关系运算符与关系表达式 .....	( 61 )
3.4.2 逻辑运算符与逻辑表达式 .....	( 63 )
3.4.3 条件运算符与条件表达式 .....	( 64 )
3.5 位运算 .....	( 65 )
3.5.1 位运算符 .....	( 65 )
3.5.2 按位逻辑运算 .....	( 66 )
3.5.3 移位运算 .....	( 68 )
3.5.4 位运算赋值运算符 .....	( 69 )
3.6 其他运算 .....	( 69 )
3.6.1 逗号运算符 .....	( 69 )
3.6.2 “.”和“->”运算符 .....	( 69 )
3.6.3 “( )”和“[ ]”运算符 .....	( 70 )
3.6.4 “*”和“&”运算符 .....	( 70 )
3.6.5 (type)运算符 .....	( 70 )
3.6.6 sizeof 运算符 .....	( 71 )
3.7 混合运算及数据类型转换 .....	( 71 )



3.7.1	混合运算	( 71 )
3.7.2	数据类型转换	( 71 )
3.8	应用实例	( 72 )
	小结	( 74 )
	习题	( 75 )
<b>第 4 章</b>	<b>顺序结构的程序设计</b>	<b>( 79 )</b>
4.1	C 语句概述	( 79 )
4.1.1	流程控制语句	( 80 )
4.1.2	表达式语句	( 81 )
4.1.3	复合语句	( 81 )
4.1.4	空语句	( 81 )
4.2	赋值语句	( 81 )
4.3	数据输出	( 82 )
4.3.1	输出一个字符的函数 putchar	( 83 )
4.3.2	格式化输出函数 printf	( 83 )
4.4	数据输入	( 86 )
4.4.1	输入一个字符的函数 getchar	( 86 )
4.4.2	格式化输入函数 scanf	( 86 )
4.4.3	关于输入方法	( 87 )
4.5	应用实例	( 87 )
	小结	( 89 )
	习题	( 90 )
<b>第 5 章</b>	<b>选择结构的程序设计</b>	<b>( 94 )</b>
5.1	用 if 语句设计选择结构程序	( 94 )
5.1.1	简单 if 语句	( 94 )
5.1.2	控制条件的表示	( 96 )
5.1.3	if _ else 语句	( 97 )
5.1.4	嵌套的 if 语句	( 98 )
5.2	用 switch 语句设计多分支结构程序	( 102 )
5.2.1	switch 语句	( 102 )
5.2.2	嵌套 switch 语句	( 105 )
5.3	goto 语句	( 106 )
5.4	应用实例	( 108 )
	小结	( 112 )

习题 .....	(113)
<b>第6章 循环结构的程序设计 .....</b>	<b>(118)</b>
6.1 循环的基本概念 .....	(118)
6.2 用 while 语句设计循环结构程序 .....	(119)
6.3 用 do _ while 语句设计循环结构程序 .....	(121)
6.4 用 for 语句设计循环结构程序 .....	(123)
6.5 break 语句与 continue 语句 .....	(125)
6.5.1 break 语句 .....	(125)
6.5.2 continue 语句 .....	(126)
6.6 几种循环语句的比较 .....	(128)
6.7 循环的嵌套 .....	(131)
6.8 应用实例 .....	(133)
小结 .....	(141)
习题 .....	(142)
<b>第7章 数组 .....</b>	<b>(147)</b>
7.1 数组的基本概念 .....	(147)
7.2 一维数组 .....	(148)
7.2.1 一维数组的定义 .....	(148)
7.2.2 一维数组的引用 .....	(149)
7.2.3 一维数组的初始化 .....	(150)
7.2.4 一维数组的应用 .....	(151)
7.3 一维字符数组 .....	(161)
7.3.1 一维字符数组的定义 .....	(161)
7.3.2 一维字符数组的初始化 .....	(162)
7.3.3 一维字符数组的引用 .....	(162)
7.3.4 字符串的输入输出 .....	(163)
7.4 字符串处理函数 .....	(166)
7.4.1 字符串复制 .....	(166)
7.4.2 求字符串的长度 .....	(167)
7.4.3 字符串连接 .....	(168)
7.4.4 字符串比较 .....	(170)
7.4.5 大小写字母的转换 .....	(172)
7.5 二维数组 .....	(172)
7.5.1 二维数组的定义 .....	(172)

7.5.2	二维数组的引用	(173)
7.5.3	二维数组的初始化	(174)
7.5.4	二维数组的应用	(175)
7.6	二维字符数组	(178)
7.6.1	二维字符数组的定义	(178)
7.6.2	二维字符数组的初始化	(179)
7.6.3	二维字符数组的引用	(179)
7.7	应用实例	(181)
	小结	(191)
	习题	(192)
<b>第8章</b>	<b>函数</b>	<b>(198)</b>
8.1	C 程序结构	(198)
8.1.1	概述	(198)
8.1.2	函数的分类	(199)
8.2	函数定义	(200)
8.2.1	函数定义的一般形式	(200)
8.2.2	无参函数的定义	(201)
8.2.3	有参函数的定义	(202)
8.3	函数的调用与返回	(203)
8.3.1	函数的调用方式	(203)
8.3.2	对被调函数的声明和函数原型	(204)
8.3.3	函数的返回	(206)
8.3.4	函数调用的执行过程	(209)
8.3.5	函数的嵌套调用	(211)
8.4	在函数之间传递数据	(212)
8.4.1	函数参数的传递规则	(212)
8.4.2	数组作为函数的参数	(216)
8.5	变量的存储属性	(219)
8.5.1	变量的生存期和作用域	(219)
8.5.2	自动变量	(221)
8.5.3	外部变量	(222)
8.5.4	静态变量	(226)
8.5.5	寄存器变量	(228)
8.5.6	变量存储类型的总结	(229)

8.6	函数的递归调用	(230)
8.6.1	递归的基本概念	(230)
8.6.2	递归程序的执行过程	(230)
8.6.3	数值型递归问题的求解方法	(233)
8.6.4	非数值型递归问题的求解方法	(234)
8.7	库函数简介	(239)
	小结	(241)
	习题	(241)
<b>第9章</b>	<b>指针</b>	<b>(246)</b>
9.1	指针及其引用	(246)
9.1.1	指针的基本概念	(246)
9.1.2	指针变量的说明	(247)
9.1.3	指针的引用和运算	(247)
9.1.4	为指针变量赋初值	(249)
9.1.5	引用指针变量	(249)
9.2	指针与函数	(250)
9.2.1	指针作函数的参数	(250)
9.2.2	函数返回指针	(252)
9.2.3	指向函数的指针	(253)
9.3	数组与指针	(255)
9.3.1	通过指针引用一维数组中的元素	(255)
9.3.2	指针基本运算	(257)
9.3.3	通过指针引用二维数组中的元素	(259)
9.4	指针与字符串	(261)
9.4.1	字符数组与字符指针	(261)
9.4.2	常见的字符串操作	(262)
9.5	指针数组	(265)
9.5.1	指针数组与数组指针	(265)
9.5.2	main 函数的参数	(268)
9.6	多级指针	(269)
9.7	应用实例	(272)
	小结	(275)
	习题	(276)

<b>第 10 章 结构、联合与枚举类型</b> .....	(281)
10.1 结构类型 .....	(281)
10.1.1 结构类型的概念与定义 .....	(281)
10.1.2 结构变量的说明 .....	(282)
10.1.3 引用结构中的成员 .....	(284)
10.1.4 结构的初始化 .....	(286)
10.2 结构数组 .....	(287)
10.3 结构指针 .....	(293)
10.4 在函数之间传递结构 .....	(298)
10.4.1 向函数传递结构的成员 .....	(298)
10.4.2 在函数之间传递整个结构 .....	(299)
10.4.3 向函数传递结构的地址 .....	(302)
10.5 联合类型 .....	(307)
10.6 枚举类型 .....	(309)
10.6.1 枚举的概念 .....	(309)
10.6.2 枚举运算 .....	(310)
10.7 用 typedef 定义类型 .....	(311)
10.8 链表基础 .....	(312)
10.8.1 链表的基本概念 .....	(313)
10.8.2 链表的基本操作 .....	(314)
10.8.3 链表的常见形式 .....	(321)
10.8.4 链表简单应用 .....	(322)
小结 .....	(326)
习题 .....	(327)
<b>第 11 章 文件</b> .....	(333)
11.1 文件概述 .....	(333)
11.1.1 什么是文件 .....	(333)
11.1.2 C 文件的分类 .....	(333)
11.2 文件的处理 .....	(335)
11.2.1 文件类型指针 .....	(336)
11.2.2 文件的一般操作过程 .....	(337)
11.2.3 打开文件函数 fopen .....	(337)
11.2.4 关闭文件函数 fclose .....	(339)
11.3 文件的顺序读写操作 .....	(339)

11.3.1	文件的字符输入/输出函数	(339)
11.3.2	文件的字符串输入/输出函数	(342)
11.3.3	文件的格式化输入/输出函数	(346)
11.3.4	文件的数据块输入/输出函数	(347)
11.4	文件的随机读写操作	(349)
11.4.1	改变文件位置指针函数 fseek	(349)
11.4.2	位置指针重返文件头函数 rewind	(350)
11.4.3	位置指针当前值函数 ftell	(351)
11.5	文件操作的状态和出错检测	(351)
11.5.1	文件状态检测函数 feof	(352)
11.5.2	报告文件操作错误状态函数 ferror	(352)
11.5.3	清除错误标志函数 clearerr	(352)
11.6	应用实例	(353)
	小结	(358)
	习题	(359)
附录 A	编译预处理	(362)
附录 B	C 语言运算符的优先级与结合性	(366)
附录 C	C 语言中的关键字	(367)
附录 D	Turbo C 2.0 简介	(367)
附录 E	Turbo C 2.0 常用库函数	(373)
附录 F	常用字符与 ASCII 代码对照表	(384)
	参考文献	(385)

# 第1章 程序设计基础知识

随着科学技术的迅猛发展,计算机技术日新月异,计算机程序设计语言也层出不穷。那么,什么是程序语言,什么是程序设计?应该学哪一种程序语言,如何进行程序设计?这些都是程序设计初学者首先遇到的问题,也是程序设计的基本问题、共性问题。

不论是什么样的计算机语言,其程序设计的基本方法是相同的。本书作为程序设计的入门教材,将以C语言程序设计为主线,介绍程序设计的基本概念和基本方法,讲述C语言的语法规则和实用的C程序设计技术。作为全书的开篇,本章就程序设计的基本知识作概括性讨论,重点介绍算法的概念、特征、设计算法的方法和策略、流程图的表示和结构化程序设计方法等内容。需要说明的是,有些概念和方法要随着后续各章的深入学习才会有深刻的理解,不必在一时不懂的问题上停滞不前。

## 1.1 程序与程序语言

### 1.1.1 程序与程序语言

#### 1. 计算机语言

什么是计算机语言?为什么要使用计算机语言?过去,一提到语言这个词,人们自然想到的是像英语、汉语等这样的自然语言,因为它是人和人相互交流信息不可缺少的工具。而今天,计算机遍布于我们生活的每一个角落,除了人和人之间的相互交流之外,我们还必须和计算机交流。用什么样的方式和计算机做最直接的交流呢?人们自然想到的是最古老也最方便的方式——语言。人和人交流用的是双方都能听懂和读懂的自然语言,同样,人和计算机交流也要用人和计算机都容易接受和理解的语言,这就是计算机语言。人们用自然语言讲述和书写,目的是给另外的人传播信息。同样,我们使用计算机语言把我们的意图表达给计算机,目的是使用计算机。

计算机语言是根据计算机的特点而编制的,它没有自然语言那么丰富多样,而只是有限规则的集合,所以它简单易学。但是,也正因为它是根据机器的特点编制的,所以交流中无法意会和言传,而更多地表现了说一不二,表现了“规则”的严谨。例如该是“;”的地方不能写成“.”,该写“a”的地方不能写成“A”,这使得人和计算机的交流在一开始会有些不习惯。不过,只要认识到计算机语言的特点,注意学习方法,把必须的严谨和恰当的灵活相结合,一切都会得心应手。



## 2. 程序

计算机是一种具有内部存储能力的自动、高效的电子设备,它最本质的使命就是执行指令所规定的操作。如果我们需要计算机完成什么工作,只要将其步骤用诸条指令的形式描述出来,并把这些指令存放在计算机的内部存储器中,需要结果时就向计算机发出一个简单的命令,计算机就会自动逐条顺序执行操作,全部指令执行完就得到了预期的结果。这种可以被连续执行的一条条指令的集合称为计算机的程序。也就是说,程序是计算机指令的序列,编制程序的工作就是为计算机安排指令序列。

但是,我们知道,指令是二进制编码,用它编制程序既难记忆,又难掌握,所以,计算机工作者就研制出了各种计算机能够懂得、人们又方便使用的计算机语言,程序就是用计算机语言来编写的。因此,计算机语言通常被称为“程序语言”,一个计算机程序总是用某种程序语言书写的。

## 3. 程序语言的发展

程序语言的产生和发展,直接推动了计算机的普及和应用。自第一个高级语言问世以来,人们已发明了上千种程序语言,常用的也有上百种。这些语言之间有什么区别,我们应该学习哪一种?

计算机语言按使用方式和功能可分为低级语言和高级语言。低级语言包括机器语言和汇编语言。机器语言就是计算机指令的集合,它与计算机同时诞生,是第一代的计算机语言;汇编语言是用符号来表示计算机指令,被称为第二代语言。机器语言和汇编语言都是围绕特定的计算机或计算机族而设计的,是面向计算机的语言。要使用这种语言必须了解计算机的内部结构,而且难学、难写、难记忆,把这种语言称为低级语言。因为低级语言是难以普及应用的,为此便产生了第三代语言——高级语言。它采用了完全符号化的描述形式,用类似自然语言的形式描述对问题的处理过程,用数学表达式的形式描述对数据的计算过程。可见,高级语言只是要求人们向计算机描述问题的求解过程,而不关心计算机的内部结构,所以把高级语言称为“面向过程语言”,它易于被人们理解和接受。典型的面向过程语言有 BASIC、FORTRAN、COBOL、C、Pascal 等。

随着计算机技术的迅猛发展,自从 20 世纪 80 年代以来,众多的第四代非过程化语言、第五代智能化语言也竞相推出。如果说第三代语言要求人们告诉计算机怎么做,那么第四代语言只要求人们告诉计算机做什么。因此,人们称第四代语言是“面向对象语言”。面向对象概念的提出是相对于“面向过程”的一次革命,面向对象技术在系统程序设计、多媒体应用、数据库等诸多领域得到广泛应用。但是,“面向过程”是程序设计的基础,尤其对于程序设计的初学者。所以,我们将以面向过程的 C 程序设计语言为背景,主要介绍程序设计的基本概念和方法。

### 1.1.2 程序设计

什么是程序设计呢? 在日常生活中我们可以看到,同一台计算机,有时可以画图,有时可



以制表、有时可以玩游戏,诸如此类,不一而足。也就是说,尽管计算机本身只是一种现代化方式批量生产出来的通用机器,但是,使用不同的程序,计算机就可以处理不同的问题。今天,计算机之所以能够产生如此之大的影响,其原因不仅在于人们发明了机器本身,更重要的是人们为计算机开发出了不计其数的能够指挥计算机完成各种各样工作的程序。正是这些功能丰富的程序给了计算机无尽的生命力,它们正是程序设计工作的结晶。而程序设计就是用某种程序语言编写这些程序的过程。

更确切地说,所谓程序,是用计算机语言对所要解决的问题中的数据以及处理问题的方法和步骤所做的完整而准确的描述,这个描述的过程就称为程序设计。对数据的描述就是指明数据结构形式;对处理方法和步骤的描述也就是下一节我们要讨论的算法问题。因而,数据结构与算法是程序设计过程中密切相关的两个方面。曾经发明 Pascal 语言的著名计算机科学家 N. Niklaus Wirth 教授关于程序提出了著名公式:程序 = 数据结构 + 算法。这个公式说明了程序设计的主要任务。但是在本书中,我们并没有以数据结构和算法为主展开讨论,因为本教材的主题是介绍用 C 语言进行编程。关于数据结构有专门的课程和教材,关于算法的问题我们在下一节给出初步的介绍。

对于程序设计的初学者来说,首先要学会设计一个正确的程序。一个正确的程序,通常包括两个含义:一是书写正确,二是结果正确。书写正确是指程序在语法上正确,符合程序语言的规则;而结果正确通常是指对应于正确的输入,程序能产生所期望的输出,符合使用者对程序功能的要求。程序设计的基本目标是编制出正确的程序,但这仅仅是程序设计的最低要求。一个优秀的程序员,除了程序的正确性以外,更要注重程序的高质量。所谓高质量是指程序具有结构化程度高、可读性好、可靠性高、便于调试维护等一系列特点。毫无疑问,无论是一个正确的程序,还是一个高质量的程序,都需要设计才能使之达到预期的目标。

那么,如何进行程序设计呢?一个简单的程序设计一般包含以下四个步骤:

(1) 分析问题,建立数学模型。使用计算机解决具体问题时,首先要对问题进行充分的分析,确定问题是什么,解决问题的步骤又是什么。针对所要解决的问题,找出已知的数据和条件,确定所需的输入、处理及输出对象。将解题过程归纳为一系列的数学表达式,建立各种量之间的关系,即建立起解决问题的数学模型。需要注意的是,有许多问题的数学模型是显然的或者简单的,以至于我们没有感觉到需要模型。但是有更多的问题需要靠分析问题来构造计算模型,模型的好与坏、对与错,在很大程度上决定了程序的正确性和复杂程度。

(2) 确定数据结构和算法。根据建立的数学模型,对指定的输入数据和预期的输出结果,确定存放数据的数据结构。针对所建立的数学模型和确定的数据结构,选择合适的算法加以实现。注意,这里所说的“算法”泛指解决某一问题的方法和步骤,而不仅仅是指“计算”。

(3) 编制程序。根据确定的数据结构和算法,用自己所使用的程序语言把这个解决方案严格地描述出来,也就是编写出程序代码。