



普通高等教育“十一五”国家级规划教材

21世纪大学计算机系列教材

C语言

大学实用教程 (第2版)

◆ 苏小红 陈惠鹏 孙志岗 等编著 ◆ 王宇颖 李东 主审



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>





普通高等教育“十一五”国家级规划教材

21世纪**大学计算机**系列教材

C语言

大学实用教程

(第2版)

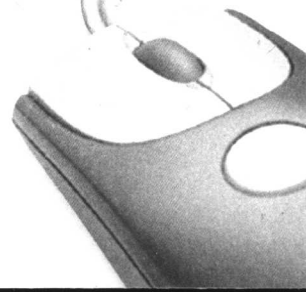
苏小红 陈惠鹏 孙志岗 等编著

王宇颖 李东 主审

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING



内 容 简 介

本书为普通高等教育“十一五”国家级规划教材。

本书是一本充满趣味性和实用性的大学 C 语言教材,适合作为大学各专业公共课教材和全国计算机等级考试参考书。全书由 11 章组成,内容包括:程序设计 ABC,数据类型、运算符与表达式,简单的 C 程序设计,程序的控制结构,函数,数组,指针,结构体与共用体,函数的高级应用,文件操作,C 程序设计常见错误及解决方案等。

本书注重教材的可读性和可用性,每章开头有内容关键词和难点提示;每章结尾安排本章小结,小结中给出了该章中的常见错误提示;典型例题一题多解,由浅入深,强化知识点、算法、编程方法与技巧;还将程序测试、程序调试与排错、软件的健壮性和代码风格、结构化与模块化程序设计方法等软件工程知识融入其中;配套教材《C 语言大学实用教程学习指导》包括三部分内容:习题指导、实验指导、错误案例与优秀案例分析;为任课教师免费提供电子课件,其中包括全部例题和习题源程序文件(可按前言提供的方式索取)。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

C 语言大学实用教程 / 苏小红,陈惠鹏,孙志岗等编著. —2 版. —北京:电子工业出版社,2007.2

普通高等教育“十一五”国家级规划教材

(21 世纪大学计算机系列教材)

ISBN 978-7-121-03782-5

I. C… II. ①苏… ②陈… ③孙… III. C 语言—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2007)第 005651 号

策划编辑:童占梅

责任编辑:章海涛

印 刷:北京季蜂印刷有限公司

装 订:三河市皇庄路通装订厂

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:28.25 字数:720 千字

印 次:2007 年 2 月第 1 次印刷

印 数:5 000 册 定价:33.00 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 zlt@phei.com.cn,盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010)88258888。

总 序

进入 21 世纪, 信息社会发展的脚步越来越快, 对人才的需求也呈现出新的变化趋势。计算机与外语成为新世纪高素质人才必须熟练掌握的工具。大学计算机公共课程也面临新的机遇和挑战, 首先是来自社会和就业市场对人才“知识—能力—素质”要求的挑战; 其次是计算机和相关领域技术及应用快速发展带来的冲击; 最后是普及计算机教育后要求高等计算机教育在教学的“难度—深度—强度”三维同步提高。在这样的大背景下, 大学计算机公共课程在“基础—技术—应用”方面呈现出层次性、通用性和专业需求多样化的特点。我们一直追踪、关注一线教师和专家的卓有成效的课程和教材改革与发展研究, 适时推出了“21 世纪大学计算机系列教材”。

该系列教材在知识结构方面力求覆盖“计算机系统与平台、程序设计与算法、数据分析与信息处理、信息系统开发”四个领域, 内容强调“概念性基础、技术与方法基础、应用技能”三个层次, 第一批教材涉及《大学计算机基础》、《程序设计与算法》、《计算机硬件技术基础》(或《计算机组成与接口技术》)、《数据库技术与应用》、《多媒体技术与应用》和《网络技术与应用》等六门核心课程。同时, 我们也在挖掘其他通用的应用课程教材, 并将陆续推出。我们特别注意到, 高校工科电类专业、理科和工科非电类专业、经管类专业和文史类专业有各自不同的特点, 可以采用“1+X”的课程解决方案, “1”指第一门计算机课程“大学计算机基础”, “X”指适合不同学校和专业特点的其他课程及其组合, 我们的系列教材为此提供了选择的灵活性。

“21 世纪大学计算机系列教材”立足体系创新、知识创新、教学设计和教学模式创新, 全面考虑读者的需求, 努力提升教材的可读性和可用性, 为教学提供尽可能完善的服务。如提供同步的“习题与实验指导”书, 一些教材还为教师提供可修改的电子教案、源程序包、教学指导手册或阶段自测题等多种类型的教学服务, 即提供“教材—教辅—课件”教学支持。读者可以通过电子工业出版社华信教育资源网站(<http://www.hxedu.com.cn> 或 <http://www.huaxin.edu.cn>) 了解该系列教材的出版和服务的动态信息。

“21 世纪大学计算机系列教材”的建设得到了很多专家和老师的热情支持, 教材作者来自哈尔滨工业大学、吉林大学、华中科技大学、中国科技大学、中山大学、北京邮电大学、浙江工业大学等高校, 这些课程都是各高校的教改优质课程和精品课程, 体现了作者对课程和教学的探索与创新。希望这套教材的出版能有力地推动大学计算机新课程体系的建立与发展, 同时也能为高等计算机教育带来与时俱进的活力和生机。

由于我们的水平和经验所限, 加之计算机和相关领域技术及应用的发展迅速, 该系列教材一定还存在不少缺点和不足, 欢迎领域专家和广大读者批评指正。我们会继续努力, 力求不断完善和提高, 以便更好地满足高等计算机教育不断变化的需求。

“21 世纪大学计算机系列教材”编委会

前 言

随便进入一家书店，来到计算机专柜，都可以看到琳琅满目的 C 语言书籍。在这种状况下写书，特色和实用性非常重要。

本书的目标是力争成为**最易懂、最专业、最时尚、最实用**的 C 语言教材和参考手册。

它首先是一本教材，适合于程序设计的初学者和想更深入了解 C 语言的人。每行文字的落笔，都以把问题讲清楚、讲明白、讲透彻，又不累赘为目标。同时抛弃了一些陈旧的内容，把程序设计领域最新、最有价值的思想和方法渗透到古老的 C 语言中，赋予 C 语言一个焕然一新面貌。

读者都有这样的体会：只追求浅显的教材，在读过一遍之后便可送人，没有长久保留的必要，也不会有常读常新的感觉。这本书不同，我们做了大量的搜集和整理工作，把各种**知识点、实际经验和常用算法**等分散渗透到各个相应章节中，并在附录 A 中提供索引，或者独立组织成文。这样做的目的就是便于读者随时查阅，使本书成为一本有保留价值的参考手册。毕竟很多深刻的内容不是简简单单读一遍就能掌握的，需要逐渐积累。愿读者每次重读本书字句，都能获得新的提高。

C 语言在本书中仅是起点，而非终点。本着面向未来的精神，我们把**程序设计中最基本的**、放之四海而皆准的**思想和方法**挖掘出来，以**C 语言为工具描述**它们，却不拘泥于 C 语言。以此培养读者无论在学习、工作中使用什么语言编程，都具有灵活应用这些思想和方法的能力。

趣味也是本书的一大特色。学习本身是一件充满乐趣的事情，它之所以使很多人感到枯燥，是因为没有人帮助他们发掘趣味。本书的作者都是有多年开发和授课经验的大学教师，并一直乐在其中。他们自然而然流露出的对 C 语言的赞叹、喜爱和沉迷之情，一定会感染读者。那些驾轻就熟的诙谐语言和生动有趣的示例，更能带给读者全新的学习体验。

全书共分 11 章，内容包括：程序设计 ABC，数据类型、运算符与表达式，简单的 C 程序设计，程序的控制结构，函数，数组，指针，结构体与共用体，函数的高级应用，文件操作，C 程序设计常见错误及解决方案等。最后一章**C 程序设计常见错误及解决方案**，可谓**本书的画龙点睛之笔**。后记则是作者多年从事程序设计的亲身体会和感悟，希望读者读后也能产生“众里寻他千百度，蓦然回首，那人却在灯火阑珊处”的共鸣。

本书注重教材的**可读性和可用性**，每章开头有**内容关键词和难点提示**，指导读者阅读；每章结尾安排**本章小结**，帮助读者整理思路，形成清晰的逻辑体系和主线，其中还给出了该章中的常见错误提示；**典型例题一题多解**，由浅入深，**强化知识点、算法、编程方法与技巧**；很多例题后面给出了思考题，不仅帮助读者了解什么是对的，还要了解什么是容易出错的。本书还将**程序测试、程序调试与排错、软件的健壮性和代码风格、结构化与模块化程序设计方法**等**软件工程知识融入其中**；习题以巩固基本知识点为目的，题型丰富，包括简答题、选择题、阅读程序写出运行结果、程序改错、程序填空和编程题等各种**全国计算机等级考试二级考试的常见题型**；附录中给出了实用学习资料列表。

本书是教育部**普通高等教育“十一五”国家级规划教材**，为方便高等院校各专业 C 语言教学和学生的自学，我们还为读者提供以下**全方位的辅教和辅学方面的信息服务**。

与本书同时配套出版《**C 语言大学实用教程学习指导**》，提供**全部习题解答、实验指导和学**

习指导内容。实验指导部分给出了在 Turbo C 环境、Visual C++ 环境及最新的 Code::Blocks+gcc+gdb 环境下的标准 C 程序调试方法编译环境下的标准 **C 程序调试方法**，相信一定会给读者以全新的体验和感受。实验指导部分以**主要知识点为主线**设计的实验题目，兼具**趣味性和实用性**，并以循序渐进的**任务驱动**方式，指导读者完成实验程序设计。该部分还包含了一个**综合应用实例**（学生成绩管理）可作为**本课程的课程设计**。学习指导部分不仅给出了部分**学生设计的优秀案例**（**游戏类程序**）分析，还给出了**常见错误案例分析**，帮助读者了解**错误发生的原因、实质、排错方法以及解决对策**。

本教材的**多媒体教学课件、全部例题和习题的源代码**都可在我们的 C 语言程序设计精品课程网站 (<http://210.46.96.21/jpsystem/properview.asp?lessonid=781>) 上**免费下载**，我们研制的**C 语言编程题自动评分系统**（已获软件著作权专利）也将免费提供给使用本教材一年以上的教学单位。有需要者可直接与作者本人联系（sxh@hit.edu.cn）。

学时建议：前面加星号*的章节和习题为有一定深度和开放性的选学内容，如果授课学时为 30 学时，教师可以留给学生自学；如果授课学时为 50 学时，则可以有选择地讲授。具体课时分配建议如右表所示。

30 学时		50 学时	
章	学时	章	学时
第 1 章	1	第 1 章	2
第 2 章	3	第 2 章	4
第 3 章	2	第 3 章	2
第 4 章	6	第 4 章	8
第 5 章	4	第 5 章	4
第 6 章	4	第 6 章	8
第 7 章	4	第 7 章	6
第 8 章	4	第 8 章	6
		第 9 章	4
第 10 章	2	第 10 章	4
		总复习	2

全书的统稿工作由苏小红教授负责，第 2、3、4、6、7 章及附录由苏小红编写，第 8、9 章由陈惠鹏编写，第 1、5、10、11 章及后记由孙志岗编写。在本次修订中，第 2、3、4、6、7、8、9 章的修订工作由苏小红完成，第 1、5、10、11 章的修订工作由孙志岗完成。

在本书的写作过程中，王宇颖和李东教授在百忙之中审阅了全部初稿，对本书提出了许多宝贵意见。在书稿的录入、校对、修订以及例题和习题程序的调试过程中，李秀坤、王庆北、温东新、傅忠传、张卫、张彦航、黄虎杰、李希然、秦兵、赵巍、郭萍、刘劲锋、张洪志、蔡江新、孙大烈、郝惠馨、单丽莉、刘国军、车万翔、刘开昌、李晗静、刘秉权、徐志明、李漾、娄久、周明德等做了大量工作。

长春理工大学光电信息学院计算机系的苗长彦、南京邮电大学计算机学院的朱立华、太原理工大学的马建芬等老师给我们的教材提出了许多宝贵的意见和建议，还有很多热心的读者给我们来信提出意见和建议，我们在此次修订过程中，部分采纳了他们的建议，在此一并向他们表示衷心的感谢。

因编者水平有限，书中错误在所难免，恳请读者批评指正。作者 E-mail 地址为 sxh@hit.edu.cn, chp@ir.hit.edu.cn, sun@hit.edu.cn。电子工业出版社华信教育资源网网址为 <http://www.huaxin.edu.cn> 或者 <http://www.hxedu.com.cn>，教材服务部联系电话为 010-88254480。我们会在每次重印时修改发现的错误，并及时将我们的教材勘误表刊登于我们的教材网站 (<http://book.sunner.cn>) 上，欢迎读者给我们发送电子邮件或在网站上留言，对教材提出宝贵意见。

编著者

于哈尔滨工业大学计算机科学与技术学院

目 录

第 1 章 程序设计 ABC	(1)
1.1 计算机与人	(2)
1.2 计算机与程序设计语言	(3)
1.3 程序设计语言的故事	(6)
1.4 C 语言的故事	(7)
1.5 程序设计语言的工作原理	(11)
1.5.1 运行	(11)
1.5.2 内存	(12)
1.6 本章小结	(13)
习题 1	(13)
第 2 章 数据类型、运算符与表达式	(14)
2.1 一个简单的 C 程序例子	(15)
2.2 C 程序常见符号分类	(16)
2.3 数据类型	(18)
2.3.1 为什么引入数据类型	(18)
2.3.2 从基本数据类型到抽象数据类型	(19)
2.3.3 类型修饰符	(19)
2.3.4 标识符命名	(21)
2.4 常量	(22)
2.4.1 整型常量	(22)
2.4.2 实型常量	(22)
2.4.3 字符常量	(23)
2.4.4 字符串常量	(24)
2.4.5 宏常量	(24)
2.4.6 枚举常量	(25)
2.5 变量	(26)
2.5.1 变量的声明与初始化	(26)
2.5.2 const 类型修饰符	(27)
2.5.3 使用变量时的注意事项	(28)
2.6 常用运算符及表达式	(32)
2.6.1 运算符的优先级与结合性	(32)
2.6.2 算术运算符	(33)

2.6.3	关系运算符	(34)
2.6.4	逻辑运算符	(36)
2.6.5	赋值运算符	(37)
2.6.6	增 1 和减 1 运算符	(38)
2.6.7	强制转换运算符	(40)
2.6.8	位运算符	(41)
2.6.9	逗号运算符	(43)
2.7	赋值和表达式中的类型转换	(44)
2.8	本章小结	(45)
	习题 2	(46)
第 3 章	简单的 C 程序设计	(48)
3.1	C 语句分类	(49)
3.2	表达式语句	(49)
3.3	复合语句和空语句的作用	(49)
3.4	基本输入/输出操作的实现	(51)
3.4.1	字符输入/输出	(51)
3.4.2	格式输入/输出	(53)
*3.4.3	使用函数 scanf()时需要注意的问题	(60)
3.5	本章小结	(70)
	习题 3	(71)
第 4 章	程序的控制结构	(75)
4.1	算法及其描述方法	(76)
4.1.1	算法的概念	(76)
4.1.2	算法的描述方法	(77)
4.2	顺序结构	(79)
4.2.1	顺序结构的流程图表示	(79)
4.2.2	应用程序举例	(80)
4.3	选择结构	(84)
4.3.1	应用场合	(84)
4.3.2	选择结构的流程图表示	(84)
4.3.3	条件语句	(85)
4.3.4	开关语句	(95)
4.4	循环结构	(100)
4.4.1	应用场合	(100)
4.4.2	循环结构的流程图表示	(100)
4.4.3	循环语句	(101)
4.4.4	单重循环应用举例	(103)

4.4.5	嵌套循环及其应用举例	(115)
4.5	流程的转移控制	(121)
4.5.1	goto 语句	(121)
4.5.2	break 与 continue 语句	(122)
4.5.3	函数 exit()	(124)
4.5.4	应用程序举例	(127)
*4.6	程序排错	(132)
4.6.1	程序中常见的出错原因	(133)
4.6.2	程序调试与排错基本方法	(134)
4.6.3	使用函数 getchar() 的问题	(136)
*4.7	结构化程序设计方法简介	(146)
4.7.1	关于 goto 论战	(146)
4.7.2	结构化程序设计的核心思想	(147)
4.7.3	自顶向下、逐步求精的程序设计方法	(147)
4.8	本章小结	(151)
	习题 4	(152)
第 5 章	函数	(162)
5.1	程序设计的艺术	(163)
5.2	函数的定义与使用	(164)
5.2.1	函数的分类	(164)
5.2.2	函数的定义	(165)
5.2.3	函数的调用、参数和返回值	(165)
5.2.4	函数原型	(167)
5.2.5	函数 main() 的特殊性	(169)
5.3	变量的作用域和存储类型	(170)
5.3.1	变量的作用域	(170)
5.3.2	全局变量	(172)
5.3.3	变量的存储类型	(174)
5.4	函数封装	(177)
5.5	预处理指令	(177)
5.5.1	#include	(177)
5.5.2	#define 和 #undef	(178)
5.5.3	条件编译	(180)
5.6	使用 assert() 查错	(181)
5.7	模块和链接	(182)
*5.8	模块化程序设计方法简介	(188)
5.8.1	功能分解	(188)

5.8.2	模块划分的原则	(189)
5.8.3	应用实例	(190)
5.9	本章小结	(193)
	习题5	(193)
第6章	数组	(196)
6.1	数组类型的应用场合	(197)
6.2	数组的定义、引用和初始化	(197)
6.2.1	数组的定义	(197)
6.2.2	数组的引用	(199)
6.2.3	数组的初始化	(200)
6.2.4	程序实例	(201)
6.3	向函数传递一维数组	(208)
6.4	向函数传递二维数组	(221)
6.5	字符数组	(229)
6.5.1	字符数组与字符串的关系	(229)
6.5.2	字符数组的输入/输出	(231)
6.5.3	字符串处理函数	(233)
6.5.4	应用举例	(234)
6.6	本章小结	(235)
	习题6	(237)
第7章	指针	(244)
7.1	指针概述	(245)
7.1.1	指针的概念	(245)
7.1.2	为什么引入指针的概念	(246)
7.1.3	指针变量作为函数参数	(249)
7.1.4	字符指针作为函数参数	(258)
7.2	指针和数组间的关系	(263)
7.2.1	一维数组的地址和指针	(264)
7.2.2	二维数组的地址和指针	(273)
7.3	指针数组	(279)
*7.4	指向指针的指针	(284)
7.5	带参数的函数 main()	(286)
*7.6	动态数组的实现	(288)
7.6.1	C程序的内存映像	(288)
7.6.2	动态内存分配函数	(288)
7.6.3	一维动态数组的实现	(290)
7.6.4	二维动态数组的实现	(291)

*7.7	使用 const 修饰指针变量	(293)
*7.8	代码风格问题	(294)
7.8.1	程序版式	(295)
7.8.2	命名规则	(298)
7.8.3	函数设计	(299)
7.8.4	防御性程序设计	(299)
7.9	本章小结	(300)
	习题 7	(302)
第 8 章	结构体与共用体	(309)
8.1	问题的提出	(310)
8.2	结构体类型与结构体变量	(312)
8.2.1	结构体类型的声明	(312)
8.2.2	关键字 typedef 的用法	(313)
8.2.3	结构体变量的定义	(314)
8.2.4	指向结构体变量的指针	(317)
8.2.5	结构体变量的引用和初始化	(318)
8.3	结构体数组	(322)
8.3.1	结构体数组的定义	(322)
8.3.2	结构体数组的应用实例	(323)
8.3.3	结构体数组与指针	(327)
8.4	结构体与函数	(329)
*8.5	动态数据结构	(332)
8.5.1	问题的提出	(332)
8.5.2	链表的定义	(334)
8.5.3	链表的特点及操作原理	(335)
8.5.4	动态链表的建立	(336)
8.5.5	链表的删除操作	(339)
8.5.6	链表的插入操作	(341)
8.6	共用体	(343)
*8.7	位段	(347)
8.8	本章小结	(350)
	习题 8	(350)
*第 9 章	函数的高级应用	(358)
9.1	递归	(359)
9.1.1	递归问题的提出	(359)
9.1.2	递归函数	(360)
9.2	返回指针值的函数	(364)
9.3	函数指针	(367)

9.4 一个综合应用实例——学生成绩管理程序	(372)
9.5 本章小结	(373)
习题 9	(374)
第 10 章 文件操作	(378)
10.1 计算机中的流	(379)
10.2 文件	(379)
10.2.1 存储设备的使用	(379)
10.2.2 目录	(380)
10.2.3 文件格式	(381)
10.3 基本文件操作	(381)
10.3.1 基本文件操作函数	(381)
10.3.2 错误处理	(386)
10.3.3 程序示例	(386)
10.3.4 基本文件操作的意义	(393)
10.4 高级文件操作	(394)
10.4.1 打开与关闭	(394)
10.4.2 读写	(394)
10.4.3 程序示例	(396)
10.4.4 标准输入与标准输出	(398)
10.5 本章小结	(400)
习题 10	(400)
第 11 章 C 程序设计常见错误及解决方案	(401)
条款 1: 使用未初始化和未赋值的变量	(402)
条款 2: 不考虑数值溢出的可能	(402)
条款 3: 不用 sizeof() 获得类型或变量的字长	(403)
条款 4: 假定类型取值范围	(403)
条款 5: 期望两个整数的运算自动得出浮点数的结果	(404)
条款 6: 不预先判断除数是否为 0	(405)
条款 7: 混淆 "&, l" 与 "&&, ll"	(405)
条款 8: 使用依赖编译器求值顺序的语句	(406)
条款 9: 使用依靠算符优先级的表达式	(406)
条款 10: 表达式过于复杂	(406)
条款 11: 用 "==" 时误用 "="	(407)
条款 12: 用 "==" 比较两个浮点数	(407)
条款 13: 使用幻数	(408)
条款 14: printf() 和 scanf() 中格式控制字符串与参数类型不匹配	(409)
条款 15: 循环或判断语句以 ";" 结尾	(409)

条款 16: 在循环体内改变循环结束条件	(410)
条款 17: case 分支不用 break 结束	(410)
条款 18: switch-case 语句没有 default 分支	(410)
条款 19: 不声明函数原型	(411)
条款 20: 不定义函数参数或返回值的类型	(411)
条款 21: 有返回值的函数不用 return 指明返回值	(412)
条款 22: 调用函数后, 不检查函数是否正确执行	(412)
条款 23: 变量、函数和模块功能不单一	(412)
条款 24: 函数过长	(413)
条款 25: 函数的参数过于复杂	(413)
条款 26: 返回指向局部变量的指针	(413)
条款 27: 随意修改全局变量的值	(413)
条款 28: 数组下标越界	(414)
条款 29: 字符串没有'\0'终结符	(414)
条款 30: 使用不限制最大处理长度的字符串处理函数	(414)
条款 31: 用 malloc()申请的内存不用 free()	(414)
条款 32: 使用已经被 free()的指针	(415)
条款 33: 文件打开后不主动关闭	(415)
条款 34: 成对函数不在同一个模块或函数内调用	(415)
条款 35: 头文件不加宏定义锁	(415)
条款 36: 忽略编译器警告	(416)
条款 37: 用“复制+粘贴”的方式复用代码	(416)
条款 38: 在字符串和注释以外的地方使用全角字符	(416)
条款 39: 代码风格不佳	(417)
条款 40: 代码与注释不一致	(417)
后记	(419)
附录 A 书中常用概念、名词、语句、运算符、数据类型和算法索引表	(421)
附录 B C 关键字	(425)
附录 C Visual C++下各数据类型所占字节数和取值范围	(426)
附录 D C 运算符的优先级与结合性	(427)
附录 E 常用 ASCII 字符表	(428)
附录 F 常用字符表	(429)
附录 G 常用的 ANSI C 标准库函数	(430)
参考文献	(437)

第1章

程序设计 ABC

内容关键词

- ✎ 计算机、计算机的工作原理
- ✎ 程序设计语言、程序设计语言的工作原理
- ✎ 内存

难点提示

- ✎ 冯·诺依曼机工作过程
- ✎ 二进制
- ✎ 编译运行与解释运行

计算机在读者手中是否还只是一个上网、聊天、游戏、看电影的工具?你是否觉得“编程”是件高不可攀的事情?“黑客”是否还是你崇拜的对象?如果你给出的答案是“是”,那么这本书将力图把它们变成“否”。

本章将用最通俗的语言向你展现计算机以及程序设计的无穷魅力。你将了解到计算机对人类生活的影响;它是如何获得如此魔力的;程序设计扮演什么样的角色;程序设计语言是怎样的语言;怎样驾驭语言成为“编程高手”。

1.1 计算机与人

计算机界的泰斗——图灵 (Turing)^① (如图 1-1 所示) 曾经做过一个梦。梦中他与隔壁的一个人聊天,谈了很多事情。聊天结束,他走到隔壁,想见见这个人,却只看到一台计算机。



图 1-1 阿兰·图灵

原来一直陪他说话的是这台计算机。这便是著名的“图灵测试^②”的一个形象化的故事。让计算机像人一样地思考,与人自然交流,也就是所谓的“人工智能 (Artificial Intelligence)”,它一直是计算机行业的梦想。所以我们给计算机又起了一个可爱的昵称——电脑,期望其能与人类相当,控制机器做出像人甚至超越人的事情。

有很多科幻故事描绘了计算机真正成为电脑时的景象。如《变形金刚》里的机器人大战,《AI (人工智能)》里探讨的人类与机器的感情关系,《终结者》里机器人企图消灭人类等。在最有想象力的《Matrix (黑客帝国)》中,计算机不仅奴役了人类的躯体,使人类成为机器的生物电池,更奴役了人类的精神,让人类生活在计算机程序构成的虚拟世界里,按照计算机的指挥去行动、思考。如此下去,计算机简直都要让上帝下岗了。

这些都是文学家、艺术家的幻想,目前及可预见的未来,计算机还不可能达到这样的智能程度。即便达到了,人类的智慧也是有能力控制它的。然而,计算机确实实战胜过人类。1997年,IBM 公司研制的深蓝 (Deep Blue) 超级计算机在一场“人机大战”中打败了国际象棋大师卡斯帕罗夫,被誉为“人工智能的一大胜利”。不过深蓝的主要研制者之一,许峰雄博士却在公开场合说:“AI is bullshit.”在他看来,胜利靠的只是不知疲倦地高速运算,并不是什么智能。目前计算机最大的本领,也确实就是不知疲倦地高速运算。然而,就是如此简单的高速运算,已经足以影响我们的生活,甚至完全改变我们的生活。

当计算机技术与通信技术融合时,就形成了信息技术 (Information Technology), 诞生了一个新名词 IT。IT 的力量,带来的是又一次的工业革命。现在有一个很流行的说法:“无所不

^① Alan Turing (1912—1954), 生于英国伦敦, 被认为是 20 世纪最著名的数学家之一。美国计算机协会 (ACM) 在 1966 年设立的每年一届的以他的名字命名的“图灵奖 (Turing Award)”, 是计算科学的最高荣誉, 是计算机界的“诺贝尔奖”。

^② 图灵测试: 图灵在 1950 年的文章《计算机器和智能》中讨论了可以把一个机器视为有智能的条件。他说, 如果一个机器可以在一个博学的人面前成功地假扮成一个人的话, 那么我们可以说这个机器是有智能的。

在的计算 (Pervasive Computing)”。无所不在的网络和网络之星罗棋布的计算机共同推动着这场革命。

如果早几年写这本书,可能作者还要列举一些轰轰烈烈的计算机推动产业的例子。但今天不同了,已经不需要了,因为几乎所有的人都已经体会过 IT 所带来的成功与便利。但我要说的是,即使你每天与计算机和网络为伴,你所体会的还只是冰山一角。

从某种程度看,IT 业更像发电厂、自来水厂,是其他行业的支持者。IT 业纵深研究的主要目的之一,是为其他行业提供更好的服务。这种努力的结果已经显现出来,各行各业几乎都开始逐渐离不开 IT 了。比如,在自动控制领域,计算机的出现简直令其改天换地。因为这一领域主要应用嵌入式系统 (Embedded System)^①,而且实时性要求高,所以这个行业里的程序员对 C/C++ 及计算机底层技术的掌握程度甚至超过了绝大多数计算机专业人士。再比如玩钢筋水泥的建筑业,对建筑师来说,鼠标已成为安全帽以外的另一个标配。如果不用计算机画设计效果图,不用软件仿真各种应力对建筑的影响,那么这个建筑师就落伍了。国外的众多建筑公司协同软件公司开发了很多非常流行的专用软件,为建筑这一最古老的工程行业带来了新的动力。国内也正在做这方面的努力。这样看来,“水泥+鼠标”绝不仅仅是网络泡沫^②下 IT 人士的理想,而在另一领域,“听诊器+鼠标”则成了医生的一个梦。

“21 世纪是生物技术的世纪”,这一论点几乎毋庸置疑。就像其他学科一样,生物医学技术的飞速发展同样依靠 IT 技术。比如当前的研究热点基因组图谱工程,其工作量巨大,如果没有计算机的帮助,几乎不可能完成。在医学上,CT、核磁共振和彩超等医疗器械也广泛使用了计算机技术,未来的远程医疗将更加依赖 IT。

IT 对其他学科有影响,而反过来,其他学科对 IT 也有影响。比如,“DNA 计算机”就是利用生物技术建造的全新架构的计算机,它成本更低、速度更快、容量更大,在一些领域很可能完全替代现有的计算机。再比如,计算机科学里很重要的一个方向——软件工程,它从建筑工程里借鉴了很多宝贵的管理、设计经验。在计算机专业,现在有越来越多的研究方向是与生物、机械、控制、管理等方向结合的交叉方向。

由此可见,不管未来是什么样的世界,它都离不开 IT。不管读者从事什么工作,IT 都将是你的有力工具。

1.2 计算机与程序设计语言

计算机很神奇,其魔力可用古老的中国哲学——《易经》来解释。“易有太极,是生两仪,两仪生四象,四象生八卦,八卦定吉凶,吉凶生大业。”“吉凶”与“大业”可代表世间万物,《易经》中的这句话阐述的哲学观点是,万物起源于“两仪”,也就是“阴阳”。按此观点,如果计算机想“什么都能干”,想控制万物,那么它只要抓住万物的本质——“阴阳”——就可

① 嵌入式系统:简单地说,将计算机嵌入到另一个系统里,控制那个系统工作,就是嵌入式系统。典型的嵌入式系统有机器人、手机和智能家电等。这是 IT 未来的主要发展方向之一。

② 20 世纪末,随着 Internet 的发展,网络经济迅速崛起,很多年轻的网络公司很快壮大起来,其股票市值扶摇直上。21 世纪初,因为这些公司中的绝大部分都不盈利,股票开始大幅下跌,投资人信心崩溃,“网络泡沫”破灭,众多网站合并、倒闭或惨淡维持。那时诞生了一个观点,就是互联网要和传统行业结合发展,也就是所谓的“水泥+鼠标”。

以了。计算机确实以“阴阳”为根本,并把它俩表示成“0”和“1”,命名为“二进制(Binary)”。

二进制是否源自于《易经》,各界人士有很多论战,并无定论,有兴趣的读者可以去网上搜索一下相关文章,但计算机靠二进制来表达万物确是真的。

计算机为什么用二进制呢?为什么不用我们日常熟悉的十进制?一方面,二进制在电器元件中容易实现。二进制只有 0 和 1 两个数,在电学中具有两种稳定状态,并可以用 0 和 1 表示的东西很多。例如,电压的高和低、电容器的充电与放电、脉冲的有与无、晶体管的导通与截止,等等;而要找出具有十种稳定状态的电子元件则是很困难的。另一方面,二进制数的加法运算规则只有 $2^2=4$ 条,非常简单,而十进制的加法运算公式从 $0+0=0$ 到 $9+9=18$ 共有加法运算规则 $10^2=100$ 条。显然,计算机进行二进制运算比进行十进制运算要简单得多。

英文里“Computer”是“Compute(计算)”加“er”后缀派生而来的。直译过来就是“计算的机器”,简称计算机。从名字上看,计算机不过是一台机器,它只会计算。事实上也确实如此。它把万物都表达成二进制的“0”与“1”的排列组合的形式,形成一个个二进制数序列,然后用各种运算手段对这些数序列进行计算,结果还是二进制数序列。这样做看上去并不复杂,你我稍加学习都能做到同样的事情,但是我们都没有计算机算得快、记得多,且不知疲倦,所以计算机大有用处。

正因为计算机只会计算,所以专门研究计算的数学在计算机科学中是很重要的。现实世界中的各种过程都被抽象成数学模型,表达成一个个数学公式;各种事物都用数字表示,代入公式中求解。这里面的奇特原理,实非本书所要阐述的重点;若读者有兴趣,可以广泛阅读其他专业资料。本书在后面的章节也会简单地介绍这方面的知识。

不管多么复杂的算式,计算机都能算。怎么让计算机知道你想让它算什么呢?最理想的情况是对着它说一声:“嗨,计算机,帮我把《高等数学》第 12 页第 25 题做了。”答案马上打印出来,甚至直接用电子邮件发给老师。不过目前计算机还做不到,但将来肯定能做到。这里面的主要问题是计算机听不懂“人话”。也许你用过 IBM ViaVoice 这样的语音输入软件,并为它总弄不清楚你说的是“例子”、“粒子”还是“栗子”而苦恼。也许你用过手机的语音拨号,但换一个人说同样的名字它就死活不知道该拨哪个号码。为了能让计算机听懂“人话”,我们先要懂计算机语言,然后用计算机语言对计算机说话,再一点点教计算机学会我们的语言。计算机语言就是“程序设计语言”。

在弄清程序设计语言之前,我们先看一看计算机是如何工作的。

1946 年,冯·诺依曼^①在总结前人工作的基础上,率先在计算机中采用二进制,并且提出了著名的“冯·诺依曼机”结构。这一结构至今仍然被几乎所有的计算机采用,他也因此被誉为“计算机之父”(如图 1-2 所示)。

冯·诺依曼机,把计算机简化为控制器、运算器、存储器、输入和输出设备五个部分。控制器和运算器就是 CPU,它被比做计算机的大脑;存储器被分为内存和外存两部分,它们使计算机具有记忆能力;输入设备就是计算机的耳朵和眼睛,是人给计算机发送指令的工具,如键盘、鼠标;输出设备是计算机给人结果反馈的设备,如显示器、打印机。

^① John Von Neumann (1903—1957), 生于匈牙利的布达佩斯, 是 20 世纪最杰出的数学家之一, 现代计算机之父。