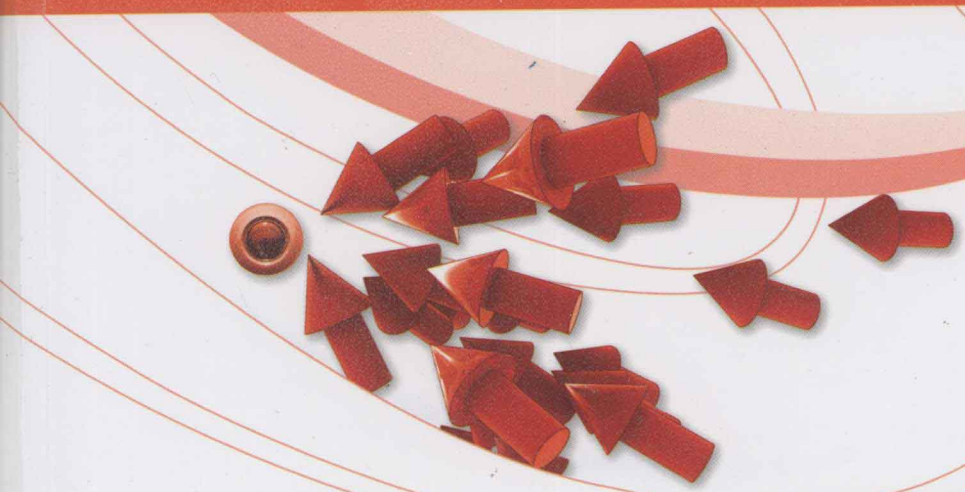
普通高等教育“十二五”规划教材



工程创新型“十二五”规划计算机精品教材

数据结构与算法
（C语言版）

胡　明　王红梅　编著

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

普通高等教育“十二五”规划教材
工程创新型“十二五”规划计算机精品教材

数据结构与算法
(C语言版)

胡　明　王红梅　编著

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

内 容 简 介

本书将基本的算法设计技术和数据结构很好地结合起来，第1章介绍数据结构和算法在程序设计中的作用，以及数据结构和算法的基本概念；第2章以初等数论作为应用实例介绍基本的算法设计技术，使学生初步理解常用的蛮力法、分治法、减治法、贪心法、动态规划法等算法设计技术的设计思想；第3～7章依次介绍线性表、栈和队列、字符串和多维数组、树和二叉树、图等数据结构，并从算法设计技术的角度讨论数据结构的基本操作；第8章和第9章是常用数据处理技术，包括查找和排序，并从算法设计技术的角度阐述查找和排序的算法思想和设计过程。

本书内容丰富，层次清晰，深入浅出，可作为高等学校计算机及相关专业数据结构课程的教材，也可供从事计算机软件开发和应用的工程技术人员阅读、参考。

图书在版编目(CIP)数据

数据结构与算法：C语言版 / 胡明，王红梅编著. —北京：电子工业出版社，2011.11
工程创新型"十二五"规划计算机精品教材
ISBN 978-7-121-14834-7

Ⅰ. ①数… Ⅱ. ①胡… ②王… Ⅲ. ①数据结构-高等学校-教材 ②算法分析-高等学校-教材 ③C语言-程序设计-高等学校-教材 Ⅳ. ①TP311.12 ②TP301.6 ③TP312

中国版本图书馆CIP数据核字(2011)第212448号

策划编辑：韩同平
责任编辑：韩同平　　特约编辑：李佩乾
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：787×1 092　1/16　印张：16.5　字数：423千字
印　　次：2011年11月第1次印刷
印　　数：3 000册　　定价：32.00元

凡所购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(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。

前　言

数据结构是计算机及相关专业的核心课程，也是计算机及相关专业考研和水平等级考试的必考科目，而且正逐渐发展成为众多理工科专业的热门选修课。算法被公认为是计算机科学的基石，利用计算机求解问题的最重要一步是将人的想法描述成算法。如何使学生真正掌握构成程序的两个重要的组成部分——数据结构和算法，提高问题求解能力是一个值得探索的教研课题。

本书将基本的算法设计技术和数据结构很好地结合起来，第1章介绍数据结构和算法在程序设计中的作用，以及数据结构和算法的基本概念；第2章以初等数论作为应用实例介绍基本的算法设计技术，使学生初步理解常用的蛮力法、分治法、减治法、贪心法、动态规划法等算法设计技术的设计思想；第3～7章依次介绍线性表、栈和队列、字符串和多维数组、树和二叉树、图等数据结构，并从算法设计技术的角度讨论数据结构的基本操作；第8章和第9章是常用数据处理技术，包括查找和排序，并从算法设计技术的角度阐述查找和排序的算法思想和设计过程。这样，一方面使学生能够学以致用，将算法设计技术应用到数据结构的实现上，另一方面能够更深刻地理解数据结构的实现方法。此外，本书还具有以下特色：

1. 紧扣《计算机学科硕士研究生入学考试专业基础综合考试大纲》，抓牢核心概念，提炼基础性知识，合理规划教学内容。

2. 定位明确，突出工程实践。减少形式化描述，注重算法设计与程序实现，每章通过两个应用实例展示数据结构和算法设计技术的实践过程。

3. 遵循认知规律，理清教学主线。根据学生的认知规律，按照从已知到未知的思维进程逐步推进教学内容，知识单元的拓扑结构安排合理，主线清晰。

4. 以知识为载体，注重能力培养。能够注意引导思维，通过讲思路讲过程讲方法，展现问题的求解过程。以算法为例，按照“提出问题→分析问题→解决问题”的过程，采用“图示理解→伪代码描述算法→C语言描述算法”的三级模式，培养计算思维能力。

5. 分析难点，针对处理。针对数据结构内容抽象的特点，全书设计了大量插图，将抽象的内容进行了具体化处理，降低了理解问题的复杂性。

6. 开扩视野，激发兴趣。本书的脚注中给出了与数据结构相关的人物小传、各种数据结构的起源，以及某些知识点的相关处理，激发学习兴趣，对学生的思维方式产生有益的影响。

本书由胡明、王红梅编著。参加本书编写的还有王涛、党源源、谷钰、刘冰等老师，2008级胡洁珺同学校对了书稿。

本书为吉林省精品课程“数据结构”的配套教材，课程网站 http://jsj.ccut.edu.cn/sjjg

本书配有电子课件，可登录华信教育资源网（www.hxedu.com.cn）免费下载。

由于作者的知识和写作水平有限，书稿虽几经修改，仍难免有缺点和错误，欢迎专家和读者批评指正。作者的电子邮箱是：

huming@mail.ccut.edu.cn

wanghm@mail.ccut.edu.cn

作　者

2011年9月于长春

目　录

第 1 章　绪论 …… (1)
1.1　问题求解与程序设计 …… (2)
1.1.1　程序设计的一般过程 …… (2)
1.1.2　数据结构在程序设计中的作用 …… (4)
1.1.3　算法在程序设计中的作用 …… (5)
1.1.4　本书讨论的主要内容 …… (6)
1.2　数据结构的基本概念 …… (8)
1.2.1　数据结构 …… (8)
1.2.2　抽象数据类型 …… (10)
1.3　算法的基本概念 …… (12)
1.3.1　算法及其重要特性 …… (12)
1.3.2　算法的描述方法 …… (13)
1.4　算法分析 …… (15)
1.4.1　算法的时间复杂度 …… (15)
1.4.2　算法的空间复杂度 …… (17)
1.4.3　算法分析举例 …… (17)
习题 1 …… (20)
第 2 章　基本算法设计技术 …… (22)
2.1　蛮力法 …… (23)
2.1.1　蛮力法的设计思想 …… (23)
2.1.2　算法设计实例——数字谜 …… (23)
2.2　分治法 …… (24)
2.2.1　分治法的设计思想 …… (24)
2.2.2　算法设计实例——数字旋转方阵 …… (25)
2.3　减治法 …… (27)
2.3.1　减治法的设计思想 …… (27)
2.3.2　算法设计实例——假币问题 …… (28)
2.4　贪心法 …… (30)
2.4.1　贪心法的设计思想 …… (30)
2.4.2　算法设计实例——埃及分数 …… (31)
2.5　动态规划法 …… (33)
2.5.1　动态规划法的设计思想 …… (33)
2.5.2　算法设计实例——数塔问题 …… (33)
习题 2 …… (36)
第 3 章　线性表 …… (38)
3.1　引言 …… (39)
3.2　线性表的逻辑结构 …… (40)
3.2.1　线性表的定义 …… (40)

3.2.2 线性表的抽象数据类型定义 …… (40)
3.3 线性表的存储结构及实现 …… (42)
3.3.1 顺序表 …… (42)
3.3.2 单链表 …… (47)
3.3.3 双链表 …… (56)
3.3.4 循环链表 …… (58)
3.3.5 静态链表 …… (60)
3.3.6 顺序表和链表的比较 …… (61)
3.4 应用实例 …… (62)
3.4.1 约瑟夫环问题 …… (62)
3.4.2 一元多项式求和 …… (65)
习题3 …… (69)
第4章 栈和队列 …… (72)
4.1 引言 …… (73)
4.2 栈 …… (74)
4.2.1 栈的逻辑结构 …… (74)
4.2.2 栈的顺序存储结构及实现 …… (75)
4.2.3 栈的链接存储结构及实现 …… (79)
4.2.4 顺序栈和链栈的比较 …… (80)
4.3 队列 …… (81)
4.3.1 队列的逻辑结构 …… (81)
4.3.2 队列的顺序存储结构及实现 …… (82)
4.3.3 队列的链接存储结构及实现 …… (85)
4.2.4 循环队列和链队列的比较 …… (87)
4.4 应用举例 …… (87)
4.4.1 括号匹配问题 …… (87)
4.4.2 表达式求值 …… (89)
习题4 …… (92)
第5章 字符串和多维数组 …… (94)
5.1 引言 …… (95)
5.2 字符串 …… (95)
5.2.1 字符串的逻辑结构 …… (95)
5.2.2 字符串的存储结构 …… (97)
5.2.3 模式匹配 …… (98)
5.3 多维数组 …… (101)
5.3.1 数组的逻辑结构 …… (101)
5.3.2 数组的存储结构与寻址 …… (102)
5.4 矩阵的压缩存储 …… (103)
5.4.1 对称矩阵的压缩存储 …… (104)
5.4.2 三角矩阵的压缩存储 …… (104)
5.4.3 对角矩阵的压缩存储 …… (105)
5.4.4 稀疏矩阵的压缩存储 …… (105)
5.5 应用实例 …… (107)
5.5.1 发纸牌 …… (107)

5.5.2 八皇后问题 …… (109)
习题5 …… (111)
第6章 树和二叉树 …… (114)
6.1 引言 …… (115)
6.2 树的逻辑结构 …… (116)
6.2.1 树的定义和基本术语 …… (116)
6.2.2 树的抽象数据类型定义 …… (117)
6.2.3 树的遍历操作 …… (118)
6.3 树的存储结构 …… (119)
6.3.1 双亲表示法 …… (119)
6.3.2 孩子表示法 …… (120)
6.3.3 孩子兄弟表示法 …… (121)
6.4 二叉树的逻辑结构 …… (122)
6.4.1 二叉树的定义 …… (122)
6.4.2 二叉树的基本性质 …… (123)
6.4.3 二叉树的抽象数据类型定义 …… (125)
6.4.4 二叉树的遍历操作 …… (126)
6.5 二叉树的存储结构 …… (127)
6.5.1 顺序存储结构 …… (127)
6.5.2 二叉链表 …… (128)
6.5.3 三叉链表 …… (131)
6.5.4 线索链表 …… (132)
6.6 二叉树遍历的非递归算法 …… (134)
6.6.1 前序遍历非递归算法 …… (135)
6.6.2 中序遍历非递归算法 …… (136)
6.6.3 后序遍历非递归算法 …… (136)
6.7 树、森林与二叉树的转换 …… (138)
6.8 应用实例 …… (140)
6.8.1 文件系统 …… (140)
6.8.2 哈夫曼树及哈夫曼编码 …… (144)
习题6 …… (148)
第7章 图 …… (150)
7.1 引言 …… (151)
7.2 图的逻辑结构 …… (152)
7.2.1 图的定义和基本术语 …… (152)
7.2.2 图的抽象数据类型定义 …… (154)
7.2.3 图的遍历操作 …… (155)
7.3 图的存储结构及实现 …… (157)
7.3.1 邻接矩阵 …… (157)
7.3.2 邻接表 …… (160)
7.3.3 邻接矩阵和邻接表的比较 …… (164)
7.4 最小生成树 …… (164)
7.4.1 Prim 算法 …… (165)
7.4.3 Kruskal 算法 …… (167)

7.5 最短路径 …… (170)
7.5.1 Dijkstra 算法 …… (171)
7.5.2 Floyd 算法 …… (172)
7.6 有向无环图及其应用 …… (174)
7.6.1 AOV 网与拓扑排序 …… (174)
7.6.2 AOE 网与关键路径 …… (177)
7.7 应用实例 …… (179)
7.7.1 七巧板涂色问题 …… (179)
7.7.2 医院选址问题 …… (180)
习题7 …… (182)
第8章 查找技术 …… (185)
8.1 概述 …… (186)
8.1.1 查找的基本概念 …… (186)
8.1.2 查找算法的性能 …… (187)
8.2 线性表的查找技术 …… (187)
8.2.1 顺序查找 …… (187)
8.2.2 折半查找 …… (188)
8.2.3 分块查找 …… (191)
8.3 树表的查找技术 …… (192)
8.3.1 二叉排序树 …… (192)
8.3.2 平衡二叉树 …… (197)
8.3.3 B 树 …… (200)
8.3.4 B + 树 …… (204)
8.4 散列表的查找技术 …… (205)
8.4.1 概述 …… (205)
8.4.2 散列函数的设计 …… (206)
8.4.3 处理冲突的方法 …… (207)
8.4.4 散列查找的性能分析 …… (211)
8.4.5 开散列表与闭散列表的比较 …… (211)
8.5 各种查找方法的比较 …… (212)
习题8 …… (212)
第9章 排序技术 …… (215)
9.1 概述 …… (216)
9.1.1 排序的基本概念 …… (216)
9.1.2 排序算法的性能 …… (217)
9.2 插入排序 …… (217)
9.2.1 直接插入排序 …… (217)
9.2.2 折半插入排序 …… (219)
9.2.3 希尔排序 …… (220)
9.3 交换排序 …… (222)
9.3.1 起泡排序 …… (222)
9.3.2 快速排序 …… (224)
9.4 选择排序 …… (228)
9.4.1 简单选择排序 …… (228)

9.4.2 堆排序 …… (229)
9.5 归并排序 …… (233)
9.5.1 二路归并排序的递归实现 …… (233)
9.5.2 二路归并排序的非递归实现 …… (235)
9.6 分配排序 …… (237)
9.6.1 桶式排序 …… (237)
9.6.2 基数排序 …… (240)
9.7 各种排序方法的比较 …… (242)
习题9 …… (244)
附录A 词汇索引 …… (247)
附录B 计算机学科硕士研究生入学考试专业基础综合考试大纲(数据结构部分) …… (251)
参考文献 …… (253)

第1章 绪 论

<table>
<tr><td>教学重点</td><td colspan="5">数据结构的基本概念;数据的逻辑结构、存储结构以及二者之间的关系;算法及特性;大 O 记号的表示</td></tr>
<tr><td>教学难点</td><td colspan="5">抽象数据类型的定义和使用;算法的时间复杂度分析</td></tr>
<tr><td rowspan="11">教学内容
和
教学目标</td><td rowspan="2">知识点</td><td colspan="4">教学要求</td></tr>
<tr><td>了解</td><td>理解</td><td>掌握</td><td>熟练掌握</td></tr>
<tr><td>程序设计的一般过程</td><td>√</td><td></td><td></td><td></td></tr>
<tr><td>数据结构在程序设计中的作用</td><td></td><td>√</td><td></td><td></td></tr>
<tr><td>算法在程序设计中的作用</td><td></td><td>√</td><td></td><td></td></tr>
<tr><td>数据结构的基本概念</td><td></td><td></td><td></td><td>√</td></tr>
<tr><td>抽象数据类型</td><td></td><td>√</td><td></td><td></td></tr>
<tr><td>算法及其重要特性</td><td></td><td></td><td></td><td>√</td></tr>
<tr><td>算法的描述方法</td><td></td><td></td><td>√</td><td></td></tr>
<tr><td>算法的时间复杂度</td><td></td><td></td><td>√</td><td></td></tr>
<tr><td>算法的空间复杂度</td><td></td><td>√</td><td></td><td></td></tr>
<tr><td>教学提示</td><td colspan="5">对本章的教学要抓住两条主线,一条主线是数据结构,包括数据结构的研究对象及相关概念;另一条主线是算法,包括算法的概念、描述方法以及时间复杂度的分析。
对于数据结构部分,从问题求解的一般过程入手,理解“数据结构 + 算法 = 程序”,注意强调数据结构与算法和程序之间的关系。数据结构的核心概念是数据元素,注意通过具体实例引申数据元素之间的关系,并抓住两个方面:逻辑结构和存储结构,强调二者之间的关系。
对于算法部分,以算法的概念和重要特性为基本出发点,并在以后的教学中注意应用算法的特性,不要孤立地讲授概念。对于算法的时间性能,强调三个要点:基本语句、执行次数、数量级。
最后,强调数据结构和算法分析都是针对大数据量,即大数据量的组织、大数据量的处理效率。</td></tr>
</table>

1.1 问题求解与程序设计

用计算机求解任何问题都离不开程序设计，只有最终在计算机上能够运行良好的程序才能解决特定的实际问题，因此，程序设计的过程就是利用计算机求解问题的过程。

1.1.1 程序设计的一般过程

用计算机求解任何问题都离不开程序设计，但是计算机不能分析问题并产生问题的解决方案，必须由人（即程序设计者）分析问题，确定问题的解决方案，采用计算机能够理解的指令描述这个问题的求解步骤（即编写程序），然后让计算机执行程序最终获得问题的解。程序设计的一般过程如图 1-1 所示。

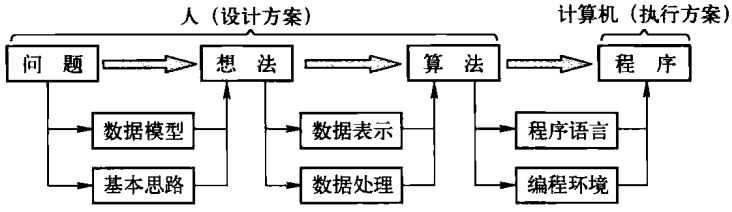


图 1-1 程序设计的一般过程

由问题到想法需要分析问题，抽象出具体的数据模型（待处理的数据以及数据之间的关系，也称数据结构），形成问题求解的基本思路。对于待求解的问题，首先搞清楚求解的目标是什么，给出了哪些已知信息、显式条件或隐含条件，应该用什么形式的数据表达计算结果。如果没有全面、准确和认真地分析问题，就急急忙忙地编写程序，结果往往是事倍功半，造成不必要的反复，甚至给程序留下严重隐患。

算法用来描述问题的解决方案，是具体的、机械的操作步骤。利用计算机解决问题的最重要一步是将人的想法描述成算法，即设计算法，也就是从计算机的角度设想计算机是如何一步一步完成这个任务的。有些问题很简单，很容易就可以得到问题的解决方案；如果问题比较复杂，就需要更多的思考才能得到问题的解决方案。由想法到算法需要完成数据表示和数据处理，即描述问题的数据模型（将数据从机外表示转换为机内表示），描述问题求解的基本思路（将问题的解决方案形成算法）。数据结构课程主要讨论非数值问题的数据表示和数据处理。

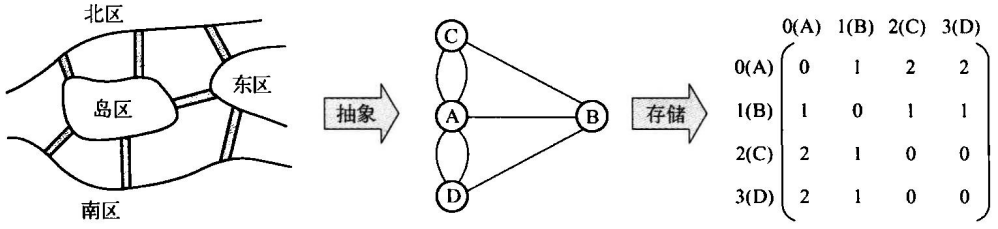
由算法到程序需要将算法的操作步骤转换为某种程序设计语言对应的语句，转换所依据的规则就是某种程序设计语言的语法，换言之，就是在某种编程环境下用程序设计语言描述要处理的数据以及数据处理的过程。

著名的计算机学者沃思[①]给出了一个公式：数据结构 + 算法 = 程序。从这个公式可以看到，数据结构和算法是构成程序的两个重要的组成部分，一个“好”程序首先是将问题抽象出一个适当的数据结构，然后基于该数据结构设计一个“好”算法。下面以著名的哥尼斯堡七桥问题为例，说明程序设计的一般过程。

---

① 沃思（Niklaus. Wirth）1934 年生于瑞士。1968 年设计并实现了 PASCAL 语言（被誉为 PASCAL 之父），1971 年提出了结构化程序设计，1976 年设计并实现了 Modula 2 语言。除了程序设计语言之外，沃思在其他方面也有许多创造，如扩充了著名的巴科斯范式，发明了语法图等。1984 年获图灵奖。

【问题】 哥尼斯堡七桥问题(以下简称七桥问题)。17 世纪的东普鲁士有一座哥尼斯堡城(现在叫加里宁格勒,在波罗的海南岸),城中有一座岛,普雷格尔河的两条支流环绕其旁,并将整个城市分成北区、东区、南区和岛区 4 个区域,全城共有七座桥将 4 个城区连接起来,如图 1-2(a)所示。于是,产生了一个有趣的问题:一个人是否能在一次步行中经过全部的七座桥后再回到起点,且每座桥只经过一次。



(a) 七桥问题　　(b) 七桥问题的数据模型　　(c) 对应的二维数组mat[4][4]

图 1-2　七桥问题的数据抽象过程

【想法】 将城区抽象为顶点,用 A、B、C、D 表示 4 个城区,将桥抽象为边,用 7 条边表示七座桥,抽象出七桥问题的数据模型如图 1-2(b)所示,从而将七桥问题抽象为一个数学问题:求经过图中每条边一次且仅一次的回路,后来人们称之为欧拉回路。欧拉回路的判定规则是:

(1) 如果通奇数个桥的城区多于两个,则不存在欧拉回路;

(2) 如果只有两个城区通奇数个桥,则可以从这两个城区之一出发找到欧拉回路;

(3) 如果没有一个城区通奇数个桥,则无论从哪里出发都能找到欧拉回路。

由上述判定规则得到求解七桥问题的基本思路:依次计算与每个顶点相关联的边的个数(称为顶点的度),根据度为奇数的顶点个数判定是否存在欧拉回路。

【算法】 进一步地,将顶点 A、B、C、D 编号为 0、1、2、3,用二维数组 mat[4][4]存储七桥问题的数据模型,如果顶点 $i(0 \leqslant i \leqslant 3)$ 和顶点 $j(0 \leqslant i \leqslant 3)$ 之间有 k 条边,则元素 mat[i][j]的值为 k,如图 1-2(c)所示。求解七桥问题的关键是求与每个顶点相关联的边数,即在二维数组 mat[4][4]中求每一行元素之和,算法描述如下:

算法:EulerCircuit
输入:二维数组 mat[n][n]
输出:度为奇数的顶点个数 count

1. count 初始化为 0;
2. 下标 i 从 0 ~ n - 1 重复执行下述操作:
   2.1　计算第 i 行元素之和 degree;
   2.2　如果 degree 为奇数,则 count ++;
3. 返回 count;

【程序】 主函数首先初始化二维数组 mat[4][4],即建立七桥问题对应的数据模型,然后调用函数 EulerCircuit 计算图模型中通奇数个桥的顶点个数,再根据欧拉规则判定是否存在欧拉回路。程序如下:

```
#include < stdio. h >                          //使用库函数 printf 和 scanf
int EulerCircuit( int mat[10][10],int n);      //函数声明
```

```
                                        //空行,以下是主函数
int main()
{
    int mat[10][10] = {{0,1,2,2},{1,0,1,1},{2,1,0,0},{2,1,0,0}};
    int num = EulerCircuit(mat,4);      //调用函数得到通奇数个桥的顶点个数
    if (num > 2)                        //多于两个顶点通奇数个桥
      printf("有%d 个地方通奇数个桥,不存在欧拉回路\n ",num);
    else if (num == 2 || num == 0)      //两个顶点通奇数个桥或没有顶点通奇数个桥
        printf("存在欧拉回路\n");
    return 0;                           //将 0 返回操作系统,表明程序正常结束
}
                                        //空行,以下是其他函数定义
int EulerCircuit(int mat[10][10],int n) //函数定义,二维数组作为形参
{
    int i,j,degree,count = 0;           //count 累计通奇数个桥的顶点个数
    for (i = 0; i < n; i ++)            //依次累加每一行的元素
    {
      degree = 0;                       // degree 存储通过顶点 i 的桥数,初始化为 0
      for (j = 0; j < n; j ++)          //依次处理每一列的元素
      {
        degree = degree + mat[i][j];    //将通过顶点 i 的桥数求和
      }
      if (degree % 2! = 0)              //桥数为奇数
        count ++;
    }
    return count;                       //结束函数,并将 count 返回到调用处
}
```

1.1.2 数据结构在程序设计中的作用

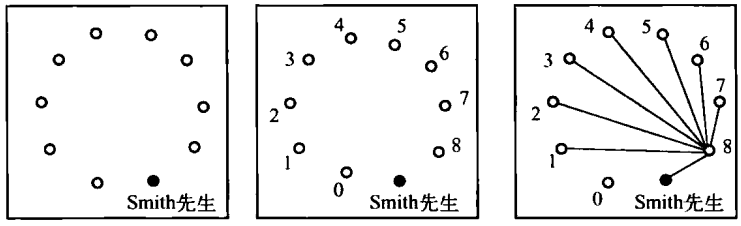
有些问题难以求解的原因是无从下手,如果能将问题抽象出一个合适的数据模型,则问题可能会变得豁然开朗。下面这个著名的握手问题是爱丁堡大学的 Peter Ross 提出来的。

【例 1-1】 握手问题。Smith 先生和太太邀请 4 对夫妻来参加晚宴。每个人来的时候,房间里的一些人都要和其他人握手。当然,每个人都不会和自己的配偶握手,也不会跟同一个人握手两次。之后,Smith 先生问每个人和别人握了几次手,他们的答案都不一样。问题是,Smith 太太和别人握了几次手?

解:这个问题具有挑战性的原因是因为它没有一个明显的起始点,但如果将此问题抽象出一个合适的数据模型,如图 1-3(a)所示,问题就变得简单了。

让我们来收集一下已知条件。Smith 先生问了房间中的 9 个人,每个人的答案都不相同,因此,每个答案都在 0 和 8 之间,相应地修改模型如图 1-3(b)所示。

让我们来列出所有的握手信息。例如,8 号除了他(她)自己和配偶,与房间里的其他人总共握了 8 次手。基于这个观察,我们可以画出 8 号的握手信息并且知道 8 号的配偶是 0 号;7



(a)原始模型　(b)握手次数　(c)配对

图 1-3　握手问题的数据模型

号除了 0 号和 1 号,与房间里的其他人总共握了 7 次手,因此 7 号的配偶是 1 号;……问题是否变得豁然开朗了?

一个"好"程序首先是将问题抽象出一个适当的数据模型,基于不同数据模型的算法,其运行效率可能会有很大差别。举例如下。

【例 1-2】 手机电话号码查询问题。假设某手机中存储了如表 1-1 所示的若干电话号码,如何在手机中查找某人的电话号码?

解:如果将电话号码集合线性排列,则某个人的电话号码只能进行顺序查找。

表 1-1　某手机中的电话号码

姓名	王靓靓	赵　刚	韩春颖	李琦勇	…	张　强
电话	13833278900	13944178123	15594434552	13833212779	…	13331688900

如果将电话号码集合进行分组,如图 1-4 所示,则查找某个人的电话号码可以只在某个分组中进行。显然,后者的查找效率更高,当数据量较大时差别就更大。

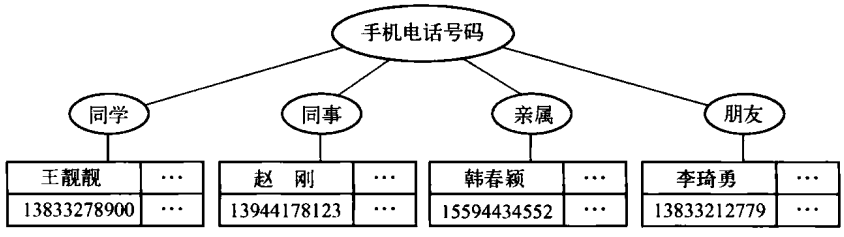


图 1-4　分组——将电话号码集合组织为树结构

1.1.3　算法在程序设计中的作用

算法是问题的解决方案,这个解决方案本身并不是问题的答案,而是能获得答案的指令序列。对于许多实际的问题,写出一个正确的算法还不够,如果这个算法在规模较大的数据集上运行,那么运行效率就成为一个重要的问题。在选择和设计算法时要有效率的观念,这一点比提高计算机本身的速度更为重要。

【例 1-3】 数组循环左移问题。将一个具有 n 个元素的数组向左循环移动 i 个位置。有许多应用程序会调用这个问题的算法,例如在文本编辑器中移动行的操作,磁盘整理时交换两个不同大小的相邻内存块等。所以,解决这个问题的算法要求有较高的时间性能和空间性能。

解法 1:先将数组中的前 i 个元素存放在一个临时数组中,再将余下的 $n-i$ 个元素左移 i

个位置，最后将前 i 个元素从临时数组复制回原数组中后面的 i 个位置，这个算法总共需要移动 $i+(n-i)+i=i+n$ 次数组元素，但使用了 i 个额外的存储单元。

解法 2：先设计一个函数将数组向左循环移动 1 个位置，然后再调用该算法 i 次，这个算法只使用了 1 个额外的存储单元，但总共需要移动 $i \times n$ 次数组元素。

解法 3：现在我们换一个角度看这个问题，将这个问题看作是把数组 AB 转换成数组 BA（A 代表数组的前 i 个元素，B 代表数组中余下的 $n-i$ 个元素），先将 A 置逆得到 $A^{-1}B$，再将 B 置逆得到 $A^{-1}B^{-1}$，最后将整个 $A^{-1}B^{-1}$ 置逆得到 $(A^{-1}B^{-1})^{-1}=BA$。设 Reverse 函数执行将数组元素置逆的操作，对 abcdefgh 向左循环移动 3 个位置的过程如下：

```
Reverse(0,i-1);          //得到 cbadefgh
Reverse(i,n-1);          //得到 cbahgfed
Reverse(0,n-1);          //得到 defghabc
```

其原理可以用一个简单的游戏来理解：将两手的掌心对着自己，左手在右手上面，将一个具有 10 个元素的数组向左循环移动 5 位的操作过程如图 1-5 所示。

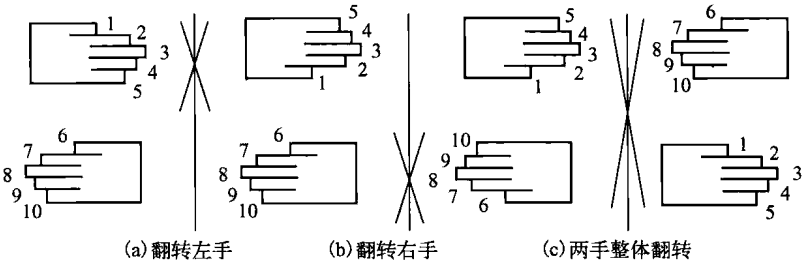


图 1-5　利用数组逆置进行循环移位的示意图

这个算法总共需要交换 $i+(n-i)+n=2n$ 次数组元素，只使用了 1 个用来交换的临时单元，并且算法简单、易懂，想出错都很难①。Brian Kernighan 在 *Software Tools in Pascal* 中使用了这个算法在文本编辑器中移动各行。

1.1.4　本书讨论的主要内容

计算机能够求解的问题一般可以分为数值问题和非数值问题。数值问题抽象出的数据模型通常是数学方程，非数值问题抽象出的数据模型通常是线性表、树、图等数据结构。下面请看几个例子。

【例 1-4】　为百元买百鸡问题抽象数据模型。已知公鸡 5 元一只，母鸡 3 元一只，小鸡 1 元三只，花 100 元钱买 100 只鸡，问公鸡、母鸡、小鸡各多少只？

解：设 x、y 和 z 分别表示公鸡、母鸡和小鸡的个数，则有如下方程组成立：

$$\begin{cases} x+y+z=100 \\ 5x+3x+z/3=100 \end{cases} \quad \text{且} \quad \begin{cases} 0\leqslant x\leqslant 20 \\ 0\leqslant y\leqslant 33 \\ 0\leqslant z\leqslant 100 \end{cases}$$

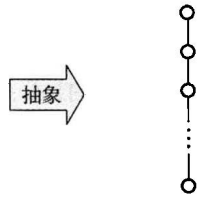
① 算法领域有一个启发式规则：不要拘泥于头脑中出现的第一个算法。一个好的算法是反复努力和重新修正的结果，即使足够幸运地得到了一个貌似完美的算法思想，我们也应该尝试着改进它。

【例 1-5】 为学籍管理问题抽象数据模型。

解:用计算机来完成学籍管理,就是由计算机程序处理学生学籍登记表,实现增、删、改、查等功能。图 1-6(a)所示为一张简单的学生学籍登记表。在学籍管理问题中,计算机的操作对象是每个学生的学籍信息——表项,各表项之间的关系可以用线性结构来描述,如图 1-6(b)所示。

学　号	姓　名	性　别	出生日期	政治面貌
0001	陆　宇	男	1986/09/02	团员
0002	李　明	男	1985/12/25	党员
0003	汤晓影	女	1986/03/26	团员
⋮	⋮	⋮	⋮	⋮

(a)学生学籍登记表

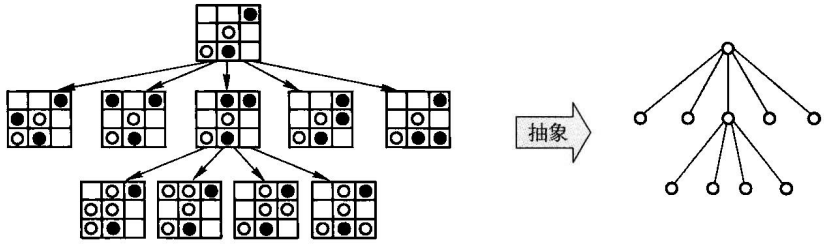


(b)线性结构

图 1-6　学籍登记表及其数据模型

【例 1-6】 为人机对弈问题抽象数据模型。

解:在对弈问题中,计算机的操作对象是对弈过程中可能出现的棋盘状态——格局,而格局之间的关系是由对弈规则决定的。因为从一个格局可以派生出多个格局,所以,这种关系通常不是线性的。例如,从三子连珠游戏的某格局出发可以派生出五个新的格局,从新的格局出发,还可以再派生出新的格局,如图 1-7(a)所示;格局之间的关系可以用树结构来描述,如图 1-7(b)所示。

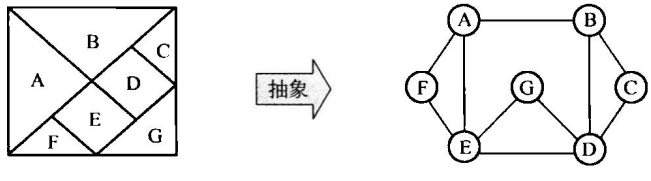


(a)对弈树的局部　　(b)树结构

图 1-7　对弈树及其数据模型

【例 1-7】 为七巧板涂色问题抽象数据模型。

解:假设有如图 1-8(a)所示的七巧板,使用至多 4 种不同颜色对七巧板涂色,要求每个区域涂一种颜色,相邻区域的颜色互不相同。为了识别不同区域的相邻关系,可以将七巧板的每个区域看成一个顶点,如果两个区域相邻,则这两个顶点之间有边相连,将七巧板抽象为图结构,如图 1-8(b)所示。



(a)七巧板　　(b)图结构

图 1-8　七巧板及其数据模型

本书讨论非数值问题的数据组织和处理，主要内容如下：

（1）数据的逻辑结构：线性表、树、图等数据结构，其核心是如何组织待处理的数据以及数据之间的关系；

（2）数据的存储结构：如何将线性表、树、图等数据结构存储到计算机的存储器中，其核心是如何有效地存储数据以及数据之间的逻辑关系；

（3）算法：如何基于数据的某种存储结构实现插入、删除、查找等基本操作，其核心是如何有效地处理数据；

（4）常用的数据处理技术：包括查找技术和排序技术等；

（5）基本的算法设计技术：包括蛮力法、分治法、减治法、贪心法和动态规划法等，并从算法设计技术的角度讨论线性表、树和图等数据结构的基本操作，以及查找算法和排序算法的设计思想和设计过程。

1.2 数据结构的基本概念

1.2.1 数据结构

1. 数据

数据（data）是信息的载体，在计算机科学中是指所有能输入到计算机中并能被计算机程序识别和处理的符号集合。可以将数据分为两大类：一类是整数、实数等数值数据；另一类是文字、声音、图形和图像[①]等非数值数据。

数据是计算机程序处理的“原料”，例如，编译程序处理的数据是源程序；学籍管理程序处理的数据是学籍登记表。

2. 数据元素

数据元素（data element）是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。构成数据元素的不可分割的最小单位称为**数据项**（data item）。例如，对于学生学籍登记表，每个学生的档案就是一个数据元素，而档案中的学号、姓名、出生日期等是数据项，如图 1-9 所示。

学　号	姓　名	性　别	出生日期	政治面貌
0001	陆　宇	男	1986/09/02	团员
0002	李　明	男	1985/12/25	党员
0003	汤晓影	女	1986/03/26	团员

数据项　　数据元素

图 1-9　数据元素和数据项

数据元素具有广泛的含义，一般来说，能独立、完整地描述问题世界的一切实体都是数据

① 在计算机中，图形和图像是两个不同的概念。图形一般是指通过绘图软件绘制的，由直线、圆、弧等基本曲线组成的画面，即图形是由计算机产生的；图像是由扫描仪、数码相机等输入设备捕捉的画面，即图像是输入计算机的真实场景或图片。