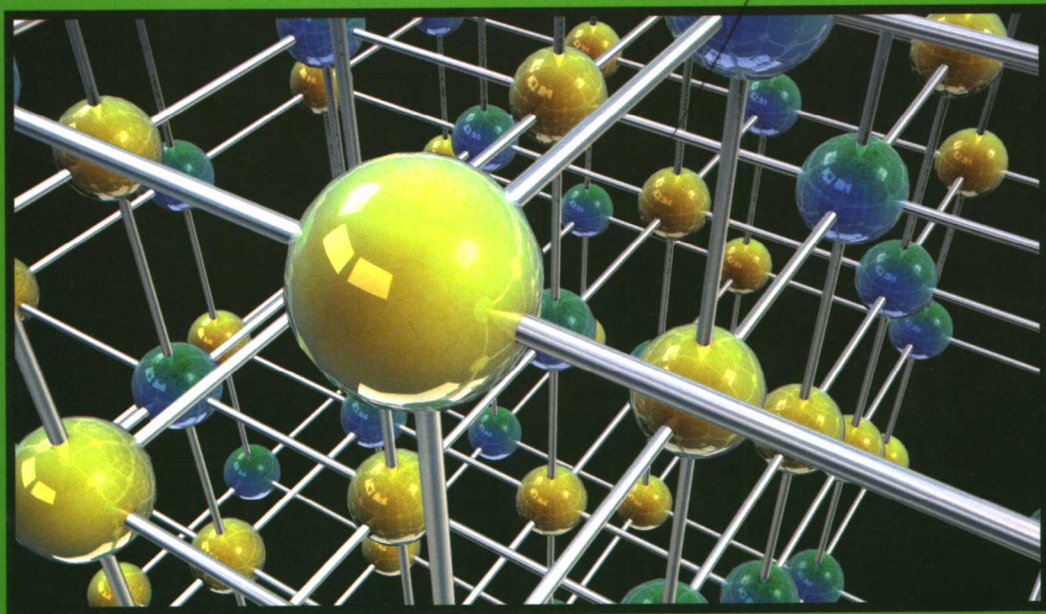


数据结构

夏克俭 王绍斌 编著



國防工業出版社

National Defense Industry Press

数 据 结 构

夏克俭 王绍斌 编著

国防工业出版社

· 北京 ·

内 容 简 介

本书系统地介绍了各种类型的数据结构、数据结构在计算机存储器中的表示以及相关的 C 语言描述算法。另外,对各类数据结构在实际中的应用作了较详细的介绍,包含例题分析及完整的 C 语言源程序。书后附有综合性习题,便于读者进一步理解和巩固所学的知识。

本书可作为大专院校计算机专业或相关专业的教材,也可供从事计算机软件开发的工程技术人员参考。

图书在版编目(CIP)数据

数据结构/夏克俭,王绍斌编著.—北京:国防工业出版社,2007.2

ISBN 978-7-118-04936-7

I. 数... II. ①夏...②王... III. 数据结构
IV. TP311.12

中国版本图书馆 CIP 数据核字(2006)第 157765 号

※

国防工业出版社 出版发行

(北京市海淀区紫竹院南路 23 号 邮政编码 100044)

天利华印刷装订有限公司印刷

新华书店经售

*

开本 787×1092 1/16 印张 20 字数 459 千字

2007 年 2 月第 1 版第 1 次印刷 印数 1—4000 册 定价 30.00 元

(本书如有印装错误,我社负责调换)

国防书店:(010)68428422

发行邮购:(010)68414474

发行传真:(010)68411535

发行业务:(010)68472764

前 言

数据结构(Data Structure)知识是计算机科学的基础,计算机学科的许多领域都构建在这个基础之上。想要更好地从事计算机软件设计、实现、测试和维护等工作,掌握数据结构的知识是非常必要的。

数据结构研究的是计算机所处理数据元素间的关系及其操作实现的算法。数据结构在计算机科学中是一门综合性的专业基础课,不仅涉及到计算机硬件(特别是编码理论、存储机制和存取方法等)的研究范围,而且和计算机软件的研究有着更密切的关系。本书在吸收现有同类教材优点的基础上,精选应用实例和习题,通过理论结合实际,加强学生解题能力和技巧的训练,以便学会分析研究数据对象的特性,学会数据的组织方法,从而在程序设计中选择合适的数据结构以及相应的算法,提高程序设计水平。

C语言是一种非常优秀的编程语言,包括操作系统在内的很多软件的设计与实现都采用C语言,所以我们选用C语言来介绍数据结构的算法。

全书包含以下内容:

- 第一章介绍数据结构的基本概念,包括数据结构研究的内容和方法、数据结构的含义、抽象数据类型的表示与实现、算法和算法分析等基本知识。
- 第二章讨论线性结构相关知识及算法描述,包括线性表的定义及其基本操作、线性表的顺序存储结构、链式存储结构以及线性表的应用举例。
- 第三章讨论软件设计中两个重要的技术——栈和队列,包括栈和队列的定义及其基本操作、栈和队列的顺序存储结构、链式存储结构以及栈与队列的应用举例。
- 第四章讨论对字符串的处理,包括串的定义及其基本操作、串的存储结构、串的模式匹配等以及串的应用举例。
- 第五章的数组和广义表是线性结构的扩充,包括数组的定义及其基本操作、数组的存储结构和矩阵的压缩存储方法、广义表的定义及其基本操作、广义表的存储结构及相关算法。
- 第六章讨论层次结构——树,包括树和二叉树的定义、基本操作、性质、存储结构,二叉树遍历算法,线索二叉树,树和森林以及二叉树应用举例——Huffman编码及译码。
- 第七章讨论网状结构——图,包括图的定义和操作、存储结构、搜索算法以及图的一些应用举例,如最小生成树、最短路径、拓扑排序和关键路径问题等等。
- 第八章讨论建立在数据结构上的一个重要操作——查找,包括顺序查找、折半

查找和分块查找的算法及分析,二叉排序树和平衡二叉树,B树和 Hash 表的查找算法及分析。

- 第九章讨论建立在数据结构上的另一个重要操作——排序,包括插入排序、交换排序、选择排序、归并排序和基数排序等内排序的算法及分析;另外,介绍了外排序的一些基本方法。

- 第十章讨论文件的基本概念和相关内容,包括信息结构、顺序文件、散列文件和索引文件等。

每章后附有一些不同难度的练习题。

本书是夏克俭教授和王绍斌博士在多年的数据结构课程教学基础上编写而成的,全书由涂序彦教授审稿。本书在编写过程中得到了北京科技大学计算机科学与技术系研究生和本科生的支持和帮助,在此深表谢意。书中不足之处,恳请广大读者批评指正。

编著者

2006 年 11 月于北京

目 录

第一章 绪论	1
1.1 数据结构研究的内容和方法	1
1.1.1 数据结构的含义	1
1.1.2 数据结构研究的内容	2
1.1.3 研究数据结构的方法	6
1.2 抽象数据类型的表示与实现	7
1.3 学习数据结构的重要性	8
1.3.1 数据结构的发展简史及在计算机科学中的地位	8
1.3.2 学习数据结构的目的是	8
1.4 算法和算法分析	9
1.4.1 算法的定义	9
1.4.2 算法的性质	9
1.4.3 算法的设计目标	10
1.4.4 算法效率的度量	11
习题一	14
第二章 线性表	16
2.1 线性表的定义及其基本操作	16
2.1.1 线性表的定义	16
2.1.2 线性表的逻辑结构和特征	16
2.1.3 线性表的抽象数据类型表示	17
2.2 线性表的顺序存储结构	19
2.2.1 顺序表	20
2.2.2 顺序表上的基本操作	21
2.2.3 顺序存储结构的基本特点	23
2.3 线性表的链式存储结构	24
2.3.1 单链表	24
2.3.2 单链表中的基本操作	26
2.3.3 单向循环链表	31
2.3.4 双向链表	32
2.3.5 静态链表	34
2.3.6 链式存储结构的特点	37

2.4 线性表应用举例.....	38
2.4.1 Josephu 问题.....	38
2.4.2 一元多项式的表示与相加	41
习题二	45
第三章 栈与队列	46
3.1 栈.....	46
3.1.1 栈的定义及其操作	46
3.1.2 栈的顺序存储结构	48
3.1.3 栈的链式存储结构	50
3.2 栈应用举例.....	51
3.2.1 数制转换	51
3.2.2 行编辑处理	53
3.2.3 表达式求值	56
3.3 栈与递归函数.....	62
3.3.1 递归定义与递归函数	62
3.3.2 递归函数到非递归函数的转化	64
3.4 队列.....	66
3.4.1 队列的定义及其操作	66
3.4.2 队列的顺序存储结构	68
3.4.3 队列的链式存储结构	70
3.5 队列应用举例.....	72
3.5.1 迷宫问题	72
3.5.2 离散事件模拟	75
3.5.3 有序事件模拟	80
习题三	83
第四章 串	84
4.1 串的定义及其操作.....	84
4.1.1 串的定义	84
4.1.2 串的抽象数据类型	84
4.2 串的存储结构.....	87
4.2.1 串的静态存储结构	87
4.2.2 串的动态存储结构	92
4.3 串模式匹配.....	98
4.4 串应用举例	101
4.4.1 文本编辑	101
4.4.2 建立词索引表	104
习题四.....	105

第五章 数组和广义表	107
5.1 数组的定义及其操作	107
5.1.1 数组的定义	107
5.1.2 数组的抽象数据类型	108
5.2 数组的存储结构	109
5.2.1 数组的静态存储方式	109
5.2.2 数组的动态存储方式	111
5.3 矩阵的压缩存储	113
5.3.1 特殊矩阵的压缩存储	113
5.3.2 稀疏矩阵的压缩存储	115
5.4 广义表的定义及其操作	120
5.4.1 广义表的定义	120
5.4.2 广义表的抽象数据类型	120
5.5 广义表的存储结构	122
5.5.1 广义表的链式存储	122
5.5.2 广义表基本操作的递归算法	124
习题五	126
第六章 树	127
6.1 树的基本概念	127
6.1.1 树的定义及基本操作	128
6.1.2 树的性质	132
6.2 二叉树	133
6.2.1 二叉树的定义及基本操作	133
6.2.2 二叉树的性质	136
6.2.3 二叉树的存储结构	138
6.3 二叉树的遍历	142
6.3.1 二叉树的遍历算法	142
6.3.2 二叉树遍历算法的非递归形式	145
6.3.3 二叉树遍历的应用	149
6.4 线索二叉树	152
6.4.1 线索二叉树的概念	152
6.4.2 建立线索二叉树	153
6.4.3 线索二叉树的遍历	155
6.4.4 线索二叉树的更新	155
6.5 树和森林	157
6.5.1 树的存储结构	157
6.5.2 森林和二叉树的转换	160

6.5.3 树和森林的遍历·····	162
6.6 二叉树应用举例·····	163
6.6.1 Huffman 树及其构造算法·····	163
6.6.2 Huffman 编码及译码·····	167
习题六·····	170
第七章 图 ·····	172
7.1 图的定义及操作·····	172
7.1.1 图的定义·····	172
7.1.2 图的抽象数据类型·····	177
7.2 图的存储结构·····	178
7.2.1 数组表示法·····	178
7.2.2 邻接表表示法·····	181
7.2.3 十字链表表示法·····	183
7.2.4 邻接多重表表示法·····	185
7.3 图的遍历·····	186
7.3.1 深度优先搜索算法·····	186
7.3.2 广度优先搜索算法·····	188
7.3.3 求连通分量的算法·····	189
7.4 最小生成树·····	190
7.4.1 Prim 算法·····	191
7.4.2 Kruskal 算法·····	194
7.5 最短路径问题·····	195
7.5.1 Dijkstra 算法·····	196
7.5.2 Floyd 算法·····	200
7.6 有向无环图的应用·····	203
7.6.1 拓扑排序·····	203
7.6.2 关键路径·····	208
习题七·····	212
第八章 查找 ·····	214
8.1 查找概述·····	214
8.2 顺序表的静态查找·····	215
8.2.1 顺序查找算法·····	215
8.2.2 折半查找算法·····	216
8.2.3 分块查找算法·····	219
8.3 树表的动态查找·····	221
8.3.1 二叉排序树和平衡二叉排序树·····	221
8.3.2 B-树·····	235

8.3.3 B+ 树和 B* 树	243
8.4 Hash 表的查找	244
8.4.1 Hash 表的含义	244
8.4.2 Hash 函数的构造方法	246
8.4.3 处理冲突的方法	248
8.4.4 Hash 表的查找及分析	250
习题八	253
第九章 排序	254
9.1 排序概述	254
9.2 插入排序	256
9.2.1 直接插入排序	256
9.2.2 折半插入排序	258
9.2.3 链表插入排序	259
9.2.4 Shell 排序	261
9.3 交换排序	263
9.3.1 起泡排序	263
9.3.2 快速排序	264
9.4 选择排序	267
9.4.1 直接选择排序	267
9.4.2 堆选择排序	269
9.5 归并排序	274
9.6 基数排序	277
9.7 各种内排序算法的比较讨论	281
9.8 外排序概述	281
习题九	283
第十章 文件	285
10.1 信息与数据的基本概念	285
10.1.1 信息与数据	285
10.1.2 信息结构	285
10.1.3 信息结构的层次	287
10.2 文件概述	288
10.2.1 文件系统的演变过程及基本概念	288
10.2.2 文件的存储介质	290
10.2.3 文件的结构	291
10.2.4 文件的基本操作	292
10.3 顺序文件	294
10.4 散列文件	297

10.5 索引文件.....	301
10.5.1 索引技术概述	301
10.5.2 ISAM 文件	302
10.5.3 VSAM 文件	304
10.6 索引链接文件.....	305
10.6.1 多重链索引文件	305
10.6.2 不定长记录的索引链接文件	307
10.7 倒排文件.....	308
习题十.....	309
参考文献.....	310

第一章 绪 论

目前,计算机业在飞速发展,其应用领域也早不限于科学计算,而是广泛深入到社会的各个部门。与此相对应,计算机加工处理的对象由纯粹的数值发展到字符、表格、图像等各种具有一定结构的数据。在计算机中如何组织数据,如何处理数据,从而更好地利用数据,是计算机科学的基本内容。为了编写出一个“好”的程序,必须分析处理对象的特性以及对象之间存在的关系。这就是数据结构这门学科形成和发展的背景。

本章主要讨论数据结构、算法及算法分析等方面的一些基本概念。

1.1 数据结构研究的内容和方法

1.1.1 数据结构的含义

数据结构(Data Structure, DS),研究的是计算机所处理数据元素间的关系及其操作实现的算法,包括数据的逻辑结构、数据的存储结构以及数据的操作。

数据结构在计算机科学中是一门综合性的专业基础课。不仅涉及到计算机硬件(特别是编码理论、存储机制和存取方法等)的研究范围,而且和计算机软件的研究有着更密切的关系,无论是编译程序还是操作系统,都涉及到数据元素在存储器中的分配问题。在研究信息检索时也必须考虑如何组织数据,以便查找和存取数据元素更为方便。因此可以认为,数据结构是介于数学、计算机硬件和计算机软件三者之间的一门核心课程(见图1.1)。在计算机科学中数据结构不仅是一般程序设计(特别是非数值计算的程序设计)的基础,而且是设计和实现编译系统、操作系统、数据库系统及其他系统程序和大型应用程序的重要基础。

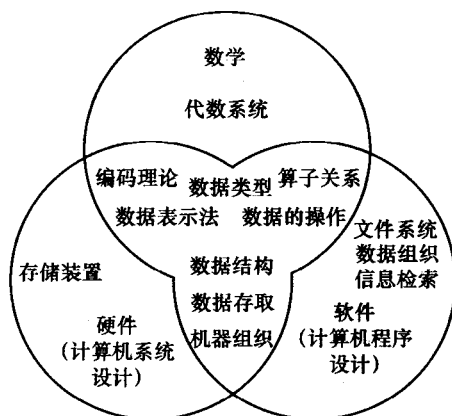


图 1.1 “数据结构”所处的地位

1.1.2 数据结构研究的内容

1. 数据的逻辑结构（Logical Structure）

简单地说，数据的逻辑结构是指数据元素之间的逻辑关系。

这里所说的数据（Data），是对客观事物的符号表示，是指输入到计算机并能被计算机程序处理的符号的总称。例如方程中的整数、实数，源程序中的字符串，以及文字、图像和声音信号等，都可作为计算机中的数据。

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处理。简单的数据元素可以是整数、字符等形式。一般，一个数据元素由若干个数据项（Data Item）组成，常称为记录（Record）。例如表 1.1 所列的图书管理表，其中： a_0 ， a_1 ， a_2 ， $\dots$ ， a_{n-1} 对应的每一行各是一个数据元素；而编号、书名等就是数据项。在计算机存取数据时，数据项是不可分割的最小存取单位。

表 1.1 图书管理表

	编号	书 名	作者	出版商	出版日期/年	...
a_0	0001	C 程序设计及应用	李益林	高等教育出版社	1998	...
a_1	0002	数据库原理与技术	周志逵	科学出版社	1998	...
a_2	0003	智能管理	涂序彦	清华大学出版社	1995	...
$\vdots$	...	...	...	...	...	...
a_{n-1}	...	...	...	...	...	...

具有相同性质的数据元素组成的集合，称为数据对象（Data Object），它是数据的一个子集。

计算机要处理的数据元素不是相互孤立的，而是有着各种各样联系的，这种数据元素之间的关系就称做逻辑结构。人们经过长期的实践和总结，根据数据元素之间关系的不同特性，归纳出 4 类基本的逻辑结构：

- (1) 集合：结构中的数据元素除了“属于同一集合”的关系外，没有其他关系（如例 1.1）。
- (2) 线性结构：结构中的数据元素存在一对一的关系（如例 1.2）。
- (3) 层次结构：结构中的数据元素之间存在一对多的关系（如例 1.3）。
- (4) 网状结构：结构中的数据元素之间存在若干个多对多关系（如例 1.4）。

例 1.1 学校举办了英语演讲比赛和歌曲演唱比赛，参赛者可以获得优胜奖和参与奖。问多少学生在两个竞赛中都获得了优胜奖？求解这个问题可以将两个竞赛中获得优秀奖的人分别构成一个整体，问题便成了求两个整体公共部分的问题。由于每个整体之中的人在这个问题中的地位是没有差别的，于是可以抽象成为两个集合，问题也就成了求两个集合的交集。

例 1.2 到医院看病的病人需要排队，而医院实行先来先服务的原则。每个病人都有自己的病历，病历上有病历的编号还有病人的其他具体信息。这些病人的信息构成一张表，如表 1.2 所列。

表 1.2 病人信息表

编号	姓名	性别	年龄/岁	...
172	张立	男	30	...
091	田方	男	52	...
007	陈丽	女	27	...
156	王军	男	9	...
...	...	...	...	...

这张表中的元素存在一个顺序关系，即谁在谁前、谁在谁后的信息。所以，可以用线性结构来刻画这种关系。本例中表示将要接受诊断的病人顺序依次为张立、田方、陈丽、王军……

例 1.3 大学系级行政机构如图 1.2 所示。其中系、办公室……教师、学生可视为数据元素；元素之间呈现的是一种层次关系，即系级下层机构为办公室、教研室和班级，而办公室、教研室和班级等单位又由若干个管理人员、教师、实验员和学生组成。

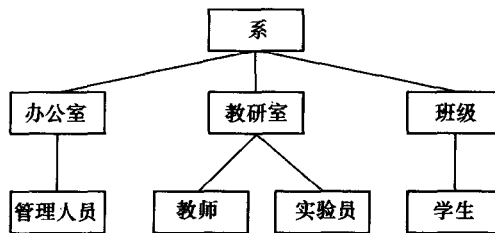


图 1.2 大学系级行政机构

例 1.4 田径比赛的时间安排问题。

设田径比赛项目有：A(跳高)，B(跳远)，C(标枪)，D(铅球)，E(100m 跑)，F(200m 跑)。参赛选手的项目表（每人限参加 3 项）如表 1.3 所列。

问如何安排比赛时间，才能使得：

- (1) 每个比赛项目都能顺利进行（无冲突）；
- (2) 尽可能缩短比赛时间。

此问题可归纳为图的“染色”问题：设项目 A~F 各表示一数据元素，以○表示。若两个项目不能同时举行，则将其连线（如项目 A 和 B 不能同时举行，否则丁一无法参赛）。由此得到如图 1.3 所示的结构。

表 1.3 选手项目表

姓名	项目 1	项目 2	项目 3
丁一	A	B	E
马二	C	D	
张三	C	E	F
李四	D	F	A
王五	B	F	

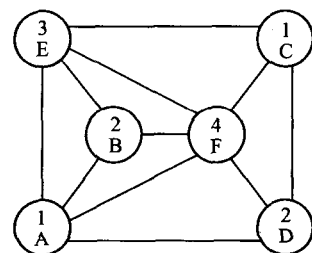


图 1.3 项目对时间安排的约束

图 1.3 为解决此问题的数学模型或数据结构的模型，数据元素之间呈现的是一种网状关系，表示项目相对时间安排的一种约束。下面要解决的问题是对此图着色。设一种颜色表示一个比赛时间片。显然，同一直线上的两个元素不能同色（否则出现冲突）。若用 1, 2, 3, 4 表示 4 种颜色，一种着色方法如图 1.3 所示，即：时间片 1 内比赛 A ,C 项目，时间片 2 内比赛 B, D 项目，时间片 3 内比赛 E 项目，时间片 4 内比赛 F 项目。当然，这种着色方法是否最优还待进一步研究。

上面的例子中所描述的逻辑结构都是从具体问题中抽象出来的数据模型，是独立于计算机存储器的，其基本逻辑关系如图 1.4 所示。

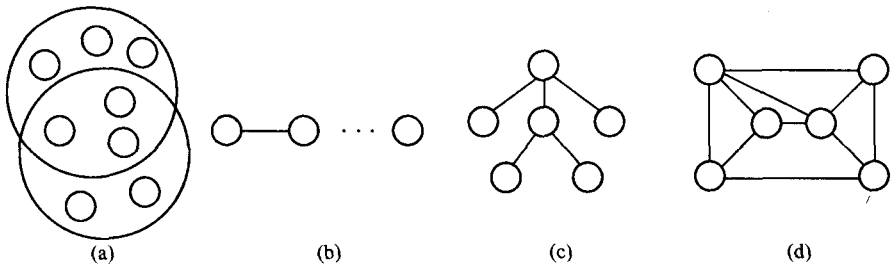


图 1.4 基本逻辑结构

(a) 集合; (b) 线性关系; (c) 层次关系; (d) 网状关系。

2. 数据的存储结构或物理结构 (Physical Structure)

数据结构在计算机中的表示 (映像) 称为数据的存储结构或物理结构，它包括数据元素的表示和关系的表示。

(1) 数据元素的表示。计算机中表示信息的最小单位是比特 (bit)，即二进制数中的一位。数据元素可由一个由若干位组成的位串来表示，通常称这个位串为节点 (Node) 或元素 (Element)。当数据元素由若干数据项组成时，位串中对应各个数据项的子串称为数据域 (Data Field)。

(2) 关系的表示。存储结构中的关系反映了数据元素在存储器中物理位置上的关系。目前，对一个数据结构常用的有 4 种存储表示。

1) 顺序存储 (Sequential Storage)

将数据结构中各元素按照其逻辑顺序存放于存储器一片连续的存储空间中 (如 C 语言的一维数组)，此时逻辑上相邻的元素在物理位置上也是相邻的，由此得到的存储结构为顺序存储结构。每种类型的逻辑结构都可以顺序存储。如表 $L=(a_1, a_2, \dots, a_n)$ 的顺序结构如图 1.5 所示。

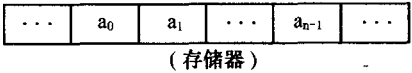


图 1.5 顺序存储

2) 链式存储 (Linked Storage)

将数据结构中各元素分布到存储器的不同点，用地址 (或链指针) 方式建立它们之间的联系，由此得到的存储结构为链式存储结构。如表 $L=(a_1, a_2, \dots, a_n)$ 的链式存储结构如图 1.6 所示。

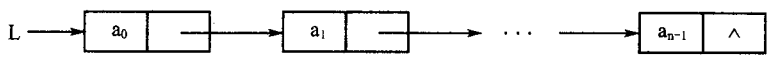


图 1.6 链式存储

此时，数据元素中除了各个数据项还要添加一个存放指针信息的子串，与数据域相对应，它被称为指针域（Pointer Field）。链式存储结构是数据结构的一个重点，因为数据结构中元素之间的关系在计算机内部很大程度上是通过地址或指针来建立的。

3) 索引存储（Indexed Storage）

在存储数据的同时，建立一个附加的索引表，即：索引存储结构=数据文件+索引表。

例 1.5 电话号码查询问题。为便于提高查询的速度，在存储用户数据文件的同时，建立一张姓氏索引表，如图 1.7(a)所示。这样，查找一个电话就可以先查索引表，再查找相应的数据文件，无疑加快了查询的速度。

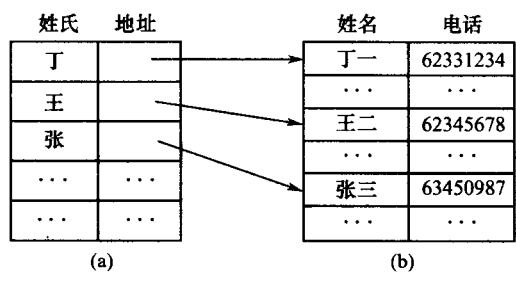


图 1.7 索引存储

(a) 索引表；(b) 数据文件。

4) 散列存储（Hash Structure）

根据数据元素的特殊字段，称为关键字（Key），计算数据元素的存放地址，然后数据元素按地址存放，所得到的存储结构为散列存储结构(Hash Structure)。

以后章节中讨论较多的是 1)、2)两种存储结构。至于建立索引和散列结构，其目的之一都是为了提高对数据文件的查询速度，这留在第八章讨论。

3. 数据的操作

一般而言，必须对数据进行加工处理才能得到问题的解。在非数值性问题中，对数据的操作（或运算）已不限于对数据进行加、减、乘、除等数学运算，数据的各种逻辑结构都有相应的操作，也就是说，在各种数据结构上都可以定义对数据实施的操作。操作的种类没有限制，可以根据需要来定义。数据的操作是定义在逻辑结构上的，而操作的具体实现是在存储结构上进行的。基本的数据操作主要有以下几种：

- (1) 查找：在数据结构中寻找满足某个特定条件的数据元素的位置或值。
- (2) 插入：在数据结构中添加新的数据元素。
- (3) 删除：删去数据结构中指定的数据元素。
- (4) 更新：改变数据结构中某个数据元素一个或多个数据项的值。
- (5) 排序：（在线性结构中）保持数据元素的个数不变，重新安排数据元素之间的逻辑顺序，使之按（某个数据项的）值由小（大）到大（小）的次序排列。

其中，查找是一种引用型操作，即它不改变现有结构，而其余 4 种均会改变现有结

构,被称为加工型操作。插入、删除和更新操作通常都包含一个查找操作,以确定插入、删除和更新的确切位置。

综上所述,数据结构是一门研究在非数值计算的程序设计中计算机操作的对象以及它们之间关系和操作的学科。在本书中,对数据结构的详细讨论,都是从数据的逻辑结构、数据的存储结构和对数据的操作这3方面展开的,读者应掌握住这个规律,以便对以后知识的学习。

1.1.3 研究数据结构的方法

研究数据结构是为了编写解决问题(或完成任务)的程序。用计算机求解一个实际问题的过程,一般可以用图1.8所示的流程加以描述。

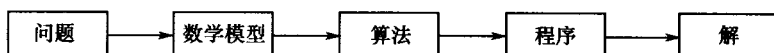


图 1.8 计算机求解问题的流程

也就是说,首先要从现实问题出发,抽象出一个适当的数学模型,然后设计一个求解此数学模型的算法,最后根据这个算法编出程序,经过测试、排错、运行直至得到最终的解答。(现实)问题、数学模型、算法和程序是问题求解过程中出现的4个不同的概念。

(1) 问题(Problem)。从直觉上讲,问题就是一个要完成的任务,即对应一组输入有一组相应的输出。例如,在例1.1和例1.2中描述的都是一个问题。在问题的定义中不应包含有关怎样解决问题的限制。只有在问题被准确定义并完全理解后才有可能研究问题的解决方法。然而在问题的定义中应该包含对所有解决方案所需要的资源(比如说,计算机的主存储器和磁盘空间以及运行时间)的限制。

(2) 数学模型(Mathematical Model)。问题的数学模型是指用数学的方法精确地把问题描述成为函数。而函数(Function),是输入(即定义域)和输出(即值域)之间的一种映射关系。函数的输入是一个值或一些信息,这些值组成的输入称为函数的参数。不同的输入可以产生不同的输出,但对于给定的输入,每次计算函数时得到的输出必须相同。

(3) 算法(Algorithm)。算法是指解决某个问题的一种方法(1.4节将给出算法更详细的定义)。如果将问题抽象为数学模型,那么它仅是精确地定义了输入和输出的映射关系,而算法则能把输入转化为输出。一个问题可以有多种算法。一个算法如果能在所要求的资源限制内将问题解决好,则称这个算法是有效率的。一个算法如果比其他已知算法需要的资源更少,则称这个算法效率更高。

(4) 程序(Program)。一个计算机程序被认为是对一个算法用某种程序设计语言的具体实现。由于使用任何一种现代计算机程序设计语言都可以实现任何一个算法,所以可能有许多程序都是同一个算法的实现。虽然算法是独立于程序的,但因为最终的目标是问题求解,所以在定义算法时,应该提供足够多的细节,以便转换为程序。在本书中使用以C语言为基本构架的程序作为算法的描述。

在问题求解模型中关键的一步是建立数学模型,而寻找(或者说将问题抽象成)数