

高等院校计算机专业教育改革推荐教材

数据结构 与算法

黄国兴 章炯民 编著

. 12

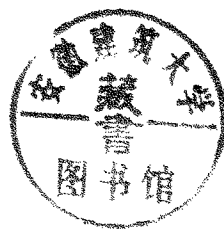


机械工业出版社
CHINA MACHINE PRESS

高等院校计算机  推荐教材

数据结构与算法

黄国兴 章炯民 编著



机械工业出版社

“数据结构与算法”是计算机科学与技术专业的一门很重要的专业基础课,也是《中国计算机科学与技术学科教程 2002》中指定的核心课程之一。本书内容覆盖了该教程中关于这门课程的所有知识点。在处理这些知识点时,本书充分考虑了数据结构和算法的融合。和传统教材相比,本教材在不增加教学学时的情况下,强化了对算法的训练,为学生学习后继课程打下扎实的基础。

本书适合计算机科学与技术学科和相关专业的本科生学习,也可作为自学教材。

图书在版编目(CIP)数据

数据结构与算法/黄国兴,章炯民编著. —北京:机械工业出版社,2004.7
(高等院校计算机专业教育改革推荐教材)

ISBN 7-111-14490-2

I. 数... II. ①黄...②章... III. ①数据结构—高等学校—教材②算法分析—高等学校—教材 IV TP311.12

中国版本图书馆 CIP 数据核字(2004)第 045702 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)

策 划: 胡毓坚

责任编辑: 王 颖

责任印制: 李 妍

北京蓝海印刷有限公司印刷·新华书店北京发行所发行

2004 年 7 月第 1 版·第 1 次印刷

787mm×1092mm $\frac{1}{8}$ ·16 印张·395 千字

0001—5000 册

定价: 23.00 元

凡购本图书,如有缺页、倒页、脱页,由本社发行部调换

本社购书热线电话:(010)68993821、88379646

封面无防伪标均为盗版

高等院校计算机专业教育改革推荐教材

编委会成员名单

主 编 刘大有

副主编 王元元

编 委 (按姓氏笔画排序)

刘晓明 李师贤 张桂芸 徐汀荣

耿亦兵 顾军华 黄国兴 薛永生

编者的话

计算机科学技术日新月异的飞速发展和计算机科学技术专业教育的相对滞后,已是不争的事实。

有两个发人深省的现象:一是,由于非计算机专业的学生既具有一门非计算机专业的专业知识,又具有越来越高的计算机应用技术水平,从而使计算机专业的学生感受到一种强烈的冲击和压力;二是,创建软件学院的工作已有近两年的历史,但软件学院的计算机专业教育的定位仍在探讨之中。

我们认为计算机科学与技术专业(以下简称计算机专业)教育的改革势在必行,正确认识和划分计算机专业教育的层次,对该专业的教育改革无疑是一个非常重要的问题。我国的计算机专业教育主要分三个层次。一般说来,这三个层次通常分布在以下三类高等院校:

第一层次主要以具有计算机一级学科博士学位授予权的教育部属重点高等院校为代表(包括具有两个博士点的大学)。这一类大学本科着重培养理论基础比较坚实、技术掌握熟练、有一定研究和开发能力的计算机专业学科型人才,其中部分学生(约本科生的10%)可攻读博士学位。

第二层次主要以具有一个计算机二级学科专业博士点的教育部属高等院校为代表。这一类高等院校本科着重培养有一定的理论基础、技术掌握比较熟练、有一定的研究或开发能力的计算机专业人才,其中一部分培养成学科型人才,另一部分培养成应用型人才,一小部分学生(约本科生的5%)可攻读博士学位。

第三层次主要以具有计算机二级学科专业硕士点的省属高等院校为代表。这一类高等院校本科面向企业应用,侧重培养对计算机技术或部分计算机技术掌握比较熟练,有一定的开发、应用能力的计算机专业应用型人才,其中很小一部分学生(约本科生的2.5%)可攻读博士学位。

国家教育部、计委批准的或省教育厅批准的示范性软件学院,就其培养目标和办学特色而言,分别与第二层次中应用型人才培养部分以及第三层次比较相近,但在如下方面有所不同:将软件工程课程作为专业教学重点;更加强调英语教学,更加重视实践能力培养,并对两者有更高的要求。

我们本着对高等院校的计算机专业状况的认识,主要面向与上述第二、第三两个层次对应的院校及与之相近的软件学院,总结多年的计算机专业的教改经验,在一定程度上溶入了ACM & IEEE CC2001和CCC2002(中国计算机科学与技术学科教程)的教改思路,组织我国一直投身于计算机教学和科研的教师,编写了这套“高等院校计算机专业教育改革推荐教材”(以下简称“推荐教材”)。自然,“推荐教材”中所贯穿的改革思路和做法,也是针对上述第二、第三两个层次对应院校的计算机专业学生。这些思路和做法可概括成以下三句话:

- 适度调整电子技术基础、计算机理论基础和系统软件的教学内容。
- 全面强化计算机工具软件、应用软件的教学要求。
- 以应用为目标大力展开软件工程的教学与实践。

电子技术基础、计算机理论基础、系统软件教学关系到学生的基本素质、发展潜力和日后

的应变能力。“推荐教材”在调整它们的教学内容时的做法是:适度压缩电子线路、数字电路和信号系统的教学内容,变三门课程为两门,并插入数字信号处理的基础内容;合并“计算机组成原理”、“微型计算机接口技术”和“汇编语言”为“计算机硬件技术基础”一门课程;注意适当放宽“离散数学”课程的知识面,使之与 CCC2002 的要求基本接轨,但适度降低其深度要求;更新系统软件课程的教学内容,以开放代码的 Linux 作为操作系统原理的讲授载体,更加关注系统软件的实践性和实用性。

为了提高计算机专业人才的计算机应用能力,全面强化计算机工具软件、实用软件的教学要求是十分重要的,这也是上述改革思路的核心。为此,“系列教材”的做法是:强化程序设计技术,强化人机接口技术,强化网络应用技术。

为强化程序设计技术,“推荐教材”支持在单片机环境、微机平台、网络平台的编程训练;支持运用程序设计语言、程序设计工具以及分布式对象技术的编程训练。大大加强面向对象程序设计课程的组合(设计了三门课程:面向对象的程序设计语言 C++,面向对象的程序设计语言 JAVA 和分布式对象技术),方便教师和读者的选择。

为强化人机接口技术,“推荐教材”设计了“人机交互教程”,“计算机图形学”和“多媒体应用技术”等可供选择的、有层次特色的课程组合。

为强化网络应用技术,“推荐教材”设计了“计算机网络技术”,“计算机网络程序设计”,“计算机网络实验教程”和“因特网技术及其应用”等可供选择的、新颖丰富的课程组合。

将软件工程课程作为专业教学重点,以应用为目标大力展开软件工程的教学与实践,是“推荐教材”改革思路的又一亮点。为改变以往软件工程课程纸上谈兵的老毛病,“推荐教材”从工程应用出发,理论联系实际,突出建模语言及其实现工具的运用,设计了“软件工程的方法与实践”,“统一建模语言 UML 导论”和“ROSE 对象建模方法与技术”等可供选择的、创新独特的软件工程课程组合。对于各类软件学院,“推荐教材”的这一特色无疑是很有吸引力的。

强调实践也是计算机学科永恒的主题,对计算机应用专业的学生来说更是如此。重应用和重实践是“推荐教材”的一个整体特点。这一特点,一方面有利于解决本文开始所指出的计算机专业学生较之非计算机专业学生,在应用开发工作中上手慢的问题;另一方面,使计算机专业的学生能在更大范围内、更高层面上掌握计算机应用技术。这一特点正是许多高等院校计算机专业教育改革追求的一个目标,也是国家教育部倡导软件学院的初衷之一。

“推荐教材”由基础知识、程序设计、应用技术、软件工程和实践环节等五个模块组成。各模块有其对应的培养目标与功能,从而构架出一个创新的、完整的计算机应用专业的课程体系。模块化的设计,使各学校可根据学生及学校的特点做自由的选择和组合,既能达到本专业的总体要求,又能体现具有特色的个性发展。整套教材的改革脉络清晰,结构特色鲜明,值得各高等院校在改革教学内容、编制教学计划、挑选教材书目时借鉴和参考。当然,很多书目也适合很多相关学科的计算机课程用作教材。

“推荐教材”的组成模块和书目详见封底。显然它不能说是完备的(实践环节模块更是如此),其改革的思路、改革的举措也可能有值得探讨的地方。我们衷心希望得到计算机教育界同仁和广大读者的批评指正。

高等院校计算机专业教育改革推荐教材

编委会

前 言

信息技术的发展日新月异,从事计算机科学与技术工作的工程技术人员越来越多,计算机科学与技术学科在国民经济建设中的地位也越来越重要。计算机科学与技术学科与电子科学、工程以及数学有着复杂的联系。计算机加工处理的对象已由纯粹的数值发展到字符、表格和图像等各种具有一定结构的数据,这就给程序设计带来一些新的问题。为了编写出一个“好”的程序,必须分析待处理对象的特性以及各处理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。在《中国计算机科学与技术学科教程 2002》为本科教学提出的 16 门核心课程中,“数据结构”是其中之一。这门课程是计算机科学与技术学科的专业基础课,它是学习编译程序、操作系统、数据库系统等专业课程的前提和基础。在《中国计算机科学与技术学科教程 2002》中对“数据结构”的具体教学知识点,提出了明确的要求,希望通过这门课程的学习能让学生掌握一些常用的数据结构和算法,并能利用学到的知识为应用服务。

本书就是按照《中国计算机科学与技术学科教程 2002》中对“数据结构”知识点的要求编写的,根据作者多年来在“数据结构”教学方面积累的经验,本书在内容的安排和对知识点教学的处理方面作了一些尝试,力求体现以下特点:

1) 涵盖《中国计算机科学与技术学科教程 2002》中对“数据结构”课程应该包含的核心知识点。

2) 对抽象概念的叙述尽量使用实际例子,使学生体会由具体到抽象的思维过程,比较牢固地掌握相关概念,并能把这些概念应用于实际。

3) 把数据结构的内容与算法设计及分析融合在一起讨论,在叙述数据结构内容时,适时穿插算法设计与分析的内容;另外把部分数据结构的内容,安排到算法设计与分析的章节讲授。这样安排有利于数据结构内容和必要的算法设计与分析内容的融合,但篇幅并没有太多增加。

4) 在叙述知识点时力求做到思路清晰、条理清楚、叙述简练,使学生容易理解,教师容易讲授。

5) 为使学生深入掌握本门课程的知识,并启发其思维,每章除了常规的习题外,还安排了一定量的应用习题。

全书共分 9 章,第 1 章 数据结构和算法概述,介绍数据结构和算法的基本概念,建议安排 5 学时;第 2 章 线性表,介绍线性表的概念、存储结构、线性表的操作及其应用,建议安排 8 学时;第 3 章 栈和队列,介绍栈和队列的基本概念及操作、存储结构及应用实例,建议安排 6 学时;第 4 章 数组、矩阵和串,介绍数组、矩阵的存储结构及变换,串的概念、存储结构及操作,建议安排 6 学时;第 5 章 树,介绍和树有关的概念、它们的存储结构及这些结构上的操作,以及堆和堆排序的内容,建议安排 15 学时;第 6 章 图,介绍图的基本概念,图的存储结构及其应用,建议安排 8 学时;第 7 章 查找,介绍各种查找方法,B 树的结构、特性及其应用,建议安排 9 学时;第 8 章 算法设计方法,介绍贪婪算法、分而治之算法、动态规划、回溯和分枝定界等 5 种最基本、最常见的算法设计方法,建议安排 10 学时;第 9 章 算法的限制,讨论算法的时间复杂

性、算法的限制或问题的难度问题,建议安排 5 学时。

由于本教材涉及的内容比较多,各学校的教师和学生的情况也不一样,在学习本书时各校可以适当调整建议教学学时;同时对其中一些章节的内容也可以根据各校的实际情况进行裁剪处理。如对某些章中有“*”号的内容,可以根据不同的要求取舍。

本书由黄国兴、章炯民编著,在编写过程中得到了鲍钰、窦亮、肖晶、赵若韵、何涛、陈强的大力协助,其间还和计算机教育界的同行就数据结构的内容和形式进行了有意义的讨论,他们提出了许多建设性的意见和建议,在此一并致谢。

由于计算机科学技术发展迅速,再加上编者水平有限,书中错误和不妥之处,恳请读者批评指正。

编 者

目 录

编者的话

前言

第 1 章 数据结构和算法概述	1
1.1 数据结构	1
1.1.1 数据结构的逻辑结构	1
1.1.2 数据结构的物理结构	3
1.1.3 抽象数据类型	4
1.2 算法	5
1.2.1 算法的概念	5
1.2.2 算法的评价	6
1.3 算法的时间复杂性和空间复杂性分析	7
1.3.1 时间复杂性分析概述	8
1.3.2 关键操作计数和执行步数计数	8
1.3.3 最好、最坏和平均情况	10
1.3.4 渐近分析	11
1.3.5 空间复杂性分析	16
1.4 习题	18
第 2 章 线性表	20
2.1 线性表的基本概念	20
2.2 顺序表	22
2.2.1 线性表的顺序存储	22
2.2.2 顺序表的操作算法	22
2.3 链表	26
2.3.1 线性表的链接存储	26
2.3.2 链表的操作算法	27
2.3.3 链表的变形	30
2.3.4 线性表实现方法的比较	34
2.4 线性表的应用	35
2.4.1 一元多项式的表示和相加算法	35
2.4.2 归并排序算法	39
2.5 广义表	41
2.5.1 广义表的概念	41
2.5.2 广义表的存储结构	42
2.5.3 广义表的递归算法	43

2.6 习题	43
第3章 栈和队列	45
3.1 栈	45
3.1.1 栈的概念	45
3.1.2 顺序栈	46
3.1.3 链接栈	48
3.2 栈的应用	49
3.2.1 数制转换	49
3.2.2 算术表达式求值	50
3.3 队列	56
3.3.1 队列的概念	56
3.3.2 链接队列	57
3.3.3 顺序(循环)队列	58
3.4 队列的应用	63
3.4.1 桶排序	63
3.4.2 多关键字排序	65
3.4.3 基数排序	66
3.5 双向队列	68
3.6 习题	69
第4章 数组、矩阵和串	71
4.1 数组的顺序存储	71
4.1.1 二维数组的顺序存储结构	72
4.1.2 n 维数组的顺序存储结构	73
4.2 矩阵的压缩存储	74
4.2.1 特殊矩阵的压缩存储	74
4.2.2 稀疏矩阵的压缩存储和操作	76
4.3 串	80
4.3.1 串的基本概念	81
4.3.2 串的存储结构	81
4.3.3 顺序串的基本操作算法	82
4.3.4 模式匹配	84
4.4 习题	88
第5章 树	89
5.1 树和森林	90
5.1.1 树和森林的概念和术语	90
5.1.2 树的存储结构	93
5.1.3 树和森林的遍历	95
5.2 二叉树	98
5.2.1 二叉树的概念	98

5.2.2	二叉树的抽象数据类型	99
5.2.3	二叉树的基本性质	100
5.2.4	几种特殊的二叉树	101
5.2.5	二叉树的存储结构	103
5.3	二叉树的遍历	105
5.4	树、森林与二叉树的转换	108
5.4.1	树、森林转换为二叉树	109
5.4.2	二叉树还原为树、森林	111
*5.5	线索二叉树	112
5.5.1	线索二叉树的概念	112
5.5.2	二叉树的线索化	113
5.5.3	线索二叉树的操作	114
5.6	二叉树的应用	115
5.6.1	表达式树及其求值	115
5.6.2	堆和堆排序	117
5.6.3	哈夫曼树及其应用	121
5.7	习题	128
第6章	图	131
6.1	图的数学基础	131
6.1.1	图的基本概念和术语	131
6.1.2	图的抽象数据类型	135
6.2	图的存储结构	136
6.2.1	邻接矩阵	136
6.2.2	邻接表	138
6.3	图的遍历	140
6.3.1	深度优先搜索法	140
6.3.2	广度优先搜索法	142
6.3.3	遍历的简单应用	144
6.4	最短路径问题	148
6.5	最小生成树	152
6.6	习题	156
第7章	查找	160
7.1	线性表的查找	161
7.1.1	顺序查找	161
7.1.2	二分查找	162
7.1.3	分块查找	164
7.2	查找树	165
7.2.1	查找树的概念	165
7.2.2	查找树的查找	166

7.2.3	查找树的插入和生成	168
7.2.4	查找树的删除	169
7.3	平衡查找树	171
7.4	B-树	178
7.4.1	B-树的查找	179
7.4.2	B-树的插入	180
7.4.3	B-树的删除	182
7.4.4	B ⁺ 树	183
7.5	散列表	185
7.5.1	散列函数	186
7.5.2	冲突处理	188
7.5.3	散列方法的性能分析	194
7.6	习题	194
第8章	算法设计方法	196
8.1	贪婪算法	196
8.1.1	直接选择排序和冒泡排序	196
8.1.2	AOV-网络和拓扑排序	200
8.1.3	0/1 背包问题	202
8.2	分而治之的算法	204
8.2.1	快速排序	204
8.2.2	排序算法综述	206
8.3	动态规划	207
8.3.1	斐波那契数	208
8.3.2	顶点对的最短路径	209
8.3.3	关键路径	211
8.4	回溯	215
8.4.1	皇后问题	216
8.4.2	迷宫问题	219
8.5	分枝定界	221
8.5.1	再论 0/1 背包问题	222
8.5.2	旅行商问题	225
8.6	随机算法	227
8.6.1	随机数的产生	228
8.6.2	蒙特卡罗积分	228
8.7	习题	229
*第9章	算法的限制	232
9.1	更快的计算机与更快的算法	233
9.2	归约	234
9.3	排序问题的时间复杂性下限	235

9.4 难解问题	237
9.4.1 NP 完全性理论	237
9.4.2 非确定性计算机	238
9.4.3 NP 完全问题的归约证明	239
9.4.4 处理 NP 难的问题	239
9.5 不可解问题	240
9.5.1 不可解问题的存在性	240
9.5.2 停机问题的不可解性	241
9.6 习题	242
参考文献	243

第 1 章 数据结构和算法概述

所谓数据结构(data structure),简单来说就是数据及其相互之间的关系。数据间的不同的关系,决定了不同种类的数据结构。本书将介绍 4 种基本数据结构:集合、线性表、树和图。

在计算机科学中研究一种数据结构是从以下 4 个方面进行的:

- 1) 数据结构的逻辑结构:特定的数据结构的定义。
- 2) 数据结构的物理结构:在计算机中存储这种数据结构的方法。
- 3) 该数据结构上常用的操作算法。
- 4) 该数据结构的应用。

在许多情况下,也把上述 4 个方面中的前 3 个方面合称为数据结构,而在另一些场合下,数据结构又单指其逻辑结构或物理结构,所以名词“数据结构”的确切含义取决于上下文。

本书将从第 2 章开始,从上述四个方面讨论常用的基本数据结构,本章介绍各种数据结构的共性问题、研究数据结构时将要使用的方法、工具和其他相关问题。

1.1 数据结构

1.1.1 数据结构的逻辑结构

数据是对客观事物的符号表示,在计算机科学中是指所有能输入到计算机中并被计算机程序处理的符号的总称。这是计算机程序加工的“原料”。例如,一个利用数值分析方法解代数方程的程序,其处理对象是整数和实数;一个编译程序或文字处理程序的处理对象是字符串,等等。因此,对计算机科学而言,数据的含义极为广泛,如图像、声音等等都可以通过编码而纳入数据的范畴。

数据元素指的是数据的基本组织单位,在计算机程序中通常将它作为一个整体进行考虑和处理。例如,在如图 1-1a 所示的组织结构中,“树”中的每一个方框所表示的数据是一个数据元素;图 1-1b 中的每一个圆圈所表示的数据也被看作一个数据元素。有时,一个数据元素又可由若干个其他数据元素组成。所以,数据元素从它们的组成结构来看是分层次的,其中不可分割的最小的数据组织单位称为数据项。注意,所谓“不可分割的最小的数据组织单位”只有相对意义,没有绝对的意义,一个数据元素是否是不可分割的取决于具体的应用及其背景。当数据元素由若干个其他数据项组成时,称能惟一标识数据元素的数据项为关键字(Key),简称为键。本书中,为叙述方便起见,在表示一个数据元素时,往往只给出该数据元素的关键字,而省略其他数据项。

表 1-1 中的每一行是一个数据元素,它由学号、姓名、各科成绩及平均成绩等数据项组成,学号是关键字。若有必要,也可以把姓名进一步分解成多个字符,即不把姓名看作数据项,而把它看作是由多个字符(数据项)组成的数据元素。

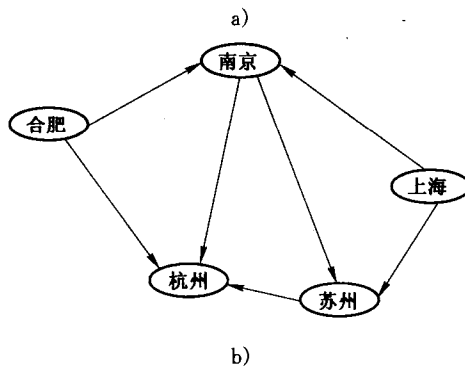
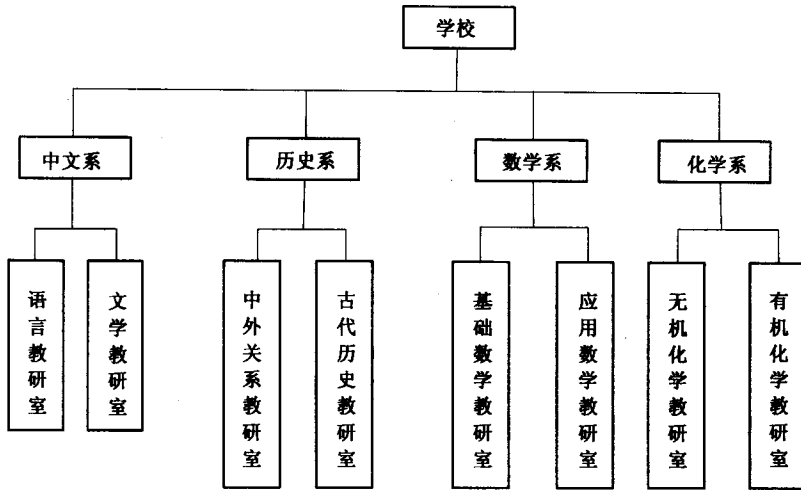


图 1-1 两个数据结构

a) 某学校的组织结构图 b) 城市之间的航线图

表 1-1 学生成绩表

学号	姓名	数据结构	C语言	高等数学	平均成绩
00122001	张三	90	85	95	90
00122002	李四	80	85	90	85
00122003	王五	70	84	86	80
00122004	马六	91	84	92	89
.....

由于数据元素所代表的客观对象在现实中不是孤立存在的,它们之间存在着或多或少的联系,所以数据元素之间也存在着某些关系,这种数据元素之间的关系称为结构,常用集合上的关系来描述。数据结构(逻辑结构)是相互之间存在一种或多种特定关系的数据元素的集合。一个数据结构可以形式地表示成一个二元组 (D, R) , D 是数据元素的集合, R 是 D 上的关系的集合,通常 R 中只有一个关系(如本书所涉及的各种数据结构)。

图 1-1a 所示的“树”是一个数据结构,其中 $D = \{\text{学校, 中文系, 历史系, 数学系, 化学系, 语言教研室, 文学教研室, 中外关系教研室, 古代历史教研室, 基础数学教研室, 应用数学教研室, 无机化学教研室, 有机化学教研室}\}$

无机化学教研室,有机化学教研室}; $R = \{r\}$, $r = \{ \langle \text{学校, 中文系} \rangle, \langle \text{学校, 历史系} \rangle, \langle \text{学校, 数学系} \rangle, \langle \text{学校, 化学系} \rangle, \langle \text{中文系, 语言教研室} \rangle, \langle \text{中文系, 文学教研室} \rangle, \langle \text{历史系, 中外关系教研室} \rangle, \langle \text{历史系, 古代历史教研室} \rangle, \langle \text{数学系, 基础数学教研室} \rangle, \langle \text{数学系, 应用数学教研室} \rangle, \langle \text{化学系, 无机化学教研室} \rangle, \langle \text{化学系, 有机化学教研室} \rangle \}$ 。图1-1b所示的“图”也是一个数据结构,其中 $D = \{ \text{上海, 南京, 苏州, 杭州, 合肥} \}$, $R = \{r\}$, $r = \{ \langle \text{上海, 南京} \rangle, \langle \text{上海, 苏州} \rangle, \langle \text{南京, 杭州} \rangle, \langle \text{南京, 苏州} \rangle, \langle \text{苏州, 杭州} \rangle, \langle \text{合肥, 南京} \rangle, \langle \text{合肥, 杭州} \rangle \}$ 。

假设有一个数据结构 (D, R) , $R = \{r\}$, a 和 b 是 D 中的数据元素。若 $\langle a, b \rangle$ 是 r 中的一个有序偶对,则称 a 是 b 的直接前驱,简称为前驱;称 b 是 a 的直接后继,简称为后继。通常,在数据结构的图形表示中,从前驱到后继画一个箭头,以表示数据元素之间的关系 r 。

如图1-1b所示,“上海”是“南京”和“苏州”的前驱,“南京”和“苏州”都是“上海”的后继;“合肥”也是“南京”的前驱,但“合肥”不是“苏州”的前驱。

注意,在图1-1a中,数据元素之间的连线上没有箭头,这是因为在用图形表示树时,约定把前驱画在上方,把后继画在下方,并省略连线上的箭头,隐含地认为所有的连线上都有一个从上方指向下方的箭头。(参见第5章)

在如图1-1a所示的数据结构中,“中外关系教研室”和“古代历史教研室”都只有一个前驱“历史系”,但“历史系”有两个后继“中外关系教研室”和“古代历史教研室”。事实上,在如图1-1a所示的数据结构“树”中,每个数据元素至多只有一个前驱,但可以有多个后继。

集合、线性表、树和图这四种基本数据结构的主要特点如下:

1) 集合结构:这种数据结构中的元素之间除了同属于一个集合的“关系”外,别无其他关系。

2) 线性结构:这种数据结构中的数据元素之间存在“一个对一个”的关系。若线性结构不是空集,则必有一个数据元素没有前驱,有一个数据元素没有后继,其余每个数据元素都有且仅有一个前驱和一个后继。表1-1给出了一个线性表,“李四”所在的数据元素的前驱和后继分别是“张三”和“王五”所在的数据元素。

3) 树形结构:这种数据结构中的数据元素之间存在“一个对多个”的关系。一个数据元素可能有多个后继,但却至多只有一个前驱。图1-1a给出了一棵树,“历史系”的前驱是“学校”,后继是“中外关系教研室”和“古代历史教研室”。

4) 图结构:这种数据结构中的数据元素之间存在“多个对多个”的关系。一个数据元素可能有多个前驱和多个后继。在图1-1b中,“南京”有两个前驱“合肥”和“上海”,有两个后继“杭州”和“苏州”。

1.1.2 数据结构的物理结构

讨论数据结构的目的是为了在计算机中实现对它的存储和处理,因此必须研究如何在计算机中存储(表示)数据结构。

数据结构在计算机中的表示称为数据的物理结构,又称为数据结构在计算机中的映象或存储结构,它包括数据元素的表示和关系的表示。

数据元素在计算机中的映象是结点。结点在计算机中表示成位串,如用一个字长的位串表示一个整数,用八位二进制表示一个字符等等,通常称这个位串为结点或记录。当数据元素

由若干其他数据元素或数据项组成时,位串中对应于各个数据元素或数据项的子位串称为数据域、域或字段。比如某个学生结点,由学号、年龄、成绩等字段组成。有些时候,数据元素也称作结点,所以术语结点的确切含义需视上下文而定。

出于表示数据元素之间的关系的需要,或出于提高处理效率的需要等,人们创造了许多不同的存储数据结构的存储方式,以适应千变万化的应用要求。常用的存储结构主要有四种:顺序、链式、散列和索引存储结构,本书将对其中的前3种存储结构作详细的讨论,对索引结构也会有所涉及。下面简要描述这4种存储结构的特点,以后各章还将对它们进行详细讨论。

1) 顺序存储结构:数据元素间的关系由它们的相对物理存储位置隐含地表示,空间利用率较高;需使用整块连续区域,可能会造成碎片;修改时要移动数据,效率较低,适合于静态场合。

2) 链式存储结构:数据元素间的关系由指针(地址)显式地表示,空间利用率较低;不需要使用整块连续区域,不会造成碎片;修改时不必移动数据,效率较高,适合于动态场合。

3) 散列存储结构:由数据元素的值直接决定(计算出)存储地址,一般操作效率高;有冲突,从而增加时空开销;有些操作不便,效率低。

4) 索引存储结构:用索引表加快操作;空间开销增加。

这些存储结构各有优缺点,对于某种特定的数据结构,应该采用哪种方法取决于具体的数据结构和具体的应用场合,通常必须进行综合平衡才能取得较好的效果。

在高级程序设计语言的层次上讨论数据结构时,不直接涉及内存地址,而是利用高级语言中提供的“数据类型”描述存储结构,例如用“一维数组”描述顺序存储结构,用“指针”描述链式存储结构等。

1.1.3 抽象数据类型

数据类型是一个值的集合以及定义在这个集合上的一组操作的总称。例如,C语言中的整数类型,其值的集合由某个区间中的整数构成(在不同的计算机上,这个区间的大小可能不一样),定义在这个集合上的操作有:加、减、乘、取模、除等等。

数据类型与数据结构之间有着密切的联系。可以把数据结构中的数据元素和关系都看作为“值”,把数据结构中的操作看作定义在这个“值”的集合上的操作,从而把数据结构当作一种数据类型。

注意,数据类型的概念不涉及值的具体存储方式,也不涉及操作的具体实现,所以数据类型只是一个逻辑上的概念,与数据类型的实现无关。事实上,根据具体的软硬件环境以及具体应用的需要,一个数据类型可以有多种不同的实现方法。

抽象数据类型(abstract data type)简称ADT,是一个数学模型,其中定义了一组操作,这组操作的行为刻画了这个数学模型的逻辑特性。与数据类型一样,ADT也是一个逻辑概念,与其在计算机上的表示和实现无关。无论具体如何表示和实现一个ADT,如果它的数学特性(行为)不变,对用户来说都是相同的。抽象数据类型和数据类型实质上是相似的概念,只是ADT的范畴更广,不再局限于某种具体的程序设计语言,可用来定义用户所需要的任何数据类型。

本书按以下的格式定义抽象数据类型,具体例子可参考以后各章。

ADT 抽象数据类型名