



普通高等学校计算机科学与技术应用型规划教材

数据结构 (C语言版)

主 编 马 睿 孙丽云
副主编 邵兰洁 靳 丽 李月军

SHUJU JIEGOU
(C YUYANBAN)



北京邮电大学出版社
www.buptpress.com

普通高等学校计算机科学与技术应用型规划教材

内容简介

本书是普通高等院校计算机科学与技术专业应用型规划教材之一，可作为高等院校计算机科学与技术专业及相关专业的教材，也可供从事计算机工作的工程技术人员参考。本书主要介绍数据结构的定义、分类、特点、应用及设计方法。全书共分5章，第1章介绍数据结构的定义、分类、特点及设计方法；第2章介绍线性表的定义、分类、特点及设计方法；第3章介绍栈、队列的定义、分类、特点及设计方法；第4章介绍树的定义、分类、特点及设计方法；第5章介绍图的定义、分类、特点及设计方法。本书可作为高等院校计算机科学与技术专业及相关专业的教材，也可供从事计算机工作的工程技术人员参考。

数据结构

(C语言版)

图例题解(C语言版)

本书可作为高等院校计算机科学与技术专业及相关专业的教材，也可供从事计算机工作的工程技术人员参考。

主编 马睿 孙丽云

副主编 邵兰洁 靳丽 李月军

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

主审：马睿、孙丽云

北京邮电大学出版社

北京邮电大学出版社

北京邮电大学出版社

内 容 简 介

“数据结构”是计算机相关专业的重要基础核心课程之一,是一门集技术性、理论性和实践性于一体的课程。本书重点介绍基本数据结构、C语言数据结构描述、数据结构的应用、算法设计与分析以及算法性能评价等内容,使读者理解数据抽象与编程实现的关系,提高用计算机解决实际问题的能力。全书共10章,主要内容包括线性表、栈、队列、串、数组、树、图、查找、排序和文件。

本书结构合理,重点突出,内容精练,通俗易懂,实例丰富,算法描述清晰。本书适用于高等院校应用型本科计算机及相关专业,可作为信息类及相关专业的教材或参考书,也可供成人高校、高职高专选用。

图书在版编目(CIP)数据

数据结构:C语言版/马睿,孙丽云主编. —北京:北京邮电大学出版社,2009

ISBN 978-7-5635-1983-5

I. 数… II. ①马…②孙… III. ①数据结构②C语言—程序设计 IV. TP311.12 TP312

中国版本图书馆 CIP 数据核字(2009)第 099041 号

书 名: 数据结构(C语言版)

主 编: 马 睿 孙丽云

责任编辑: 王丹丹 陈士燕

出版发行: 北京邮电大学出版社

社 址: 北京市海淀区西土城路 10 号(邮编:100876)

发 行 部: 电话: 010-62282185 传真: 010-62283578

E-mail: publish@bupt.edu.cn

经 销: 各地新华书店

印 刷: 北京源海印刷有限责任公司

开 本: 787 mm×1 092 mm 1/16

印 张: 18.75

字 数: 441 千字

印 数: 1—3 000 册

版 次: 2009 年 8 月第 1 版 2009 年 8 月第 1 次印刷

ISBN 978-7-5635-1983-5

定 价: 32.00 元

· 如有印装质量问题,请与北京邮电大学出版社发行部联系 ·

前 言

“数据结构”是高等院校计算机、信息管理与信息系统等相关专业的一门必修、经典的核心课程之一,是计算机软件的基础课程。

本书主要讨论现实世界中数据的各种逻辑结构、在计算机中的存储结构以及进行各种非数值运算的算法,旨在使软件开发和维护人员学习分析研究计算机加工的数据对象的特性,学会数据的组织方法,以便选择合适的逻辑结构、存储结构及相应的运算,把现实中的问题转化为计算机内部的数据表示和数据处理。

本书是在深入研究国内外有关数据结构的教材和大量资料的基础上,结合作者科研任务及多年教学经验组织编写而成的。本教材注重理论与实践的统一,逻辑性强,内容精练。在讲解具体内容时,特别注重实用性,尽量列举实例,在叙述上力求深入浅出,通俗易懂。

书中所有算法的描述均采用 C 语言函数的形式来描述,这些函数均已在 Turbo C 3.0 上调试运行通过,并加以详细的注释,以利于读者理解算法的基本思想,同时方便学生在计算机上验证相关算法。

全书分成 10 章,系统地介绍了线性表、栈、队列、串、数组、树、图等基本数据结构的内在逻辑关系、在计算机中的存储表示及其各种运算的实现,介绍了查找和排序的各种算法以及算法的实现。

本书各章配有丰富的习题和例题。习题的类型包括选择题、填空题、操作题和算法设计题。每章后的习题能够加深对各章内容的认识,让学生通过动手操作掌握知识。

本书由马睿和孙丽云任主编,其中马睿编写第 1 章和第 7 章,孙丽云编写了第 3 章和第 9 章,邵兰洁编写了第 6 章,靳丽编写了第 2 章,白艳霞编写了第 8 章,李珊编写了第 4 章,李立芬编写了第 5 章,云彩霞编写了第 10 章,李月军,裘琳玲等也参与了本书的编写工作。

非常感谢北京化工大学北方学院莫德举院长和北京邮电大学出版社在本书编写过程中给予的指导和帮助。

由于作者水平有限,加之时间仓促,书中难免有疏漏和错误之处,恳请同行专家及读者提出批评意见。作者电子邮箱为:marui05@sina.com。

编 者

目 录

第1章 绪论

1.1 引言	1
1.1.1 为什么要学习数据结构	1
1.1.2 数据结构课程的主要内容	1
1.2 基本概念和常用术语	2
1.2.1 基本概念和术语	2
1.2.2 数据结构	3
1.2.3 抽象数据类型	8
1.3 算法和算法分析	9
1.3.1 算法的定义	9
1.3.2 算法评价与分析	10
1.3.3 算法效率的度量	10
1.4 本章小结	14
习题	14

第2章 线性表

2.1 线性表的逻辑结构	17
2.1.1 线性表的定义	17
2.1.2 线性表的基本运算	18
2.2 线性表的顺序存储及运算实现	19
2.2.1 线性表的顺序存储结构	19
2.2.2 顺序表上基本运算的实现	20
2.3 线性表的链式存储及运算实现	25
2.3.1 单链表	26
2.3.2 循环链表	32
2.3.3 双向链表	33
2.3.4 静态链表	35
2.3.5 单链表应用举例	36
2.4 顺序表和链表的比较	37
2.5 本章小结	37

习题	38
第3章 栈和队列	
3.1 栈	41
3.1.1 栈的定义及其基本运算	41
3.1.2 栈的存储结构和基本运算的实现	42
3.2 栈的应用举例	48
3.2.1 应用栈解决数制的转换问题	48
3.2.2 表达式求值	49
* 3.3 栈与递归	56
3.3.1 递归的概念	56
3.3.2 栈与递归	59
3.4 队列	59
3.4.1 队列的定义和基本运算	59
3.4.2 循环队列——队列的顺序表示和实现	60
3.4.3 链队——队列的链式表示和实现	64
3.5 本章小结	66
习题	66
第4章 串	
4.1 串及其基本运算	72
4.1.1 串的基本概念	72
4.1.2 串的基本运算	72
4.2 串的存储结构	74
4.2.1 串的顺序存储结构	74
4.2.2 串的链式存储结构	76
4.3 串的模式匹配	79
4.3.1 朴素的模式匹配算法	79
4.3.2 KMP 算法	81
* 4.4 串的应用举例	85
4.5 本章小结	87
习题	87
第5章 数组和广义表	
5.1 数组的概念	89
5.1.1 数组的定义	89
5.1.2 二维数组	89
5.2 数组的顺序存储	90

5.3 矩阵的压缩存储	92
5.3.1 特殊矩阵	92
5.3.2 稀疏矩阵	95
*5.4 广义表	103
5.4.1 广义表的定义	103
5.4.2 广义表的存储结构	105
5.5 本章小结	107
习题	108
第6章 树	
6.1 树的概念和操作	112
6.1.1 树的定义	112
6.1.2 树的基本术语	112
6.1.3 树的基本操作	113
6.1.4 树的表示	114
6.2 二叉树	114
6.2.1 二叉树的概念	114
6.2.2 二叉树的性质	115
6.2.3 二叉树的存储结构	117
6.2.4 二叉树的基本操作	119
6.3 二叉树的遍历	120
6.3.1 二叉树的遍历方法及递归实现	120
6.3.2 二叉树遍历的非递归实现	122
6.3.3 二叉树的层次遍历	126
6.3.4 二叉树遍历算法的应用	127
6.4 线索二叉树	134
6.4.1 线索二叉树的基本概念	134
6.4.2 线索二叉树的基本操作	136
6.5 树和森林	139
6.5.1 树的存储结构	139
6.5.2 树、森林与二叉树的相互转换	142
6.5.3 树和森林的遍历	145
6.6 二叉树的应用	146
6.6.1 霍夫曼树及其应用	146
6.6.2 表达式求值	155
6.7 本章小结	158
习题	159

第7章 图

7.1 图的定义和术语	166
7.1.1 图的基本概念	166
7.1.2 图的基本操作	168
7.2 图的存储结构	169
7.2.1 邻接矩阵	169
7.2.2 邻接表	170
7.2.3 邻接多重表	172
7.2.4 十字链表	173
7.3 图的遍历	174
7.3.1 深度优先搜索	175
7.3.2 广度优先搜索	176
7.4 最小生成树	178
7.4.1 普里姆(Prim)算法	179
7.4.2 克鲁斯卡尔(Kruskal)算法	181
7.5 有向无环图及其应用	184
7.5.1 拓扑排序	185
* 7.5.2 关键路径	188
7.6 最短路径	194
7.6.1 从某个源点到其他各顶点的最短路径	194
7.6.2 每一对顶点之间的最短路径	198
7.7 本章小结	202
习题	202

第8章 查找

8.1 基本概念	208
8.2 线性表查找	209
8.2.1 顺序查找	210
8.2.2 在顺序存储的有序表上查找	212
8.3 树表查找	215
8.3.1 二叉排序树	215
8.3.2 平衡二叉树	222
8.3.3 B-树	228
8.4 散列表查找	233
8.4.1 散列表	233
8.4.2 常用的散列函数	234
8.4.3 处理冲突的方法及散列表的构造	235

8.4.4 散列表上的查找	239
8.4.5 散列表上的删除	241
8.5 本章小结	241
习题	242

第9章 排序

9.1 排序的基本概念及分类	247
9.1.1 排序概念	247
9.1.2 排序分类	247
9.1.3 排序数据的数据类型说明	248
9.2 插入排序	249
9.2.1 直接插入排序	249
9.2.2 折半插入排序	250
9.2.3 希尔排序	251
9.3 交换排序	253
9.3.1 冒泡排序	253
9.3.2 快速排序	255
9.4 选择排序	257
9.4.1 简单选择排序	257
9.4.2 堆排序	258
9.5 归并排序	261
9.6 基数排序	262
9.7 内部排序的比较与选择	262
9.7.1 内部排序算法性能比较	262
9.7.2 内部排序算法的选择	263
9.8 外部排序简介	264
9.9 本章小结	264
习题	265

第10章 文件

10.1 文件的基本知识	268
10.1.1 文件的基本概念	268
10.1.2 文件的逻辑结构与物理结构	269
10.1.3 文件的操作	270
10.2 顺序文件	271
10.2.1 存储在顺序存储器上的顺序文件	271
10.2.2 存储在直接存储器上的顺序文件	272
10.3 索引文件	272

10.3.1	索引文件构成	273
10.3.2	索引文件的存储	273
10.3.3	索引文件的操作	274
10.3.4	利用查找表建立多级索引	274
10.4	索引顺序文件	275
10.4.1	ISAM 文件	275
10.4.2	VSAM 文件	277
10.5	散列文件	280
10.5.1	散列文件的组织方式	280
10.5.2	散列文件的操作	281
10.5.3	散列文件的特点	281
10.6	多关键字文件	282
10.6.1	多关键字文件概念	282
10.6.2	多关键字文件与其他文件的区别	282
10.6.3	多重表文件	283
10.6.4	倒排文件	284
10.7	本章小结	285
	习题	286
	参考文献	288

第 10 章 索引和文件

第 11 章 数据库系统概论

第 12 章 数据库系统性能优化

第 13 章 数据库系统安全

第 14 章 数据库系统应用

第 15 章 数据库系统开发

第 16 章 数据库系统维护

第 17 章 数据库系统测试

第 18 章 数据库系统部署

第 19 章 数据库系统迁移

第 20 章 数据库系统备份与恢复

第 21 章 数据库系统灾难恢复

第 22 章 数据库系统安全审计

第 23 章 数据库系统性能监控

第 24 章 数据库系统性能调优

第 25 章 数据库系统性能分析

第 26 章 数据库系统性能优化案例

第 27 章 数据库系统性能优化工具

第 28 章 数据库系统性能优化案例

第 29 章 数据库系统性能优化案例

第 30 章 数据库系统性能优化案例

第 31 章 数据库系统性能优化案例

第 32 章 数据库系统性能优化案例

第 33 章 数据库系统性能优化案例

第 34 章 数据库系统性能优化案例

第 35 章 数据库系统性能优化案例

第 36 章 数据库系统性能优化案例

第 37 章 数据库系统性能优化案例

第 38 章 数据库系统性能优化案例

第 39 章 数据库系统性能优化案例

第 40 章 数据库系统性能优化案例

第 41 章 数据库系统性能优化案例

第 42 章 数据库系统性能优化案例

第 43 章 数据库系统性能优化案例

第 44 章 数据库系统性能优化案例

第 45 章 数据库系统性能优化案例

第 46 章 数据库系统性能优化案例

第 47 章 数据库系统性能优化案例

第 48 章 数据库系统性能优化案例

第 49 章 数据库系统性能优化案例

第 50 章 数据库系统性能优化案例

第 51 章 数据库系统性能优化案例

第 52 章 数据库系统性能优化案例

第 53 章 数据库系统性能优化案例

第 54 章 数据库系统性能优化案例

第 55 章 数据库系统性能优化案例

第 56 章 数据库系统性能优化案例

第 57 章 数据库系统性能优化案例

第 58 章 数据库系统性能优化案例

第 59 章 数据库系统性能优化案例

第 60 章 数据库系统性能优化案例

第 61 章 数据库系统性能优化案例

第 62 章 数据库系统性能优化案例

第 63 章 数据库系统性能优化案例

第 64 章 数据库系统性能优化案例

第 65 章 数据库系统性能优化案例

第 66 章 数据库系统性能优化案例

第 67 章 数据库系统性能优化案例

第 68 章 数据库系统性能优化案例

第 69 章 数据库系统性能优化案例

第 70 章 数据库系统性能优化案例

第 71 章 数据库系统性能优化案例

第 72 章 数据库系统性能优化案例

第 73 章 数据库系统性能优化案例

第 74 章 数据库系统性能优化案例

第 75 章 数据库系统性能优化案例

第 76 章 数据库系统性能优化案例

第 77 章 数据库系统性能优化案例

第 78 章 数据库系统性能优化案例

第 79 章 数据库系统性能优化案例

第 80 章 数据库系统性能优化案例

第 81 章 数据库系统性能优化案例

第 82 章 数据库系统性能优化案例

第 83 章 数据库系统性能优化案例

第 84 章 数据库系统性能优化案例

第 85 章 数据库系统性能优化案例

第 86 章 数据库系统性能优化案例

第 87 章 数据库系统性能优化案例

第 88 章 数据库系统性能优化案例

第 89 章 数据库系统性能优化案例

第 90 章 数据库系统性能优化案例

第 91 章 数据库系统性能优化案例

第 92 章 数据库系统性能优化案例

第 93 章 数据库系统性能优化案例

第 94 章 数据库系统性能优化案例

第 95 章 数据库系统性能优化案例

第 96 章 数据库系统性能优化案例

第 97 章 数据库系统性能优化案例

第 98 章 数据库系统性能优化案例

第 99 章 数据库系统性能优化案例

第 100 章 数据库系统性能优化案例

第1章 绪 论

1.1 引言

1.1.1 为什么要学习数据结构

“数据结构”是计算机相关事业的一门专业基础课,它可以为后续专业课程的学习提供必要的知识和技能准备,是十分重要的核心课程。

计算机相关事业的很多后续课程都要用到本书所涉及的知识和技能,例如,编译原理要使用堆栈、散列表及语法树,操作系统会用到队列、存储管理表及目录树,数据库系统将运用线性表、多链表以及索引树等基本数据结构及其相关的算法。本课程讨论的其他一些数据结构,如广义表、集合以及图的知识也是很多领域经常涉及的。所有的计算机系统软件和应用软件都要用到各种类型的数据结构。因此,要想更好地运用计算机来解决实际问题,仅掌握几种计算机程序设计语言是难以应付众多复杂的课题的,要想有效地使用计算机、充分发挥计算机的性能,还必须学习和掌握数据结构的有关知识。

数据结构作为一门独立的课程,在国外是从 1968 年才开始的,但在此之前,其有关内容已散见于编译原理及操作系统之中。20 世纪 60 年代中期,美国的一些大学开始设立有关课程,但当时的课程名称并不叫数据结构。1968 年美国唐·欧·克努特教授开创了数据结构的最初体系,他所著的《计算机程序设计技巧》第一卷《基本算法》是第一本较系统地阐述数据的逻辑结构和存储结构及其操作的著作。从 20 世纪 60 年代末到 70 年代初,出现了大型程序,软件也相对独立,结构程序设计成为程序设计方法学的主要内容,人们越来越重视数据结构。从 70 年代中期到 80 年代,各种版本的数据结构著作相继出现。目前,数据结构的发展并未终结,一方面,面向各专业领域中特殊问题的数据结构得到研究和发展,如多维图形数据结构等;另一方面,从抽象数据类型和面向对象的观点来讨论数据结构已成为一种新的趋势,越来越被人们重视。

1.1.2 数据结构课程的主要内容

数据结构与数学、计算机硬件和软件有十分密切的关系。数据结构是介于数学与计算机硬件和软件之间的一门计算机科学与技术专业的核心课程,是高级程序设计语言、编译原理、操作系统、数据库及人工智能等课程的基础。同时,数据结构技术也广泛应用于信息科学、系统工程、应用数学以及各种工程技术领域。

数据结构课程的教学目的是介绍一些最常用的数据结构,阐明数据结构内在的逻辑关系,讨论它们在计算机中的存储表示,并结合各种数据结构,讨论它们各种运算(操作)的实现算法。本课程集中讨论软件开发过程中的设计阶段,同时涉及编码和分析阶段的若干基本问题。此外,为了构造出好的数据结构及其实现,还需考虑数据结构及其实现的评价与选择。因此,数据结构的内容包括 3 个层次的 5 个“要素”,如表 1-1 所示。

表 1-1 数据结构课程内容体系

方面 层次	数据表示	数据处理
抽象	逻辑结构	基本运算
实现	存储结构	算法
评价	不同数据结构的比较及算法分析	

数据结构的核心技术是分解与抽象。通过对问题的抽象,舍弃数据元素的具体内容,就得到逻辑结构。类似地,通过分解将处理要求划分成各种功能,再通过抽象舍弃实现细节,就得到运算的定义。上述两个方面的结合将问题变换为数据结构。这是一个从具体(即具体问题)到抽象(即数据结构)的过程。然后,通过增加对实现细节的考虑进一步得到存储结构和实现运算,从而完成设计任务。这是一个从抽象(即数据结构)到具体(即具体实现)的过程。熟练地掌握这两个过程是数据结构课程在专业技能培养方面的基本目标。

1.2 基本概念和常用术语

1.2.1 基本概念和术语

数据(Data)是对信息的一种符号表示,是人们利用文字符号、数字符号以及其他规定的符号对现实世界的事物及其活动所做的抽象描述。因此,一个文档、记录、数组、句子、单词以及算式等统称为数据。在计算机科学中,人们把所有能输入到计算机中并被计算机程序处理的信息都叫做数据,包括文字、表格和图像等。

例如,一个学生管理程序所要处理的数据可能是一张表格,如表 1-2 所示。

表 1-2 学生信息查询表

序 号	学 号	姓 名	性 别	专 业	出 生 日 期
1	200802056	刘文杰	男	软件工程	1990.1
2	200803011	白萍	女	信息工程	1989.3
3	200701055	刘丽	女	计算机科学与技术	1990.5
4	200802033	李莎莎	女	软件工程	1990.10
5	200701028	徐涛	男	计算机科学与技术	1989.6
6	200801065	田磊	男	计算机科学与技术	1989.2

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。在不同的条件下,数据元素又可称为元素、结点、顶点、记录等。例如,在表 1-2 所示的学生信息查询表中,为了便于处理,把其中的每一行(代表一名学生)作为一个基本单位来考虑,故该数据由 6 个数据元素构成。一个数据元素又可由若干个数据项组成,数据项(Data Item)是数据的不可分割的最小单位。对于表 1-2 所示的学生信息查询表,每个数据元素由序号、学号、姓名、性别、专业和出生日期等数据项组成。

数据对象(Data Object)是性质相同的数据元素的集合,是数据的一个子集。在某个具体问题中,数据元素具有相同的性质(元素值不一定相等),是数据对象集合中的数据成员。例如,表 1-2 所示的学生信息查询表就是一个数据对象。整数集合和复数集合都是数据对象。

1.2.2 数据结构

数据结构(Data Structure)是指互相之间存在一种或多种关系的数据元素的集合。数据的描述对象是现实世界的事物及其活动,而任何事物及其活动都不是孤立存在的,在一定意义上都是相互联系、相互影响的,所以数据之间必然存在着联系。数据之间的相互联系称为数据的逻辑结构。在计算机中存储数据时,不仅要存储数据本身,而且要存储它们之间的联系(即逻辑结构)。数据结构在存储器中的存储方式称为数据的物理结构或存储结构。由于存储方式有顺序、链接、索引和散列等多种形式,所以一种数据结构可以根据应用的需要表示成任一种或几种存储结构。每种数据结构都有一个运算的集合,例如最常见的运算有检索、插入和排序等,这些运算在数据的逻辑结构上定义,只规定“做什么”,而在数据的存储结构上考虑运算的具体实现,规定“如何做”。

综上所述,按某种逻辑关系组织起来的一批数据,按一定的存储方式存储在计算机的存储器中,并在这些数据上定义一个运算的集合,就称为一个数据结构。

因此,对于数据结构,一般研究数据的逻辑结构、数据的存储结构和数据的运算 3 个方面。

1. 逻辑结构

数据的逻辑结构包含两个要素:一个是数据元素的集合,另一个是关系的集合。在形式上,数据的逻辑结构通常可以采用一个二元组来表示:

$$\text{Data_Structure} = (D, R)$$

Data_Structure 是一种数据结构,它由数据元素的集合 D 和 D 上二元关系的集合 R 所组成:

$$D = \{d_i \mid 1 \leq i \leq n, n \geq 0\}$$

$$R = \{r_j \mid 1 \leq j \leq m, m \geq 0\}$$

其中 d_i 表示集合 D 中的第 i 个数据元素, n 为 D 中数据元素的个数,特别地,若 $n=0$,则 D 是一个空集,因而 Data_Structure 也就无结构而言; r_j 表示集合 R 中的第 j 个二元关系(下面简称关系), m 为 R 关系的个数,特别地,若 $m=0$,则 R 是一个空集,表明集合里的元素之间不存在任何关系,彼此是独立的。在本书所讨论的数据结构中,一般只讨论 $m=1$ 的情况,即 R 中只包含一个关系($R=\{r\}$)的情况。对于 R 中包含多个关系的情况,

可以用类似的方法进行讨论。

在一个数据结构中,每个数据元素可称为一个结点,数据结构中所包含的数据元素之间的关系就是结点之间的关系。

D 上的关系 r 是序偶的集合。对于 r 中的任一序偶 $\langle x, y \rangle (x, y \in D)$, 我们把 x 叫序偶的第一元素, 把 y 叫序偶的第二元素, 又称 x 为 y 的直接前驱(简称前驱), 称 y 为 x 的直接后继(简称后继)。 x 和 y 互为相邻结点; 如果 x 没有前驱, 则称 x 为开始结点。如果 y 没有后继, 则称 y 为终端结点; 如果 x 既不是开始结点, 也不是终端结点, 则称 x 为中间结点。

根据数据元素间关系的不同特性,数据的逻辑结构通常分为以下 4 类:

(1) 集合。在集合中,数据元素间的关系是“属于同一个集合”。集合是元素关系极为松散的一种结构。

(2) 线性结构。该结构的数据元素之间存在一对一的关系。除了开始结点和终端结点外,任何一个结点都有一个唯一的前驱和一个唯一的后继。

(3) 树形结构。该结构的数据元素之间存在一对多的关系。除了树根结点,任何一个结点最多有一个前驱,可以有多个后继。树形结构是典型的非线性结构。

(4) 图形结构。该结构的数据元素之间存在多对多的关系。这种结构的特征是一个元素可以有多个前驱,也可以有多个后继,是一种多对多的前驱后继关系。图形结构也称作网状结构。

图 1-1 给出了上述 4 类逻辑结构的示意图。

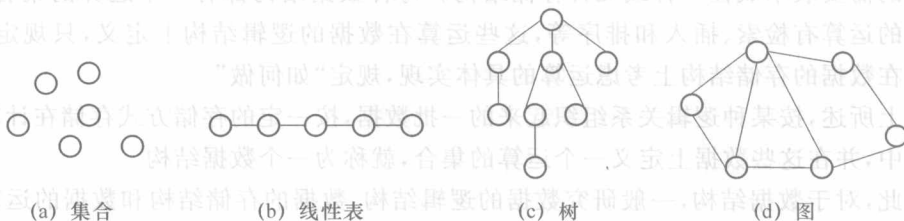


图 1-1 基本的逻辑结构

数据结构还能够利用图形形象地表示出来,图形中的每个结点对应着一个数据元素,两结点之间带箭头的连线(称作有向边或弧)对应着关系中的一个序偶,其中序偶的第一元素为有向边的起始结点,第二元素为有向边的终止结点,即箭头所指向的结点。

【例 1-1】 某数据结构 $L=(D, R)$, 其中

$$D=\{1,2,3,4,5,6\}$$

$$R=\{r\}$$

$$r=\{\langle 1,2 \rangle, \langle 2,3 \rangle, \langle 3,4 \rangle, \langle 4,5 \rangle, \langle 5,6 \rangle\}$$

试画出对应的逻辑结构图,说明它是何种数据结构,并给出哪些是开始结点,哪些是终端结点。

【解】 L 对应的逻辑结构图如图 1-2 所示。



图 1-2 数据的线性结构示意图

在图 1-2 中,1 是开始结点,它没有前驱,其余每个结点有且仅有一个直接前驱;6 是终端结点,它没有后继,其余每个结点有且仅有一个直接后继;2、3、4、5 是中间结点,因此图 1-2 所示的是线性结构。这种数据结构的特点是数据元素之间的 1 对 1(1:1)联系,即线性关系。

表 1-2 表示的是线性结构,表中有 6 个数据元素,且每个数据元素排列位置有先后次序,所以在表中会按序号形成一种次序关系,即整个二维表就是学生数据的一个线性序列。诸如此类的还有电话自动查号系统、考试查分系统和仓库库存管理系统等。在这类文档管理的数学模型中,一般用表格表示数据和数据之间的关系,整张表格形成一个线性序列。

【例 1-2】 某数据结构 $T=(D, R)$, 其中

$D=\{1,2,3,4,5,6,7,8\}$

$R=\{r\}$

$r=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 1,4\rangle,\langle 4,5\rangle,\langle 4,6\rangle,\langle 5,7\rangle,\langle 1,8\rangle\}$

试画出对应的逻辑结构图,说明它是何种数据结构,并给出哪些是开始结点,哪些是终端结点。

【解】 T 对应的逻辑结构图如图 1-3 所示。

在图 1-3 中,1 是开始结点,它没有前驱,其余每个结点有且仅有一个直接前驱;3、7、6、8 是终端结点,它们没有后继,其余每个结点可以有多个直接后继;2、4、5 是中间结点,因此图 1-3 所示的是树形结构。这种数据结构的特点是数据元素之间的 1 对 $N(1:N)$ 联系($N \geq 0$),即层性关系。

这是不同于线性结构的另一种类型的数据结构。在计算机系统中,描述磁盘目录和文件结构时,如图 1-4 所示,某磁盘 E: 盘包括一个根目录(root)和若干个一级子目录,如电子图书、电子教案和自编教材等,每个一级子目录中又包含若干个二级子目录,如 C++ 程序设计、数据结构和计算机网络等。这种关系很像自然界中的树,所以称为目录树。

在这种结构中,目录和目录以及目录和文件之间不再是前面所列举的那种一一对应关系,而呈现出一对多的非线性关系,即根目录有多个子目录(也称为孩子),每个孩子又有属于自己的孩子,而任一子目录或文件都只有一个唯一的上级(也称为双亲)。这种数学模型称为树形数据结构。

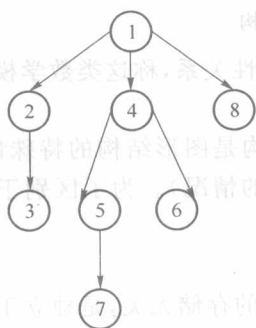


图 1-3 数据的树形结构示意图

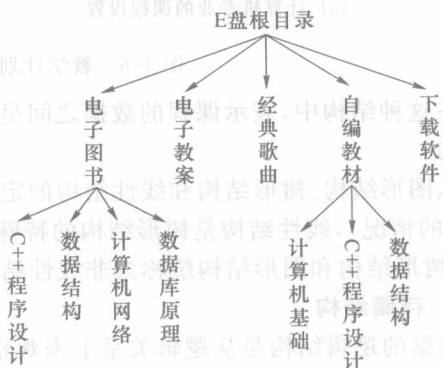


图 1-4 磁盘目录和文件的树形结构示意图

【例 1-3】 某数据结构 $G=(D, R)$, 其中
 $D=\{1,2,3,4,5,6\}$
 $R=\{r\}$
 $r=\{\langle 1,2\rangle,\langle 3,4\rangle,\langle 4,5\rangle,\langle 5,6\rangle,\langle 1,6\rangle,\langle 3,2\rangle,\langle 4,1\rangle,\langle 1,5\rangle\}$
试画出对应的逻辑结构图,并说明它是何种数据结构。

【解】 G 对应的逻辑结构图如图 1-5 所示。

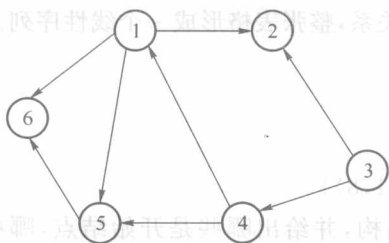


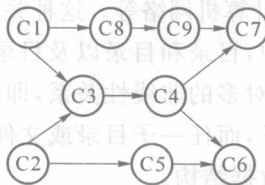
图 1-5 数据的图形结构示意图

从图 1-5 可以看出,每个结点均可以有多个前驱和后继,结点之间的联系是 M 对 N ($M:N$) 联系 ($M\geq 0, N\geq 0$), 即网状关系。也就是说,每个结点可以有任意多个直接前驱和任意多个直接后继。因此图 1-5 所示的是图型结构。

在图 1-6(a) 所示的教学计划中,包括许多课程,在这许多课程之间,有些必须按规定的先后次序排课,有些则没有次序要求。如一门课程可能以一些先修课程为基础,而它本身又可能成为另一些课程的先修课程。即这些课程之间存在先修与后续的关系,任意一门课可以和其他多门课之间存在这种先修与后续的关系。各门课程之间的这种次序关系可用一个顶点和表示先后课关系的有向边组成的图来表示。图 1-6(b) 即为这些课程和它们之间关系的表示,图中的顶点表示课程,有向边表示课程之间先修和后续的关系。

课程编号	课程名称	先修课程
C1	高等数学	无
C2	程序设计基础	无
C3	离散数学	C1, C2
C4	数据结构	C3, C5
C5	算法语言	C2
C6	编译技术	C4, C5
C7	操作系统	C4, C9
C8	普通物理	C1
C9	计算机原理	C8

(a) 计算机专业的课程设置



(b) 表示课程之间先修与后续关系的有向图

图 1-6 教学计划编排问题的数据结构

在这种结构中,表示课程的数据之间呈现多对多的非线性关系,称这类数学模型为图型结构。

从图形结构、树形结构和线性结构的定义可知,树形结构是图形结构的特殊情况(即 $M=1$ 的情况),线性结构是树形结构的特殊情况(即 $N=1$ 的情况)。为了区别于线性结构,把树形结构和图形结构统称为非线性结构。

2. 存储结构

数据的逻辑结构是从逻辑关系上来观察数据,它与数据的存储无关,是独立于计算机的。数据的存储结构是逻辑结构在计算机存储器里的实现,它是依赖于计算机的。

计算机的存储器(主存)是由有限多个存储单元组成,每个存储单元有唯一的地址,各存储单元的地址是连续编码的,也就是说每个存储单元都有唯一的后继单元。一片相邻

的存储单元的整体叫做存储区域。

基本的存储映像方法有 4 种,下面分别加以介绍。

(1) 顺序存储方法

顺序存储方法是把逻辑上相邻的元素存储在物理位置相邻的存储单元中,结点间的逻辑关系由存储单元的邻接关系来体现,由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法,通常借助于程序设计语言中的数组来实现。

【例 1-4】 某数据结构 $L=(D, R)$, 其中

$D=\{A, B, C, D, E\}$

$R=\{r\}$

$r=\{\langle A, B \rangle, \langle B, C \rangle, \langle C, D \rangle, \langle D, E \rangle\}$

假设每个结点占一个存储单元,第一个结点存放在地址为 1000H 的单元中,则顺序存储实现如图 1-7 所示。

1000H	A
1001H	B
1002H	C
1003H	D
1004H	E

图 1-7 顺序存储的线性结构

(2) 链式存储方法

链式存储方法对逻辑上相邻的元素不要求其物理位置相邻,元素间的逻辑关系通过附设的指针字段来表示,即将结点所占用的存储单元分为两部分,一部分存放数据元素本身的值,称数值域;另一部分存放此结点的后继结点所对应的存储单元的地址,称为指针域。由此得到的存储表示称为链式存储结构。

前面例 1-4 所示的逻辑结构可采用链式存储的方法实现如图 1-8 所示。

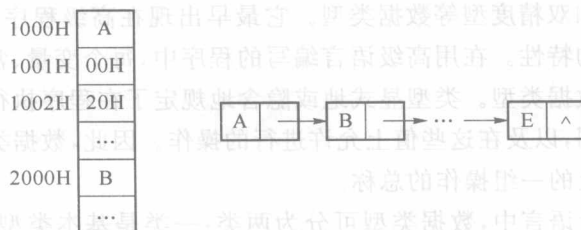


图 1-8 链式存储的线性结构

(3) 索引存储方法

索引存储方法是在存储结点信息的同时,还建立附加的索引表。索引表中的每一项包含关键字和地址,关键字是能够唯一标识一个数据元素的数据项,地址指示出数据元素所在的存储位置。索引存储主要是针对数据内容的存储,而不强调关系的存储。索引存储方法主要面向查找操作。

(4) 散列存储方法

散列存储方法是以数据元素的关键字的值为自变量,通过某个函数(散列函数)计算出该元素的存储位置。散列存储也是针对数据内容的存储方式。

以上 4 种存储方法中,顺序存储方法和链式存储方法是最基本最常用的,索引存储方法和散列存储方法在具体实现时需要用到前两种方法。存储结构是数据结构的 3 个方面之一,若逻辑结构相同但存储结构不同,则为不同的数据结构,有时这种不同是相当大的。