

高·等·学·校·计·算·机·教·材

数据结构

实用教程

(C语言版)

郑阿奇 主编/王琼 审稿



Data Structure



电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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



数据结构实用教程

(C 语言版)

郑阿奇 主 编
王 琼 审 稿

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书系统介绍线性表、栈和队列、串、数组和广义表、树和二叉树、图、排序、查找等内容，以 C 语言作为数据结构和算法的描述语言。一般先通过应用实例引入，在介绍数据类型基本操作后，通过综合应用实例进行应用和消化。所有的综合应用实例在 Visual C++ 6.0 环境下调试并且运行通过。通过本书的学习，不但比较容易理解数据结构的主要内容，而且应用数据结构知识解决了若干小应用问题，大大提高了学生解决问题的能力。使用本书作为“数据结构”课程的教材，教师容易教，学生容易学。

本书可作为计算机专业本科、高职高专“数据结构”课程的教材，也可作为其他有关专业“数据结构”选修课的教材或参考书。

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

数据结构实用教程：C 语言版 / 郑阿奇主编. —北京：电子工业出版社，2011.7

高等学校计算机教材

ISBN 978-7-121-14036-5

I. ①数… II. ①郑… III. ①数据结构—高等学校—教材②C 语言—程序设计—高等学校—教材
IV. ①TP311.12②TP312

中国版本图书馆 CIP 数据核字 (2011) 第 132402 号

责任编辑：郝黎明 特约编辑：田领红

印 刷：三河市鑫金马印装有限公司
装 订：

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：19.25 字数：492 千字

印 次：2011 年 7 月第 1 次印刷

印 数：3000 册 定价：36.00 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，联系及邮购电话：(010) 88254888。

质量投诉请发邮件至 zltz@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

目前,“数据结构”不仅是计算机专业核心课程,其他许多理工类专业也把它作为选修课。但是,“数据结构”课程难学和难懂始终困扰着老师和学生们。通过长时间的探索和思考,我们认为,把数据结构主要内容的介绍与实际应用结合起来可以更容易理解相关的知识,也把学习数据结构的目的是落到了实处。

本书主要包括下列特点。

(1) 绪论从几个典型实例出发,回答“数据结构”研究的是什么内容,主要用于解决什么问题。在这个基础上介绍数据结构的基本概念。抽象数据类型从高级语言的数据类型出发,通过类比把数据类型变得不抽象。数据的逻辑结构和存储结构、算法的问题在后面章节与之相呼应。

(2) 线性表首先介绍数据结构,由于对 $(a_1, a_2, \dots, a_n)$ 线性表的表示太抽象,能够让学生理解这个表示就显得很重要。我们从扑克牌入手理解 $(a_1, a_2, \dots, a_n)$,在这个基础上介绍线性表的定义和抽象数据类型。然后从顺序存储结构和链式存储结构分别介绍抽象数据类型及其基本操作,通过扑克牌游戏和学生信息管理系统理解线性表数据类型及其应用方法。

(3) 从栈和队列开始主要介绍数据类型及其基本操作,然后通过综合应用实例理解和消化这些知识。栈和队列的综合应用实例是“表达式求值”、“老鼠钻迷宫”、“打印文档”和“火车车厢重排”;“串”的综合应用实例是“简易的记事本”;“广义表”的综合应用实例是“文件目录结构”;“树及二叉树”的综合应用实例是“算术表达式的计算”和“电报编码”;“图”的综合应用实例是“高速公路行驶最佳方案”和“计算机课程学习的优先顺序”。排序和查找仍然从理解操作数据入手,算法描述尽可能图形化。

(4) 综合应用实例中的基本操作尽可能采用相应抽象数据类型的基本操作,这样不但减少了综合应用实例的代码重复,而且更好地理解数据类型的基本操作。所有的数据类型和基本操作代码均采用C语言描述,所有的综合应用实例在Visual C++ 6.0环境下调试、运行通过。

本书由南京师范大学郑阿奇主编,南京师范大学王琮审稿。参加本书编写的还有梁敬东、顾韵华、王洪元、刘启芬、丁有和、曹弋、徐文胜、殷红先、张为民、姜乃松、彭作民、王一莉、徐斌、王志瑞、周怡明、刘博宇、周怡君、郑进、刘毅等。

使用本书作为教材教师容易教,学生容易学。本书配有教学课件和所有综合应用实例源文件,需要者可以通过华信教育资源网免费下载。网站地址为 <http://www.hxedu.com.cn>。通过适当确认,出版社可为教师提供本书的习题参考答案。

由于编著水平有限,错误之处在所难免,敬请读者批评指正。

意见、建议邮箱: easybooks@163.com

《数据结构实用教程（C语言版）》读者意见反馈表

尊敬的读者：

感谢您购买本书。为了能为您提供更优秀的教材，请您抽出宝贵的时间，将您的意见以下表的方式（可从 <http://www.hxedu.com.cn> 下载本调查表）及时告知我们，以改进我们的服务。对采用您的意见进行修订的教材，我们将在该书的前言中进行说明并赠送您样书。

姓名：_____ 电话：_____

职业：_____ E-mail：_____

邮编：_____ 通信地址：_____

1. 您对本书的总体看法是：

☐ 很满意 ☐ 比较满意 ☐ 尚可 ☐ 不太满意 ☐ 不满意

2. 您对本书的结构（章节）：☐ 满意 ☐ 不满意 改进意见_____

3. 您对本书的例题：☐ 满意 ☐ 不满意 改进意见_____

4. 您对本书的习题：☐ 满意 ☐ 不满意 改进意见_____

5. 您对本书的实训：☐ 满意 ☐ 不满意 改进意见_____

6. 您对本书其他的改进意见：

7. 您感兴趣或希望增加的教材选题是：

请寄：100036 北京市万寿路 173 信箱 郝黎明 收

电话：010-88254565 E-mail: gaozhi@phei.com.cn

目 录

第1章 绪论	(1)
1.1 数据结构的基本概念	(1)
1.1.1 基本概念和术语	(3)
1.1.2 数据的逻辑结构	(5)
1.1.3 数据的存储结构	(6)
1.2 抽象数据类型	(7)
1.2.1 什么是抽象数据类型	(7)
1.2.2 抽象数据类型的描述	(7)
1.3 算法的定义及特征	(8)
1.3.1 算法的定义及特征	(8)
1.3.2 算法设计的要求	(9)
1.4 算法的度量及分析	(9)
1.4.1 算法效率的度量	(9)
1.4.2 时间复杂度分析	(10)
1.4.3 常见的时间复杂度	(12)
1.4.4 空间复杂度分析	(12)
习题	(13)
第2章 线性表	(15)
2.1 定义	(16)
2.1.1 什么是线性表	(16)
2.1.2 线性表的抽象数据类型	(16)
2.2 顺序线性表	(17)
2.2.1 线性表的顺序存储	(17)
2.2.2 顺序表的基本操作	(19)
2.2.3 综合应用实例——扑克牌游戏	(24)
2.3 链式线性表(一)	(29)
2.3.1 线性表的链式存储	(29)
2.3.2 单链表的基本操作	(31)
2.3.3 综合应用实例——学生信息管理系统	(36)
2.4 链式线性表(二)	(40)
2.4.1 循环链表	(40)
2.4.2 双向链表	(40)
2.4.3 综合应用实例——约瑟夫问题	(44)
2.5 顺序表与链表的比较	(48)
习题	(48)

第3章 栈和队列	(52)
3.1 栈	(52)
3.1.1 栈的定义	(52)
3.1.2 栈的抽象数据类型	(53)
3.2 顺序栈	(54)
3.2.1 顺序栈的存储结构	(54)
3.2.2 顺序栈的基本操作	(54)
3.2.3 综合应用实例——表达式求值	(57)
3.3 链栈	(62)
3.3.1 链栈的存储结构	(62)
3.3.2 链栈的基本操作	(63)
3.3.3 综合应用实例——老鼠钻迷宫	(66)
3.4 队列	(72)
3.4.1 队列的定义	(73)
3.4.2 队列的抽象数据类型	(73)
3.5 顺序队列	(74)
3.5.1 顺序队列的存储结构	(74)
3.5.2 循环队列的逻辑结构	(75)
3.5.3 循环队列的基本操作	(77)
3.5.4 综合应用实例——打印文档	(79)
3.6 链队列	(81)
3.6.1 链队列的存储结构	(81)
3.6.2 链队列的基本操作	(81)
3.6.3 综合应用实例——火车车厢重排	(85)
习题	(89)
第4章 串	(93)
4.1 串的定义	(93)
4.1.1 什么是串	(93)
4.1.2 串的抽象数据类型	(94)
4.2 串的顺序存储	(94)
4.2.1 串的顺序存储结构	(95)
4.2.2 顺序存储的基本操作	(95)
4.3 串的堆存储	(97)
4.3.1 串的堆存储结构	(97)
4.3.2 堆存储的基本操作	(97)
4.4 串的链式存储	(100)
4.4.1 串的链式存储结构	(100)
4.4.2 链式存储的基本操作	(101)
4.5 串的模式匹配	(103)
4.5.1 Brute-Force 算法	(103)

4.5.2	KMP 算法	(104)
4.5.3	综合应用实例——简易的“记事本”	(110)
习题		(114)
第 5 章	数组和广义表	(116)
5.1	数组	(116)
5.1.1	数组的定义及存储	(116)
5.1.2	数组的抽象数据类型	(118)
5.2	特殊矩阵	(118)
5.2.1	对称矩阵	(118)
5.2.2	三角矩阵	(119)
5.2.3	对角矩阵	(120)
5.3	稀疏矩阵	(121)
5.3.1	稀疏矩阵的存储结构	(121)
5.3.2	稀疏矩阵的基本操作	(122)
5.3.3	稀疏矩阵的十字链表表示	(126)
5.4	广义表	(129)
5.4.1	广义表的定义	(129)
5.4.2	广义表的抽象数据类型	(131)
5.4.3	广义表的存储结构	(131)
5.4.4	广义表的基本操作	(132)
5.4.5	综合应用实例——文件目录结构	(136)
习题		(139)
第 6 章	树及二叉树	(142)
6.1	树及其表示	(143)
6.1.1	树的定义和术语	(143)
6.1.2	树的逻辑表示	(145)
6.1.3	树的抽象数据类型	(146)
6.1.4	树的存储结构	(146)
6.2	二叉树	(149)
6.2.1	二叉树的定义	(149)
6.2.2	二叉树的性质	(150)
6.2.3	二叉树的抽象数据类型	(151)
6.2.4	二叉树的存储结构	(152)
6.2.5	二叉树的基本操作	(154)
6.2.6	二叉树的遍历	(157)
6.2.7	综合应用实例——算术表达式的计算	(158)
6.3	二叉树的线索化	(162)
6.3.1	线索二叉树的定义	(163)
6.3.2	中序线索化二叉树	(163)
6.3.3	线索二叉树的操作	(165)

6.4	树、森林与二叉树的转换	(167)
6.4.1	树与二叉树的相互转化	(167)
6.4.2	森林与二叉树的相互转换	(168)
6.5	哈夫曼树及哈夫曼编码	(170)
6.5.1	哈夫曼树	(170)
6.5.2	哈夫曼编码	(173)
6.5.3	综合应用实例——电报编码	(176)
	习题	(179)
第7章	图	(184)
7.1	定义	(184)
7.1.1	什么是图	(184)
7.1.2	图的常用术语	(185)
7.1.3	图的抽象数据类型	(188)
7.2	图的存储结构	(189)
7.2.1	邻接矩阵	(189)
7.2.2	邻接表	(191)
7.2.3	十字链表	(194)
7.3	图的遍历	(195)
7.3.1	深度优先搜索	(195)
7.3.2	广度优先搜索	(197)
7.4	最小生成树	(199)
7.4.1	普里姆算法	(200)
7.4.2	克鲁斯卡尔算法	(202)
7.5	最短路径	(204)
7.5.1	从某个源点到其余各顶点的最短路径	(205)
7.5.2	每一对顶点之间的最短路径	(207)
7.5.3	综合应用实例——高速公路行驶最佳方案	(209)
7.6	拓扑排序和关键路径	(212)
7.6.1	拓扑排序	(213)
7.6.2	关键路径	(215)
7.6.3	综合应用实例——计算机课程学习的优先顺序	(219)
	习题	(221)
第8章	排序	(227)
8.1	定义	(228)
8.1.1	什么是排序	(228)
8.1.2	排序算法的分类	(229)
8.2	插入排序	(229)
8.2.1	直接插入排序	(230)
8.2.2	二分插入排序	(231)
8.2.3	希尔排序	(232)

8.3	交换排序	(233)
8.3.1	冒泡排序	(233)
8.3.2	快速排序	(236)
8.4	选择排序	(238)
8.4.1	直接选择排序	(238)
8.4.2	堆排序	(240)
8.5	归并排序	(244)
8.6	基数排序	(245)
8.7	几种排序算法的比较	(247)
	习题	(248)
第9章	查找	(250)
9.1	定义	(250)
9.1.1	什么是查找	(250)
9.1.2	查找的性能分析	(251)
9.2	静态查找	(252)
9.2.1	顺序表的查找	(252)
9.2.2	有序表的查找	(253)
9.2.3	索引顺序表的查找	(255)
9.3	动态查找	(256)
9.3.1	二叉排序树	(257)
9.3.2	平衡二叉树	(262)
9.3.3	B树和B ⁺ 树	(265)
9.4	哈希表	(269)
9.4.1	定义	(270)
9.4.2	哈希函数的构造方法	(271)
9.4.3	处理冲突的方法	(273)
9.4.4	哈希表的基本操作	(275)
	习题	(277)
附录A	认识 Visual C++ 6.0 中文版开发环境	(280)
附录B	程序组织、预处理和调试	(290)
B.1	设置断点	(290)
B.2	控制程序运行	(292)
B.3	查看变量或数组的内容	(293)

在问世的初期，计算机主要用于科学计算，即处理数值数据。随着计算机技术的飞速发展，计算机已深入到人类社会的各个领域。计算机应用的范围不断扩大，更多地用于控制、管理和数据处理等非数值计算的处理工作。计算机加工的对象扩展到字符、表格、图形图像等各种数据。为了通过编程处理这些数据，就需要研究非数值计算程序设计问题中计算机的操作对象、对象之间的关系和操作方法，这也是“数据结构”这门学科研究的内容。

1.1 数据结构的基本概念

为了更好地了解非数值计算中计算机处理的对象及计算机对它们进行的操作，下面先看几个例子。

【例 1.1】 学生成绩表。假设某班有若干名学生，现在要通过计算机来管理这些学生的成绩信息，包括学生的学号、姓名、成绩和总学分，其中每个学生通过学号来进行标识。这些信息如表 1.1 所示。

表 1.1 学生成绩表

学 号	姓 名	成 绩	总 学 分
101101	李林	89	13
101102	李枫兰	97	14
101103	蓝军	78	12
101104	柳中宝	56	10
101105	张一红	61	13
101106	赵红	82	13
101107	黄日申	88	14
101108	金平	50	11
101119	施善中	86	12
101110	胡民新	75	12

从表 1.1 中可以看出,学生成绩表由 10 行组成。其中每行的数据是表的数据元素,描述的是一个学生的基本信息。这些数据元素之间有先后次序,它们形成一种简单的线性关系,称为“线性表”。对这个线性表的操作包括向表中增加一个学生、从表中删除一个学生、按照学号或者姓名查找学生、显示表中所有学生数据等。

在日常生活中,需要计算机进行管理的工作很多,如图书管理、产品管理、设备管理、合同管理等,它们的共同特点就是处理数据元素之间存在的线性关系。

有些线性表很简单,数据元素只有一两个简单数据,例如, (“2”、“A”、“K”、“Q”、“Q”、“10”、“9”、“6”) 线性表表示某人手上的扑克牌,对它们的操作与前面的表有许多共同之处。因此,只要研究一般线性表的存储和操作,就可以解决这一类的问题。

【例 1.2】 老鼠钻迷宫。把老鼠放在迷宫的入口,把食物放在迷宫的出口。如果老鼠想吃到食物,就必须找到一条从入口到出口的路径,如图 1.1 所示。

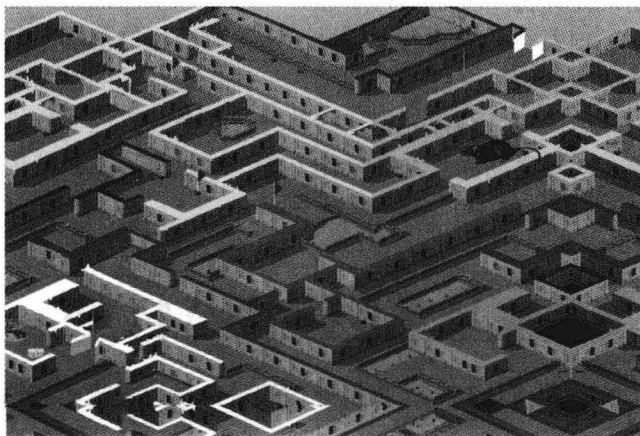


图 1.1 老鼠钻迷宫

老鼠怎样才能找到从入口到出口的路径呢?通过观察知道,老鼠从入口出发,顺着某一方向向前探索。如果能走通,则继续往前走;否则沿原路退回,从另一个方向再继续探索,直到找到一条通路为止。

如果要编写程序,用计算机来模仿老鼠钻迷宫,必须要解决迷宫的表达和老鼠钻迷宫的过程,最后得到老鼠钻迷宫的路径。迷宫可以通过二维数组来描述,而记录老鼠钻迷宫的过程和路径需要用到一种称做“栈”的数据结构。

【例 1.3】 计算机中文件目录的逻辑结构。在 Windows 操作系统中,计算机的硬盘可以分成 C 盘、D 盘、E 盘等逻辑盘。其中每一个逻辑盘的文件夹和文件形成了目录文件结构,如图 1.2 所示。

计算机磁盘上的目录结构可以看成是一棵树。在图 1.2 中,“本地磁盘 (D:)”节点是树根,其下还包含“Student”、“Data”、“资料.tar”等多个子节点。而在其中的某些子节点下又分别包含若干个节点,以此类推。因此,树中节点之间的关系不再是顺序的,而是分层、分叉的一对多的非线性结构,称为“树形结构”。

对于图 1.2 所示的树结构,可以对它进行相应的操作,其中包括增加一个目录或者文件、删除一个目录或者文件、显示当前目录下有哪些目录或者文件、显示整个树包含的目录和文件等。

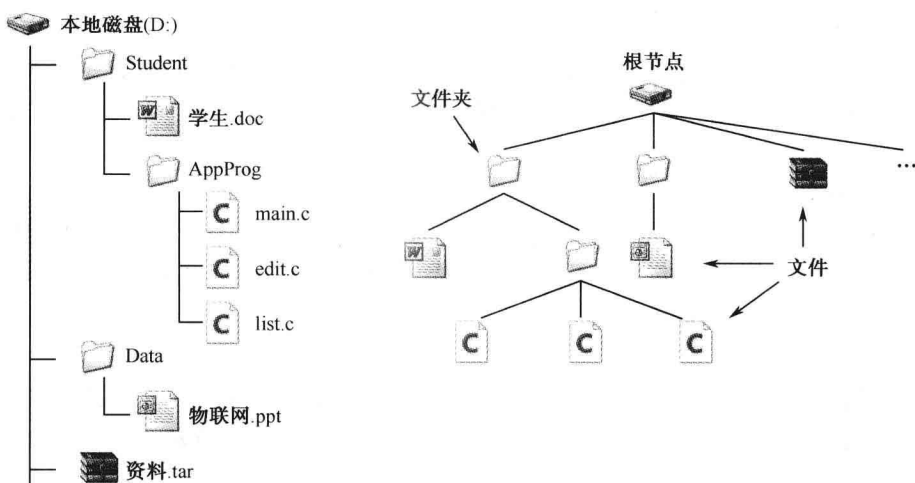


图 1.2 计算机磁盘上的目录结构

【例 1.4】 高速公路路线图。连接各大城市的高速公路可以通过图的结构来进行描述，其中城市为图中的节点。若两个城市间有高速公路，那么对应的节点之间就有连线和一个描述距离长短的值，如图 1.3 所示。

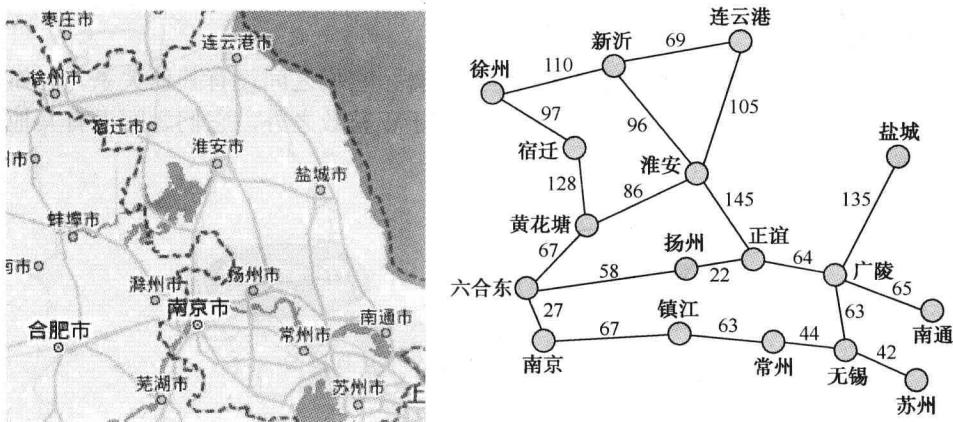


图 1.3 用图描述的高速公路

对这种图形结构进行的操作包括计算以某个城市为起点，到其他所有城市的最短路径。

综上 4 个例子可见，描述这类非数值计算问题的数学模型不再是数学方程，而是诸如表、栈、树和图等。简单说来，数据结构就是研究诸如此类非数值计算问题中数据对象、关系和对它们的操作。

1.1.1 基本概念和术语

1. 数据

数据是用来描述现实世界的数字、字符、图像、声音，以及能够输入到计算机中并能被计算机处理的符号集合。

2. 数据元素

数据元素是数据的基本单位, 在计算机程序中通常作为一个整体进行考虑和处理, 通常也称为“元素”、“记录”、“节点”等。例如, 在学生基本信息表中, 每个学生的信息作为一个记录, 也就是一个数据元素。

3. 数据对象

数据对象是具有相同性质的数据元素的集合, 是数据的一个子集。例如, 集合{“数据结构”, “C 语言”, “操作系统”}是课程数据对象, 集合{“2”, “A”, “K”, “Q”, “Q”, “10”, “9”, “6”}是扑克牌数据对象。

4. 数据结构

数据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。在需要解决的问题中, 数据元素都不是孤立存在的。在它们之间存在着某种关系, 这种数据元素相互之间的关系称为“结构”。

1.1.2 数据的逻辑结构

数据的逻辑结构是指数据对象中数据元素之间的相互关系, 它可以分为集合结构、线性结构、树形结构和图状结构, 而后 3 种逻辑结构是数据结构研究的主要内容。

1. 集合结构

集合结构中的数据元素除了同属于一个集合外, 它们之间没有其他关系。例如, 对于集合{13, 45, 77, 29, 58}而言, 其中的数除了都属于正整数之外, 它们之间没有其他关系, 如图 1.4 所示。

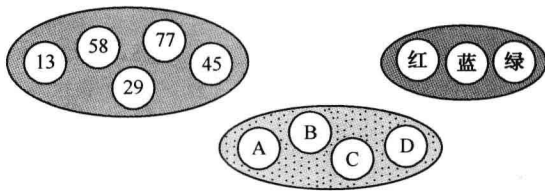


图 1.4 集合结构示意图

2. 线性结构

线性结构中的数据元素之间存在“一对一”的关系。若结构为非空集, 则除了第 1 个数据元素和最后一个数据元素以外, 其他每个数据元素都只有一个直接前驱和一个直接后继。直接前驱是指与该数据元素相邻的前一个数据元素; 直接后继是指与该数据元素相邻的后一个数据元素, 如图 1.5 所示。

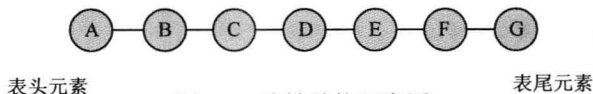


图 1.5 线性结构示意图

3. 树形结构

树形结构中的数据元素之间存在“一对多”的关系。若结构为非空集, 则除了第 1

个数据元素以外，其他每个数据元素都只有一个直接前驱，以及零个或多个直接后继，如图 1.6 所示。

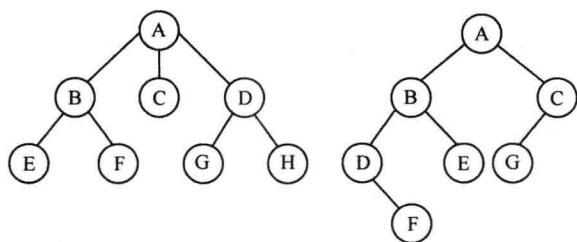


图 1.6 树形结构示意图

4. 图状结构

图状结构中的数据元素之间存在“多对多”的关系。若结构为非空集，则每个数据元素可有多（零）个直接前驱和多个（零）个直接后继，如图 1.7 所示。

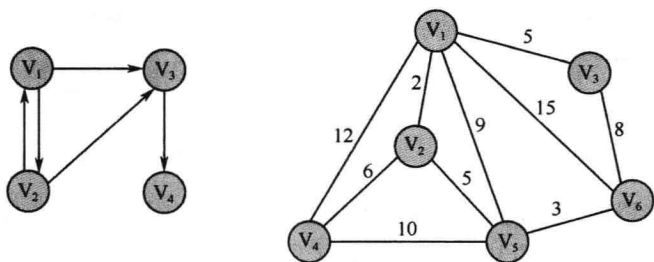


图 1.7 图状结构示意图

按照数据的逻辑结构，一般将数据结构分为线性结构和非线性结构。常用的数据结构及其分类如图 1.8 所示。

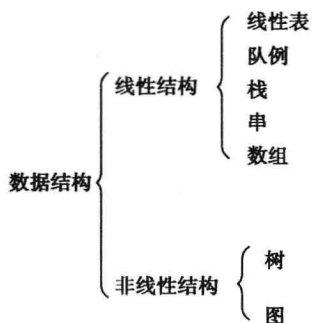


图 1.8 常用的数据结构及其分类

按照数据的存储结构，可以将数据结构分为顺序存储结构、链式存储结构、索引存储结构和散列存储结构。

1.1.3 数据的存储结构

数据的存储结构就是数据元素及其相互关系的存储方式。要想用计算机处理数据，

就必须把数据的逻辑结构映射为数据的存储结构。逻辑结构可以映射为以下 4 种存储结构。

(1) 顺序存储结构。把逻辑上相邻的数据元素存储在物理位置也相邻的存储单元中, 借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系, 如图 1.9 所示。

(2) 链式存储结构。借助指针表达数据元素之间的逻辑关系, 不要求逻辑上相邻的数据元素在物理位置上也相邻, 如图 1.10 所示。

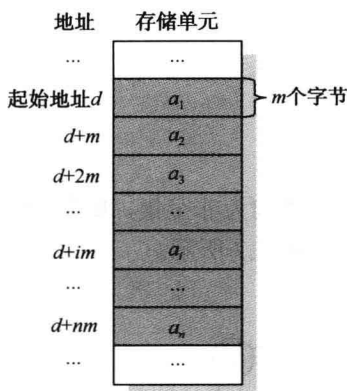


图 1.9 顺序存储结构

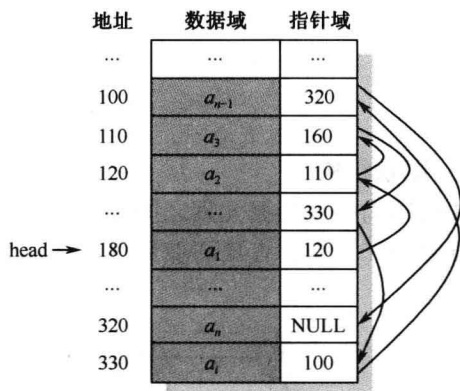


图 1.10 链式存储结构

(3) 索引存储结构。在存储数据元素的同时, 还建立附加的索引表。通过索引表, 可以快速地找到存储数据元素的节点, 如图 1.11 所示。

(4) 散列存储结构。根据散列函数和处理冲突的方法确定数据元素的存储位置, 如图 1.12 所示。

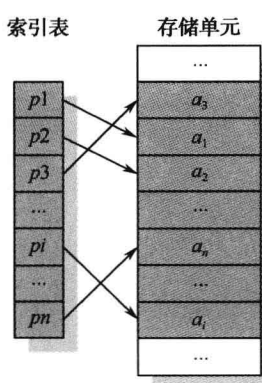


图 1.11 索引存储结构

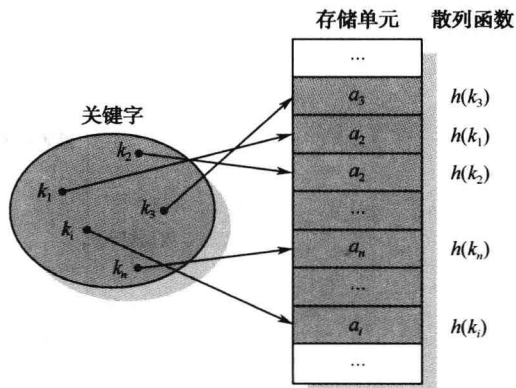


图 1.12 散列存储结构

需要注意的是, 数据的逻辑结构和数据的存储结构是指一个事物的两个方面, 而不是两个事物。二者相辅相成, 不可分割; 同时, 一种逻辑结构可以映射为多种存储结构, 具体映射为哪种存储结构, 要根据具体情况而定, 主要考虑运算方便及算法对时间空间效率的要求。

1.2 抽象数据类型

1.2.1 什么是抽象数据类型

在C语言中,系统提供了一些数据类型和基于每一个数据类型的运算(操作)。例如,系统定义的整数数据类型用int表示,在计算机中以16位定点有符号二进制补码形式存储,它对应的操作为加(+)、减(-)、乘(*)、除(/)等。但程序员用C语言编程使用整数数据类型时,并不需要知道它在计算机内如何表示及其操作是如何实现的,而只需要关注应用整数解决具体问题。下面是计算 $ax^2+bx+c=0$ 方程根的部分程序段:

```
int a,b,c,t;
scanf(&a,&b,&c);
t=b*b-4*a*c;
...
```

对于上面的程序,程序员仅仅需要考虑求解一元二次方程根的步骤,而不需要关心键盘上输入的a、b、c数据放在内存的什么地方及其如何表示的,也不需要关心t的值是如何计算出来的。程序员只需要直接应用这个数据类型解决实际问题。这里,整数就是C语言中一个抽象数据类型。

在C语言中提供的数据类型仅能解决部分应用问题,在很多非数值计算的应用中,经常需要用到表、栈、队列、树、图等。在数据结构中,把它们定义成抽象数据类型(Abstract Data Type, ADT),分别介绍这些抽象数据类型及对它操作的实现方法。在此基础上,通过实例应用这些抽象数据类型解决实际问题。

1.2.2 抽象数据类型的描述

抽象数据类型的描述包括定义抽象数据类型的名称、数据对象、数据之间的关系和基本操作等。抽象数据类型的设计者将根据这些描述给出操作的具体实现,而抽象数据类型的使用者则依据这些描述来使用抽象数据类型。

抽象数据类型描述的一般形式如下:

```
ADT <抽象数据类型名> {
    数据对象: <数据对象的定义>
    数据关系: <数据关系的定义>
    基本操作: <基本操作的定义>
} ADT <抽象数据类型名>;
```

由于数据结构可以由一个二元组(D, R)来表示,其中,D是某个数据对象,R是该对象中所有数据元素之间关系的有限集合。那么在抽象数据类型的描述中,数据对象的定义表示为D,而数据关系的定义表示为R。

在描述抽象数据类型时,可以任意给定抽象数据类型名,但最好是使用意义贴近的文字表示。例如,“线性表”抽象数据类型的描述如下: