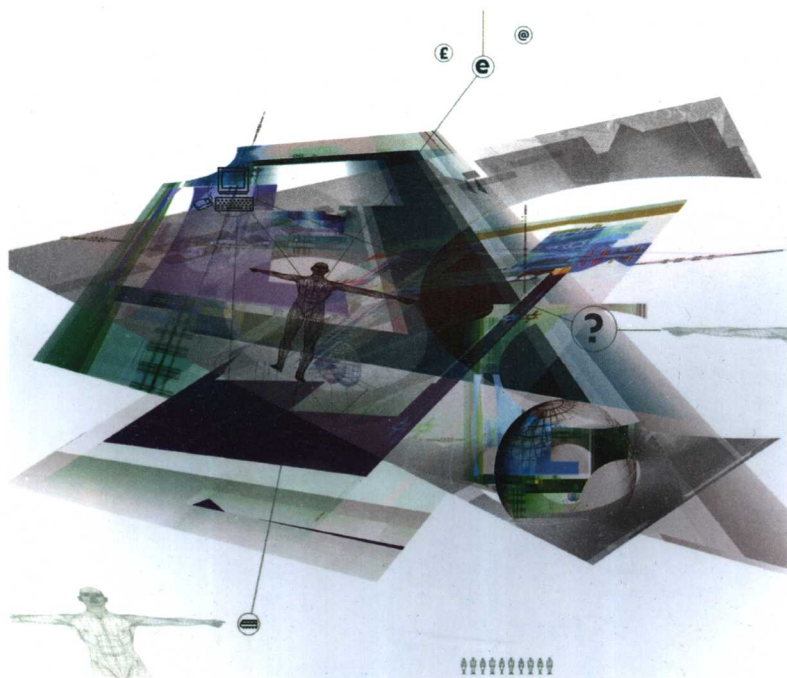


河南省高职高专计算机规划教材

数据结构

主 编 商信华



西北大学出版社
NORTHWEST UNIVERSITY PRESS

河南省高职高专计算机规划教材

数 据 结 构

主 编 商信华
副主编 尚展全 申志军
编 者 (以姓氏笔画为序)
申志军 吴华芹 李 昕
李 杰 杨俊红 谷海红
尚展全 赵素萍 商信华
谢伟增

西北大学出版社

图书在版编目(CIP)数据

数据结构 / 商信华主编. — 西安: 西北大学出版社, 2006.2

高职高专计算机规划教材

ISBN 7-5604-2097-4

I. 数... II. 商... III. 数据结构 - 高等学校: 技术学校 - 教材 IV. TP311.12

中国版本图书馆 CIP 数据核字(2005)第 140266 号

数据结构

主 编 商信华

出版发行 西北大学出版社

通讯地址 西安市太白北路 229 号 邮编 710069

经 销 新华书店经销

印 刷 河南新华印务有限公司

开 本 787mm × 1092mm 1/16

印 张 13.75

字 数 310 千字

版 次 2006 年 2 月第 1 版 2006 年 2 月第 1 次印刷

书 号 ISBN 7-5604-2097-4/TP · 47

定 价 21.00 元

出版说明

当前,我国正在走新型工业化道路,坚持以信息化带动工业化,以工业化促进信息化,加快发展现代服务业,全面建设小康社会。走新型工业化道路,既需要一大批拔尖创新人才,也需要数以千万计的专门人才和数以亿计的高素质劳动者。根据劳动力市场技能型人才短缺的状况,国家决定实施技能型紧缺人才培养培训工程,其中计算机人才的培养培训是其重要的组成部分。

为适应高职高专计算机教育发展的需要,促进教育教学改革和教材建设,满足经济和社会发展对计算机人才的需求,根据教育部等六部门印发的《关于实施职业院校制造业和现代服务业技能型紧缺人才培养培训工程的通知》精神,按照教育部《关于制定〈2004~2007年职业教育教材开发编写计划〉的通知》要求,在教育部有关部门的支持和指导下,经河南省教育厅批准,我们组织有关专家,对计算机高职高专教育的培养目标和模式、课程体系、教学内容、教学方法和手段、教学实践等方面,进行了广泛而深入的调研。

在充分调研的基础上,在教育部有关部门和河南省教育厅的大力支持下,我们组织有关专家召开了计算机高职高专教育教学研讨会、教学大纲审定会和主编人会议,确定了教材编写的指导思想、原则和要求,组织全省近40所院校的一线教师,吸取了最新的计算机高职高专教育教学经验和成果,编写了这套教材。本套教材充分体现了以就业为导向,以职业技能培养为根本的编写指导思想,突出了思想性、科学性、先进性、可读性和适用性的编写原则,较好地处理了“三基”(基本理论、基本知识、基本技能)关系,学历教育与职业认证、职业准入的关系。

这套教材虽经广泛调研与精心编撰,但一定还会存在这样或那样的不足,我们诚挚欢迎广大读者,尤其是选用该教材的教师和学生对其中的不足之处给予批评指正,以便我们在重印和修订时采纳有益的建议和意见,使之不断完善。

河南省高等学校计算机教育研究会

2006年1月

河南省高职高专计算机规划教材

参加编写学校

(以汉语拼音排序)

安阳师范学院
华北水利水电学院水利职业学院
黄河水利职业技术学院
河南财政税务高等专科学校
河南纺织高等专科学校
河南公安高等专科学校
河南工业大学
河南工业大学化学职业学院
河南经贸职业学院
河南农业职业学院
河南司法警官职业学院
河南商业高等专科学校
河南质量工程职业学院
济源职业技术学院
焦作大学
开封教育学院
开封人民警察学校

漯河职业技术学院
洛阳师范学院
平顶山工业职业技术学院
商丘师范学院
铁道警官高等专科学校
许昌职业技术学院
信阳农业高等专科学校
中原工学院广播影视职业学院
中州大学
郑州航空工业管理学院信息统计职业学院
郑州经济管理干部学院
郑州交通职业学院
郑州牧业工程高等专科学校
郑州轻工业学院
郑州轻工业学院轻工职业学院
郑州师范高等专科学校
郑州铁路职业技术学院

前 言

数据结构是计算机及相关专业中一门重要的专业基础课程。当用计算机来解决实际问题时,就要涉及数据的表示及数据的处理,而数据表示及数据处理正是数据结构课程的主要研究对象。通过这两方面内容的学习,为后续课程,特别是软件方面的课程学习打下坚实的知识基础,同时也提供了必要的技能训练。因此,数据结构课程在计算机应用专业中具有举足轻重的作用。本课程的目的:在基础方面,要求学生掌握常用数据结构的基本概念及其不同的实现方法;在技能方面,学会分析研究计算机的数据对象的特性,以便选择适当的数据结构和存储结构以及相应的算法,并运用各种数据结构解决实际问题。

本书共分9章,主要讲述了线性表、栈和队列、串、数组、树和二叉树、图、查找、排序等数据结构以及实现这些数据组织的典型算法,采用C语言作为算法描述语言。本书结合高职高专教学特点,理论“必须够用”,突出实践,所以概念叙述简洁,深入浅出,概念讨论和实际例子相结合,且实际例子典型完整。

本书由信阳农业高等专科学校商信华任主编,郑州轻工业学院尚展垒、信阳农业高等专科学校申志军任副主编。第1章由商信华和河南工业大学李昕编写,第2章由信阳农业高等专科学校申志军编写,第3章由郑州轻工业学院尚展垒编写,第4章由济源职业技术学院李杰编写,第5章由鹤壁职业技术学院谷海红编写,第6章由司法警官学院谢伟增编写,第7章由洛阳师范学院赵素萍编写,第8章由郑州铁路职业学院杨俊红编写,第9章由河南工业大学化学职业学院吴华芹编写。此外,信阳农业高等专科学校杜永强、郑州纺织工业高专的张胜利参加了本书部分章节的编写工作。

尽管作者在写作过程中非常认真和努力,但由于时间仓促,错误和不足仍在所难免,恳请读者批评指正。

编 者

2005年6月

目 录

第1章 绪论	(1)
1.1 数据结构的基本概念	(1)
1.1.1 为什么要学数据结构	(1)
1.1.2 数据结构的基本概念	(2)
1.2 抽象数据类型的表示与实现	(5)
1.3 算法和算法分析	(7)
1.3.1 算法	(7)
1.3.2 算法的设计要求	(8)
1.3.3 算法效率的度量	(8)
小结	(13)
习题	(13)
第2章 线性表	(15)
2.1 线性表的概念及运算	(15)
2.1.1 线性表的逻辑结构	(15)
2.1.2 线性表的抽象数据类型	(16)
2.1.3 操作举例	(17)
2.2 线性表的顺序存储及运算实现	(18)
2.2.1 线性表的顺序存储	(18)
2.2.2 顺序存储下的线性表操作的实现	(19)
2.3 线性表的链接存储和实现	(25)
2.3.1 线性表的链接存储	(26)
2.3.2 线性表操作在单链表上的实现	(28)
2.3.3 循环链表、双向链表、静态链表	(36)
小结	(39)
习题	(39)
第3章 栈和队列	(42)
3.1 栈	(42)
3.1.1 栈的定义及基本运算	(42)
3.1.2 栈的顺序表示和实现	(43)
3.1.3 栈的链式表示和实现	(46)
3.2 栈的简单应用	(48)
3.3 队列	(52)
3.3.1 队列的定义及基本运算	(52)

3.3.2	队列的顺序表示和实现	(54)
3.3.3	队列的链式表示和实现	(56)
3.3.4	队列的应用	(58)
小结		(61)
习题		(62)
第4章	串	(63)
4.1	串及其基本运算	(63)
4.1.1	串的基本概念	(63)
4.1.2	串的基本运算	(64)
4.2	串的定长顺序存储及基本运算	(65)
4.2.1	串的定长顺序存储	(65)
4.2.2	定长顺序串的基本运算	(68)
4.2.3	串的模式匹配及算法	(70)
4.3	串的堆存储结构	(72)
4.3.1	串名的存储映像	(72)
4.3.2	堆存储结构	(73)
4.3.3	基于堆结构的基本运算	(73)
小结		(76)
习题		(76)
第5章	数组	(77)
5.1	多维数组	(77)
5.1.1	数组的定义和操作	(77)
5.1.2	数组的内存映象	(78)
5.2	特殊矩阵的压缩存储	(79)
5.2.1	三角矩阵	(79)
5.2.2	对称矩阵	(81)
5.2.3	带状矩阵	(81)
5.3	稀疏矩阵	(81)
5.3.1	稀疏矩阵的三元组表存储	(81)
5.3.2	稀疏矩阵的十字链表存储	(85)
5.4	广义表	(88)
5.4.1	广义表的定义和基本运算	(88)
5.4.2	广义表的存储	(89)
5.4.3	广义表基本操作的实现	(90)
小结		(94)
习题		(95)
第6章	树结构	(96)
6.1	一般树的定义、基本操作与存储实现	(97)
6.1.1	一般树的定义	(97)

6.1.2	一般树的基本操作	(98)
6.1.3	一般树的存储实现	(99)
6.2	二叉树的定义、基本操作与存储实现	(102)
6.2.1	二叉树的定义	(102)
6.2.2	二叉树的存储	(105)
6.2.3	二叉树的基本操作及实现	(107)
6.3	二叉树的遍历	(109)
6.3.1	二叉树的遍历方法及递归实现	(110)
6.3.2	二叉树遍历的非递归实现	(112)
6.3.3	由遍历序列恢复二叉树	(115)
6.4	线索二叉树	(116)
6.4.1	线索二叉树的定义及结构	(117)
6.4.2	线索二叉树的基本操作实现	(118)
6.5	二叉树的应用	(119)
6.5.1	二叉树遍历的应用	(119)
6.5.2	最优二叉树—哈夫曼树	(121)
小结		(125)
习题		(125)
第7章	图结构	(127)
7.1	图的基本概念	(127)
7.1.1	图的定义和术语	(127)
7.1.2	图的基本操作	(130)
7.2	图的存储表示	(131)
7.2.1	邻接矩阵	(131)
7.2.2	邻接表	(133)
7.2.3	十字链表	(135)
7.2.4	邻接多重表	(136)
7.3	图的遍历	(138)
7.3.1	深度优先遍历	(138)
7.3.2	广度优先遍历	(141)
7.4	最小生成树	(144)
7.4.1	最小生成树的基本概念	(144)
7.4.2	构造最小生成树的 Prim 算法	(145)
7.4.3	构造最小生成树的 Kruskal 算法	(148)
7.5	最短路径	(149)
7.5.1	最短路径的基本概念	(149)
7.5.2	从一个顶点到其他各顶点的最短路径	(149)
小结		(151)
习题		(151)

第 8 章 查找	(155)
8.1 查找的基本概念	(155)
8.2 静态查找表	(156)
8.2.1 顺序表的查找	(156)
8.2.2 有序表的查找	(157)
8.2.3 索引顺序表的查找	(160)
8.3 动态查找表	(163)
8.3.1 二叉排序树	(163)
8.3.2 平衡二叉树	(169)
8.3.3 B-树和 B+树	(172)
8.4 哈希表	(176)
8.4.1 哈希表的基本概念	(176)
8.4.2 哈希函数的构造方法	(177)
8.4.3 哈希冲突解决方法	(178)
8.4.4 哈希表的查找分析	(179)
小结	(181)
习题	(182)
第 9 章 排序	(184)
9.1 基本概念	(184)
9.2 插入排序	(185)
9.2.1 直接插入排序	(185)
9.2.2 折半插入排序	(187)
9.2.3 希尔排序	(187)
9.3 选择排序	(188)
9.3.1 简单选择排序	(188)
9.3.2 堆排序	(190)
9.4 交换排序	(193)
9.4.1 冒泡排序	(193)
9.4.2 快速排序	(194)
9.5 综合应用举例	(195)
小结	(198)
习题	(198)
参考文献	(200)

第 1 章 绪论

计算机软件的发展有力地推动了计算机系统的发展和应用,给人们的工作和生活带来了深刻的变化。为了能够以最少的成本、最快的速度、最好的质量开发出合乎要求的计算机软件,设计有效的数据结构是最关键的因素之一。因此作为软件设计人员就必须了解如何组织各种数据在计算机中的存储、传递和转换,而这正是数据结构这门课程要研究的问题。

本章主要对数据结构中的基本知识做概括性的描述,这些内容将贯穿整个课程的学习。在学习本章之前,应当掌握一种或几种高级语言作为算法描述语言学习的基础。

1.1 数据结构的基本概念

1.1.1 为什么要学数据结构

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。在计算机科学中,数据结构不仅是一般程序设计的基础,而且是设计和实现编译程序、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。学习“数据结构”既为进一步学习其他软件课程提供必要的准备知识,又有助于提高软件设计和程序编制水平。著名的瑞士计算机科学家沃思(N. Wirth)教授曾提出:算法+数据结构=程序。这里的数据结构是指数据的逻辑结构和存储结构,而算法则是对数据运算的描述。由此可见,程序设计的实质是对实际问题选择一种好的数据结构,加之设计一个好的算法,而好的算法在很大程度上取决于描述实际问题的数据结构。

在计算机发展的初期,人们使用计算机的目的主要是处理数值计算问题,很多问题的求解最后都转化为求解数学方程或数学方程组(主要是线性或非线性的代数方程组或微分方程组),即使是不需要用计算机求解的简单问题也需要一个数学模型来描述。例如“鸡兔同笼”问题,可转化为对二元一次方程组进行求解;又如,在房屋设计或桥梁设计中的结构应力分析计算可化解为线性代数方程组求解的问题,该方程组可以使用迭代算法来求解;再如,人们在电视里天天看到的天气预报,它的数学模型是一个环流模式方程。由于数值计算所涉及的运算对象是简单的整型、实型或布尔类型数据,所以程序设计者的主要精力是集中于程序设计的技巧上,而无须重视数据结构。

而当计算机进入非数值计算领域、特别是用在管理上的时候,计算机的操作对象之间的关系就无法用数学方程加以描述了。下面所列举的就是属于这一类的具体问题。

例 1.1 八皇后问题:八皇后问题源于几百年前,内容是这样的:在标准国际象棋棋盘上,放上八个皇后,使其互相不能够攻击。这个问题是高斯最先提出的,每一行,每一列,或一斜线上只能有一个皇后,否则就违反了规则,若不考虑旋转对称因素,共有 92 种解法。

例 1.2 最短路径问题:举例来说,一些城市之间有道路相连,道路的长度已知,现在要求有这么一个系统来满足旅客提出的一些问题:如一旅客要从 A 城到 B 城去旅行,他希望选择一条总路程最短或是途中中转次数最少的路径。

例 1.3 Hanoi 塔(汉诺塔)问题:假设有三个分别命名为 A,B 和 C 的塔座,在塔座 B 上插有 n 个直径大小各不相同、依小到大编号为 $1, 2, \dots, n$ 的圆盘。现要求将 B 轴上的 n 个圆盘移至塔座 A 上并仍按同样顺序叠排,圆盘移动时必须遵循下列规则:

- ①每次只能移动一个圆盘;
- ②圆盘可以插在 A,B 和 C 中的任一塔座上;
- ③任何时刻都不能将一个较大的圆盘压在较小的圆盘之上。

例 1.4 学生信息检索系统。当需要查找某个学生的有关情况的时候,或者想查询某个专业或年级学生的有关情况的时候,只要建立了相关的数据结构,按照某种算法编写了相关程序,就可以实现计算机自动检索。计算机的主要操作便是按照某个特定要求(如给定姓名)对学生信息进行查询,如表 1.1 所示。

表 1.1 学生情况表

学号	姓名	性别	专业	系别
2004001001	张红娇	女	计算机科学与技术	计算机科学系
2004001002	苗金霞	女	计算机科学与技术	计算机科学系
2004001003	李立斌	男	计算机科学与技术	计算机科学系
2004002001	卫 东	女	会计电算化	经济管理系
2004002002	黄伟兴	男	会计电算化	经济管理系
2004003001	赵小年	男	商务英语	外语系
2004003002	程 成	女	商务英语	外语系

由上面所举的例子可见,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、树、图之类的数据结构。随着计算机应用领域的扩大和软、硬件的发展,非数值计算问题越来越显得重要。据统计,当今处理非数值计算性问题占用了 90% 以上的机器时间。这类问题涉及到的数据结构更为复杂,数据元素之间的相互关系一般无法用数学方程式加以描述。因此,解决这类问题的关键不再是数学分析和计算方法,而是要设计出合适的数据结构,才能有效地解决问题。大量非数值计算问题的数学模型正是这门课程所要讨论的对象。因此,概括地说,数据结构是一门讨论“描述现实世界实体的数学模型(非数值计算)及其上的操作在计算机中如何表示和实现”的学科,或者说数据结构主要是一门研究非数值计算的程序设计问题中所出现的计算机操作对象以及它们之间的关系和操作的学科。

1.1.2 数据结构的基本概念

在系统地学习数据结构知识之前,首先对一些基本概念和术语赋予确切的含义。

1. 数据(Data)

数据这个词的本义是指在科学实验、检验、统计等过程中所获取的和用于科学研究、技术设计、查证、决策等的数值,这个初始含义已经不能反映当代计算机应用的现实,因此当代计算机科学把计算机系统所处理的一切信息,包括数值和非数值的信息,都统称为数据。本课程内所称的数据是计算机操作的对象 的总称,它的范围既包括数值型数据(整数、实数、复数、双精度数数据等),也包括非数值型数据(字符、字符串、声音、图像、图片、语音等)。

2. 数据元素(Data Element)

数据元素简称元素,是数据(集合)中的一个“个体”,它是数据结构讨论的基本单位,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:

学生信息检索系统中学生信息表中的一个记录、八皇后问题中状态树的一个状态等,都是一个数据元素。对一文件来说,记录就是数据元素;对一字符串来说,单个的字符就是数据元素;对一数组来说,数组元素就是数据元素。数据与数据元素并没有严格的区分。

3. 数据项(Data Item)

数据项是组成数据元素的单位,是数据的不可分割的最小单位,如结构中的域。图 1.1 是以形象的形式指出了数据元素和数据项的关系。

学 号	姓 名	英 语	高等数学	C 语言程序设计
20050001	张三	85	73	89
20050002	李 四	79	84	92
20050003	王 五	68	58	75
一个数据项				一个数据元素

图 1.1 学生成绩表

注意“最小单位”的含义,比如把 20050001 号学生的姓名“张三”再分作“张”和“三”,这个分割结果对于该结构而言是无意义的,但是对于另一个分析学生姓名的中文处理程序而言,分割是有意义的,因此数据项的层次具有的是相对意义。

4. 数据对象(Data Object)

数据对象是性质相同的数据元素的有限集合,比如:一个班级的成绩表可以看作一个数据对象;整数、实数等也可以看作数据对象。它是数据的一个子集。

在某个具体问题中,当在同一个数学模型中数据元素都具有相同的性质(元素值不一定相等),就可以看成属于同一数据对象或者说数据元素类(Data Element Class),数据元素是数据元素类的一个实例。

5. 数据结构(Data Structure)

(1) 数据结构的定义

数据结构是在整个计算机科学与技术领域上广泛被使用的术语,简写为 DS,也称信息结构,指的是数据之间的相互关系,即数据的组织形式。①数据元素之间的逻辑关系,也称为数据的逻辑结构(Logical Structure);②数据元素及其关系在计算机存储器内的表示,称为数据的存储结构(Storage Structure);③数据的运算,即对数据施加的操作。

(2) 数据的逻辑结构

逻辑结构是指数据及其相互间的联系。数据的逻辑结构可以看作是从具体问题抽象出来的数学模型,它与数据的存储无关。逻辑结构可归结为以下四种基本类型:

① 集合结构

如果数据结构中的元素之间没有或者不考虑关系问题,则称这种结构是集合。集合中的元素没有前驱后继之分,它们除了彼此“同属于一个集合”外,没有别的关系,元素彼此是平等的,元素可以按任一序列排列。用一些离散的圆圈来表示集合的元素,比如五个元素的集合如图 1.2 所示。

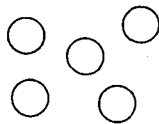


图 1.2 五个元素的集合

② 线性结构

线性结构的特点是:结构中有且仅有一个始结点和一个终结点,始结点只有一个后继结点,终结点只有一个前趋结点,每个内结点有且仅有一个前趋结点和一个后继结点。线性结构最一般的情形是线性表,如图

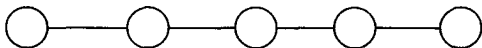


图 1.3 五个元素的线性结构

③ 树型结构

树型结构的特点是:树结构中的结点有且仅有一个没有前趋结点的结点,称之为根(root)结点,它是树型结构的最高层次的结点。其他结点都仅有一个前趋结点,但允许有多个后继结点。从根结点到任一非根结点,都有且仅有一条路径,如图 1.4 所示是五个元素的树型结构。

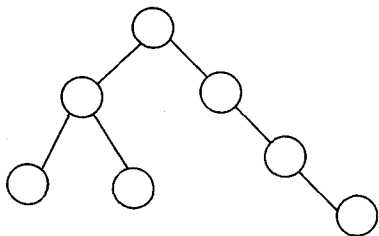


图 1.4 五个元素的树型结构

树型结构存在着很多变种,如二叉树结构,堆结构等。

④ 图形结构

图形结构简称图结构,又称作网状结构,如果数据结构中的每一个元素都可以有 0 个或者多个前驱以及 0 个或者多个后继,则称这种结构为图,它是相对比较复杂的数据结构。

图形结构有时称为结点互联的网络结构——因特网的网页链接关系就是一个非常复杂的图形结构。图 1.5 是五个元素的图形结构。

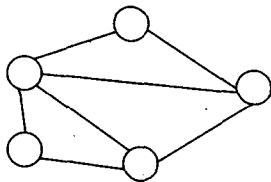


图 1.5 五个元素的图形结构

(3) 数据的物理结构

研究数据结构的目的是为了在计算机中实现对它的操作,任何需要计算机进行管理和处理的数据元素都必须首先按某种方式存储在计算机中。数据元素在计算机中的存储方式称为数据的存储结构,也称为数据的物理结构。

数据存储结构的基本形式有两种:其一为顺序存储结构,用一组地址连续的存储单元依次存储数据元素,把逻辑上相邻的元素存储在物理位置相邻的存储单元中,数据间的逻辑关系表现在数据元素的存储位置上,通常借助于程序设计语言中的数组来实现。其二为链式存储结构,借助于指示元素存储地址的指针来单独表示数据元素之间的逻辑关系。即对逻辑上相邻

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,链式存储结构通常借助于程序设计语言中的指针类型来实现。

顺序存储结构和链式存储结构是两种基本的存储结构,如图 1.6 所示。由上面的分析可见,在顺序存储结构中结点只包含数据元素本身的信息,而链式存储结构中由数据元素域和指针域组成的结构体称为结点。

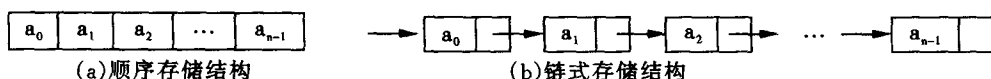


图 1.6 基本存储结构形式

除了通常采用的顺序存储方法和链式存储方法外,有时为了查找的方便还采用索引存储方法和散列存储方法。

索引(indexed)存储方法:在存储结点的同时,还建立附加的索引表,索引表中的每一项称为索引项,形式为:关键字,地址。

散列(hashing)存储方法:根据结点的关键字直接计算出该结点的存储地址。

四种存储方法可结合起来对数据结构进行存储映像。

例 1.5 成绩表数据可用 C 语言的结构体数组 `classonestu[50]` 来存储:

```
struct stu
{
    int stuno; /* 数据项,也称 stu 位串中的一个子位串,或叫做数据域 */
    char name[20];
    int maths;
    int language;
    int c_language;
}classonestu[50];
```

6. 结构类型(Structure Data Type)

其值可分解为若干个成份(或称为份量),如 C 语言的数组结构等类型,通常是由标准类型派生的,故它也是一种导出类型。

1.2 抽象数据类型的表示与实现

抽象数据类型的定义可以由一种数据结构和定义在其上的一组操作组成,而数据结构又包括数据元素及元素间的关系,因此抽象数据类型一般可以由元素、关系及操作三个要素来定义,用三元组表示。用数学方法定义对象集合和运算集合,仅通过运算的性质来刻画数据对象,而独立于计算机中可能的表示方法。

抽象数据类型的形式化定义:

$ADT = (D, S, P)$

其中: D 是数据对象, S 是 D 上的关系集, P 是对 D 的基本操作集。

其中 (D, S) 即为相应的数据结构的定义。

P 的基本格式如下:

基本操作名(参数表)

初始条件:<初始条件描述>

操作结果:<操作结果描述>

基本操作有两种参数:赋值参数只为操作提供输入值;引用参数以 & 打头,除可提供输入值外,还将返回操作结果。

“初始条件”描述了操作执行之前数据结构和参数应满足的条件,若不满足,则操作失败,并返回相应出错信息。

“操作结果”说明了操作正常完成之后,数据结构的变化状况和应返回的结果。

若初始条件为空,则可省略之。

由于数据结构是和它的操作集分不开的,因此以后对数据结构的定义均包括它们操作的定义,即直接将它们定义为抽象数据类型。

而 ADT 的描述格式如下:

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

其中,数据对象和数据关系的定义用伪码描述。

抽象数据类型的实现包括数据结构的实现和操作的实现,因此不仅要借用高级语言中的数据类型来描述它的存储结构,也要利用高级编程语言中已经实现的数据类型(通常称之为固有数据类型)来实现抽象数据类型的操作,即利用处理器中已存在的数据类型来说明新的结构,用已经实现的操作来组合新的操作。

由于高级语言为抽象数据类型的定义提供了很好的环境和工具,再复杂的数据模型都可以通过构造数据类型来表达,再复杂的运算都可以借助过程或函数来描述。因此,上述由数据模型和数据模型上定义的运算综合起来的抽象数据类型很容易用高级语言来定义。

但是,只要抽象数据类型的定义不变,不管上述构造数据类型的表示和过程与函数的实现如何改变,主程序的表达都不会改变;反过来,不管主程序在哪里调用抽象数据类型上的函数或过程,上述构造数据类型的表示和过程与函数的实现都不必改变。算法顶层的设计与底层的设计之间的这种独立性,显然得益于抽象数据类型的引入。而这种独立性给算法和程序设计带来了许多好处。

本书采用的类 C 语言精选了 C 语言的核心子集,同时为了算法设计和描述的方便,又作了若干扩充,增强了语言的描述功能,以下对其作简要说明。

①预定义常量和类型

函数执行结果状态码(Status):

# define	TRUE	1
# define	FALSE	0
# define	OK	1
# define	ERROR	0

```
#define INFEASIBLE -1
```

```
#define OVERFLOW -2
```

Status 是函数结果的类型,其值是函数结果状态代码

```
typedef int Status;
```

②数据结构的表示用类型 typedef 描述,数据元素约定为 ElemType,在具体的应用中由用户自行定义。

③输入输出语句可简记为:

```
scanf([格式串], &变量1, &变量2, ..., &变量n);
```

```
printf([格式串], 表达式1, 表达式2, ..., 表达式n);
```

格式串通常省略。

④算法中的注释采用单行注释“//...”或多行注释“/* ... */”

1.3 算法和算法分析

1.3.1 算法

1. 算法的概念

算法(Algorithm)是对特定问题求解操作步骤的一种描述。算法要用某种语言来描述。描述算法的语言主要有三种形式:文字形式、伪码形式、程序设计语言形式。文字形式是用某种文字语言如中文或英文等文字来描述。伪码形式是用一种仿程序设计语言来描述算法。程序设计语言形式,是用某种程序设计语言来描述算法,其优点是直接作为程序语句输入计算机后,计算机就能调用和运行。本教材采用C语言描述算法。

算法与数据结构的关系非常密切,数据结构与算法两者互为存在。“算法+数据结构=程序”的公式正揭示了程序设计的本质:对实际问题的求解就是,选择一种好的数据结构,加上设计一个好的算法。

2. 算法的特性

严格说来,一个算法必须满足以下五个重要特性:

①有穷性。一个算法总是对于任意一组合法的输入值,在执行有穷步骤之后一定能够结束,且每一步骤是在有穷时间内结束的。这里所说的“有穷”是针对实际而言的,而非抽象意义上的有穷,若一程序的运行时间远远超出了所能接受的程度,则不再被认为是具有有穷的。

②确定性。算法中的每一条指令必须有确切的含义,不能产生多义性。算法的执行对应着的相同的输入仅有惟一的一条路径。对于相同的输入必须执行相同的路径,得到相同的执行结果(无二义性);对于每种情况下,下一步骤所应执行的操作都必须明确,即在算法中有确切的规定,使算法的执行者或阅读者都能明确其含义及如何执行。并且在任何条件下,算法都只有一条执行路径。

③可行性。算法中的每一步都可以通过已经实现的基本运算的有限次执行得以实现。即用于描述算法的操作都是“足够基本的”,引入新的操作必须是已有基本操作的有限组合,这时算法的每种运算都可以至少在理论上用手工计算的方式在有限时间内完成。

④输入。一个算法有0个或者多个输入,这些输入取自特定的数据对象集合,作为算法的