

21 世纪 高等学校本科系列教材

总主编 吴中福

数 据 结 构

(12)

周 娅 主 编

张振宇 副主编



重庆大学出版社

数 据 结 构

周 娅 主 编
张振宇 副主编

重 庆 大 学 出 版 社

内 容 简 介

本书以“培养抽象思维能力,面向实际应用”为指导思想,以“逻辑结构→存储结构→基本运算的实现→典型应用”为主线,全面系统地介绍了线性表、栈、队列、数组、树与二叉树、图等几种常用的数据结构,以“分析算法思想→运用C语言描述算法→进行算法分析”为思路,解析了各种算法。还介绍了递归设计技术和程序设计常用的查找、排序方法,分析比较了各种查找、排序算法的运行效率,最后介绍了文件。

本书概念清楚,逻辑性好,应用性强。每章配有学习方法指导和习题,便于读者掌握各章的重点和难点,并进行必要的训练。本书可作为计算机类相关专业学生的教材或参考书,也可供从事计算机工程与应用的科技工作者参考。

图书在版编目(CIP)数据

数据结构/周娅主编. —重庆:重庆大学出版社,2003.3

计算机科学与技术专业本科系列教材

ISBN 7-5624-2339-3

I. 数... II. 周... III. 数据结构—高等学校—教材 IV. TU311.12

中国版本图书馆 CIP 数据核字(2002)第 062273 号

数 据 结 构

周 娅 主 编

张振宇 副主编

责任编辑:周 立 版式设计:周 立

责任校对:蓝安梅 责任印制:秦 梅

*

重庆大学出版社出版发行

出版人:张鸽盛

社址:重庆市沙坪坝正街174号重庆大学(A区)内

邮编:400044

电话:(023) 65102378 65105781

传真:(023) 65103686 65105565

网址: <http://www.cqup.com.cn>

邮箱: fxk@cqup.com.cn (市场营销部)

全国新华书店经销

自贡新华印刷厂印刷

*

开本:787×1092 1/16 印张:17.5 字数:436千

2003年3月第1版 2003年3月第1次印刷

印数:1—5 000

ISBN 7-5624-2339-3/TP·295 定价:19.50元

本书如有印刷、装订等质量问题,本社负责调换

版权所有 翻印必究

著名计算机科学家 Niklaus Wirth 提出:算法+数据结构=程序。有关数据结构和算法的研究是计算机科学与工程的基础性研究之一。数据结构课程是计算机类专业最基础最核心的课程,也是一个程序员或软件工程师的必修课程。

在计算机的实际应用中,面对问题,首先需要抽象出合适的数据模型,选择恰当的数据结构描述,即考虑如何在计算机内存中以某种结构组织存储数据,以方便对数据的查询和操作;然后在已确定的“数据存储结构”的基础上考虑采用哪种算法,其算法的效率如何。因此,本书以“培养抽象思维能力,面向实际应用”为指导思想,以“逻辑结构→存储结构→基本运算的实现→典型应用”为主线,全面系统地介绍了线性表、栈、队列、数组、树与二叉树、图等几种常用的数据结构,以“分析算法思想→运用 C 语言描述算法→进行算法分析”为思路,解析了各种算法。还在第 5 章介绍了递归模型、递归过程等递归设计技术,在第 8 章、第 9 章介绍了程序设计中常用的查找、排序方法,分析比较了各种查找、排序算法的运行效率,最后在第 10 章介绍了文件。从培养读者根据实际问题的要求,选择合适的数据结构设计算法的分析,解决实际问题能力出发,以“提出问题→分析问题→解决问题”为线索,力求“由简及繁、由易到难、循序渐进”。内容的组织思路清晰,算法思想叙述深入浅出,算法描述清晰自然,所有的算法用 C 语言描述,尽可能使算法的描述接近可运行的程序,缩短从算法到程序的转换过程。

本书的最大特点就是强调应用。通过现实生活中的许多应用实例,具体运用各种数据结构和算法设计方法。根据实例,使学生不但可以印证许多基本概念,而且能加深理解,从而更好地掌握相应的数据结构和算法,并能达到熟练应用。把应用与理论知识紧密结合,以激发学生学习数据结构和算法的兴趣。每章配有学习方法指导和习题,便于读者掌握各章的重点和难点,并进行必要的训练。

本书由周娅担任主编,张振宇担任副主编,与吴晓红、方娇莉共同编写。周娅编写了第 1 章、第 5 章、第 6 章以及实验指导,张振宇编写了第 2 章、第 7 章、第 8 章,吴晓红编写了第 4 章、第 9 章,方娇莉编写

了第3章、第10章。全书由周娅统稿。参与本书编写的作者都是高校中从事数据结构教学多年的老师,书中包含了他们多年的工作与思考,但书中难免会有或大或小、或多或少的错误,恳请读者批评指正。

作 者

2003年2月

目录

第1章 绪论	1
1.1 什么是数据结构	1
1.2 数据的逻辑结构和存储结构	4
1.2.1 数据的逻辑结构	4
1.2.2 数据的存储结构	5
1.3 存储结构的表示与实现	6
1.3.1 数据类型	6
1.3.2 抽象数据类型	6
1.3.3 抽象数据类型的表示与实现	7
1.4 问题、算法和程序	8
1.5 算法的描述与分析	9
1.5.1 算法的描述	9
1.5.2 算法分析	14
学习指导	15
习 题	16
第2章 线性表	18
2.1 线性表的逻辑特点	18
2.2 线性表的顺序存储结构——顺序表	19
2.2.1 顺序表的类型定义	19
2.2.2 基本操作的实现	20
2.2.3 典型应用——多项式求值	23
2.3 线性表的链式存储结构——链表	24
2.3.1 单链表的类型定义	24
2.3.2 基本运算的实现	25
2.3.3 循环链表与双向链表	29
2.3.4 典型应用——多项式相加	33
2.4 链表的数组描述与实现	36
学习指导	38

习 题	39
第3章 栈和队列	40
3.1 栈的逻辑特点	40
3.2 栈的顺序存储结构——顺序栈	41
3.2.1 顺序栈的类型定义	41
3.2.2 基本运算的实现	42
3.2.3 多栈共享空间	44
3.2.4 典型应用举例	46
3.3 栈的链式存储结构——链栈	54
3.3.1 链栈的类型定义	54
3.3.2 基本运算的实现	54
3.3.3 典型应用举例	57
3.4 队列的逻辑特点	58
3.5 队列的链式存储结构——链队列	58
3.5.1 链队列的类型定义	58
3.5.2 基本运算的实现	59
3.6 队列的顺序存储结构——循环队列	61
3.6.1 循环队列的类型定义	62
3.6.2 基本运算的实现	63
3.6.3 典型应用举例	65
学习指导	68
习 题	69
第4章 数组和串	70
4.1 数组的逻辑特点	70
4.2 数组的顺序存储结构	71
4.3 特殊矩阵的压缩存储	73
4.4 稀疏矩阵运算的实现	75
4.4.1 矩阵转置	75
4.4.2 矩阵相乘	77
4.5 串	80
4.5.1 串的存储表示与操作实现	80
4.5.2 串的模式匹配算法	83
学习指导	87
习 题	88
第5章 递归	90
5.1 递归模型与递归算法	90

5.2	递归过程与递归工作栈	93
5.3	递归程序的阅读方法	98
* 5.4	递归问题的非递归算法	99
5.5	广义表	102
5.5.1	广义表的递归定义	102
5.5.2	广义表的存储结构	103
5.5.3	广义表的递归算法	106
5.5.4	典型应用—— m 元多项式的表示	109
	学习指导	111
	习 题	111
第 6 章	树和二叉树	113
6.1	树的逻辑结构	113
6.1.1	树的递归定义	113
6.1.2	树的基本术语	114
6.1.3	树的表示	115
6.2	二叉树	117
6.2.1	二叉树的递归定义	117
6.2.2	二叉树的性质	117
6.2.3	二叉树的存储结构	120
6.3	二叉树遍历	123
6.3.1	二叉树遍历的定义	123
6.3.2	前序遍历算法描述	124
6.3.3	中序遍历算法描述	126
6.3.4	后序遍历算法描述	128
6.3.5	遍历算法的应用	131
6.4	线索化二叉树	135
6.4.1	二叉树的线索化	136
6.4.2	线索二叉树的遍历	138
6.5	树和森林	140
6.5.1	树的存储结构	140
6.5.2	树、森林与二叉树的转换	144
6.5.3	树和森林的遍历	146
6.6	哈夫曼树及其应用	147
6.6.1	基本术语	147
6.6.2	构造哈夫曼树	148
6.6.3	哈夫曼树的应用	150
	学习指导	152
	习 题	153

第7章 图	156
7.1 图的定义和术语	156
7.2 图的存储结构	158
7.2.1 邻接矩阵	158
7.2.2 邻接表	159
7.2.3 十字链表	160
7.2.4 邻接多重表	161
7.3 图的遍历	162
7.3.1 深度优先搜索	162
7.3.2 广度优先搜索	165
7.4 图的应用	166
7.4.1 图的连通性	166
7.4.2 最小生成树	166
7.4.3 拓扑排序	171
7.4.4 最短路径	173
学习指导.....	177
习 题.....	179
第8章 查找	181
8.1 查找的基本概念	181
8.2 顺序表查找	182
8.2.1 顺序查找	182
8.2.2 折半查找	183
8.3 索引表查找	185
8.3.1 索引查找	185
8.3.2 分块查找	185
8.4 树表查找	186
8.4.1 二叉排序树	186
8.4.2 平衡二叉树	189
8.4.3 B-树	192
8.5 哈希表查找	196
8.5.1 哈希表与哈希函数	196
8.5.2 处理冲突的办法	197
8.5.3 哈希表的查找	199
学习指导.....	201
习 题.....	202
第9章 排序	204
9.1 排序的基本概念	204

9.2 插入排序	205
9.2.1 直接插入排序	205
9.2.2 折半插入排序	206
9.2.3 希尔(shell)排序	207
9.3 交换排序	209
9.3.1 冒泡排序	209
9.3.2 快速排序	210
9.4 选择排序	212
9.4.1 简单选择排序	212
9.4.2 堆排序	213
9.5 归并排序	216
9.5.1 两个有序文件的归并	216
9.5.2 归并排序	217
9.6 基数排序	219
9.7 各种排序方法的比较	219
学习指导	221
习 题	222
第10章 文件	223
10.1 外部存储设备	223
10.1.1 磁带	223
10.1.2 磁盘	224
10.2 文件的基本概念	225
10.2.1 文件的定义及其分类	225
10.2.2 文件的逻辑结构及物理结构	226
10.3 顺序文件	228
10.4 索引文件	230
10.5 索引顺序文件	232
10.6 散列文件	236
10.7 多关键字文件	237
10.7.1 多重链表文件	237
10.7.2 倒排文件	238
10.8 典型应用举例	239
10.8.1 顺序文件的批量处理实例	239
10.8.2 文件的随机访问实例	241
学习指导	244
习 题	244

第 11 章 《数据结构》实验指导	245
11.1 实验一 线性表及其应用	247
11.1.1 课程成绩管理	247
11.1.2 一元稀疏多项式简单计数器	247
11.2 实验二 栈和队列及其应用	249
11.2.1 停车场管理	249
11.2.2 车厢调度	251
11.2.3 汽车加油	251
11.3 实验三 串及其应用——简单行编辑程序	252
11.4 实验四 树及其应用	254
11.4.1 二叉树的中序遍历及线索化	254
11.4.2 重言式判别	254
11.5 实验五 图及其应用——跳马将军	255
附录	257
源程序 1	257
源程序 2	263
参考文献	269

第 1 章

绪 论

众所周知,著名计算机科学家 Niklaus Wirth 提出:算法+数据结构=程序。

“程序”是大家熟悉的概念,计算机程序是人们利用计算机解决某一问题时向计算机提供的、用一种程序设计语言书写并能被计算机执行的一组规则和步骤,以语句(或指令)的形式回答了“做什么”和“怎么做”的问题。程序被计算机运行,运行的对象是数据,运行的目的是加工、处理数据以得到问题的解。因此,程序包括两部分内容:数据(按一定方式存储的数据)和操作(处理数据的过程)。数据的存储方式即数据结构研究的内容,而数据的处理过程即程序的设计,首要的是算法的设计。数据结构、算法、程序是密不可分的,程序的研制开发实际上是数据结构和算法的研究。在程序的开发过程中通常要做到两点:一是如何高效描述数据;二是如何设计一个好的算法,该算法最终可用程序来实现。程序设计的实质就是对一个确定的问题选择出一种好的数据结构和设计一种好的算法。要想高效描述数据,必须具备数据结构的专门知识;要想设计一个好的算法,则需要算法设计领域的专门知识。

因此,本章围绕什么是数据结构,什么是算法展开讨论。

1.1 什么是数据结构

计算机突飞猛进的发展,不仅体现在计算机本身运算速度的提高,信息存储量的增大,而且更重要的是其应用范围的拓广。从最初的工程和科学计算逐步发展到信息处理、事务管理、工业控制、CAD/CAM/CIMS 等人类活动的各个领域。计算机处理的对象也从纯数值性数据,如整数、实数、复数、双精度数,扩展到字符、图形、图像、声音等非数值性数据,因此,我们可以把计算机的处理对象统称为数据,同时给数据下一个定义:

数据(data):数据是客观事物的符号表示,在计算机领域中,是指所有能输入到计算机中并被计算机程序处理的符号总称。

用计算机对应用领域的数据进行处理,必须由软件人员根据需要编制程序,一般经过这样几个阶段:①对具体问题进行分析,抽象出一个适当的数学模型;②设计针对该数学模型的算法;③根据算法编制程序;④对程序进行调试、测试,直至得到最终解答。4个步骤中,抽象出问题的可描述数学模型是求解实际问题关键的一步,而建立数学模型的实质是分析具体问题,从中提取操作对象——数据,并找出数据之间的关系,用数学语言描述。对于工程和科学计算问题,其数学模型往往可用微分方程、代数方程等数学方法描述,如预报人口增长率的数学模型是微分方程。而应用领域的很多非数值性数据处理问题无法找到相应的数学公式建立其数

数据结构

学模型,则需要选择合适的数据结构来描述和存储这些非数值性数据及其关系。下面的几个例子分别选用不同的数据结构加以描述。

例 1.1 同学通讯录管理

要利用计算机管理同学通讯录,首先需将每个同学通讯情况存入计算机中。最简单的方法是建立一张如图 1.1 所示的表,表中的每一行存放一个同学的通讯信息,每个同学在表中占一行。

学 号	姓 名	性 别	出生日期	家庭住址	电话	E-mail
003201	王 昆	男	83/02/21	桂林市中山北路 1000 号	0773-2821356	Wangkun@263.net
003202	赵 铭	女	82/10/08	上海市国权路 508 号	021-65432222	Zhaoming@sina.net
⋮						
⋮						

图 1.1 同学通讯录

同学通讯录以表的形式存入计算机后,计算机的处理对象为同学的相关信息即数据。每个同学的信息,在计算机程序中作为一个整体进行考虑和处理,是数据的基本单位,称作数据元素(Data Element)。图 1.1 同学通讯录中的一行就是一个数据元素,这样的数据元素是由若干数据项组成的,构成一个同学通讯信息的数据项有学号、姓名、性别、出生日期、家庭住址、电话、Email 等。数据项(Data Item)是使数据有意义的最小单位,不能再分割。若将数据项如姓名再分割就没有实际意义了。当然,数据元素也可能仅由一个数据项构成,如英文字母表中的数据元素就是 A,B,C…各个字母,季节表中的数据元素为春、夏、秋、冬,整数表{12,35,46,25,78}中的数据元素分别为 12,35,46,25,78。总之,在数据处理领域中,每一个需要处理的对象都可以抽象成数据元素。

存储在计算机中的同学通讯录,每个数据元素即每个同学的信息是一个接一个存放的,数据元素之间是一种简单的线性关系,逻辑上是一种线性结构,这类数学模型可称为线性数据结构,存储通讯录的表为线性表。对通讯录的管理就转化为对线性表的操作即线性表的运算。如转入新同学,即在线性表中增加一行新同学的通讯信息;转出一位同学,即在线性表中删除该同学的通讯信息;查询某同学的电话号码或电话号码改变时,即在线性表中查找该同学所对应的数据元素。也就是说,对通讯录的查询、修改、同学信息的增加、减少等动作表现为定义在线性表中数据元素的查找、修改、插入、删除等运算。对于这些运算,要求设计“查找”、“修改”、“插入”、“删除”等算法。可以发现,算法的设计完全依赖于线性表的数据元素在计算机内的存储方式。就“查找”运算而言,从直观上来说,“查询某同学的电话号码”的操作与通讯录的组织方式有关,通讯录有多种组织方式,可随意排列,也可按学号排列,或按姓名的字母顺序排列,对于不同的组织形式,该操作的执行效率是不同的。同样,“查找”运算的算法设计与线性表的存储方式密切相关。

图书馆书目检索、仓库账目管理等常见的二维表管理如同通讯录管理一样,数据元素之间都是一种线性关系,在计算机处理时都可抽象为线性表这一数据结构。

例 1.2 建立人事档案

建立某学校教师的人事档案也是数据结构的典型例子。逻辑上这种结构如图 1.2 所示。很显然,它实际上是一种树状结构。计算机处理的对象是树中结点上的数据,即为数据元素,数据元素之间的关系是一种层次关系,教师的信息存放在树的叶子上。

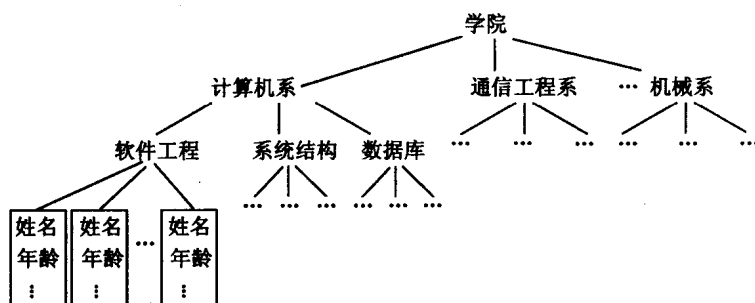


图 1.2 教师人事档案结构

下面讨论关于询问某教师有关情况的处理问题。为了处理这个问题，首先必须解决如何把这些数据存储到计算机中，也就是要解决数据的物理存储问题。假如这个问题已解决，那么又如何实现对于某教师有关情况的查询？若查询给出了该教师的所在的系和专业，查询可方便地从树根沿某一分支找到该教师的信息。若查询者根本不知道该教师属于何系何专业，则问题的解决就复杂得多，必须按系按专业逐一查找，这就是所谓的树的遍历问题。还会出现这样的问题：当增加一个系或专业时，需在树中增加结点，有新教师调入时需增加叶子结点，这就是树结构的插入问题。若某系或某专业被取消时，必定会影响树结构，怎样将该系或该专业的所有教师的数据从树中删除？这实际上是树的删除问题，而删除的过程势必引起有关的系和专业的变动，需要进行重新整理，就会涉及有关结点的删除和插入问题。因此，为了适应数据的查询、增加、减少，必须定义相应的查找、插入、删除的运算及其算法，并保证在插入、删除之后不破坏树结构。

另一个树状结构的例子是近几年来较热的人机对弈问题。计算机能够和人进行对弈，是因为有人将对弈的策略存入计算机。而对弈是在一定规则下随机进行的，不仅要看棋盘当时的格局，还要能够预测将来将可能发生的趋势。这是一个相当复杂的问题，不仅需要数据结构的知识，还深入涉及到人工智能的领域。如图 1.3(a)所示为井字棋，计算机处理的对象是对弈过程中某时刻出现的格局，当前格局对应的下一个格局可能有 5 种，而从每个新格局又可派生出 4 个新格局。格局与被派生的格局之间是一种层次关系，构成树状结构。

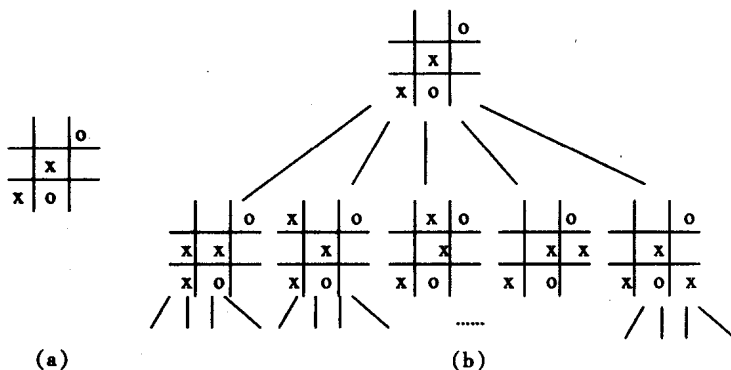


图 1.3 井字棋对弈树

(a) 棋盘格局 (b) 对弈树的局部

例 1.3 旅行家旅行最短路径的问题

举例来说，一些城市之间有道路相连，道路的长度已知，如图 1.4 所示，现在要求有这么一

数据结构

个系统来满足旅行家提出的一些问题:如要从 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。这是一个以城市为顶点,以城市和城市之间相连的道路为连线构成的图状结构或网状结构,这类道路交通问题的数学模型就是一种称为“图”的数据结构。

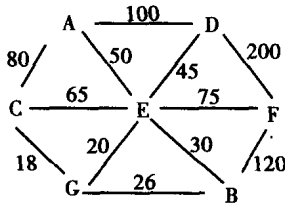


图 1.4 城市道路交通结构图

综上所述,描述这类非数值计算问题的数学模型不再是数学方程,而是诸如线性表、树、图之类的数据结构。从上述的例子中,大家应该知道数据结构是处理哪些领域的问题,同时,也可看出,作为计算机的操作对象是一组具有同样性质的数据元素的集合,可称为数据对象(Data Object)。而且数据元素不是孤立存在的,相互之间存在某种结构关系,因此,把相互之间存在一种或多种特定关系的数据元素的集合定义为数据结构(Data Structure)。

数据结构和数据对象不同,在描述一种数据结构时,不但要描述数据对象,还要描述数据元素之间的相互关系。

1.2 数据的逻辑结构和存储结构

1.2.1 数据的逻辑结构

在了解了数据结构的定义之后,再来看一看数据结构所研究的对象:数据元素。而要对数据元素进行抽象研究,就要对其进行分类。在本课程中,选择的分类标准就是数据元素之间的关系,根据数据元素之间关系的不同特性,通常有下列 4 类基本结构:

集合:结构中的数据元素除了同属于一个集合的关系外,并无其他逻辑关系,即数据元素之间的关系是一种空关系,这是一种最松散的组织形式。

线性结构:结构中的数据元素之间存在一个对一个的关系,是按线性关系或前后关系来组织的。

树形结构:结构中的数据元素之间存在一个对多个的关系,具有分支及层次特性,其形态像一颗倒挂的树。

图状结构或网状结构:结构中的数据元素之间存在多个对多个的关系,任意两个数据元素都可邻接;是一种最复杂的关系。

图 1.5 为上述 4 类基本结构的关系图,使我们直观感受了“数据结构”,严格地说,数据结构是一个二元组 $\text{Data_Structure} = (D, R)$,其中, D 是数据元素的有限集, R 是 D 上关系的有限集。这里, R 集合中的关系指的是数据元素之间的逻辑关系,因而这种结构又称为数据的逻辑结构。

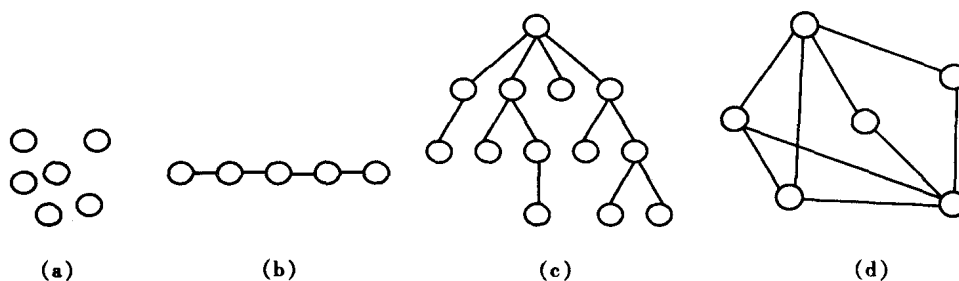


图 1.5 4 类基本结构关系图

(a)集合 (b)线性结构 (c)树 (d)图

数据的逻辑结构是从解决问题的需要出发,为实现必要的功能所建立的数学模型即数据结构,属于用户视图,是面向问题的。其实,讨论数据结构的目的是为了在计算机中实现它的操作,因此,重点还需研究如何在计算机中表示它。

1.2.2 数据的存储结构

数据结构在计算机中的表示(又称映像)称为数据的物理结构,又称存储结构,是数据逻辑结构的物理存储方式,属于具体实现的视图,是面向计算机的。数据的存储结构包括数据元素的存储表示和数据元素之间逻辑关系的存储表示。例如,一个简单字母表(A,B,C,D,E)是一个线性表,其存储可以实现为一字符数组 $ch[0..maxn]$,每个数据元素存储在一个存储单元(字节),数据元素之间的线性关系则隐式地表示为数组元素的顺序连续存放,如图 1.6(a)所示。显然,从逻辑结构到存储结构的映像中,数据元素可以存储在字节、字或多个字构成的位串中,这个位串称为结点。结点可以看成数据元素在计算机中的映像,而存储结构关键取决于逻辑关系的存储表示,有两种方式:

(1) 顺序存储方式

如图 1.6(a)所示,每个数据元素对应一个结点,所有存储结点的物理位置相继存放在一个连续的存储区内。也就是说,在顺序存储方式下,数据元素之间的逻辑关系隐含体现在存储结点的位置关系中,这样的存储结构称为顺序存储结构。

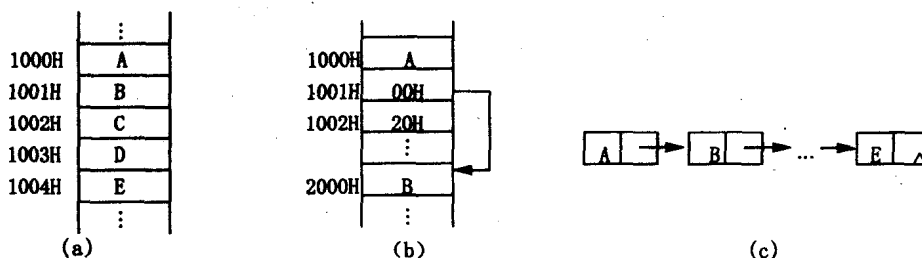


图 1.6 简单字母表的存储结构

(a)顺序存储结构 (b)链式存储结构 (c)链式存储结构形象描述

(2) 链式存储方式

在这种方式下,每个数据元素对应的存储结点包括数据元素和指针两部分,借助指示元素存储地址的指针(Pointer)显式表示数据元素的逻辑关系,如图 1.6(b)、图 1.6(c)所示。这样的存储结构称为链式存储结构。

1.3 存储结构的表示与实现

如何描述存储结构呢?虽然存储结构涉及数据元素及其关系在存储器中的物理位置,但不可能直接以内存单元地址来描述存储结构,可以借用高级程序语言中的“数据类型”来描述。如可以用“一维数组”类型描述顺序存储结构,以“指针”类型描述链式存储结构。

1.3.1 数据类型

首先,以C语言为例,回顾一下程序设计语言中的数据类型。C语言有5种基本的数据类型:字符型(char)、整型(int)、浮点型(float)、双精度型(double)和无值(void)。这些数据类型不但规定了使用该类型的取值范围,而且规定了取值范围内的数据可进行的一组操作。如C语言中的整型,取值范围为 $-32\ 767 \sim 32\ 767$ 的整数,相关的操作有 $+$ 、 $-$ 、 $*$ 、 $/$ 等。

事实上,数据类型(Data Type)是一组性质相同的值的集合以及定义在这个值集上的一组操作的总称,分为原子类型和结构类型两种。原子类型为值在逻辑上不可分解的数据类型,如上述C语言的基本数据类型。同时,在C语言或PASCAL语言这样的程序设计语言中,不但规定了一些基本的数据类型,还提供了一些组合类型(如数组型、构造型、文件型等)的规则。数据元素的值由若干成分按某种结构组成,称为结构类型。程序员可以利用结构类型的规则,自行定义为解决应用问题所必需的数据类型。

数据类型封装了数据存储与操作的具体细节。程序员对“两整数求和”,只关心其“数学上求和”的抽象特性,而不需了解“整数”在计算机中如何表示,相加操作如何实现,这是信息隐藏或封装的概念。

1.3.2 抽象数据类型

从数据类型到抽象数据类型,“抽象”的意义在于数据类型的数据抽象特性的反映。数据的抽象,以一个例子说明:数据在计算机中以二进制数存储和运算的,而汇编语言程序设计中可使用各种数据的自然表示,如十进制数65.3,100等,而不必考虑十进制到二进制的转换,这是二进制数的抽象;在高级语言程序设计中,出现字符型、整型、浮点型、双精度型等,这是更高一级的数据抽象;待到抽象数据类型出现,可以定义更高级的数据抽象,如线性表、栈、队列、树、图等。抽象数据类型可以使我們更容易描述现实世界。例:用线性表描述学生成绩表,用树或图描述遗传关系。

抽象数据类型通常是指由用户定义,用以表示应用问题的数学模型以及定义在该模型上的一组操作。抽象数据类型由数据类型组成,并包括一组相关的操作。使用它的人可以只关心它的逻辑特征,不需要了解它的存储方式,定义它的人同样不必要关心它如何存储。如线性表这样的抽象数据类型,其数学模型是数据元素的集合,该集合内的元素有这样的关系:除第一个和最后一个外,每个元素有惟一的前趋和惟一的后继。可以有这样一些操作:插入一个元素、删除一个元素等。

抽象数据类型可以用三元组表示: (D, R, P) ,其中D是数据对象,R是D上的关系集,P是对D的基本操作集。其定义格式为: