数据结构(PASCAL 语言)

杜世培　主编

重庆大学出版社

内容简介

本书系统地介绍各种常用的数据结构和排序、查找的各种方法。阐述了各种数据结构的逻辑结构、存储结构及运算操作，并对类 PASCAL 语言描述的算法作了详细的注解和简要的性能分析。全书既注重原理又注重实践，配有大量的例题、习题和上机实习作业，内容丰富，概念讲解清楚，逻辑性强，可读性好。书中针对不同层次教学的特点和需要，用“ * ”号标明不同要求的区别。

本书可作为高等学校计算机及应用专业学生的教材，亦可作为成人教育(面授或函授)的教材，还可供广大从事计算机应用的科技人员参考。

数据结构(PASCAL 语言)

杜世培　主编

责任编辑：曾令维　版式设计：曾令维

责任印制：张立全

*

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：400030

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

重庆升光电力印务有限公司印刷

*

开本：787×1092　1/16　印张：15.25　字数：380 千

1997 年 6 月第 1 版　2003 年 7 月第 4 次印刷

印数：17 001- 20 000

ISBN 7-5624-1342-8/TP・120　定价：17.00 元

序

面对知识爆炸, 社会学家们几乎都开出了一个相同的药方: 计算机。计算机也深孚众望, 以其强大的功能, 对人类作出了巨大的贡献, 取得了叹观止矣的成就。自它 1946 年 2 月 14 日在美国费城诞生以来, 至今已过“ 知天命 ”的年龄了。现在, 计算机已是一个庞大的家族。如果说, 它的成员占据了世界的每一个角落和每一个部门也并不过分, 甚至找不到这样一个文明人, 他的生活不直接或间接与计算机有关。 目前, 全世界计算机的总量已达数亿台, 而且, 现在正以每年几千万台的速度增长。

作为计算机在信息传递方面的应用, 计算机加上网络, 被认为是和能源、交通同等重要的基础设施。这种设施对信息的传递起着异常重要的作用。西方发达国家和我们国家对此都非常重视。例如, 美国的信息高速公路计划, 全球通讯的“ 铱 ”计划, 我国也开始实行一系列“ 金 ”字头的国民经济管理信息化计划。这些计划中唱主角的设备便是计算机。计算机在各个方面的应用不胜枚举, 我们每个人都自觉不自觉地处于计算机包围中。

计算机对社会生产来说是一个产业大户, 对每个现代人来说是一种工具, 对学生们来说, 它是一个庞大的知识系统。 面对计算机知识的膨胀, 面对计算机及其应用产业的膨胀, 计算机各个层次的从业人员的需要也在不断膨胀, 计算机知识的教育也遍及从小学生到研究生的各个层次。

为了适应计算机教学的需要, 重庆大学出版社近几年出版了大量的计算机教学用书, 这一套教材就是一套适应专科层次的系列教材。我们将会看到, 这一套教材以系列、配套、适用对路, 便于教师和学生选用。 如果再仔细研究一下, 将会发现它的一系列编写特色:

1. 这些书的作者们是一些长期从事计算机教学和科研的教师, 不少作者在以前都有大量计算机方面的著作出版。 例如本系列书中的《Visual Fox Pro中文版教程》的作者, 十年前回国后最早将狐狸软件介绍到祖国大陆, 这一本书已是他的第八本著作了。 坚实的作者基础,

是这套书成功的最根本的保证。

2. 计算机科学是发展速度惊人的科学, 内容的先进性、新颖性、科学性是衡量计算机图书质量的重要标准, 这一套书的作者们在这方面花了极大的功夫, 力求让读者既掌握计算机的基础知识, 又让读者了解最新的计算机信息。

3. 在内容的深度和知识结构上, 从专科学生的培养目标出发, 在理论上, 从实际出发, 满足本课程及后续课程的需要, 而不刻意追求理论的深度。在知识结构上, 考虑到全书结构的整体优化, 而不过分强调单本书的系统性。这样, 在学过这一套系列教材后, 学生们就可在浩瀚的计算机知识中, 建立起清晰的轮廓, 就会知道这些知识的前因后果, 就会了解这些知识的前接后续。使学生们能在今后的工作实践中得心应手。

4. 计算机是实践性很强的课程, 仅靠坐而论道是学习不了这些知识的。所以从课程整体设置来讲, 包括有最基本的操作技能的教材。对单本书来说, 在技术基础课和专业课中, 都安排有一定的上机实习或实验, 这样可使学生既具备一定的理论知识以利今后发展和深造, 又掌握实际的工作技能胜任今后的实际工作。

编写一套系列教材, 这是一个巨大的工程。这一套书的作者们, 重庆大学出版社的领导和编辑们, 都为此付出了辛勤的劳动。作为计算机工作者, 以此序赞赏他们的耕耘, 弘扬他们的成绩。

1997 年 6 月 15 日

前　言

《数据结构》在计算机科学技术中是一门专业基础课，它的教学要求是：学会分析研究计算机加工的数据对象的特性，以便选择适当的数据结构、存储结构以及相应的算法，初步掌握算法性能分析方法；使学生通过本课程的学习，获得更进一步的程序设计训练。因此，在我国计算机专业的教学计划中，它是核心课程之一。

本书属于计算机专业系列教材。全书共十章，详细介绍了线性表、栈和队列、串、数组、树和二叉树以及图等常用的数据结构，以及在程序设计中经常遇到的两个问题——查找和排序。讨论上述数据结构的逻辑结构，在计算机中的存储结构以及在这些数据结构上的操作算法，其中大多数算法都用类 PASCAL 语言进行描述，并对算法的性能进行简要的分析和讨论。

本书可作为高等学校计算机及应用专业学生的教材，讲授 60～80 学时。在学时少的情况下，讲授教师可根据本校学生的情况适当删去目录中用“＊”号标志的内容。本书在每一章后面，都给出了习题，在第二、三、六、七、八和九章后面给出了上机实习题，供使用者选用。教师可根据实际情况安排学生上机 20～30 机时。本书亦可作为成人教育（面授或函授）的教材，还可供广大从事计算机应用的科技人员参考。为了便于读者自学，本书在文字叙述上力求做到语言通俗、简明易懂，其中大多数的算法都用类 PASCAL 语言进行描述，只要稍加修改，便可变成能上机执行的 PASCAL 程序。本书还直接给出个别算法的可上机执行的 PASCAL 程序，便于读者使用。

本书的第一章、第八章、第九章和第十章由杜世培编写，第二章、第三章由曾潇编写，第四章、第六章由卢霞编写，第五章、第七章由宋文编写。全书由杜世培统稿、修改。

由于编著者水平有限，书中缺点或错误在所难免，殷切希望广大读者批评指正。

编著者

1997 年 3 月

目　　录

第一章　绪论 …… 1
§ 1.1　数据结构的基本概念 …… 1
§ 1.2　数据结构与算法 …… 6
§ 1.3　数据结构发展概况及其在计算机科学中的地位…… 10
习题一 …… 10

第二章　线性表 …… 12
§ 2.1　线性表的逻辑结构…… 12
§ 2.2　线性表的顺序存储结构…… 13
§ 2.3　线性表的链式存储结构…… 18
§ 2.4　线性表实现的综合评价及其应用…… 27
§ 2.5　线性表的应用举例: 多项式的表示和相加 …… 34
习题二 …… 37
上机实习一 …… 39

第三章　栈和队列 …… 41
§ 3.1　栈…… 41
§ 3.2　栈的应用举例…… 46
§ 3.3　队列…… 50
习题三 …… 57
上机实习二 …… 57

第四章　串 …… 59
§ 4.1　串的概念…… 59
§ 4.2　串的存储结构…… 61
§ 4.3　串基本操作的实现…… 65
§ 4.4　串的应用举例——文本编辑…… 69
习题四 …… 71

第五章　数组 …… 72
§ 5.1　数组的定义与操作…… 72
§ 5.2　数组的顺序存储结构…… 73
§ 5.3　稀疏矩阵的压缩存储方法…… 74
习题五 …… 86

第六章　树和二叉树 …… 88
§ 6.1　树 …… 88
§ 6.2　二叉树 …… 90
§ 6.3　遍历二叉树 …… 94
§ 6.4　线索二叉树 …… 100
§ 6.5　树和森林 …… 105
§ 6.6　哈夫曼树及其应用 …… 109
习题六 …… 113
上机实习三 …… 117

第七章　图 …… 118
§ 7.1　图的定义和术语 …… 118
§ 7.2　图的存储结构 …… 121
§ 7.3　图的遍历 …… 126
§ 7.4　图的连通性 …… 130
§ 7.5　有向无环图及其应用 …… 136
§ 7.6　最短路径及其应用 …… 143
习题七 …… 148
上机实习四 …… 150

第八章　查找 …… 151
§ 8.1　线性表的查找 …… 151
§ 8.2　树表的查找 …… 157
§ 8.3　哈希(Hash)技术 …… 166
习题八 …… 172
上机实习五 …… 172

第九章　排序 …… 174
§ 9.1　基本概念 …… 174
§ 9.2　插入排序 …… 175
§ 9.3　交换排序 …… 178
§ 9.4　选择排序 …… 181
§ 9.5　归并排序 …… 186
*§ 9.6　基数排序 …… 188
*§ 9.7　外部排序简介 …… 191
习题九 …… 193
上机实习六 …… 194

第十章　文件…… 195
§ 10.1　文件的基本概念…… 195
§ 10.2　顺序文件…… 197
§ 10.3　索引文件…… 199
§ 10.4　直接存取文件…… 205
*§ 10.5　多关键字文件 …… 206
习题十…… 208

附录一　类 PASCAL 语言语法概要 …… 209
附录二　名词索引…… 212
附录三　上机实习报告提纲及范例…… 218

参考文献…… 234

第一章　绪　论

本章介绍了数据结构课程的研究对象; 基本概念和术语; 算法的概念, 描述算法的类 PASCAL 语言; 数据结构的发展概况及其在计算机科学中的地位。

§ 1.1　数据结构的基本概念

1.1.1 “ 数据结构 ”课程的研究对象

用计算机解决一个实际问题时, 一般的步骤是: 首先得到实际问题的数学模型, 然后设计相应的算法, 最后编写程序, 调试、完善, 直至得到问题的答案。当人们用计算机处理数值计算问题时, 所用的数学模型是用数学方程描述的, 所涉及的运算对象一般是简单的整型、实型和逻辑型数据, 因此程序设计者的主要精力是集中于程序设计技巧上, 而不是数据的存储和组织上; 然而, 目前计算机应用的更多领域是“ 非数值计算问题 ”, 它们的数学模型却无法用数学方程描述, 而是用“ 数据结构 ”描述, 解决此类问题的关键是设计出合适的数据结构, 下面用两个例子对此作一说明。

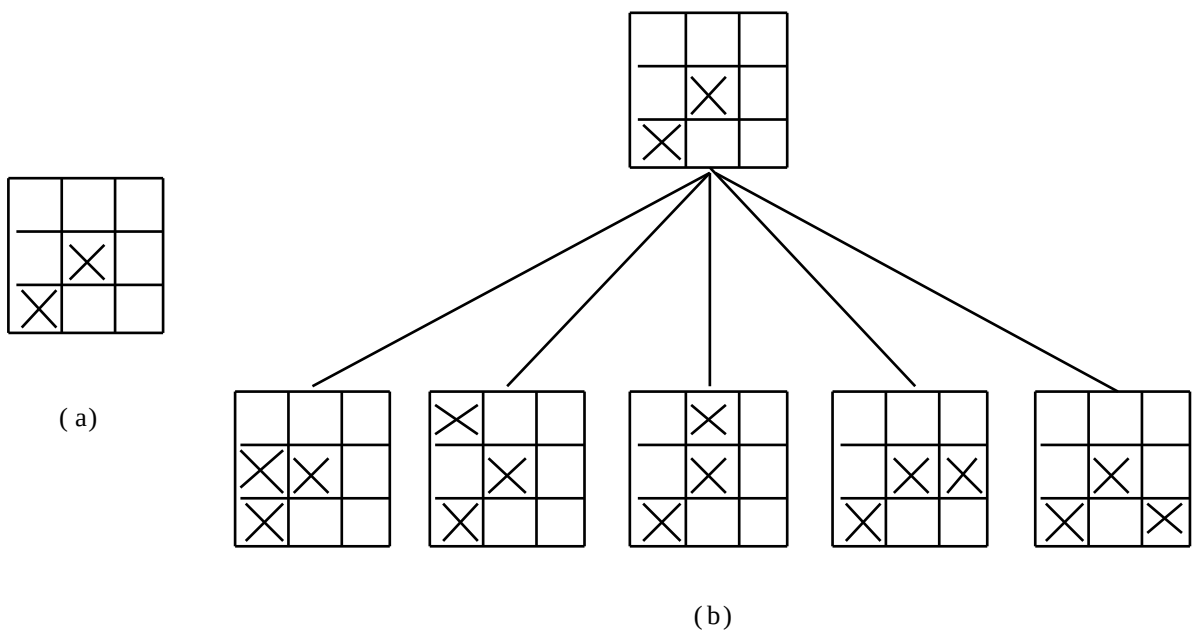
例 1.1　高校学生学籍管理自动化问题。高校的学生管理部门要负责全校学生的管理工作, 其中, 学籍管理是一个重要的工作。为此, 经常要查询学生的学号、姓名、性别、年龄及其他信息并作各种统计、汇总工作, 其他部门及相应的系也需要了解这些信息。如何利用计算机来辅助管理呢? 首先要考虑的是对学生的各种信息如何加以组织和存储。常用的方法是建立一个表, 每个学生的信息由学号、姓名、性别、年龄等项目组成, 占表的一行, 如表 1.1 所示, 在表中, 各个学生的信息是一个接一个地排列的, 这种表称为线性表, 它构成了计算机辅助管理学籍的数学模型。这样, 上面的问题就归结为用计算机对线性表这种数学模型所进行的一系列操作, 用户可以根据需要, 按各种不同的项目进行查询、统计和汇总等工作, 从而快速、准确地得到结果。

表 1.1　学生名册线性表示例

学　　号	姓　名	性别	年龄	备　　注
9501450021	张明生	男	19	
9501450022	李　萍	女	20	
…	…	…	…	

例 1.2　计算机和人对奕问题。计算机能和人对奕是由于有人事先已将对奕的规则、策略存入计算机, 使计算机既能“ 看到 ”棋盘当前的状态(这也就是计算机的操作对象, 称为格局), 又能预测棋局发展的趋势, 这一切是通过计算机操作处理格局及格局的相互关系来实现的。现

以井字棋为例。这种棋由两人在 3× 3 的方格棋盘上对奕,当一方的三个棋子占同一行、或同一列、或同一对角线时为胜方。用× 表示先行方,用○表示后行方,图 1.1(a)表示其中一个格局,图 1.1(b)表示由这个格局派生出来的五个格局,而从每一个新格局又可派生出四个可能出现的格局。若将从对奕开始到结束的过程中所有可能出现的格局都画在一张图上,则可得到一棵倒长的“树”。“树根”是对奕开始之前的棋盘格局,而所有的“叶子”,就是可能出现的结局,对奕的过程就是从树根沿树叉到某个叶子的搜索过程。因此,树就是解决计算机和人对奕问题的数学模型。



(a)

(b)

图 1.1　井字棋对奕“树”

(a) 棋盘格局示例　(b) 对奕树的局部

从以上两个例子可以看出,描述非数值计算问题的数学模型是诸如线性表、树等等之类的数据结构,这也就是数据结构课程的研究对象。一般说来,数据结构是一门研究非数值计算的程序设计问题中计算机的操作对象以及它们之间的关系和操作等等的学科。

1.1.2　基本概念和术语

数据(data)　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计算机中并被计算机程序识别和处理的符号的总和。它是计算机程序加工处理的对象。客观事物包括数值、字符、声音、图形、图像等等,它们本身并不是数据,只有通过编码变成能被计算机识别、存储和处理的符号形式才是数据。

数据元素(element)　是数据的基本单位,在计算机程序中通常是作为一个整体进行考虑和处理的。例如,例 1.1 的表中的一行,例 1.2 的“树”中的一个棋盘格局都是一个数据元素。有时,一个数据元素可由若干个数据项(item)组成,例 1.1 中构成一个学生的信息的每一项目(如学号、姓名、性别、年龄等)是一个数据项。数据项是数据的不可分割的最小单位。

数据是一个集合的概念,有时需要对这个集合中某些具有相同性质的元素组成的子集进行讨论,这时就要使用数据对象这个术语。

数据对象(object)　是性质相同的数据元素的集合,是数据的一个子集。例如整数数据对象是集合 N= {0, ±1, ±2, …},字母字符数据对象是集合 C= {‘A’,‘B’, ……,‘Z’}。

数据结构要研究的不是几个孤立的数据元素,而是大量的相互关联的数据。将数据元素之

间的相互关系称为结构(structure)。“数据结构”这个概念的含义可以认为是由“数据”和“结构”有机组成的。

数据结构(data structure)是相互之间存在一种或多种关系的数据元素的集合。这个概念涉及了两个内容:一个是数据元素,另一个是数据元素之间的相互关系,数据元素不是孤立存在的,在它们之间总存在某种相互关系。

从集合论的观点出发,数据结构是由两个集合构成的一个二元组

$$Data\_Structure = (D, S) \tag{1-1}$$

其中:D 是同属一个数据对象的数据元素的有限集合,S 是 D 上关系的有限集合。式(1-1)称为数据结构的形式定义。下面举一个例子说明这种定义方法。

例 1.3　在例 1.1 中,学生名册可以看作一种数据结构

$$Student = (D, S) \tag{1-2}$$

其中:数据 $D = \{S_1, S_2, \cdots\cdots, S_n\}$,$S_1$ 表示学生 1,S_2 表示学生 2,其余类推。

结构 $S = \{\langle S_1, S_2\rangle, \langle S_2, S_3\rangle, \cdots\cdots, \langle S_{n-1}, S_n\rangle\}$,$\langle S_1, S_2\rangle$表示学生 2 排在学生 1 的后面,$\langle S_2, S_3\rangle$表示学生 3 排在学生 2 的后面,其余类推。需要说明的是,在式(1-2)中关系的集合 S 中只有一种关系:学生之间的关系。本书以后的讨论,如不特别声明,一般只考虑一种关系的情况。

在以后的讨论中,关于数据结构将多次使用逻辑结构和物理结构这两个术语。前者不涉及数据结构在机内的表示,比较抽象,而后者则涉及它在机内的表示,比较具体。

上述数据结构定义中的“关系”描述的是数据元素之间的抽象化的相互关系,这种数据元素之间的相互逻辑关系通常称为逻辑结构,它与数据的存储无关,是独立于计算机的。

根据数据元素之间关系的不同特性,有下列四类基本逻辑结构:(1)集合结构中的元素之间除了“同属一个集合”的相互关系之外,别无其他关系;(2)线性结构中的元素之间存在一个对一个的相互关系;(3)树形结构中的元素之间存在一个对多个的相互关系;(4)图形结构中的元素之间存在多个对多个的相互关系。图 1.2 为上述四种基本逻辑结构的关系图。

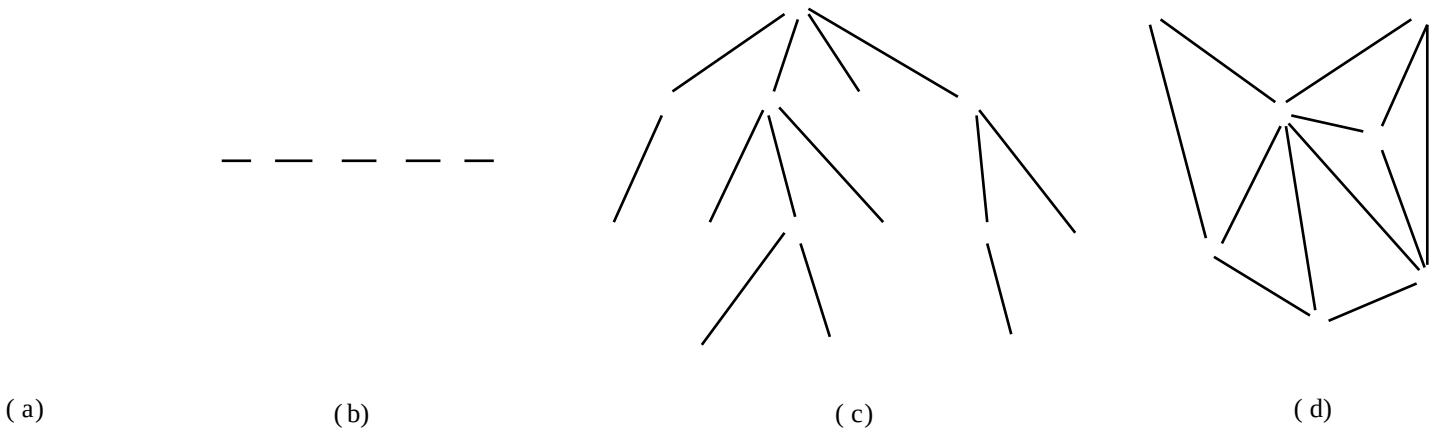


图 1.2　四种基本逻辑结构关系图

(a) 集合　(b)线性结构　(c)树形结构　(d) 图形结构

数据结构在计算机中的表示(又称映象)称为数据的物理结构,又称为存储结构。它包括数据元素的机内表示和关系的机内表示,下面分别讨论一下。

数据元素的机内表示　在计算机中表示信息的最小单位是二进制的位(bit)。在计算机中,可以用由若干位组合起来形成的一个位串,表示一个数据元素(如用一个八位二进制位串表示一个字符),通常称这个位串为结点(node)。当数据元素由若干数据项组成时,位串中与各

数据项对应的子位串称为数据域(data field)。因此,结点是数据元素的机内表示(或机内映象)。

数据元素之间关系的机内表示 可以分为顺序映象和非顺序映象,由此可以得到两种不同的存储结构:顺序存储结构和链式存储结构。顺序映象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。非顺序映象借助指示元素存储位置的指针(pointer)来表示数据元素之间的逻辑关系。在下面的例子中,说明一下这两种存储方法。

例 1.4 序列 A=(12, 23, 42, 33, 45)的两种存储结构。假设用两个字节存储一个数据元素,图 1.3(a)表示顺序存储结构,其中首元素 12 存放在地址为 0100 的单元中,占两个字节,第二个元素存放在地址为 0102 的单元中,占两个字节,其余类推。

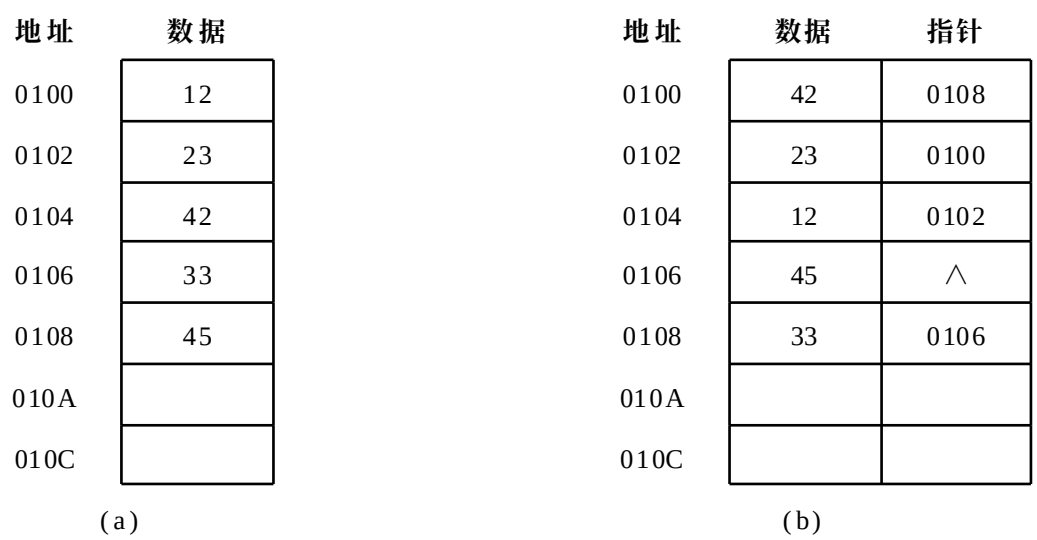


图 1.3 序列 A 存储结构示意图

(a)顺序存储结构 (b) 链式存储结构

由这种存储方式,容易确定序列中任一个元素的位置,如第三个元素 42 的位置是 0100+(3- 1)* 2= 0104,可以通过计算公式直接确定。图 1.3(b)表示链式存储结构,其中首元素 12 存放在地址为 0104 的单元中,第二个元素 23 的存储位置由首元素的指针域指示出来,为地址是 0102 的单元,而第三个元素的存储位置又由第二个元素的指针域指示出来,为地址是 0100 的单元,其余类推。最后一个元素 45 的存储位置为地址是 0106 的单元,它的指针域存放一个空指针∧。在这种存储方式中,要确定序列中任一个元素的位置,都必须从第一个元素开始,顺藤摸‘瓜”,如要确定第三个元素的位置,必须先由首元素 12 的指针域找到第二个元素 23 的存储位置,再由第二个元素的指针域找到第三个元素 42 的存储位置 0100。这两种存储结构各有优缺点,在以后的章节中进一步讨论。

数据的逻辑结构和存储结构是密切相关的两个方面,任何一个算法的设计取决于选定的数据的逻辑结构,而算法的实现则依赖于所采用的存储结构。

存储结构怎样具体表示呢?由于本书是在高级语言的层次上讨论对数据的操作,因此不可能直接将数据元素及其关系用它们存储器中的物理位置(内存地址)来表示。但是,可以借助于高级语言中提供的‘数据类型”来表示。考虑到本书用类 PASCAL 语言描述算法,用这种语言所具有的‘数组”类型来表示顺序存储结构,用‘指针”来表示链式存储结构。这样与存储结构的本来意义有一些差别,但如果将类 PASCAL 语言看成是一个用 PASCAL 数据类型表示数据并执行 PASCAL 语句的虚拟处理器,那么,本书中讨论的存储结构就是数据结构在这个虚拟处理器中的表示,因此,它是一种‘虚拟”存储结构。

数据类型 是一个值的集合和定义在这个值集合上的一组操作的总称。

数据类型是和数据结构密切相关的概念,按“值”是否可分解,高级语言中的数据类型可以分为两类:原子类型和结构类型。原子类型的值是不可分解的,如 PASCAL 语言中的标准类型(整型、实型、字符型和布尔型)、枚举类型、子界类型和指针类型等。结构类型的值是由若干成分按某种结构组成的,因而是可以分解的,并且它的成分既可以是非结构的,也可以是结构的,如 PASCAL 语言中的数组类型、集合类型、文件类型、记录类型和对象类型等。

在分析复杂的情况或处理复杂的事物时,经常使用的抽象思维方法是:舍去复杂系统中非本质的细节,只把其中某些本质的、能够反映系统重要宏观特性的东西提炼出来,构成系统的模型,并深入研究这些模型,得出一般的、具有规律性的东西,用以指导对实际问题的解决过程。这种抽象思维方法同样适用于软件系统的设计和研究。任何实际问题都包括数据和对数据的处理。一个软件系统也是由数据和处理(操作)组成。因此,在复杂软件系统的设计中应用抽象思维方法,就要进行数据抽象和操作抽象。将数据抽象和操作抽象紧密结合起来就形成了抽象数据类型的概念。

抽象数据类型(Abstract Data Type 简称 ADT)是指一个数学模型以及定义在该模型上的一组操作。

抽象数据类型的定义只与它的一组逻辑特性有关,而与它在计算机内部如何表示和实现无关。换句话说,不论它的内部结构如何变化,只要它的数学特性不变,都不影响它的外部使用。例如,各种类型的计算机都拥有的“整数”类型就是一个抽象数据类型,尽管处理器不同,它的实现方法可以不同,但由于其定义的数学特性相同,从用户使用的角度来看都是相同的。注意,如果在相同的数学模型上定义两个不同的操作集,则认为它们代表两个不同的抽象数据类型。

对软件系统的模块定义抽象数据类型的作用是可以降低软件模块对具体数据的依赖程度,从而提高软件模块的复用程度。现代程序设计方法要求,一个软件系统的框架应建立在数据之上,而不是建立在操作之上。在构成软件系统的每个相对独立的模块上,定义一组数据和作用于这些数据上的一组操作。在模块外部使用的只是抽象的数据和抽象的操作,而在模块内部才给出这些数据的表示及其上操作的实现。因此,在模块外,看到的数据类型的抽象层次越高,模块对具体数据的依赖程度自然就越低,该模块的可复用程度也就越高。

抽象数据类型概念的引入,降低了大型软件设计的复杂性,并使软件设计中普遍遵循的模块化,信息隐蔽,代码共享,软件复用等思想得到更充分的体现。因此,抽象数据类型是计算机科学的一个重要的新概念。

抽象数据类型的实现方法:用给定的程序设计语言描述抽象数据类型的数学模型并用一组过程或函数来实现该模型上定义的各种操作。在抽象数据类型中采用什么样的数据结构来表示数学模型,要根据是否能够方便和有效地实现该抽象数据类型上定义的大多数操作来确定。数据结构一般利用该语言的基本数据类型和复合数据类型的构造机制来构成。

抽象数据类型的实现可以使用 Turbo PASCAL 提供的单元(unit)。一个单元是常数、数据类型、变量、过程及函数的集合,它是一个可以独立编译的程序单位。单元通过过程和函数提供一系列的功能。

单元由单元首部、接口部分、实现部分和初始化代码组成。

单元首部以保留字 UNIT 开头,后跟单元名称。Turbo PASCAL 要求单元名与存放该单元的文件名一致。

单元接口部分以保留字 INTERFACE 开头,可以说明常量、数据类型、变量、过程和函数,

这些内容对任何使用本单元的程序和其他单元都是“可见”的,即可被使用本单元的程序和其他单元引用。

单元实现部分以保留字 IMPLEMENTATION 开头,定义单元提供的功能的实现细节。它的内容对任何使用本单元的程序和其他单元都是“不可见”的。

初始化代码出现在单元最后的 BEGIN 和 END 之间。若程序使用了单元,在运行时,则先调用单元的初始化代码,再运行程序体。

单元的实例可以参看第二章的 2.4.3 节,抽象数据类型线性表的实现及应用。

§ 1.2 数据结构与算法

著名的计算机科学家,PASCAL 语言的发明者 N.Wirth 教授曾提出一个公式:

算法+ 数据结构= 程序

这个有名的公式清楚地揭示了算法和数据结构这两个计算机科学的重要支柱的重要性和统一性。这就是说,既不能离开数据结构去抽象地分析求解问题的算法,也不能脱离算法去孤立地研究程序的数据结构。因此,在初步了解数据结构的基本概念和术语之后,还应该讨论算法的概念。

1.2.1 算法的概念

算法(algorithm) 是对特定问题求解步骤的一种描述,它是指令的有限序列,其中每一条指令表示一个或多个操作。

通常,一个算法必须具备以下五个重要特性:

1. 有穷性 一个算法对任何合法的输入值必须总是在执行有穷步之后结束,而且每一步都可在有穷时间内完成。注意,有的算法,操作步骤看起来似乎有穷,但不能保证在动态环境下实际执行次数的有穷性。请看例 1.5,它给出的算法不具备动态环境下实际执行次数的有穷性。

例 1.5 一个非算法的计数过程

(1)开始

(2)n:= 0

(3)n:= n+ 1

(4)重复(3)

(5)结束

该“算法”之所以不具备动态环境下实际执行次数的有穷性,是因为第(4)步的重复是无条件的。如果将(4)改成:若 n= 10000,则执行(5),否则执行(3),就可以得到一个正确的算法。

2. 确定性 算法中每一条指令必须有确切的含义,其他人读程序时不会产生二义性。并且,在任何条件下,算法只有惟一的一条执行路径,即对于相同的输入只能得出相同的输出。

3. 可行性 一个算法是能行的,算法中描述的操作都是可以通过已经实现的基本运算执行有限次来实现。

4. 输入 一个算法有 n(n≥0)个数据的输入。

5. 输出　一个算法必须有一个或多个有效信息的输出，它是同输入有某个特定关系的量。

1.2.2　描述算法的方法

考虑到本书中讨论的算法，既要便于阅读，又要能容易地转换为用高级语言书写的程序，因而采用类 PASCAL 语言作为算法的描述工具。所谓类 PASCAL 语言，是对标准 PASCAL 语言的一种简单的扩充，其扩充部分的语法图如附录一所示，下面做一些简单的说明，并与标准 PASCAL 语言作一对比，读者容易从中得到启发。

1. 所有的算法都以过程或函数的形式表示，下面左面所列的是类 PASCAL 语言的形式表示，右面是对应的标准 PASCAL 语言的形式表示(以下同)。

类 PASCAL	标准 PASCAL

PROC 过程名(参数表);
　{算法说明}
　　语句组
ENDP; {过程名}
 	
FUNC 函数名(参数表):类型;
　　{算法说明}
语句组
ENDF; {函数名}
	PROCEDURE 过程名(参数表);
	　VAR 局部变量说明;
	BEGIN
	　　语句组
	END;
	FUNCTION 函数名(参数表):类型;
	　VAR 局部变量说明;
	BEGIN
	　　语句组
	END;

注: 参数表可含有若干值参和变参; 语句组由一个或一个以上的语句组成，两语句之间用“; ”作分隔符。在正常情况下，左面的过程结束于 ENDP，右面的过程结束于 END; 在函数过程中，左面的函数值 f 以 RETURN(f)形式返回，右面的返回值是通过函数赋值的形式实现的，即: 函数名: = f。左面除了参数表中的参数需要说明类型外，算法中出现的其他变量可以不作说明，右面不仅要对参数表中的参数需说明类型，而且算法中出现的其他变量也需加以说明。	

2. 在赋值语句上，类 PASCAL 语言的形式要多一些。

类 PASCAL	标准 PASCAL
简单赋值　变量名: = 表达式;	变量名: = 表达式;
串联赋值 　变量名 1: = 变量名 2: = …: = 表达式 0;	变量名 1: = 表达式 0; 变量名 2: = 表达式 0; …
成组赋值 (变量名 1, …, 变量名 k): = (表达式 1, …, 表达式 k);	变量名 1: = 表达式 1; … 变量名 k: = 表达式 k;

续表

类 PASCAL	标准 PASCAL
交换赋值　　变量名 1←→变量名 2	中间变量名:= 变量名 1; 变量名 1:= 变量名 2; 变量名 2:= 中间变量名;

3. 条件语句和情况语句

类 PASCAL	标准 PASCAL
IF 条件 THEN 语句 1 IF 条件 THEN 语句 1 ELSE 语句 2 CASE 　条件 1: 语句 1; 　… 　条件 n: 语句 n; 　(ELSE　语句 n+ 1) ENDC; CASE 变量名 OF 　值 1: 语句 1; 　… 　值 n: 语句 n; 　(ELSE　语句 n+ 1) ENDC;	IF 条件 THEN 语句 1 IF 条件 THEN 语句 1 THEN 语句 2 CASE〈表达式〉OF 　〈值表 1〉:〈语句 1〉; 　〈值表 2〉:〈语句 2〉; 　… 　〈值表 n〉:〈语句 n〉; 　(ELSE 语句 n+ 1) END

注:其中“语句”可以是任意的语句组,当语句组包含一个以上的语句时,左面需用一对黑体方括弧“【】”括起来,右面则应用语句括号“begin end”括起来。以下说明中出现的“语句”类同。

4. 循环语句及过程调用和函数调用语句,类 PASCAL 语言与标准 PASCAL 语言的形式是类似的。

循环语句有:

WHILE 条件 DO 语句;

REPEAT 语句组 UNTIL 条件;

FOR 变量名:= 初值 TO 终值 DO 语句; (终值大于初值,增量为 1)

FOR 变量名:= 初值 DOWNTO 终值 DO 语句; (终值小于初值,增量为- 1)

过程和函数调用语句,过程和函数调用均允许嵌套或递归调用。

过程名(参量表)

变量名:= 函数名(参量表)

5. 出错处理语句,类 PASCAL 语言的形式是:

ERROR(’文字’)

其功能为:结束本算法过程并带回表示错误的文字串。在标准 PASCAL 语言中没有相应的语

句,读者只能在过程中,用程序结构和输出语句来实现。

6. 输入和输出,类 PASCAL 语言有两个标准过程

read(变量表)

write(变量表)

变量表由若干变量名或字符串组成,中间用逗号隔开。在标准 PASCAL 语言中有相应的过程,读者可以原样照搬使用。

7. 基本函数

在类 PASCAL 语言中有:求最大值:max()、求最小值:min()、求绝对值:abs()、下取整:⌊ ⌋、上取整:⌈ ⌉、判文件结束:eof()、判行结束:eoln()。在标准 PASCAL 语言中除了下取整和上取整函数没有相应的形式,需要用别的函数或过程来实现外,其他函数均有相应的形式可直接采用。

8. 注释　类 PASCAL 语言和标准 PASCAL 语言均可以在算法中任何处插入用一对花括弧括起来的文字注释。

9. 布尔运算符 AND、OR、NOT 在类 PASCAL 语言及标准 PASCAL 语言中均有相同的形式。而在类 PASCAL 语言中还有 CAND 和 COR 两种运算符是标准 PASCAL 语言没有的。设 A 和 B 分别是布尔表达式,若 A = true 与 B = true,则 A CAND B = true,否则 A CAND B = false,而且当 A = false 时不再对 B 求值;类似地,若 A = true 或 B = true,则 A COR B = true,否则A COR B = false,并且,当 A = true 时不再对 B 求值,这就是说,第一表达式 A 优先于第二表达式 B,两者不可交换。在标准 PASCAL 语言中,必须根据具体情况,用适当的程序结构来实现。

1.2.3　算法分析

通常从以下几个方面评价算法的质量:

(1)正确性　算法应能正确地实现预定的功能(即处理要求)。

(2)可读性　算法应易于阅读和理解,以便于调试时修改和扩充。

(3)健壮性　当输入数据非法时,算法也能适当地作出反应或进行处理,而不会产生莫名其妙的输出结果。

(4)高效率　即要求执行算法的时间短,所需要的存储空间少。

要有一个高效的算法,在设计算法时,就要对算法的执行时间有一个客观的分析和判断。为此,通常将算法的执行时间看成为问题规模(一般用整数 n 表示)的函数。问题的规模在不同的具体问题中有不同的含义,它可以是矩阵的阶,线性表的长度,图的顶点数等等。

往往用时间复杂度来度量算法的效率。从算法中选取一种对于所研究的问题来说是基本运算的原操作,该原操作重复执行的次数的量级称为时间复杂度,记为 T(n)。在下面例子中

```
(1) x:= x+ 1;
(2) FOR i:= 1 TO n DO
        x:= x+ 1;
(3) FOR i:= 1 TO n DO
        FOR j:= 1 TO n DO
          x:= x+ 1;
```