



高等学校计算机类“十二五”规划教材

数据结构教程

◎主 编 胡元义
◎副主编 宁耀斌 孙旭霞 罗作民
雷西玲 陈 曦

SHUJU JIEGOU
JIAOCHENG



西安电子科技大学出版社
<http://www.xduph.com>

内 容 简 介

本书系统地介绍了数据结构的有关内容,主要包括:线性表、栈、队列、串、数组、广义表、树、图等常用数据的逻辑结构和存储结构,使用各种数据结构的基本操作,以及查找、排序算法等。

本书采用的算法全部用 C 语言描述,各章节均附有大量习题。与本书配套的《数据结构教程习题解析与算法上机实现》详细给出了本书习题的解题思路和参考答案,对书中的所有算法和涉及算法的示例都给出了完整的 C 语言实现程序,并且在 VC++ 6.0 环境下上机通过。

本书结构清晰,既强调知识的实用性,也注重理论的完整性,化繁为简,将理论融入具体实例中,化难为易,以达到准确、清楚地阐述相关概念和原理的目的。书中给出的例子也具有较强的实用性与连贯性。

本书可作为高等院校相关专业本科生及硕士研究生的专业教材或参考书,也可作为相关技术人员的自学用书。

图书在版编目(CIP)数据

数据结构教程/胡元义主编. —西安:西安电子科技大学出版社, 2012.8

高等学校计算机类“十二五”规划教材

ISBN 978-7-5606-2890-5

I. ① 数… II. ① 胡… III. ① 数据结构—高等学校—教材 IV. ① TP311.12

中国版本图书馆 CIP 数据核字(2012)第 173038 号

策 划 胡华霖

责任编辑 薛 媛 王 瑛

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfxb001@163.com

经 销 新华书店

印刷单位 陕西华沐印刷科技有限责任公司

版 次 2012 年 8 月第 1 版 2012 年 8 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印 张 18.5

字 数 438 千字

印 数 1~3000 册

定 价 32.00 元

ISBN 978-7-5606-2890-5/TP · 1365

XDUP 3182001-1

如有印装问题可调换

本社图书封面为激光防伪覆膜,谨防盗版。

前 言

如果把程序设计比作一棵大树，数据结构无疑是大树的躯干。可以说，程序设计的精髓就是数据结构。计算机各个领域的应用都要用到各种数据结构，只有较好地掌握了数据结构知识，才能在程序设计中做到游刃有余，从而在计算机应用领域进行研究和开发时做到胸有成竹。

数据结构课程是计算机专业的一门核心课程，它是在长期的程序设计实践中提炼、升华而成的，反过来又应用于程序设计。数据结构课程同时又是操作系统、编译原理等计算机核心课程的基础，在计算机专业课程的学习中起着承上启下的作用。数据结构是一门应用广泛且最有实用价值的课程，不掌握数据结构知识就难以成为一名合格的软件工程师或计算机工作者。

由于数据结构的原理和算法都比较抽象，且数据结构又具有概念性强、内容灵活、不易掌握和难度大等特点，因此学习起来比较困难。本书在内容的组织和编排上注意掌握内容的难易程度，把握理论的深度，追求应用的广度；在内容的讲解上注重条理性和连贯性，做到化繁为简、化难为易，引导学生由基本概念出发，思考数据结构的实质内容以及相应算法求解的思路和实现方法，并由此深化对数据结构基本概念的理解，达到举一反三，培养分析问题和解决问题的能力这一目的。

本书共分9章，采用的算法均用C语言描述。第1章简要介绍了数据结构的基本概念与算法基础；第2章重点介绍了两种存储结构——顺序表和单链表的有关概念；第3章介绍了栈和队列的概念、特点及应用；第4章涉及到串的概念，特别是串的模式匹配的有关内容；第5章介绍了数组和广义表的有关概念；第6章重点介绍了二叉树的有关概念；第7章重点介绍了图的有关知识和应用；第8章给出了各种静态和动态查找表的方法；第9章详细讨论了各种排序的方法和特点。

为了便于正确理解数据结构的概念和相关内容，本书除了给出大量有针对性的实例外，还采用了图解的方式来解析数据结构中的算法和示例的实现过程。本书的另一个特点是各章的习题概念性强、覆盖面广、内容全面，具有典型性和代表性，这对加深理解本章的知识是必不可少的。习题中的算法设计题选材得当、针对性强，可提高学生的动手能力并培养其程序设计的素质。

本书与配套的教材《数据结构教程习题解析与算法上机实现》配合使用将会得到更好的学习效果。书中带“*”的章节为选学内容。

此外，本书中的汉诺塔非递归算法和二叉树后序遍历的第一种非递归算法，是作者在1996年出版的《TURBO PASCAL 6.0 精讲、题解及应用》(西安电子科技大学出版社)一书

中首次提出来的。在本书中，作者又设计了平衡二叉树、建立广义表等算法，并对一些经典算法进行了完善和修改。特别是平衡二叉树算法，真正实现了无需改动二叉树存储结构(不使用平衡因子)就可以对不平衡的子树进行调正，目前国内还未见到此类算法。

由于编者水平有限，书中难免存在不足，敬请广大读者批评指正。

编 者

2012 年 1 月

目 录

第 1 章 绪论.....	1	3.3 队列.....	50
1.1 数据结构的概念.....	2	3.3.1 队列的定义及基本运算.....	50
1.1.1 数据与数据元素.....	2	3.3.2 队列的存储结构和运算实现.....	51
1.1.2 数据结构.....	3	*3.4 递归转化为非递归的研究.....	56
1.2 逻辑结构与存储结构.....	4	3.4.1 汉诺塔问题递归解法.....	56
1.2.1 逻辑结构.....	4	3.4.2 汉诺塔问题非递归解法.....	59
1.2.2 存储结构.....	5	3.4.3 八皇后问题递归解法.....	62
1.3 算法与算法分析.....	6	3.4.4 八皇后问题非递归解法.....	64
1.3.1 算法的定义和描述.....	6	习题 3.....	66
1.3.2 算法分析和复杂度计算.....	7		
习题 1.....	8	第 4 章 串.....	70
		4.1 串的概念及基本运算.....	70
第 2 章 线性表.....	11	4.1.1 串的基本概念.....	70
2.1 线性表及其逻辑结构.....	11	4.1.2 串的基本运算.....	71
2.1.1 线性表的定义.....	11	4.2 串的顺序存储结构及基本运算.....	72
2.1.2 线性表的基本操作.....	12	4.2.1 串的顺序存储结构.....	72
2.2 线性表的顺序存储结构及运算实现.....	12	4.2.2 顺序串的基本运算.....	73
2.2.1 线性表的顺序存储——顺序表.....	12	4.3 串的链式存储结构及基本运算.....	75
2.2.2 顺序表上基本运算的实现.....	14	4.3.1 串的链式存储结构.....	75
2.3 线性表的链式存储结构及运算实现.....	19	4.3.2 链串的基本运算.....	76
2.3.1 单链表.....	19	4.4 串的模式匹配.....	78
2.3.2 单链表上基本运算的实现.....	21	4.4.1 简单模式匹配.....	79
2.3.3 循环链表.....	27	4.4.2 无回溯的 KMP 匹配.....	80
2.3.4 双向链表.....	28	*4.4.3 next 函数的改进.....	85
2.3.5 静态链表.....	31	习题 4.....	86
2.3.6 单链表应用示例.....	33		
习题 2.....	36	第 5 章 数组与广义表.....	89
		5.1 数组的概念与存储结构.....	89
第 3 章 栈和队列.....	40	5.1.1 数组的基本概念.....	89
3.1 栈.....	40	5.1.2 数组的存储结构.....	90
3.1.1 栈的定义及基本运算.....	40	5.2 特殊矩阵的压缩存储.....	92
3.1.2 栈的存储结构和运算实现.....	41	5.2.1 对称矩阵.....	92
*3.2 栈与递归.....	46	5.2.2 三角矩阵.....	93
3.2.1 递归及其实现.....	46	5.2.3 对角矩阵.....	95
3.2.2 递归调用过程分析.....	47	5.3 稀疏矩阵.....	96

5.3.1 稀疏矩阵的三元组表示	96	第7章 图	162
5.3.2 稀疏矩阵的十字链表表示	100	7.1 图的基本概念	162
5.4 广义表	104	7.1.1 图的定义	162
5.4.1 广义表的基本概念	104	7.1.2 图的基本术语	163
5.4.2 广义表的存储结构	105	7.2 图的存储结构	165
5.4.3 广义表基本操作实现算法	109	7.2.1 邻接矩阵	166
习题5	114	7.2.2 邻接表	167
第6章 树与二叉树	118	*7.2.3 有向图的十字链表存储方法	170
6.1 树的基本概念	118	*7.2.4 无向图的邻接多重表存储方法	171
6.1.1 树的概念与定义	118	7.3 图的遍历	172
6.1.2 树的基本术语	119	7.3.1 深度优先搜索	172
6.2 二叉树	120	7.3.2 广度优先搜索	175
6.2.1 二叉树的定义	120	7.3.3 图的连通性问题	177
6.2.2 二叉树的性质	121	7.4 生成树与最小生成树	178
6.2.3 二叉树的存储结构	123	7.4.1 生成树和生成森林	178
6.3 二叉树的遍历	125	7.4.2 最小生成树与构造最小生成树的 Prim 算法	181
6.3.1 二叉树的遍历方法	125	7.4.3 构造最小生成树的 Kruskal 算法	185
6.3.2 遍历二叉树的递归算法及 遍历示例	126	7.5 最短路径	188
6.3.3 遍历二叉树的非递归算法	129	7.5.1 从一个源点到其他各点的 最短路径	188
6.3.4 二叉树的层次遍历算法	133	7.5.2 每一对顶点之间的最短路径	192
6.3.5 由遍历序列恢复二叉树	134	7.6 拓扑排序和关键路径	195
6.3.6 二叉树遍历的应用	136	7.6.1 AOV 网与拓扑排序	195
6.4 线索二叉树	139	7.6.2 AOE 网与关键路径	199
6.4.1 线索二叉树的定义及结构	139	习题7	204
6.4.2 线索化二叉树	141	第8章 查找	211
6.4.3 访问线索二叉树	143	8.1 查找的基本概念	211
6.5 哈夫曼树	145	8.2 静态查找表	212
6.5.1 哈夫曼树的基本概念及 构造方法	145	8.2.1 顺序查找	212
6.5.2 哈夫曼算法的实现	148	8.2.2 有序表的查找	213
6.5.3 哈夫曼编码	150	8.3 树表形式的动态查找表	218
6.6 树和森林	153	8.3.1 二叉排序树	218
6.6.1 树的定义与存储结构	153	8.3.2 平衡二叉树	225
6.6.2 树、森林与二叉树之间的转换	155	8.3.3 B 树和 B+树	231
6.6.3 树和森林的遍历	156	8.4 地址映射方式下的动态 查找表——哈希表	238
习题6	157	8.4.1 哈希表与哈希方法	238

8.4.2 哈希函数的构造方法	239	9.4 选择排序	264
8.4.3 处理冲突的方法	241	9.4.1 直接选择排序	264
8.4.4 哈希表的查找	243	9.4.2 堆排序	266
习题 8	246	9.5 归并排序	269
第 9 章 排序	253	9.6 基数排序	274
9.1 排序的基本概念	253	9.6.1 多关键字排序	274
9.2 插入排序	254	9.6.2 链式基数排序	275
9.2.1 直接插入排序	254	*9.7 外排序简介	278
9.2.2 折半插入排序	256	9.8 各种内排序方法的比较	281
9.2.3 希尔(Shell)排序	257	习题 9	283
9.3 交换排序	259	参考文献	288
9.3.1 冒泡排序	259		
9.3.2 快速排序	261		

第
1
章

绪 论

计算机发展的初期其应用主要是数值计算问题，即所处理的数据都是整型、实型及布尔型等简单数据。但随着计算机应用领域的不断扩大，非数值计算问题占据了目前计算机应用的绝大部分，计算机所处理的数据也不再是简单的数值，而是扩展到字符串、图形、图像、语音、视频等复杂的数据，这些复杂的数据不仅量大，而且具有一定的结构。例如，一幅图像是由简单的数值组成的矩阵；一个图形中的几何坐标可以组成表；语言编译程序中使用的栈、符号表和语法树，操作系统中所用到的队列、树形目录等，都是有结构的数据。为了有效地组织和管理好这些数据，设计出高质量的程序以及高效率地使用计算机，就必须深入研究这些数据自身的特性以及它们之间的相互关系，这正是数据结构这门课程形成与发展的原因。也即，数据结构的研究涉及构筑计算机求解问题过程的两大基石：刻画实际问题中信息及其关系的数据结构和描述问题解决方案的逻辑抽象算法。

数据结构从 1968 年开始作为一门独立的课程，但在此之前其有关内容已出现在编译原理和操作系统的课程中。20 世纪 60 年代中期，美国的一些大学已经开设了有关数据结构的课程，但当时的课程名称并不叫数据结构。1968 年 D. E. Knuth 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》(Art of Computer Programming)第一卷《基本算法》是第一本较系统阐述数据的逻辑结构、存储结构以及相应操作的著作。从 20 世纪 60 年代末到 70 年代初，出现了大型程序并且软件也相对独立，结构程序设计成为程序设计方法学的主要内容，数据结构越来越受到人们的重视。20 世纪 70 年代中期到 80 年代，各种版本的数据结构著作相继问世，这对此后的计算机应用产生了深远影响。至今数据结构的发展并未终结：一方面，面向各种专门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。面对当今的信息化时代，计算机处理的数据越来越多，越来越复杂，数据结构和算法也越来越受到重视。数据结构和算法的研究也成为面向专业方向的研究，比如遗传算法的研究、云计算的并行算法等。

现在，数据结构已是计算机及相关专业必不可少的专业基础课程，主要学习用计算机实现数据组织和数据处理的方法。数据结构课程将为计算机相关后继课程，如操作系统、编译原理、数据库原理、人工智能和软件工程等打下坚实的基础。

数据是人们利用文字符号、数学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。简而言之，数据是信息的载体，是对客观事物的符号化表示。从计算机的角度看，数据是计算机程序对所描述客观事物进行加工处理的一种表示，凡是能够被计算机识别、存取和加工处理的符号、字符、图形、图像、声音、视频信号等都可以称为数据。

我们日常涉及到的数据主要分为两类：一类是数值数据，包括整数、实数和复数等，它们主要用于工程和科学计算以及商业事务处理；另一类是非数值数据，主要包括字符和字符串以及文字、图形、语音等，它们多用于控制、管理和数据处理等领域。

数据元素是数据集合中的一个“个体”，是数据的基本单位。在计算机中，数据元素通常被作为一个整体来进行考虑和处理。在有些情况下，数据元素也称为元素、结点、顶点和记录等。一个数据元素可以由一个或多个数据项组成，数据项是具有独立含义的数据最小单位，有时也称为字段或域。

例 1.1 一个学生信息(数据)表如表 1.2 所示，请指出表中的数据、数据元素及数据项，并由此得出三者之间的关系。

表 1.2 学生信息表

姓 名	性 别	年 龄	专 业	其 他
刘小平	男	21	计算机	...
王 红	女	20	数 学	...
吕 军	男	20	经 济	...
⋮	⋮	⋮	⋮	⋮
马文华	女	19	管 理	...

【解】表 1.2 中是全部学生信息数据。表中的每一行即为记录一个学生信息的数据元素，而该行中的每一项则为一个数据项。数据、数据元素和数据项实际上反映了数据组织的三个层次，数据可以由若干个数据元素构成，而数据元素则又可以由若干数据项构成。

1.1.2 数据结构

数据结构是指数据元素以及数据元素之间的相互关系，也即数据的组织形式，可以看做是相互之间存在着某种特定关系的数据元素集合。也即，可以把数据结构看做是带结构的数据元素集合。进一步说，数据结构描述按照一定逻辑关系组织起来的待处理数据元素的表示及相关操作，涉及数据的逻辑结构、存储结构和运算。因此，数据结构包含以下三个方面的内容：

- (1) 数据元素之间的逻辑关系，即数据的逻辑结构。
- (2) 数据元素及其关系在计算机存储器中的存储方式，即数据的存储结构(物理结构)。
- (3) 施加在数据上的操作，即数据的运算。

数据的逻辑结构是从逻辑关系上(主要指相邻关系)来描述数据的，它与数据如何存储无关，是独立于计算机的。因此，数据的逻辑结构可以看做是从具体问题抽象出来的数学模型。

数据的存储结构是指数据的逻辑结构在计算机存储器中的映像表示，即在能够反映数据逻辑关系的前提下数据在存储器中的存储方式。

数据的运算是在数据上所施加的一系列操作，称为抽象运算，它只考虑这些操作的功能，而暂不考虑如何完成，只有在确定了存储结构后，才会具体实现这些操作。也即，抽象运算是定义在逻辑结构上的，而实现则是建立在存储结构上的。最常用的运算有：检索、插入、删除、更新以及排序等。

1.2 逻辑结构与存储结构

1.2.1 逻辑结构

数据的逻辑结构是对数据元素之间逻辑关系的描述,它与数据在计算机中的存储方式无关。根据数据元素之间关系的不同特性,可以划分出以下四种基本逻辑结构,如图 1-1 所示。

(1) 集合结构:数据元素之间除了“属于同一集合”的联系之外,没有其他关系。

(2) 线性结构:数据元素之间存在着“一对一”的关系。数据之间存在前后顺序关系,除第一个元素和最后一个元素外,其余元素都有唯一的一个前驱元素和唯一的一个后继元素。

(3) 树形结构:数据元素之间存在着“一对多”的关系。数据之间存在层次关系,除了一个根结点(元素)外,其余元素(结点)都有唯一的一个前驱元素,并且可以有多个后继元素。

(4) 图结构(或称网状结构):数据元素之间存在着“多对多”的关系。也即,每个元素都可以有多个前驱元素和多个后继元素。

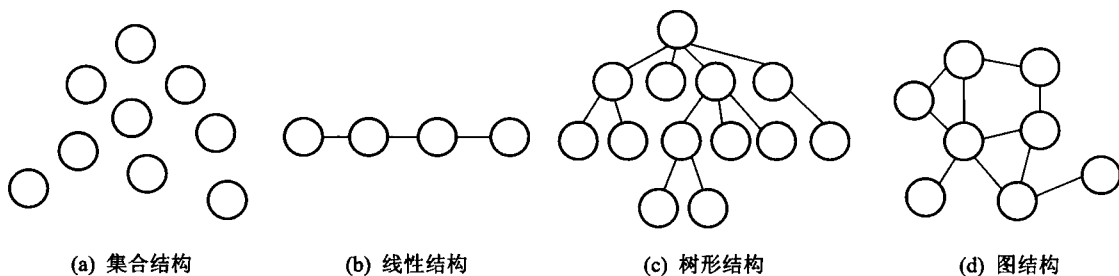


图 1-1 逻辑结构的四种基本形态

由于集合结构具有简单性和松散性,因此通常只讨论其他三种逻辑结构。数据的逻辑结构可以分为线性结构和非线性结构两类。若数据元素之间的逻辑关系可以用一个线性序列简单地表示出来,则称为线性结构,否则称为非线性结构。树形结构和图结构就属于非线性结构。现实生活中的楼层编号就属于线性结构,而省、市、地区的划分属于树形结构,城市交通图则属于图结构。

关于逻辑结构需要注意以下几点:

(1) 逻辑结构与数据元素本身的形式和内容无关。例如,给表 1.2 中的每个学生增加一个数据项“学号”,就得到另一个数据,但由于所有的数据元素仍是“一个接一个排列”,故新数据的逻辑结构与原来数据的逻辑结构相同,仍是一个线性结构。

(2) 逻辑结构与数据元素的相对位置无关。例如,将表 1.2 中的学生按年龄由大到小的顺序重新排列就得到另一个表格,但这个新表格中的所有数据元素“一个接一个排列”的性质并没有改变,其逻辑结构与原表格相同,还是线性结构。

(3) 逻辑结构与所含数据元素的个数无关。例如,在表 1.2 中增加或删除若干学生信息(数据元素),所得到的表格仍为线性结构。

1.2.2 存储结构

数据的存储结构是数据结构在计算机中的表示方法，也即数据的逻辑结构到计算机存储器的映像，包括数据结构中数据元素的表示以及数据元素之间关系的表示。数据元素及数据元素之间的关系在计算机中可以有以下四种基本存储结构：

(1) 顺序存储结构：借助于数据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。通常顺序存储结构是利用程序语言中的数组来描述的。

(2) 链式存储结构：在数据元素上附加指针域，并借助指针来指示数据元素之间的逻辑关系。链式存储结构通常是利用程序语言中的指针类型来描述的。

(3) 索引存储结构：在存储所有数据元素信息的同时建立附加索引表。索引表中表项的一般形式是：(关键字，地址)。关键字是数据元素中某个数据项的值，它唯一标识该数据元素；地址则是指向该数据元素的指针。由关键字可以立即通过地址找到该数据元素。

(4) 哈希(或散列)存储结构：此方法的基本思想是根据数据元素的关键字通过哈希(或散列)函数直接计算出该数据元素的存储地址。

顺序存储结构的主要优点是节省存储空间，即分配给数据的存储单元全部用于存放数据元素的数据信息，数据元素之间的逻辑关系没有占用额外的存储空间。采用这种存储结构可以实现对数据元素的随机存取，即每个数据元素对应有一个序号，并由该序号可以直接计算出数据元素的存储地址(例如对于数组 A 其序号为数组元素的下标，数组元素 $A[i]$ 可以通过 $*(A+i)$ 进行存取)。但顺序存储结构的主要缺点是不便于修改，对数据元素进行插入、删除运算时，可能要移动一系列的数据元素。

链式存储结构的主要优点是便于修改，在进行插入、删除运算时仅需修改相应数据元素的指针值，而不必移动数据元素。与顺序存储结构相比，链式存储结构的主要缺点是存储空间的利用率较低，因为除了用于数据元素的存储空间外还需要额外的存储空间来存储数据元素之间的逻辑关系。此外，由于逻辑上相邻的数据元素在存储空间中不一定相邻。所以不能对数据元素进行随机存取。

线性结构采用索引存储方法后就可以对结点进行随机访问。在进行插入、删除运算时，只需改动存储在索引表中数据元素的存储地址，而不必移动数据元素，所以仍保持较高的数据修改和运算效率。索引存储结构的缺点是增加了索引表，这也降低了存储空间的利用率。

哈希(或散列)存储结构的优点是查找速度快，只要给出待查数据元素的关键字，就可以立即计算出该数据元素的存储地址。与前面三种存储方法不同的是，哈希存储结构只存储数据元素的数据而不存储数据元素之间的逻辑关系。哈希存储结构一般只适合于对数据进行快速查找和插入的场合。

图 1-2 给出了表 1.2 在顺序存储结构和链式存储结构下的示意。

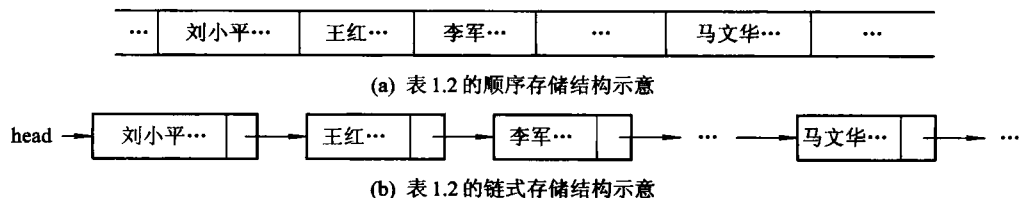


图 1-2 表 1.2 在不同的存储结构下的存储示意

1.3 算法与算法分析

概要的说, 算法是程序的逻辑抽象, 是解决某类客观问题的过程。计算机求解问题的核心是算法设计, 而算法设计又高度依赖于数据结构, 因为在算法设计时必须先确定相应的数据结构, 而在讨论某一种数据结构时也必然会涉及相应的算法。

1.3.1 算法的定义和描述

算法是建立在数据结构基础上对特定问题求解步骤的一种描述, 是若干条指令组成的有限序列。其中, 每一条指令表示一个或多个操作。算法必须满足以下性质:

- (1) 有穷性: 一个算法必须在有穷步之后结束, 即必须在有限时间内完成。
- (2) 确定性: 算法的每一步必须有确切的含义而没有二义性。对于相同的输入, 算法执行的路径是唯一的。
- (3) 可行性: 算法所描述的操作都可以通过可实现的基本运算在有限次执行后得以完成。
- (4) 输入: 一个算法可以有零个或多个输入。
- (5) 输出: 一个算法具有一个或多个输出, 且输出与输入之间存在某种特定的关系。

算法的含义与程序十分相似但又有区别。一个程序不一定满足有穷性, 例如操作系统程序只要不停机执行就永不停止, 因此, 操作系统不是一个算法。此外, 程序中的语句最终都要转化(编译)成计算机可执行的指令, 而算法中的指令则无此限制, 即一个算法可采用自然语言如英语、汉语描述, 也可以采用图形方式如流程图、拓扑图描述。算法给出了对一个问题的求解, 而程序仅是算法在计算机上的实现。一个算法若用程序设计语言来描述, 则此时算法也就是一个程序。

对某个特定问题的求解究竟采用何种数据结构及选择什么算法, 要看问题的具体要求和现实环境的各种条件; 数据结构的选择是否恰当将直接影响到算法的效率, 只有把数据结构与算法有机地结合起来才能设计出高质量的程序来。

例 1.2 对两个正整数 m 和 n , 给出求它们最大公因子的算法。

【解】 算法设计如下:

(1) 求余数: 以 n 除 m , 余数为 r , 且 $0 \leq r < n$ 。

(2) 判断余数 r 是否等于零: 如果 r 为零, 则输出 n 的当前值(即为最大公因子), 算法结束; 否则执行(3)。

(3) 将 n 值传给 m 、将 r 值传给 n , 转(1)。

也可以用流程图描述该算法, 如图 1-3 所示。

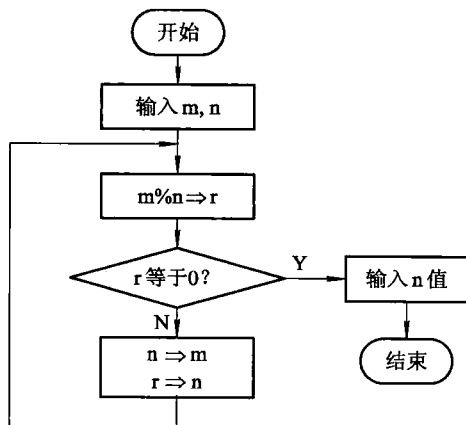


图 1-3 求最大公约数的算法

上述算法给出了三个计算步骤,而且每一步骤意义明确并切实可行。虽然出现了循环,但 m 和 n 都是已给定的有限整数,并且每次 m 除以 n 后得到的余数 r 即使不为零也总有 $r < \min(m, n)$, 这就保证循环执行有限次后必然终止,即满足算法的所有特征,所以是一个正确的算法。

1.3.2 算法分析和复杂度计算

算法设计主要是考虑可解算法的设计,而算法分析则是研究和比较各种算法的性能与优劣。算法的时间复杂度和空间复杂度是算法分析的两个主要方面,其目的主要是考察算法的时间和空间效率,以求改进算法或对不同的算法进行比较。

(1) 时间复杂度:一个程序的时间复杂度是指程序运行从开始到结束所需要的时间。

(2) 空间复杂度:一个程序的空间复杂度是指程序运行从开始到结束所需的存储量。

在复杂度计算中,实际上是把求解问题的关键操作,如加法、减法和比较运算指定为基本操作,然后把算法执行基本操作的次数作为算法的时间复杂度,而算法执行期间占用存储单元的数量作为算法的空间复杂度。

在此,涉及到频度的概念,即语句(指令)的频度是指它在算法中被重复执行的次数。一个算法的时间耗费就是该算法中所有语句(指令)的频度之和(记作 $T(n)$),它是该算法所求解问题规模 n 的某个函数 $f(n)$ 。当问题规模 n 趋向无穷大时, $T(n)$ 的数量级称为时间复杂度,记作:

$$T(n) = O(f(n))$$

上述式子中“ O ”的文字含义是 $T(n)$ 的数量级,其严格的数学定义是:若 $T(n)$ 和 $f(n)$ 是定义在正整数集合上的两个函数,则存在正常数 C 和 n_0 ,使得当 $n \geq n_0$ 时都满足:

$$0 \leq T(n) \leq C \cdot f(n)$$

例如,一个程序的实际执行时间为: $T(n) = 2.7n^3 + 8.3n^2 + 5.6$, 则 $T(n) = O(n^3)$ 。也即,当 n 趋于无穷大时, n^3 前的 2.7 可以忽略,即该程序的时间复杂度的数量级是 n^3 。

算法的时间复杂度采用这种数量级的形式表示后,将给分析算法的时间复杂度带来很大的方便。即对一个算法,只需分析影响该算法时间复杂度的主要部分即可,而无需对该算法的每一个语句都进行详细的分析。

若一个算法中的两个部分其时间复杂度分别为 $T_1(n) = O(f(n))$ 和 $T_2(n) = O(g(n))$, 则:

(1) 在“ O ”下的求和准则为: $T_1(n) + T_2(n) = O(\max(f(n), g(n)))$ 。

(2) 在“ O ”下的乘法准则为: $T_1(n) \times T_2(n) = O(f(n) \times g(n))$ 。

当算法转换为程序后,每条语句执行一次所需的时间取决于机器的指令性能、速度以及编译所生成的代码质量,这是难以确定的。因此,我们假设每条语句执行一次所需的时间均是单位时间,则程序计算的时间复杂度法则如下:

(1) 执行一条读写语句或赋值语句所用的时间为 $O(1)$ 。

(2) 依次执行一系列语句所用的时间采用求和准则。

(3) 条件语句 `if` 的耗时主要是当条件为真时执行语句体所用的时间,而检测条件是否为真还需耗费 $O(1)$ 。

(4) 对 `while`、`do...while` 和 `for` 这样的循环语句,其运行时间为每次执行循环体及检测是否继续循环的时间,故常用乘法准则。

例 1.3 试求下面程序段的时间复杂度。

```
for(i=0;i<n;i++)
    for(j=0;j<n;j++)
    {
        C[i][j]=0;
        for(k=0;k<n;k++)
            C[i][j]=C[i][j]+A[i][k]*B[k][j];
    }
```

【解】我们给程序中的语句进行编号，并在其右侧列出该语句的频度：

(1)	for(i=0;i<n;i++)	n+1
(2)	for(j=0;j<n;j++)	n(n+1)
	{	
(3)	C[i][j]=0	n ²
(4)	for(k=0;k<n;k++)	n ² (n+1)
(5)	C[i][j]=C[i][j]+A[i][k]*B[k][j];	n ³
	}	

语句(1)的 i 值由 0 递增到 n，并且测试到 i 等于 n 时(即条件“i<n”为假)才会终止，故它的频度是 n+1，但它的循环体却只能执行 n 次。语句(2)作为语句(1)循环体中的一个语句应该执行 n 次，而语句(2)自身又要执行 n+1 次，所以语句(2)的频度是 n(n+1)。同理，可得语句(3)、(4)和(5)的频度分别是 n²、n²(n+1)和 n³。也即，该程序段所有语句的频度之和为

$$T(n)=2n^3+3n^2+2n+1$$

最后得到： $T(n)=O(n^3)$ 。实际上，由算法的三重 for 循环且每重循环进行 n 次以及“O”下的乘法准则可直接得到： $T(n)=O(n^3)$ 。

此外要说明的是，时间复杂度按数量级递增排列的顺序如下：

$$O(1)<O(\lg n)<O(n)<O(n \lg n)<O(n^2)<O(n^3)<O(2^n)$$

习 题 1

1. 单项选择题

(1) 研究数据结构就是研究_____。

- A. 数据的逻辑结构
- B. 数据的存储结构
- C. 数据的逻辑结构和存储结构
- D. 数据的逻辑结构、存储结构及其数据在运算上的实现

(2) 下面关于算法的说法，错误的是_____。

- A. 算法最终必须由计算机程序实现
- B. 为解决某问题的算法和为该问题编写的程序含义是相同的
- C. 算法的可行性是指指令不能有二义性

D. 以上说法都是错误的

(3) 数据的_____包括集合、线性、树和图四种基本类型。

A. 存储结构 B. 逻辑结构 C. 基本运算 D. 算法描述

(4) 数据的存储结构包括顺序、链接、散列和_____四种基本类型。

A. 向量 B. 数组 C. 集合 D. 索引

(5) 关于逻辑结构，以下说法错误的是_____。

A. 逻辑结构与数据元素本身的形式和内容无关

B. 逻辑结构与数据元素的相对位置有关

C. 逻辑结构与所含结点的个数无关

D. 一些表面上很不相同的数据可以有相同的逻辑结构

(6) 根据数据元素之间关系的不同特性，以下四类基本逻辑结构反映了四类基本数据的组织形式。下面解释中错误的是_____。

A. 集合中任何两个结点之间都有逻辑关系，但组织形式松散

B. 线性结构中结点按逻辑关系依次排列成一条“锁链”

C. 树形结构具有分支、层次的特点，其形态有点像自然界中的树

D. 图状结构中各结构点按逻辑关系互相缠绕，任何两个结点都可以邻接

(7) 下面程序的时间复杂度为_____。

```
for(i=0;i<m;i++)
    for(j=0;j<n;j++)
        A[i][j]=i*j;
```

A. $O(m^2)$

B. $O(n^2)$

C. $O(m \times n)$

D. $O(m+n)$

2. 多项选择题

(1) 数据元素是_____。

A. 数据集合中的一个个体

B. 数据的基本单位

C. 数据的最小单位

D. 一个结点

E. 一个记录

(2) 数据结构被形式地定义为(K, R)，其中 K 是 ① 的有限集，R 是 K 上的 ② 有限集。

A. 算法

B. 数据元素

C. 数据操作

D. 逻辑结构

E. 操作

F. 映像

G. 存储

H. 关系

(3) 线性结构的顺序存储结构是一种 ① 的存储结构，线性结构的链式存储结构是一种 ② 的存储结构。

A. 随机存取

B. 顺序存取

C. 索引存取

D. 散列存取

E. 随机存取和索引存取

(4) 算法分析的目的是 ①，算法分析的两个主要方面是 ②。

A. 找出数据结构的合理性

B. 研究算法中输入和输出关系

C. 分析算法的效率以求改进

D. 分析算法的易懂性和文档性

E. 空间复杂度和时间复杂度

F. 正确性和简单性

G. 可读性和文档性

H. 数据复杂性和程序复杂性

(5) 算法指的是 ①，它必须是具备输入、输出和 ② 等五个特性。

- | | |
|-------------------|----------------|
| A. 计算方法 | B. 排序方法 |
| C. 解决问题的有限运算序列 | D. 调度方法 |
| E. 可执行性、可移植性和可扩充性 | F. 可行性、确定性和有穷性 |
| G. 确定性、有穷性和稳定性 | H. 易读性、稳定性和安全性 |

3. 填空题

- (1) 一个数据结构在计算机中的_____称为存储结构。
- (2) 对于给定的 n 个元素，可以构造出的逻辑结构有_____、_____、_____和_____四种。
- (3) 数据是描述客观事物的数、字符以及所有_____计算机中并被计算机程序所_____的符号集合。
- (4) 线性结构中的元素之间存在_____关系，树形结构中元素之间存在_____关系，图形结构中的元素之间存在_____关系，而集合结构中的元素之间不存在_____关系。
- (5) 数据结构是研究数据的_____和_____以及它们之间的相互关系，并对这种结构定义相应的_____且设计出相应的_____。
- (6) 数据的_____结构与数据元素本身的内容和形式无关。
- (7) 一个算法的时空性能是指该算法的_____和_____。前者是算法包含的_____，后者是算法需要的_____。

4. 判断题

- (1) 顺序存储方式只能用于存储线性结构。
- (2) 数据元素是数据的最小单位。
- (3) 算法可以用不同的语言描述，如果用 C 语言编写一个程序，则程序就是算法了。
- (4) 数据结构是带有结构的数据元素的集合。
- (5) 数据的逻辑结构是指各数据元素之间的逻辑关系，是用户根据需要而建立的。
- (6) 数据结构、数据元素、数据项在计算机中的表示(映像)分别称为存储结构、结点、数据域。

5. 名词解释

- (1) 数据 (2) 数据元素 (3) 数据项 (4) 数据结构 (5) 逻辑结构 (6) 存储结构

6. 写出下面程序段的时间复杂度。

```
y=0;
while((y+1)*(y+1)<=n)
    y=y+1;
```

7. 对下面程序段：

```
for(i=1; i<=n; i++)
    for(j=1; j<=i; j++)
        for(k=1; k<=j; k++)
            s=s+1;
```

试分析每一条语句执行的次数及时间复杂度。