数据结构
学习指导

第二版

曹桂琴 郭芳 编著

- 重点内容概要
- 典型例题解析
- 同步测试
- 考研试题精析

大连理工大学出版社

TP311.12/165

2008

数据结构学习指导

（第二版）

曹桂琴　郭　芳　编著

大连理工大学出版社

图书在版编目(CIP)数据

数据结构学习指导 / 曹桂琴,郭芳编著. —2版. —大连:大连理工大学出版社,2008.3
ISBN 978-7-5611-2381-2

Ⅰ.数… Ⅱ.①曹… ②郭… Ⅲ.数据结构—高等学校—教学参考资料 Ⅳ.TP311.12

中国版本图书馆CIP数据核字(2008)第007309号

大连理工大学出版社出版
地址:大连市软件园路80号 邮政编码:116023
发行:0411-84708842 邮购:0411-84703636 传真:0411-84701466
E-mail:dutp@dutp.cn URL:http://www.dutp.cn
大连理工印刷有限公司印刷 大连理工大学出版社发行

幅面尺寸:140mm×203mm 印张:11.5 字数:283千字
2008年3月第2版 2008年3月第2次印刷

责任编辑:梁 锋 唐立敏 责任校对:黎 玉
封面设计:孙宝福

ISBN 978-7-5611-2381-2 定 价:18.00元

前 言

随着计算机的普遍应用，计算机软件的作用日见突出。在进行程序设计时，不仅要对程序的构造进行系统而科学的研究，同时要求对程序处理的复杂数据进行系统的研究，使其有利于解决问题。数据结构它主要研究的正是如何合理地组织数据，怎样在计算机中有效地表示数据和处理数据，是开发出质量好、效率高的程序的重要基础环节。因此，数据结构课程已经成为计算机专业的核心课程，是从事计算机软件开发、应用人员应当必备的专业基础。

本书是作者在高校教学多年长期积累的教学经验的结晶，不仅包括数据结构相关概念及内容的归纳和总结，而且包括大量的典型例题和同步测试，使读者在深入理解和掌握数据结构的各部分内容的精髓的基础上，通过分析典型例题，强化自测训练，形成较系统而全面的数据结构的解题思路和算法设计思想。

全书共分9章，分别介绍数据结构基本概念及算法分析基础知识、线性表、栈和队列、串、数组和广义表、树和二叉树、图、查找、排序、文件等数据结构基本课程内容。每一章主要由重点内容概要、典型例题解析、同步测

试、同步测试参考答案四部分组成，部分章还附有上机实习题及程序。重点内容部分对各章涉及的数据结构相关内容做出归纳和总结，给出该章的学习要点，力求言简意赅。典型例题是针对各章内容而精心编选的，除了有作者多年教学过程中积累的例题，还选择部分自学考试试题、高级程序员考试试题、著名高校部分考研试题等，并且都给出了详尽的分析、解题过程及方法。同步测试和参考答案是为读者在学习本章内容后进行自我检验学习效果而设计的，通过强化做题训练，可以巩固所学知识。附录部分给出部分工学硕士、工程硕士入学考试试题，计算机本科生期末考试模拟试卷、成人教育学士学位课程考试模拟试卷等，供不同层次的读者参考。

因为许多高校选择的数据结构教材的算法描述使用的是C语言，因此本书算法用C语言描述，并且大多数算法都已经在计算机上调试通过。

本书第1章到第8章由大连工业大学郭芳编写，第9章由大连理工大学曹桂琴编写，全书由曹桂琴统稿。另外张绍武，王珺、林晓惠、李国禄、孙王杰、金玉子提供了部分试题及解答。

再版时，由郭芳对书中的内容进行全面修改，增加了一些典型例题及解答，还增加了部分最新考研试题。

编 者

2008.1

此为试读，需要完整PDF请访问：www.ertongbook.com

目 录

第1章 绪 论

● 重点内容概要 ●

本章主要讨论数据结构的基本概念和方法，并贯穿整个课程的学习过程，因此很有必要重点掌握，算法分析是学习的难点。

1. 基本概念和术语

(1)数据：计算机化的现实世界的事物的抽象描述。

(2)数据元素：数据的基本单位。通常由若干个数据项组成。数据项是具有独立含义的数据的最小可命名单位。

(3)数据对象：具有相同特性的数据元素的集合。

(4)数据结构：是带有结构的数据对象。结构是数据元素之间相互关系的集合。数据结构包括三个方面的内容：数据的逻辑结构、物理结构和数据的运算。

(5)数据的逻辑结构：只抽象地描述数据元素间的逻辑关系而与在计算机中如何存储无关。可描述为：S = (D,R)，其中 D 为数据对象，R 为数据元素之间相互关系的集合。

数据的逻辑结构可划分为两类：线性结构和非线性结构。

①线性结构的数据元素呈现为线性序列，即有且仅有一个开始结点和一个终端结点，除开始结点外所有结点都有唯一的一个直接前驱，除终端结点外所有结点都有唯一的一个直接后继。典型的线性结构包括线性表、栈、队列等。

②非线性结构的逻辑特征是一个结点可能有零个或多个直接前驱和零个或多个直接后继结点。典型的非线性结构有树形结构、图结构等。

(6)数据的物理结构:数据的逻辑结构在计算机存储器内的实现,数据元素自身值和数据元素之间关系的表示,又称为存储结构。

主要有以下几种存储方法:

·顺序存储方法:把逻辑上相邻的结点存储在物理位置上相邻的存储单元中。

·链接存储方法:逻辑上相邻的结点不要求物理位置亦相邻。

·索引存储方法:除存储结点信息外,还建立附加的索引表。

·散列存储方法:根据结点的关键字直接计算该结点的存储地址。

(7)数据的运算:对数据元素施加的操作。数据的运算是定义在数据的逻辑结构上的,但数据运算的具体实现与存储结构相关联。常用的数据运算有:查找、插入、删除、更新和排序等。

2. 算法的描述和分析

(1)算法

算法是解决某一特定问题的步骤和方法。特性:①有穷性:必须在执行有限步骤后结束;②确定性:算法的每一步骤必须是确切地规定的,无二义性;③可行性:算法的每一个运算都是可实现的;④输入:有0个和多个输入;⑤输出:有1个和多个输出。

算法设计可与高级语言无关,算法的实现必须借助于具体高级语言提供的数据类型及其运算。判断算法好坏的标准:①正确性:算法应实现具体问题需求的功能和性能。②可读性:算法应当便于阅读,以便理解和修改。③健壮性:要求算法具有查错和处理功能。④效率:算法的效率主要指算法运行时所需要的计算机资源的多少,包括运行时间和存储空间的消耗,称为算法的时间代价和空间代价。

(2)算法的时间复杂度

设 $T(n)$ 是问题规模 n 的时间函数,采用"大 O 表示法"表示

算法代价增长率的上界，设$f(n)$和$T(n)$是定义在正整数集合上，当且仅当存在正整数c和n_0，使得对所有的$n \geqslant n_0$，都满足$T(n) \leqslant cf(n)$。当n充分大时，随n的增加，如果函数$T(n)$存在一个增长的上界$cf(n)$，则算法的渐近时间复杂度表示为$O(f(n))$，简称为时间复杂度，算法时间复杂度的渐近阶为$f(n)$，记为$T(n) = O(f(n))$。

当算法的时间复杂度$T(n)$与问题规模n无关时，$T(n) = O(1)$；当$T(n)$与问题规模n为线性关系时，$T(n) = O(n)$；当$T(n)$与问题规模n为平方关系时，$T(n) = O(n^2)$；另外，常见的时间复杂度还有$O(\log_2 n)$、$O(n\log_2 n)$等。通常简单分析算法的时间复杂度可以依据算法中基本语句的重复执行次数来估算。

算法的时间复杂度不仅仅依赖于问题的规模，也取决于输入实例的初始状态。最坏时间复杂度是指在最坏情况下算法的时间复杂度。平均时间复杂度是指所有可能的输入实例均以等概率出现的情况下，算法的期望运行时间。

(3)算法的空间复杂度

算法的空间复杂度$S(n)$是问题规模n的空间函数，反映为算法所消耗的存储空间。算法的渐近空间复杂度(常称为空间复杂度)表示方法与时间复杂度类似，表示为$O(f(n))$，记为$S(n) = O(f(n))$。一个程序运行时占用的空间包括存放程序本身需要的存储空间和解决问题所需要的辅助存储空间，如临时工作单元及递归算法中所需要的递归工作栈等。通常如果输入数据所占空间只取决于问题本身和算法无关，则只需分析所占辅助存储空间的多少。

● 典型例题解析 ●

【例1】 设数据的逻辑结构定义为S=(D,R)，其中D={a,b,c,d,e}，R={r}，r={⟨a,b⟩,⟨b,c⟩,⟨b,d⟩,⟨e,b⟩}，请画出对

应的逻辑结构，说明是何种数据结构。

解 S 对应的数据逻辑结构图见图 1-1。由于结点 b 有两个直接前驱结点 a 和 e，有两个直接后继结点 c 和 d，所以它是一个有向图。

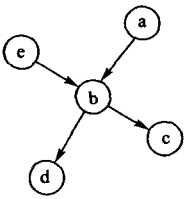

图 1-1 数据逻辑结构图

【例 2】 试将序列 2^{100}，$(2/3)^n$，$(3/2)^n$，n^n，$n!$，2^n，$\log_2 n$，$n^{\log_2 n}$，$n^{3/2}$，$\sqrt{n}$ 按增长率由小到大进行排序。

解 2^{100} 是常数阶，$(2/3)^n$ 和 $(3/2)^n$ 是指数阶，n^n 是指数方阶，$n!$ 相当于 n 次方阶，2^n 是指数阶，$\log_2 n$ 是对数阶，$n^{\log_2 n}$ 是对数方阶，$n^{3/2}$ 是 3/2 次方阶，$\sqrt{n}$ 是平方根阶。

按增长率由小至大的顺序可排列如下：

$(2/3)^n$，2^{100}，$\log_2 n$，$\sqrt{n}$，$n^{3/2}$，$n^{\log_2 n}$，$(3/2)^n$，2^n，$n!$，n^n。

【例 3】 试将序列 $2n$，n^2，n^3，2^n，$n!$，n^n 按增长率由小到大的顺序排列。

解 表 1-1 给出各函数值的列表，当 $n \geqslant 10$ 时，按增长率由小到大排列次序为

$$2n, n^2, n^3, 2^n, n!, n^n$$

表 1-1　　各函数值增长表

n	$2n$	n^2	n^3	2^n	$n!$	n^n
1	2	1	1	2	1	1
2	4	4	8	4	2	4
3	6	9	27	8	6	27
4	8	16	64	16	24	256
5	10	25	125	32	120	3125
6	12	36	216	64	720	46656
7	14	49	343	128	5040	823543
8	16	64	512	256	40320	16777216
9	18	81	729	512	362880	3.9×10^8
10	20	100	1000	1024	3628800	1.0×10^{10}

此为试读，需要完整PDF请访问：www.ertongbook.com

【例 4】　将下列算法的时间复杂度，按由低到高的顺序排列（n 是问题的规模），$O(n)$，$O(2^n)$，$O(\log_2 n)$，$O(n\log_2 n)$，$O(n^5)$，$O(n^2+1)$，$O(n^3-n^2)$。

解　$O(n^2+1)=O(n^2)$，$O(n^3-n^2)=O(n^3)$，所以由低到高的顺序排列如下：

$O(\log_2 n)$，$O(n)$，$O(n\log_2 n)$，$O(n^2+1)$，$O(n^3-n^2)$，$O(n^5)$，$O(2^n)$。

【例 5】　计算多项式 $P_n(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$ 值的算法如下，请分析算法的时间复杂度。

解

```
float value (float a[ ], float x, int n)
{ int i;
  float f=1, p=a[0];
  for (i=1;i<=n;i++)
  { f*=x;
    p=p+a[i]*f;
  }
  return p;
}
```

由于循环体内的语句重复执行 n 次，故算法的时间复杂度为 $O(n)$。

● 同步测试 ●

1. 下面是几种数据的逻辑结构 S=(D,R)，分别画出对应的数据逻辑结构，并指出它们分别属于何种结构。

D={a,b,c,d,e,f}　R={r}

(a)　r={⟨a,b⟩,⟨b,c⟩,⟨c,d⟩,⟨d,e⟩,⟨e,f⟩}

(b)　r={⟨a,b⟩,⟨b,c⟩,⟨b,d⟩,⟨d,e⟩,⟨d,f⟩}

(c) r={⟨a,b⟩,⟨b,c⟩,⟨d,a⟩,⟨d,b⟩,⟨d,e⟩,⟨d,f⟩}

2. 分析下列程序段的时间复杂度

```
(a)  for (i=0; i<m; i++)
         for (j=0; j<n; j++)
             b[i][j]=0;
(b)  s=0;
     for (i=0; i<n; i++)
         for (j=i; j<n; j++)
             s+=b[i][j];
(c)  i=1;
     while (i<n)
         i*=2;
```

3. 在数据结构中,与所使用的计算机无关的是_____。

A. 存储结构　　B. 物理结构

C. 物理和存储结构　　D. 逻辑结构

4. 非线性结构中每个结点_____。

A. 无直接前驱结点

B. 只有一个直接前驱和直接后继结点

C. 无直接后继结点

D. 可能有多个直接前驱和多个直接后继结点

5. 可以把数据的逻辑结构划分成_____。

A. 内部结构和外部结构　　B. 动态结构和静态结构

C. 紧凑结构和非紧凑结构　　D. 线性结构和非线性结构

● 同步测试参考答案 ●

1. (a)是线性结构,对应的数据逻辑结构图见图 1-2。

(b)是树形结构,对应的数据逻辑结构图见图 1-3。

(c)是有向图,其数据逻辑结构图见图 1-4。

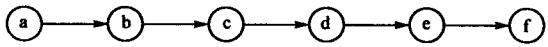

图 1-2　线性结构

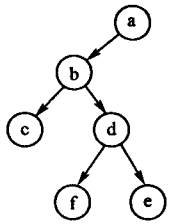

图 1-3　树形结构

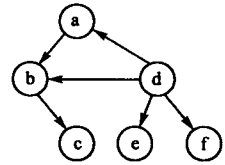

图 1-4　有向图

2. (a) $O(m \times n)$

(b) $s+ = b[i][j]$ 的重复执行次数是 $n(n+1)/2$,时间复杂度是 $O(n^2)$

(c) $O(\log_2 n)$

3. D　　**4.** D　　**5.** D

第2章　线性表

● 重点内容概要 ●

1. 线性表的逻辑结构和基本运算

线性表：是一种典型的线性结构，它是由 $n(n \geqslant 0)$ 个结点 a_1，a_2，…，a_n 组成的有限序列。其中，结点个数 n 定义为表的长度：当 $n = 0$ 时称为空表，当 $n > 0$ 时，常将非空表记做 $L = (a_1, a_2, \cdots, a_n)$。显然，$a_1$ 为开始结点，a_n 为终端结点，除了开始结点外，其余每个结点 $a_i(2 \leqslant i \leqslant n)$ 都有且仅有一个直接前驱结点 a_{i-1}，除了终端结点外，其余每个结点 $a_i(1 \leqslant i \leqslant n-1)$ 都有且仅有一个直接后继结点 a_{i+1}。

在线性表这种逻辑结构上定义了六种基本的运算。

(1) InitList(L)：对表 L 初始化。

(2) ListLength(L)：求表 L 的长度。

(3) GetNode(L, i)：取表 L 中的第 i 个结点，$1 \leqslant i \leqslant$ ListLength(L)。

(4) LocateNode(L, x)：在 L 中查找值为 x 的结点。若找到，则返回首次找到的结点位置；若没找到，则返回一个特殊值(或 NULL)表示查找失败。

(5) InsertList(L, x, i)：在 L 中的第 i 个位置上插入一个值为 x 的新结点，表 L 的长度增 1，$1 \leqslant i \leqslant n+1$。

(6) DeleteList(L, i)：删除表 L 的第 i 个结点，表 L 的长度减 1，$1 \leqslant i \leqslant n$。

由于存储方式的不同，六种基本运算实现的方式亦不同。线

性表主要有两种存储结构:顺序存储结构和链式存储结构。

2. 线性表的顺序存储结构

把线性表的结点按逻辑次序依次存放在一组地址连续的存储单元里,这种方法存储的线性表称为顺序表。顺序表是线性表的顺序存储结构。顺序表的特点是:逻辑上相邻的数据元素其物理存储位置必须紧邻。顺序表可以随机存取。通过数据元素序号可直接计算出存储位置。设线性表中所有结点的类型相同,每个结点占用 c 个存储单元,开始结点 a_1 的存放的首地址(基地址)是 $LOC(a_1)$,则 a_i 的存储地址 $LOC(a_i) = LOC(a_1) + (i-1) * c(1 \leqslant i \leqslant n)$。

顺序表的类型定义如下:

```
#define M 100/*线性表的最大容量*/
typedef struct {
        int data[M];  /*结点类型,可为 int 或 char 等,为了说明
                        问题方便,设为 int*/
    int n;/* 表长 */
    } Slist;
```

在顺序表中插入一个结点时,需要先将从表尾到插入位置的所有结点依次向后移动一个位置,然后将新的结点插入应存位置,表长加 1。在等概率的条件下,插入运算平均需要移动结点的次数为 $n/2$,其时间复杂度为 $O(n)$。删除一个结点时,需要将删除位置后的所有结点依次向前移动一个位置,表长减 1。在等概率的条件下,删除运算平均需要移动结点的次数为 $(n-1)/2$,其时间复杂度为 $O(n)$。

3. 线性表的链式存储结构

链式存储结构的线性表称为链表,是用任意的存储单元存放线性表中的数据元素,链表中逻辑上相邻的数据元素其物理存储位置不要求紧邻。用附加的指针域指示结点间的逻辑关系。下面

介绍常用的三种链表。

(1)单链表:每个结点附加一个指针(链)域,用来指向该结点的直接后继结点(地址)。对线性表(a_1,a_2,…,a_i,…,a_n)的单链表的图形表示见图 2-1。head 是单链表头指针,指向开始结点 a_1;a_n 是终端结点,其指针域为空。一个单链表由头指针 head 唯一确定,因此可用头指针来命名单链表。

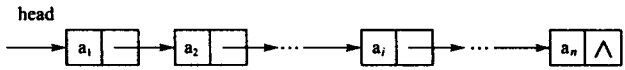

图 2-1 单链表示意图

结点类型定义为

```
typedef struct node { int data;/*结点的数据域*/
                      struct node * next;/*结点的链域*/
                    }Lnode;
```

为了使单链表中第一个结点与其余各结点的基本运算统一,通常在单链表前另加一个表头结点,表头结点的链域指针指向单链表中第一个结点。本书如不特别说明,单链表都带表头结点。带头结点的单链表的图示如图 2-2 所示。

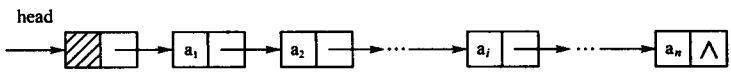

图 2-2 带头结点的单链表示意图

(2)双向链表:每个结点中有两个指针域,其中 next 指向其直接后继结点,prior 指向其直接前驱结点。由于双向链表是一种对称结构,故可以方便地向前或向后访问结点。双向链表一般也由头指针 head 唯一确定,也可增加头结点。带头结点的双向链表的图示如图 2-3 所示。

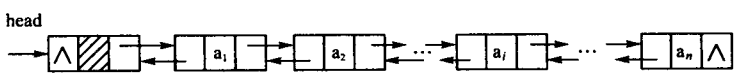

图 2-3 带头结点的双向链表示意图

此为试读,需要完整PDF请访问: www.ertongbook.com