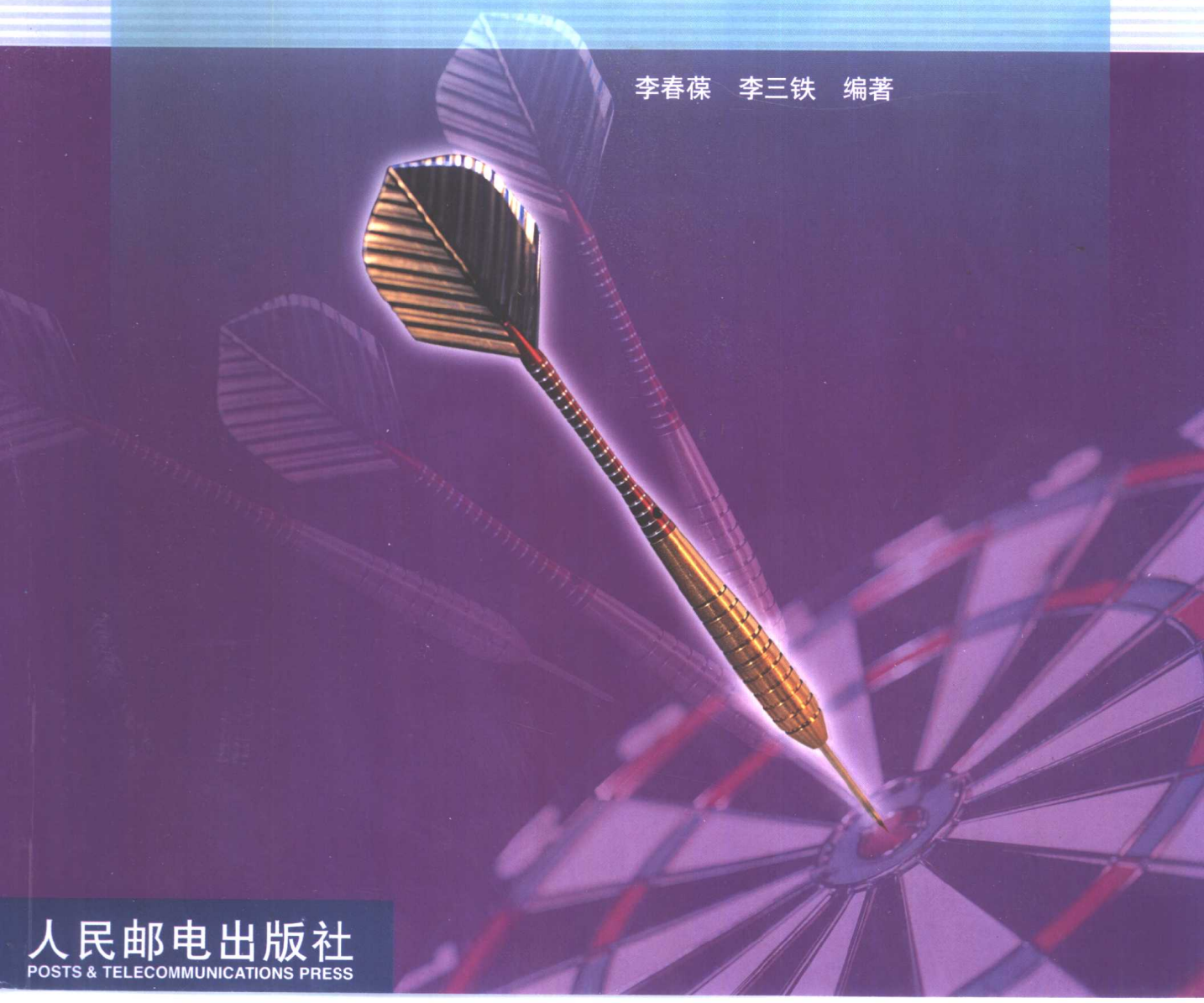
计算机专业考研指导丛书

数据结构

考点精要与解题指导

李春葆　李三铁　编著

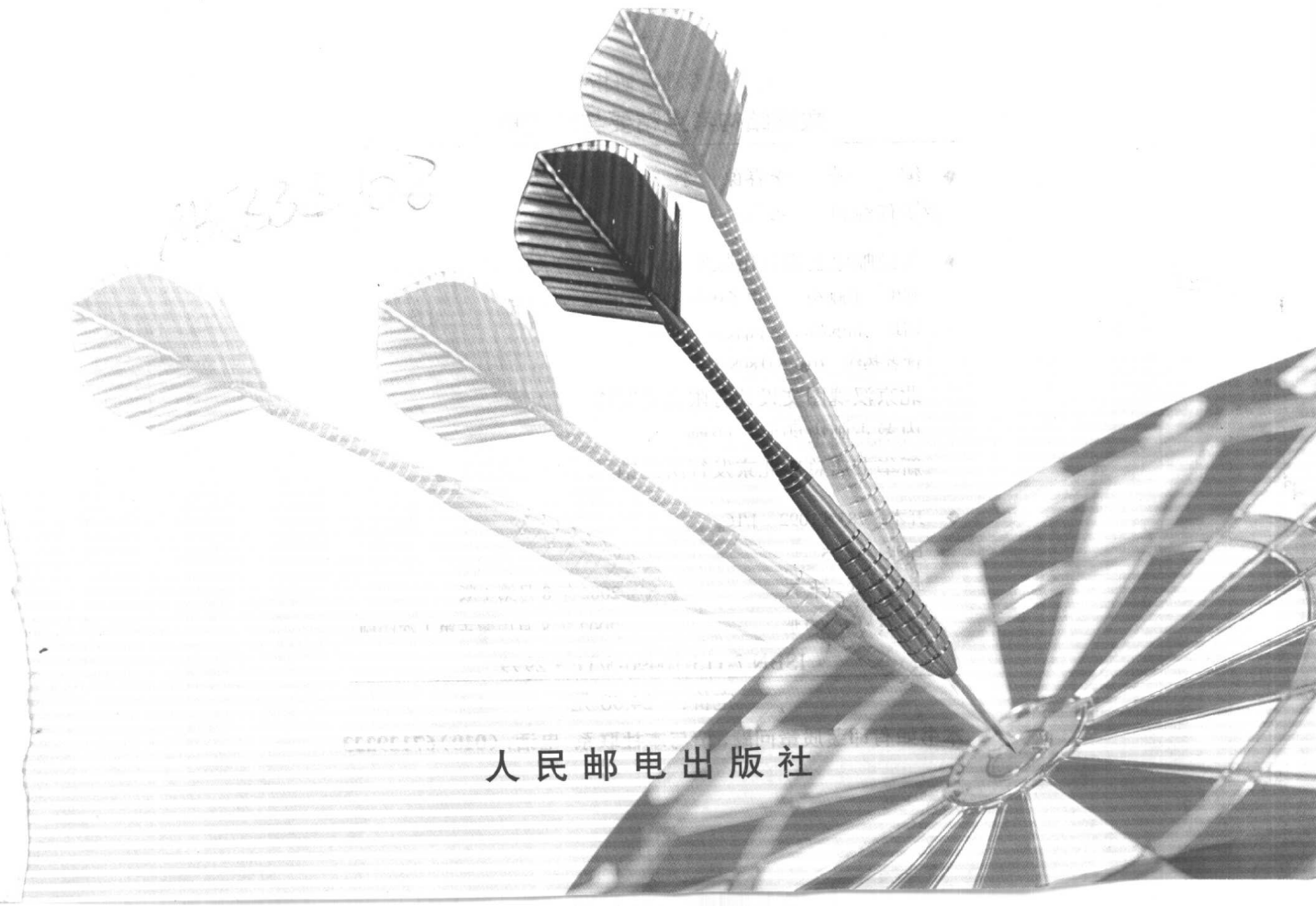


人民邮电出版社
POSTS & TELECOMMUNICATIONS PRESS

计算机专业考研指导丛书

数据结构
考点精要与解题指导

李春葆　李三铁　编著



人民邮电出版社

图书在版编目(CIP)数据

数据结构考点精要与解题指导/李春葆，李三铁编著. 北京：人民邮电出版社，2002.8
(计算机专业考研指导丛书)
ISBN 7-115-10490-5

I. 数… II. ①李…②李… III. 数据结构—研究生—入学考试—自学参考资料 IV. TP311.12

中国版本图书馆 CIP 数据核字(2002)第 054213 号

计算机专业考研指导丛书

数据结构考点精要与解题指导

◆ 编　　著　李春葆　李三铁
责任编辑　邓革浩

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www. ptpress.com.cn
读者热线　010-67180876
北京汉魂图文设计有限公司制作
内蒙古邮电印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：16.75
字数：399 千字　　2002 年 8 月第 1 版
印数：1-6 000 册　　2002 年 8 月内蒙古第 1 次印刷

ISBN 7-115-10490-5/TP・2999

定价：　24.00 元

本书如有印装质量问题，请与本社联系　电话：(010)67129223

内容提要

目前数据结构是各大专院校计算机专业的核心课程,也是很多高校招收计算机专业研究生必考的科目之一。

本书是针对考研者编写的，书中高度概括和总结了数据结构的基本考点，收集了大量的研究生入学考试试题并给出了分析和解答。全书分为 11 章，其内容包括：绪论、线性表、栈和队列、串、数组和广义表、树和二叉树、图、查找、内排序、外排序、文件。每章由三部分构成，即考点精要、例题解析、自测题及参考答案。考点精要部分高度概括了本章考试内容及注意要点；例题解析部分详尽地解答了精选的考研试题，各题都包含有相关知识、例题分析和例题答案；自测题及参考答案收集了大量的相关试题并给出了相应的参考答案。

本书的特点是概念清晰，文字简洁明了，解题思路完整，极便于考研者短时间内掌握解题要点，提高考试成绩。

本书适合于考研者应试复习和提高，同样也适合于作为大专院校各专业数据结构课程的复习参考书，还可供计算机软件水平考试者研习。

丛书序

计算机专业是当今最热门也是发展最迅速的学科之一，很多学生为了进一步提高专业水平和应用能力，纷纷报考计算机专业研究生。据统计，近几年报考计算机软件与理论、计算机应用和计算机与通信专业硕士研究生的考生远远超过报考其他专业的考生，其中有相当一部分考生原来所学专业并非计算机专业，还有很多考生是工作多年的在职人员。为了方便报考者复习计算机专业课程，我们特地组织一批计算机专业教学第一线的教授和副教授（其中大多数编写者多年参加硕士研究生入学试题命题工作）编写了本丛书。本丛书包含如下课程：

- 《C 程序设计考点精要与解题指导》
- 《离散数学考点精要与解题指导》
- 《数据结构考点精要与解题指导》
- 《操作系统考点精要与解题指导》
- 《编译原理考点精要与解题指导》
- 《计算机组成原理考点精要与解题指导》

本丛书具有以下特点：

◎ 讲述全面而详实

本丛书涵盖各门专业课程的内容，不是针对个别学校的命题特点，而是充分地讲授课程中的重点、难点和考点，并通过例题进行扩充与深化，使读者得以全面温习，不留“死角”。

◎ 阐述简洁而明了

不同于本、专科教材，本丛书的目的是使考生花较少的时间温习各门课程的内容，因此，不过多地解释简单的术语，只对基本知识点进行高度概括和总结，使读者将主要精力花在解题过程上。

◎ 重点突出解题思路

本丛书重点介绍解题的方式和方法，不仅授人以“鱼”，更在于授人以“渔”，选择的例题和习题大多是计算机专业研究生入学考试试题（题目前标有“★”号），并配上详解，具有很强的实战性。

◎ 强调内容的综合与提高

一般的教科书多是按照内容的先后顺序按步就班地介绍，这种方式有助于初学者学习，但不便于复习和综合，因为考研题一般都具有很强的综合性，往往一个题涉及好几章的概念，所以本丛书打破了一般教科书的教学模式，将相关的概念有机地融为一体，从而提高考生的解题能力。

◎ 答疑解惑

本丛书选择的例题和习题大部分具有较高的难度，书中不仅给出了答案，而且详细介绍了解题思路和解题过程，有助于考生纠正以往的概念误区。

本丛书希望在考研指导方面作一些探索和尝试，起到抛砖引玉的作用，书中的不妥之处敬请广大的读者和同行指教。

李春葆

2002.6

此为试读，需要完整PDF请访问：www.ertongbook.com

前　言

“数据结构”是计算机及其相关专业的重要基础课，也是大多数高校招收计算机及相关专业硕士研究生的必考科目之一。

本书是结合考研的特点编写的。全书分为 11 章，第 1 章为绪论，第 2 章介绍线性表，第 3 章介绍栈和队列，第 4 章介绍串，第 5 章介绍数组和广义表，第 6 章介绍树和二叉树，第 7 章介绍图，第 8 章介绍查找，第 9 章介绍内排序，第 10 章介绍外排序，第 11 章介绍文件。

每章由三部分构成，即考点精要、例题解析、自测题及参考答案。考点精要部分高度概括了本章考试内容及注意要点；例题解析部分详尽地解答了精选的考研试题，各题都包含有相关知识、例题分析和例题答案（少数比较简单的例题没有给出例题分析）；自测题及参考答案收集了大量的相关试题并给出了相应的参考答案。

书中涵盖了近几年来大量的高校数据结构考研试题（这些题目前加有“★”号），并给出了详解或参考答案，为了统一，将原题中用 Pascal 语言编写的算法均改用 C 语言编写。另外，对于有些考研试题中同一概念采用不同术语的情况，本书也进行了统一，如二叉查找树均统一为二叉排序树，堆栈均统一为栈，环形队列均统一为循环队列等。

书中的部分程序在 Visual C++ 6.0 环境下调试通过，读者只需要具备简单的 C++基础就能阅读这些程序。为了照顾只熟悉 C 语言的读者，书中尽量采用 C 语言的语法，并在附录中给出了本书程序使用的 C++语法的说明。

本书的特点是概念清晰，文字简洁明了，所有题目都给出了详细的解答，极便于读者在短时间内掌握解题要点。

尽管本书主要针对考研者应试复习和提高，但同样也适合于作为大专院校各专业数据结构的复习参考书，还可供计算机软件水平考试者研习。

由于作者水平有限，书中难免存在不足之处，敬请有关专家和广大读者不吝指正。

编者

2002.6

目　录

第1章 绪 论

基本知识点：数据结构和算法的概念。

重点：数据结构的逻辑结构、存储结构、数据运算三方面的概念及相互关系；算法时间复杂度分析。

难点：分析算法的时间复杂度。

1.1 考点精要

1.1.1 什么是数据结构

1. 数据

数据：是指能够被计算机识别、存储和加工处理的信息的载体。

数据元素：是数据的基本单位，也称为元素、结点、顶点、记录等。一个数据元素可以由若干个数据项组成。数据项是具有独立含义的最小标识单位。

2. 数据结构

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。它一般包括以下3个方面的内容。

（1）数据的逻辑结构：指数据元素之间的逻辑关系，即从逻辑关系上描述数据。它与数据的存储无关，是独立于计算机的。

（2）数据的存储结构：指数据元素及其关系在计算机存储器内的表示（也称为映像）。数据的存储结构是逻辑结构用计算机语言的实现，它依赖于计算机语言。

（3）数据的运算：指对数据施加的操作。数据的运算是定义在数据的逻辑结构上的，而实现则要在存储结构上进行。

3. 数据的逻辑结构

在不引起混淆的前提下，通常将数据的逻辑结构简称为数据结构。数据的逻辑结构有两大类：

（1）线性结构

线性结构的逻辑特征是：若结构是非空集，则有且仅有一个开始结点和一个终端结点，并且所有结点都最多只有一个直接前趋和一个直接后继。例如，线性表是一个线性结构。

（2）非线性结构

非线性结构的逻辑特征是一个结点可能有多个直接前趋和直接后继。例如，树和图都是非线性结构。

4. 数据的存储结构

数据通常有以下4种常用的存储表示方法：

（1）顺序存储方法

该方法是把逻辑上相邻的结点存储在物理位置上相邻的存储单元里，结点间的逻辑关系由存储单元的邻接关系来体现，由此得到的存储结构称为顺序存储结构，通常顺序存储结构是借助于程序语言的数组来描述的。

（2）链接存储方法

该方法不要求逻辑上相邻的结点在物理位置上亦相邻，结点间的逻辑关系是由附加的指针字段表示的，由此得到的存储表示称为链式存储结构，通常要借助于程序语言的指针类型来描述它。

（3）索引存储方法

该方法通常是在存储结点信息的同时，还建立附加的索引表。索引表中的每一项称为索引项，索引项的一般形式是：（关键字，地址）。其中关键字惟一标识结点，地址作为指向结点的指针。

（4）散列存储方法

该方法的基本思想是根据结点的关键字直接计算出该结点的存储地址。

1.1.2 算法和算法分析

1. 什么是算法

算法是对特定问题求解步骤的一种描述，它是指令的有限序列，其中每条指令表示一个或多个操作。算法有以下5个主要特征：

（1）有穷性：一个算法必须总是（对任何合法的输入）在执行有穷步之后结束，且每一步都可在有穷时间内完成；

（2）确定性：算法中每一条指令必须有确切的含义，确保不会产生二义性。

（3）可行性：一个算法是可行的，即算法中描述的操作是可以通过基本运算的有限次执行来实现的。

（4）输入性：一个算法有0个或多个的输入。

（5）输出性：一个算法有一个或多个的输出。

2. 算法的时间复杂度

一个语句的频度，是指该语句在算法中被重复执行的次数。算法中所有语句的频度之和记作 $T(n)$，它是该算法所求解问题规模 n 的函数。当问题的规模趋向无穷大时，$T(n)$ 的数量级称为渐近时间复杂度，简称为时间复杂度，记作 $T(n)=O(f(n))$。

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。一个问题的输入实例是满足问题陈述中所给出的限制和为计算该问题的解所需要的所有输入构成的。

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。

上述表达式中“O”的含义是 $T(n)$ 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，则存在正的常数 C 和 $n0$，使得当 $n \geqslant n0$ 时，都满足 $0 \leqslant$

此为试读，需要完整PDF请访问：www.ertongbook.com

$T(n) \leqslant C \times f(n)$。

一般总是考虑在最坏的情况下的时间复杂度，以保证算法的运行时间不会比它更长。

3. 算法的空间复杂度

算法的空间复杂度 $S(n)$，定义为该算法所耗费的存储空间，它是问题规模 n 的函数。渐进空间复杂度也常常简称为空间复杂度。

1.2 例题解析

例 1.1 逻辑结构和存储结构之间的关系?

【相关知识】 数据结构的概念。

【例题答案】 对于已经建立的逻辑结构是设计人员根据解题需要选定的数据组织形式，因此建立的机内表示应遵循选定的逻辑结构，所建立数据的机内表示称为数据存储结构。

例 1.2 ★常用的存储表示方法有哪几种?

【相关知识】 数据的存储结构。

【例题答案】 常用的存储表示方法有 4 种：

(1) 顺序存储方法：它是把逻辑上相邻的结点存储在物理位置相邻的存储单元里，结点的逻辑关系由存储单元的邻接关系来体现，由此得到的存储结构称为顺序存储结构。

(2) 链式存储方法：它不要求逻辑上相邻的结点在物理位置上亦相邻，结点之间的逻辑关系是由附加的指针字段表示的。由此得到的存储结构称为链式存储结构。

(3) 索引存储方法：除建立存储结点信息外，还建立附加的索引表来标识结点的地址。

(4) 散列存储方法：根据结点的关键字直接计算出该结点的存储地址。

例 1.3 ★有实现同一功能的两个算法 A1 和 A2，其中 A1 的时间复杂度为 $T1(n)=O(2^n)$，A2 的时间复杂度为 $T2(n)=O(n^2)$，仅就时间复杂度而言，请具体分析这两个算法哪一个更好。

【相关知识】 算法的时间复杂度。

【例题答案】 显然算法 A2 好于 A1，因为随着问题规模 n 的增长，算法 A2 所花费的时间远远少于算法 A1 所花的时间。具体分析如表 1.1 所列。

表 1.1　　算法 A1 与 A2 的比较

n	A1 所花时间	A2 所花时间
1	2	1
2	4	4
3	8	9
4	16	16
5	32	25
6	64	36

续表

n	A1 所花时间	A2 所花时间
7	128	49
8	256	64
9	512	81
10	1024	100
……	……	……

例 1.4 ★设 n 为正整数，给出以下程序段的时间复杂度。

```
（1）void func1(int n)
    {
        int i=1;
        int k=0;
        while (i<n) {
            k=k+lO*i;i++;
        }
    }
（2）void func2(int n)
    {
        int y=1;
        int x=n;
        while(x>=(y+1)*(y+1))
            y++;
    }
（3）void func3(int n)
    {
        int x=91，y=100;
        while (y>0)
            if (x>100) {
                x=x-10;y--;
            }
            else x++;
    }
（4）void func4(int n)
    {
        i=1;
        while (i<=n)
```

```
        i=i*3;
}
```

【相关知识】 算法时间复杂度分析。

【例题答案】 设$f(n)$为各算法的频度，$T(n)$为对应的时间复杂度。

(1) $f(n)=n-1$，所以$T(n)=O(n)$。

(2) $f(n)=\log_2 n$，所以$T(n)=O(\log_2 n)$。

(3) $f(n)=O(1)$，因为while语句与n无关，所以$T(n)=O(1)$。

(4) $f(n)^3 \leqslant n$，则$f(n) \leqslant \log_3 n$，$T(n)=O(\log_3 n)$

例1.5 ★按增长率由小至大的顺序排列下列各函数：

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

【相关知识】 时间复杂度。

【例题分析】 2^{100}是常数阶，$(2/3)^n$和$(3/2)^n$是指数阶，其中前者是随n的增大而减小的；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；$n^{\log 2n}$是对数方阶；$n^{3/2}$是3/2次方阶；$\sqrt{n}$是平方根阶。

【例题答案】 根据以上分析按增长率由小至大的顺序可排列如下：

$(2/3)^n < 2^{100} < \log_2 n < \sqrt{n} < n^{3/2} < n^{\log 2n} < (3/2)^n < 2^n < n! < n^n$

例1.6 ★设计一个算法求解Hanoi问题：有3根柱子A、B、C，有n个半径不同的中间有孔的圆盘，这n个圆盘在柱子A上，从上往下半径依次增大，要求把所有圆盘移至目标盘C上，可将柱子B作为辅助柱，移动圆盘时必须服从以下规则：

(1) 每次只可搬动一个圆盘。

(2) 任何柱子上都不允许大圆盘在小圆盘的上面。

并分析算法的时间复杂度。

【相关知识】 算法设计与时间复杂度分析。

【例题分析】 求解Hanoi问题的思路为：当$n=1$时，搬动一次即可；当$n>1$时，分3步进行：

(1) 将$n-1$个圆盘（除最大的圆盘外）从柱子A移到柱子B；

(2) 将最大的圆盘从柱子A移动到柱子C；

(3) 将柱子B上的$n-1$个圆盘从柱子B移动到柱子C。

【例题答案】 由此得到以下求解算法：

```
void hanoi（int n，char a，char b，char c）
{
    if (n==1)
        printf("move %d disk from %c to %c\n"，n，a，c);
    else {
        Hanoi(n-1，a，c，b);
        Printf("move %d disk from %c to %c\n"，n，a，c);
        hanoi（n-1，b，a，c）;
    }
}
```

由上述算法得到时间复杂度的递归关系如下：

$$T(n)=\begin{cases}1 & n=1\\ 2T(n-1)+1 & n>1\end{cases}$$

所以 $T(n)=2T(n-1)+1$

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

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

$=\cdots$

$=2^{n-1}T(1)+2^{n-1}+1=2^n+1$

$=O(2^n)$

1.3 自测题及参考答案

1. 运算与运算的实现是什么关系?有哪些相同点和不同点?

答：一个运算的实现是指完成该运算功能的程序，而运算实现的核心是处理步骤规定，即算法设计，它们二者之间一个抽象，一个具体。

2. ★举一个数据结构的例子，叙述其逻辑结构、存储结构和运算 3 个方面的内容。

答：例如有一张学生成绩表，如表 1.2 所列，记录了一个班的学生各门课的成绩。按学生的姓名为一行构成的表，这个表就是一个数据结构。每个记录（包括学号，姓名，成绩）就是一个结点，对于整个表来说，只有一个开始结点（前面无记录）和一个终端结点（后面无记录），其他的结点则各有一个也只有一个直接前趋和直接后继（它的前面和后面均有且有一个记录）。这几个关系就确定了这个表的逻辑结构。

表 1.2　　学生成绩表

学　号	姓　名	成　绩
1	王华	90
2	李明	84
……	……	……

那么我们怎样把这个表中的数据存储到计算机里呢?用高级语言如何表示各结点之间的关系呢?是用一片连续的内存单元来存放这些记录（如用数组表示），还是随机存放各结点数据再用指针进行链接呢?这就是存储结构的问题。我们都是从高级语言的层次来讨论这个问题的。例如，若用链式存储方式，结点的数据类型定义如下：

```
struct node {
   int no;                 //存放学号的数据域
   char name[10];          //存放姓名的数据域
   int score;              //存放成绩的数据域
   struct node *next;      //指向下一个结点的指针
}
```

最后，我们有了这个表（数据结构），肯定要用它，那么就要对这张表中的记录进行查询、修改、删除等操作，对这个表可以进行哪些操作以及如何实现这些操作就是数据的运算问题了。若采用链式存储方式，其运算的实现参见第2章中单链表的相关算法。

3. ★将下列算法的时间复杂度级别，按照由低到高的顺序排成一行（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：其中$O(n^2+1)=O(n^2)$，$O(n^3-n^2)=O(n^3)$，所以由低到高的顺序如下：

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

4. ★设n是偶数，试计算运行下列程序段后m的值并给出该程序段的时间复杂度。

```
m=0;
for (i=1;i<=n;i++)
    for (j=2*i;j<=n;j++)
        m++;
```

解：n为偶数，外循环$i=1$时，内循环j从$2\sim n$，语句m++执行$n-2+1$即$n-1$次；外循环$i=2$时，内循环j从$4\sim n$，语句m++执行$n-4+1$即$n-3$次……外循环$i=n/2$时，内循环j从$n\sim n$，语句m++执行1次；当外循环中$i>n/2$时，内循环不执行，而m的值即为内循环执行的次数，所以：

$$m=(n-1)+(n-3)+\cdots+1 \quad (\text{共 } n/2 \text{ 项})$$
$$=n^2/4$$

该程序段的时间复杂度为$O(n^2)$。

5. 下面程序段的时间复杂度为多少？

```
void mergesort(int i, int j)
{
    int m;
    if (i!=j) {
        m=(i+j)/2;
        mergesort(i, m);
        mergesoft(m+1, j);
        merge(i, j, m);
    }
}
```

其中mergesort()用于数组a[n]的归并排序，调用方式为mergesort(0, n-1)；merge()用于两个有序子序列的合并，是非递归函数，它的时间复杂度为O(n)。

解：分析得到以下求时间复杂度的递归关系：

$$T(n)=\begin{cases}0(1)\\2T(n/2)+0(n)\end{cases}$$

O(n)为merge()所需的时间，设为cn(c为常量)。因此：

T(n)=2T(n/2)+cn

=2(2T(n/4)+cn/2)+cn

$=2^2T(n/4)+cn+cn$

$=2^2T(n/2^2)+cn+cn$

$=2^3T(n/2^3)+cn+cn+cn$

$=\cdots$

$=2^kT(n/2^k)+cn+...+cn$　（有 k 个 cn 相加）

$=2^kO(1)+kcn$

由 $n/2^k=1$ 得 $k=\log_2 n$

所以 $T(n)=2^{\log_2 n}O(1)+cn\log_2 n=n+cn\log_2 n=O(n\log_2 n)$

6. ★编写一个函数，用不多于 $3n/2$ 的平均比较次数，在一个向量 A 中找出最大和最小值的元素。

解：如果在查找出最大和最小值的元素时各扫描一遍所有元素，则至少要比较 $2n$ 次，为此，使用一趟扫描找出最大和最小值的元素。算法如下：

```
void maxmin(int A[], int n)
{
    int max, min, i;
    max=A[0]; min=A[0];
    for (i=1; i<n; i++) {
        if (A[i]>max)
            max=A[i];
        else if (A[i]<min)
            min=A[i];
    }
    printf("max=%d, min=%d\n", max, min);
}
```

在这个函数中，最坏的情况是数组 A 的元素以递减顺序排列，这时（A[i]>max）条件均不成立，这时比较的次数为 n-1，另外每次都要比较 A[i]<min，所花比较次数同样为 n-1，因此，总的比较次数为：2（n-1）。

最好的情况是数组 A 的元素以递增次序排列，这时（A[i]>max）条件均成立，不会再执行 else 的比较，所以总的比较次数为：n-1。

平均比较次数为：（2（n-1）+n-1）/2=3n/2-3/2

所以该函数的平均比较次数不多于 3n/2。

7. ★运算是数据结构的一个重要方面，试举一例说明两个数据结构的逻辑结构和存储方式完全相同，只是对于运算的定义不同，因而两个结构具有显著不同的特性，是两个不同的结构。

解：在数据结构中这类例子较多，如顺序表和字符串等。下面以二叉树和二叉排序树进行说明。

二叉树的定义为：二叉树是有限的结点集合，这个集合或者是空，或者由一个根结点和两棵互不相交的称为左子树和右子树的二叉树组成。

二叉排序树的定义为：二叉排序树或者是空树，或者是满足如下性质的二叉树：

此为试读，需要完整PDF请访问：www.ertongbook.com