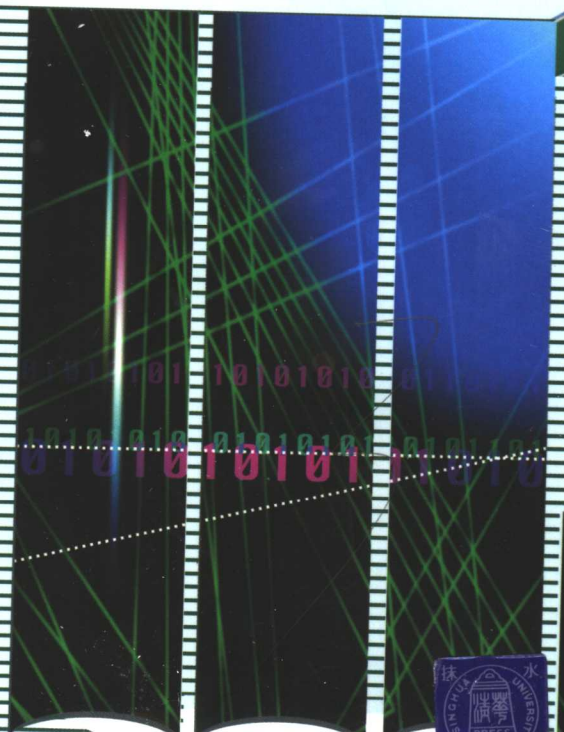
数据结构
复习指南与题解

刘海英　马征兵　李俊杰　编著

全面提升水平的应试辅导书:

- 涵盖知识重点
- 精选典型实例
- 题解详尽透彻



清华大学出版社

数据结构复习指南与题解

刘海英　马征兵　李俊杰　编著

清华大学出版社
北　京

内 容 简 介

计算机是被广泛使用的工具，数据结构课程是学习计算机软件设计的基础课程。本书是作者在长期教学经验积累的基础上精心编著的数据结构课程的学习参考书。全书共分 10 章，主要有数据结构基础知识、线性表、栈和队列、串、数组和广义表、树与二叉树、图、查找、排序和文件等内容，各章均包括基本概念、基本理论、典型实例和习题。本书使用类 C 语言作为算法描述语言，且所有算法都可以在任意一种 C 语言的开发环境中实现。

本书是计算机本科和专科学生的学习参考书，并可作为报考计算机专业硕士研究生、参加国家高等教育自学考试、高等学校专升本或计算机等级三级和四级考试考生的复习参考书。

图书在版编目(CIP)数据

数据结构复习指南与题解/刘海英，马征兵，李俊杰编著. —北京：清华大学出版社，2005.1

ISBN 7-302-07609-X

Ⅰ.数… Ⅱ.①刘…②马…③李… 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(2003)第 103782 号

出 版 者：清华大学出版社 **地 址**：北京清华大学学研大厦

http://www.tup.com.cn **邮 编**：100084

社 总 机：010-62770175 **客户服务**：010-62776969

组稿编辑：文 新

文稿编辑：张彦青

封面设计：陈刘源

印 刷 者：国防工业出版社印刷厂

装 订 者：三河市李旗庄少明装订厂

发 行 者：新华书店总店北京发行所

开 本：185×260 **印张**：21.5 **字数**：511 千字

版 次：2005 年 1 月第 1 版 2005 年 1 月第1 次印刷

书 号：ISBN 7-302-07609-X/TP・5600

印 数：1～4500

定 价：28.00 元

前　言

“数据结构”是计算机程序设计的重要理论技术基础，它是前人在程序设计方面经验的积累。计算机在各个领域的应用过程中，都会或多或少地涉及到数据的组织与程序结构的编排问题，这就要用到各种各样的数据结构。数据结构课程不仅是计算机专业的核心课程，而且也日益成为其他理工专业的热门选修课。

本书共分为 10 章。第 1 章综述了数据结构、存储方式、算法评论等基本概念和一些数据结构中常用的理论，第 2 章至第 7 章从不同数据类型的角度，分别讨论了线性表、串、数组与广义表、树和二叉树、图等基本类的数据结构概念、算法、实现及应用。第 8 章至第 10 章分别就查找、排序和文件进行了讨论。各章都在最后给出了例题与解析，使读者能够在全面掌握“数据结构”知识点的基础上，更好地将“数据结构”的知识灵活运用到实际的“数据结构”中。通过数据结构应用的实例，引导读者对各种不同数据类型的理论和使用加深理解，并通过 “数据结构”试题帮助读者进一步巩固对“数据结构”的理解。

本书文字通俗、简明易懂、便于自学或作为高等院校计算机专业的教学辅导用书，也可以作为从事计算机应用工作的科技人员的实用参考书。

参与本书编写的除了刘海英、马征兵和李俊杰外，还有李艳、郭宇波等人。由于编写时间比较仓促，错误之处在所难免，还望广大的读者批评指正。

编　者

2004.11

目　录

第1章 概 论

随着计算机技术日新月异的发展，计算机已经深入到人类社会的各个领域，其应用已经不再局限于科学计算，而更多地用于控制、管理及数据处理等非数值计算的处理工作。另外，计算机加工处理的对象由纯粹的数值发展到字符、表格和图像等各种具有一定结构的数据，这就必须分析待处理对象的特性以及各处理对象之间存在的关系。数据结构就是研究数据组织、存储和运算的学科。本章主要总结数据结构的基本概念和理论，并对典型例题进行讲解。

1.1 基 本 概 念

数据结构指的是相关之间存在一种或多种特定关系的数据元素的集合。本节将对一些概念和术语赋予确定的含义，以便读者能进一步加深对数据结构含义的理解。

1.1.1 数据的有关概念

- 【数据】数据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到计算机中并被计算机程序处理的符号的总称。它是计算机程序加工的“原料”。
- 【数据项】数据项是数据处理中的最小单位。
- 【数据元素】数据元素是由若干数据项组成的数据的基本单位，也称为结点。在计算机程序中通常作为一个整体进行考虑和处理。
- 【数据类型】数据类型是和数据结构密切相关的一个概念，它最早出现在高级程序设计语言中，用以刻画(程序)操作对象的特性。在程序中，形式不同的数据采用数据类型来标识。变量的数据类型说明变量可能取得的值的集合，以及可能施于变量的操作的集合。所以，数据类型不仅定义了一组形式相同的数据集，也定义了对这组数据可施行的一组操作集。
- 【抽象数据类型】数据类型概念的引申。指一个数学模型以及在其上定义的操作集合，面向逻辑层次。
- 【数据表示】将数据从机外表示转化为机内表示称为数据表示。
- 【数据处理】编制程序使计算机对机内表示的数据进行各种操作，得到所需的结果，这项工作称为数据处理。
- 【数据对象】数据对象是性质相同的数据元素的集合，是数据的一个子集。

1.1.2 数据结构的有关概念

- 【数据逻辑结构】数据逻辑结构是对操作对象的一种数学描述，也就是说，是从操作对象抽象出来的数学模型。它反映的是数据元素之间的逻辑(数学)关系，并不依赖于计算机。
- 【数据物理结构】数据物理结构，又称存储结构，是数据结构在计算机中的表示。它包括数据本身在计算机中的存储方式和数据之间的逻辑关系在计算机中的表示。与数据逻辑结构不同的是，它是依赖于计算机的。其主要有两种：顺序存储结构和链式存储结构。
- 【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。它研究的是数据逻辑结构和数据物理结构以及它们之间的相互关系，并定义与这种结构相适应的运算和设计出相应的算法，且确保经过这些运算以后所得到的新结构仍然是原来的结构类型。
- 【逻辑关系】逻辑关系是指数据元素间的关联方式或称“邻接关系”。在对数据进行整理的过程中，可以发现数据元素都不是孤立存在的，它们之间总是存在某种关系，这种关系的整体称之为逻辑结构，逻辑关系是逻辑结构的核心。
- 【存储结构】(或称为物理结构)：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原则下，数据在存储器中的存储方式。
- 【线性结构】数据逻辑结构中的一类。它的特征是若结构为非空集，则该结构有且只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驱和直接后继。线性表就是一个典型的线性结构。
- 【非线性结构】数据逻辑结构中的一大类。它的逻辑特征是一个结点可能有多个直接前驱和直接后继。

1.1.3 算法的有关概念

- 【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，其中每一条指令表示一个或者多个操作。
- 【最坏时间复杂度】以算法在所有输入下的计算量的最大值作为算法的计算量，这种计算量称为算法的最坏情况时间复杂性或最坏情况时间复杂度。
- 【平均时间复杂度】以算法在所有输入下的计算量的加权平均值作为计算量，这种计算量称为算法的平均时间复杂性或平均时间复杂度。
- 【空间复杂度】对数据进行操作的工作单元和储存一些为实现计算所需信息的辅助空间的大小。
- 【运算实现】在确定的存储结构下，用计算机语言(如“类 C 语言”)描述实现某种操作(或称运算)的算法，称为运算实现。
- 【算法的时空性能】算法的时空性能是指该算法的时间性能(或时间效率)和空间性能(或空间效率)。前者是算法包含的计算量，后者是算法需要的存储量。

1.2 基本理论

上节总结了数据结构的有关概念，这一节将总结数据的结构、数据的运算以及运算的实现等算法方面的理论知识，以便读者能对数据结构的应用加深理解。

1.2.1 数据结构的研究目的和研究内容

(1) 数据结构研究的目的是寻求有效地组织和处理非数值数据的理论、技术和方法，这类数据具有量大且具有一定内在逻辑关系的特点。

(2) 数据结构的核心研究内容包括 3 个方面：数据的逻辑结构、存储结构以及相应的基本操作运算的定义和实现。

1.2.2 逻辑结构的 4 种基本形态及特点

现实世界中非数值数据对象的逻辑关系可以划分为以下 4 种基本结构，它们的复杂程度依次递增(例如，以某班级学生作为数据对象，学籍档案为数据元素为例，考察数据元素之间的关系)。

(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。例如，认定一个学生是否为班级的成员。

(2) 线性结构：数据元素之间存在一对一的关系。例如，以学生入学报到的时间先后顺序排列数据元素。

(3) 树型结构：数据元素之间存在一对多的关系。例如，班级中的管理体系，班长管理多个组长，每个组长管理多个组员。

(4) 图形结构(或称网状结构)：数据元素之间存在多对多的关系。例如，同学之间的朋友关系。

特点：

集合结构中任何两个结点之间都没有逻辑关系，组织形式松散。线性结构中结点按逻辑关系依次排列形成一条“锁链”。树型结构具有分支、层次特性，其形态有点像自然界中的树。图形结构最复杂，其中的各个结点按逻辑关系互相缠绕，任何两个结点都可以邻接。

1.2.3 引入抽象数据类型概念的好处

抽象数据类型概念的引入可以将数据结构的理论分成两部分：一部分是其概念所涵盖的数据逻辑结构的说明和运算的定义，突出的是在某种关系的数据对象上可以做什么；另一部分是在计算机中如何存储这些数据并具体实现定义的运算，解决的是怎样做的问题。

抽象数据类型面向使用层次，隐蔽了实现层次，主要的优点在于实现方法的改变，不会影响用户的使用，可以提高含有该抽象数据类型的软件模块的复用程度。抽象数据类型的概念与面向对象方法的思想是一致的。

1.2.4 逻辑结构的特点及意义

1. 特点

(1) 逻辑结构与数据元素本身的形式、内容无关。
(2) 逻辑结构与数据元素的相对位置无关。
(3) 逻辑结构与所含结点个数无关。

2. 意义

一些表面上很不相同的数据可以有相同的逻辑结构，因此，逻辑结构是数据组织的某种“本质性”的东西。事实上，逻辑结构是数据组织的主要方面。抓住这种本质，不仅有利于分析数据机外表示的组织形式，而且有利于程序设计人员根据解题需要重新组织数据，即把数据中的所有数据元素按所需的逻辑结构重新安排。

1.2.5 算法的特征及设计要求

算法规定了给定类型问题所需的求解步骤及其执行顺序，使得给定类型的任何问题能在有限时间内被机械地求解。

算法有以下 5 个重要特征：

(1) 有穷性：一个算法必须总是(对任何合法的输入值)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，读者理解时不会产生二义性。
(3) 可行性：一个算法是能行的，即算法中描述的操作都是可以通过已经实现的基本运算的有限次执行来实现。
(4) 输入性：一个算法有零个或多个的输入。
(5) 输出性：一个算法有一个或多个的输出。

算法的设计要求：

(1) 正确性：算法应能正确地实现预定的功能(即处理要求)。
(2) 易读性：算法应易于阅读和理解，以便于调试、修改和扩充。
(3) 健壮性：当环境发生变化(如遇到非法输入)时，算法能适当地做出反应或进行处理，不会产生不需要的运行结果。
(4) 效率与低存储量需求：即达到算法执行所需要的时间与空间性能。效率指的是算法执行时间，对于同一个问题如果有多个算法可以解决，执行时间短的算法效率高。存储量需求指算法执行过程中所需要的最大存储空间。

1.2.6　算法的计算量的含义及估算的方法

在计算机上运行依据算法编制的程序所耗费的时间叫做该算法在给定输入下的计算量。通常用下述办法来估算求解某类问题的各个算法在给定输入下的计算量：

(1) 根据该类问题的特点合理地选择一种或几种操作作为“标准操作”。

(2) 确定每个算法在给定输入下共执行了多少次“标准操作”，并将此次数规定为该算法在给定输入下的计算量。

算法的时间复杂度不仅与问题的规模有关，还与输入实例中的元素取值等相关，但在最坏的情况下，其时间复杂度就是只与求解问题的规模相关。讨论时间复杂度时，一般是以最坏情况下的时间复杂度为准。

1.2.7　数据的存储方式

存储结构是逻辑结构的存储实现，即数据按逻辑结构规定的关系在计算机存储器中的存放方式。

数据的存储方式有 4 种：顺序存储方式、链式存储方式、索引存储方式和散列存储方式。

(1) 顺序存储方式是把逻辑上相邻的结点存储在物理上相邻的存储单元里，结点之间的关系由存储单元的邻接关系来体现。其优点是占用最少的存储空间；其缺点是由于只能使用相邻的一整块存储单元，因此可能产生较多的碎片现象。

(2) 链式存储方式是将结点所占的存储单元分为两部分，一部分存放结点本身的信息，即为数据项；另一部分存放该结点的后继结点所对应的存储单元的地址，即为指针项。其优点是不会出现碎片现象，可充分利用所有的存储单元；其缺点是每个结点占用较多的存储空间。

(3) 索引存储方式是用结点的索引号来确定结点存储地址。其优点是检索速度快。其缺点是附加的索引表会占用较多的存储空间；另外，在增加和删除数据时由于要修改索引表会花费较多时间。

(4) 散列存储方式是根据结点的值确定它的存储地址。其优点是检索、增加和删除结点的操作速度都很快；其缺点是采用不好的散列函数时可能出现结点存储单元的冲突，为解决冲突需要附加的时间和空间开销。

1.2.8　算法的分类

算法可分为以下 3 类：

(1) 运行终止的程序可执行部分。运行终止的程序可执行部分是用程序设计语言描述的算法，这种算法可直接在计算机上运行，从而使给定问题在有限时间内被机械地求解。为了叙述方便，有时又将这种算法称为程序。

(2) 伪语言算法。采用某种“伪程序设计语言”描述的算法称为伪语言算法。伪语言算法不可直接在计算机上运行(可在某种“抽象机”上运行)，但容易编写和阅读，

适合于教学。

(3) 非形式算法。用自然语言(如汉语)，同时可能还使用了程序设计语言或伪程序设计语言描述的算法称为非形式算法。

1.2.9 数据结构的评价和选择

对数据结构的评价和选择通常从以下 3 个方面进行：

(1) 一般地，对任何给定的实际问题，可以建立不同的数据结构。这一方面是由于可以选择不同的数据组织形式(即逻辑结构)；另一方面是由于可以设计不同的非形式算法及相应地选择不同的基本运算集合。为了评价这些不同的数据结构，可从以下两个方面入手：

- 是否准确、完整地描述了问题的基本特性？
- 是否易于实现？

  具体地说，可以考虑以下问题：

  - 对数据的分解是否恰当？分解的恰当与否不仅要看它是否满足我们提出的标准，而且要看它是否有利于处理功能的规定(即运算的定义)，是否有利于存储实现，是否有利于提高运算实现的时空性能。
  - 逻辑结构的选择是否恰当？这里，恰当性的主要标准是有利于减小系统开发代价，有助于获得好的系统结构和高的系统性能。
  - 基本运算的选择是否恰当？这里，恰当性的主要标准是有利于减小系统开发代价，有助于获得好的系统结构和高的系统性能。因此，不一定要求所有基本运算都是功能独立的(即不能由其他基本运算实现的)。

(2) 对于给定的数据结构，设计人员可以选择不同的存储实现，即采用不同的存储结构。通过采用 4 种不同的存储方式，原则上可以建立同一个数据结构的 4 种不同的存储结构。为了评价这些存储结构以便从中选出一个较好的，可从以下几个方面去考虑：

- 是否有利于基本运算的实现？必须注意，运算的实现要求以存储结构的确定为前提，即运算的实现只能是在给定的存储结构上的实现。因此，存储结构将在一定程度上、一定范围内决定运算实现是否方便，得到的算法是否高效。
- 是否适合于数据的其他特性？数据的组织形式是其逻辑特性，而其他特性应在选择存储结构时加以考虑。例如，数据的规模是选择存储结构时要考虑的重要因素。
- 是否适合于预期的软、硬件环境。比如，数据是放在内存中还是外存中对存储结构有不同的要求，故应根据情况适当选择。

(3) 在给定数据结构、给定存储结构的条件下，仍可设计出实现同一运算的不同算法。对同一运算的不同算法的评价可以从以下两个角度进行：一是开发(设计)产品的角度，着眼于产品(算法)的质量；二是这种算法运行速度的快慢。

1.3　典 型 例 题

我们进一步了解了数据的概念和理论知识，那我们应怎样灵活运用这些知识呢？下面就通过一些典型例题向读者介绍解题的方法和步骤。

例题 1

设有如图 1.1 所示的逻辑结构图，试给出其数据结构表示：

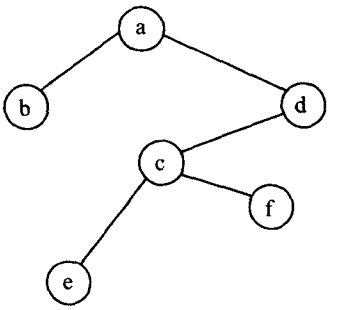


图 1.1

【解】

数据结构是数据元素相互之间存在的一种或多种特定关系的集合。

其数据结构定义为(K，R)，其中 K 是数据元素的有限集，R 是 K 上关系的有限集。

所以依所给图形可知：

K = {a, b, d, c, e, f}

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

例题 2

求两个 n 阶矩形的乘法 C=A×B，其算法如下：

```
 # define MAX 100
void maxtrixmult( int n, float  a [MAX][MAX],b[MAX][MAX],float c[MAX][MAX])
{
  int i,j,k;
  float x;
  for(i=1;i<=n;i++)                    ①
  {
   for(j=1;j<=n;j++)                   ②
   {
     x=0;                              ③
     for(k=1;k<=n;k++)                 ④
        x+=a[i][k]*b[k][j];            ⑤
     c[i][j]=x;                        ⑥
   }
  }
}
```

分析该算法的时间复杂度。

【解】

在通常情况下，算法的时间复杂度指的是该语句重复执行的次数即频度。

所以该算法中主要语句的频度分别是：

①n+1 ②n(n+1) ③n^2 ④$n^2(n+1)$ ⑤n^3 ⑥n^2

则该算法的时间复杂度为所有语句的频度之和 $T(n)=2n^3+3n^2+2n+1=O(n^3)$。

例题 3

给出以下算法的时间复杂度。

```
Void sort(int j, int n , int a[ ])
{
  if (j==n)                  //子数组只有一个元素        ①
    printf("%d",a[n]);
  else
  {
    for(i=j+1;i<=n;i++)                                  ②
      if(a[i]<a[j])
      {
        temp=a[i];
        a[i]=a[j];
        a[j]=temp;
      }
    printf("%d",a[j]);                                   ③
    sort(j+1,n,a);                                       ④
  }
}
```

【解】

从上述算法中看出：当 j=n 时，子数组只有一个元素，此时只需打印一个数 a[n]，时间为常量，用 O(1)表示，不然则执行 else 子句，其中语句②需要执行 n-j 次，语句③是打印一个数，时间为常量，用 O(1)表示，语句④是递归调用语句，用 T(j,n)表示 sort(j, n)的工作时间，语句④的工作时间就是 T(j+l, n)，由于 a[j+1]～a[n]的排序时间与 a[j]～(n-1)的排序时间相等，所以

T(j+l, n)=T(j,n-1)

则有

T(j, n)=T(j,n-1)+n-j+1

其中，1 为语句③的工作时间，n-j 为语句②的工作时间。若用 T(n)表示 sort(l,n)的工作时间，则得到如下递归表达式：

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

则

T(n)=T(n-1)+n

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

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

$=\cdots$

$=T(1)+2+3+\cdots+(n-2)+(n-1)+n$

$=1+2+3+\cdots+n$

$=\frac{n(n+1)}{2}$

$=O(n^2)$

例题 4

下述函数中渐近时间最小的是哪些？

(1) $T_1(n)=n\log_2 n-1000\log_2 n$

(2) $T_2(n)=n\log_2^3-1000\log_2 n$

(3) $T_3(n)=n^2-1000\log_2 n$

(4) $T_4(n)=2n\log_2 n-1000\log_2 n$

【解】

$T_1(n)=o(n\log_2 n)$，$T_2(n)=o(n\log_2{}^3)$，$T_3(n)=o(n^2)$，$T_4(n)=o(n\log_2 n)$

从中可看到，$T_1(n)$、$T_4(n)$最小。但 $T_1(n)=(n-1000)\log_2 n$,$T_4(n)=(2n-1000)\log_2 n$，显然，n 足够大时，$T_1(n)<T_4(n)$。所以渐近时间最小的是 $T_1(n)$。

例题 5

设有数据逻辑结构为：

B=(K,R)

K={k1,k2,…,k9}

R={<k1,k3>,<k1,k8>,<k2,k3>,<k2,k4>,<k2,k5>,<k3,k9>,<k5,k6>,<k8,k9>,<k9,k7>,<k4,k7>,<k4,k6>}

画出这个逻辑结构的图示，并确定相对关系 R，哪些结点是开始结点？哪些结点是终端结点？

【解】

该题的逻辑结构如图 1.2 所示。

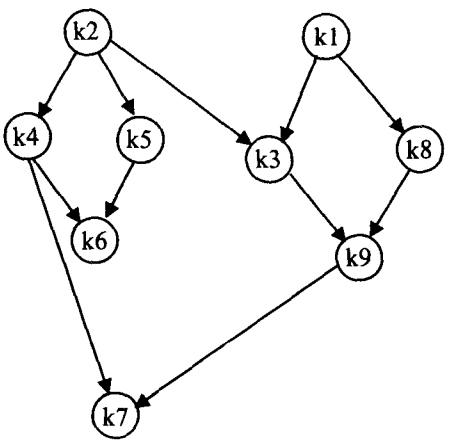


图 1.2