



高等学校经典教材配套辅导丛书

数据结构

辅导及习题精解

C 语言版

吴志坚 陶东辉 周则明 蒋琳 编著

- ★ 学习要点归纳
- ★ 基础知识详述
- ★ 教材习题精解
- ★ 全真考题剖析

新版



陕西师范大学出版社
SHAANXI NORMAL UNIVERSITY PRESS



高等学校经典教材配套辅导丛书

数据结构

辅导及习题精解

C 语言版

吴志坚 陶东辉 编著
周则明 蒋琳



陕西师范大学出版社
SHAANXI NORMAL UNIVERSITY PRESS

图书代号:JF6N0829

图书在版编目(CIP)数据

数据结构辅导及习题精解/陶东辉主编. —西安:陕西师范大学出版社, 2006. 8
(高等学校经典教材配套辅导丛书)

ISBN 7-5613-3538-5/T·17

I. 数… II. 陶… 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(2006)第 073091 号

责任编辑 陈光明 彭 青

装帧设计 王静婧

出版发行 陕西师范大学出版社

社 址 西安市陕西师大 120# (邮政编码:710062)

网 址 <http://www.snuph.com>

经 销 新华书店

印 刷 南京金阳彩色印刷有限公司

开 本 787×960 1/16

印 张 18.25

字 数 315 千

版 次 2006 年 9 月第 1 版

印 次 2006 年 9 月第 1 次印刷

定 价 21.90 元

开户行:光大银行西安电子城支行 账号:0303080—00304001602

读者购书、书店添货或发现印装问题,请与本社营销中心联系、调换。

电 话:(029)85307864 85233753 85251046(传真)

E-mail:if-centre@snuph.com

前言

“数据结构”是计算机程序设计的重要理论技术基础,它不仅是计算机学科的核心课程,而且已经成为其他理工科专业的热门选修课程,同时它也许许多信息类相关专业的研究生入学考试课程。

在多年的《数据结构》教学过程中,我们深切体会到,学生从理解数据逻辑结构的概念和方法,到具体在存储结构上的实现以及对数据结构的应用还有相当大的距离,对算法,初学者普遍有“能理解,但不会写”的感觉,为了帮助读者掌握数据结构的基本理论和基本方法,掌握综合运用分析各种数据结构以及算法的技巧和方法、提高分析问题和解决问题的能力,我们根据严蔚敏教授编写的《数据结构题集》(C语言版)编写了本辅导教材。

本辅导教材由以下几部分组成:

1. 学习要点及难点:列出各章的基本概念、重要性质和重要存储结构,突出必须掌握和理解的核心内容。

2. 课后习题解:原题集中习题数量大、层次多,特别是一些算法设计题从多个角度帮助理解基本概念和基本理论,适合训练读者的基本应用能力,一些层次较高的问题更有助于广大读者进一步的提高和应用。如果能够独立设计、完成这些算法设计题目,对学生的程序设计能力无疑是极好的训练,但是对于很多初学者,在解题和设计算法时感觉有很大困难。针对这种情况,我们对习题给出了详细的解答,以帮助读者理解提高,但是我们强烈建议读者还应参考解答以掌握算法思路为主。

3. 考研试题精讲:精选历年各院校研究生入学考试试题中具有代表性的试题进行了详细的解答,这些例题涉技巧性强,可以使广大读者举一反三,触类旁通,开拓解题思路,更好地掌握数据结构的基本内容和方法。

全书共分为11章,采用类C语言作为数据结构和算法的描述语言,绝大多数算法都调试验证过,解题思路清晰,算法描述规范、严谨。当然,算法设计题的解法不是唯一的,我们给出的也只是其中的一种解法,仅供参考。

另外,考虑到篇幅的原因,忽略了个别非常简单的、类同的或者难度太大的、纯数学的证明题目,对个别解题过程太过繁琐、冗长的内容也进行了删减。

本书由吴志坚、陶东辉、周则明和蒋林等编写。谭明超同志参与了部分程序调试。在本书的策划、编写、审稿等方面得到了滕家俊同志以及陕西师范大学出版社的大力支持和热情帮助,在此表示感谢。由于作者的水平有限,加之时间仓促,书中不足之处敬请广大同行和读者批评指正。

编者

2006年7月

目 录

第1章 绪 论	(1)
一、本章学习要点	(1)
二、基础知识题	(2)
三、算法设计题	(7)
四、考研典型试题	(11)
第2章 线 性 表	(13)
一、本章学习要点	(13)
二、基础知识题	(13)
三、算法设计题	(16)
四、考研典型试题	(40)
第3章 栈和队列	(43)
一、本章学习要点	(43)
二、基础知识题	(44)
三、算法设计题	(46)
四、考研典型试题	(66)
第4章 串	(69)
一、本章学习要点	(69)
二、基础知识题	(70)
三、算法设计题	(73)
四、考研典型试题	(87)
第5章 数组和广义表	(90)
一、本章学习要点	(90)

二、基础知识题	(91)
三、算法设计题	(96)
四、考研典型试题	(108)
第6章 树和二叉树	(111)
一、本章学习要点	(111)
二、基础知识题	(113)
三、算法设计题	(122)
四、考研典型试题	(154)
第7章 图	(158)
一、本章学习要点	(158)
二、基础知识题	(159)
三、算法设计题	(165)
四、考研典型试题	(196)
第8章 动态存储管理	(199)
一、本章学习要点	(199)
二、基础知识题	(199)
三、算法设计题	(203)
四、考研典型试题	(207)
第9章 查找	(209)
一、本章学习要点	(209)
二、基础知识题	(210)
三、算法设计题	(216)
四、考研典型试题	(232)
第10章 排序	(235)
一、本章学习要点	(235)
二、基础知识题	(235)

三、算法设计题.....	(245)
四、考研典型试题.....	(264)
第 11 章 外部排序	(267)
一、本章学习要点.....	(267)
二、基础知识题.....	(267)
三、考研典型试题.....	(271)
数据结构模拟考题一	(273)
数据结构模拟考题二	(281)

第1章 绪论

一、本章学习要点

1. 概念和术语

- 数据:就是指能够被计算机识别、存储和加工处理的符号的总称。
- 数据元素:是数据的基本单位,在计算机中通常作为一个整体进行考虑和处理。
- 数据项:是数据不可分割的最小单位。
- 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。其形式定义为:数据结构

是一个二元组

$$\text{Data_Structure} = (D, S)$$

其中:D是数据的有限集,S是D上关系的有限集。

- 集合:结构中的数据元素之间除了“同属于一个集合的关系外”,别无其他关系。
 - 线性结构:结构中的元素之间存在一对一的关系。
 - 树形结构:结构中的元素之间存在一对多的关系。
 - 图状结构或网状结构:结构中的元素之间存在多对多的关系。
 - 逻辑结构:从逻辑结构上描述数据,独立于计算机。
 - 存储结构:是逻辑结构用计算机语言的实现。
 - 数据类型:是一个值的集合以及在这些值上定义的一组操作的总称。
 - 抽象数据类型 ADT:是抽象数据的组织和与之的操作。相当于在概念层上描述问题。优点是
- 将数据和操作封装在一起实现了信息隐藏。

• 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,具备特性:有穷性、确定性、可行性、输入、输出5个特性。

- 评价算法的好坏的因素:正确性、可读性、健壮性、效率和低存储量
- 时间复杂度:是某个算法的时间耗费,它是该算法所求解问题规模 n 的函数。
- 渐近时间复杂度:是指当问题规模趋向无穷大时,该算法时间复杂度的数量级。
- 时间复杂度按数量级递增排列依次为:常数阶 $O(1)$ 、对数阶 $O(\log_2 n)$ 、线性阶 $O(n)$ 、线性对数阶 $O(n \log_2 n)$ 、平方阶 $O(n^2)$ 、立方阶 $O(n^3)$ 、…… k 次方阶 $O(n^k)$ 、指数阶 $O(2^n)$ 。

- 空间复杂度:是某个算法的空间耗费,它是该算法所求解问题规模 n 的函数。

2. 难点

- 理解抽象数据结构的定义、表示和实现方法。
- 估算算法的时间复杂度。

二、基础知识题

题 1.1 简述下列术语:数据.数据元素.数据对象.数据结构.存储结构.数据类型和抽象数据类型。

【解】 略,参见要点。

题 1.2 试描述数据结构和抽象数据类型的概念与程序设计语言中数据类型概念的区别。

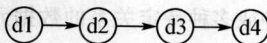
【解】 数据结构定义了一组按某些关系结合在一起的数组元素。数据类型不仅定义了一组带结构的数据元素,还在其上定义了一组操作。程序设计语言的数据类型是一个值的集合和定义在这个值的集合上的一组操作的总称。而抽象数据类型是定义了一个抽象的数学模型和该模型上的一组操作。

题 1.3 设有数据结构(D, R),其中

$$D = \{d1, d2, d3, d4\}, R = \{r\}, r = \{(d1, d2), (d2, d3), (d3, d4)\}.$$

试按图论中图的画法惯例画出其逻辑结构图。

【解】



题 1.4 试仿照三元组的抽象数据类型分别写出抽象数据类型复数和有理数的定义(有理数是其分子/分母均为自然数且分母不为零的分数)。

【解】 复数抽象数据类型定义

ADT Complex{

数据对象: $D = \{\text{real}, \text{image} \mid \text{real}, \text{image} \in \mathbb{R}\}$

数据关系: $R = \{ \langle \text{real}, \text{image} \rangle \}$

基本操作:

Initcomplex(&c, r, I)

操作结果:复数 c 被创建并以 r, I 分别赋给 real 和 image

Destroycomplex(&c)

操作结果:复数 c 被销毁

Getreal(c, &real)

初始条件:复数 c 存在

操作结果:用 real 返回复数 c 的实部

Getimage(c, &image)

初始条件:复数 c 存在

操作结果:用 image 返回复数 c 的虚部

Add(&c1, c2, c3)

初始条件:复数 c3、c2 存在

操作结果:复数 c1 为复数 c3 和 c2 的和

Sub(&c1, c2, c3)

初始条件:复数 c3、c2 存在

操作结果:复数 c1 为复数 c2 和 c3 的差

Mul(&c1, c2, c3)

初始条件:复数 c3、c2 存在

操作结果:复数 c1 为复数 c3 和 c2 的积

Div(&c1, c2, c3)

初始条件:复数 c3、c2 存在

操作结果:复数 c1 为复数 c2 和 c3 的商

}ADT Complex

有理数抽象数据类型定义:

ADT Rational {

数据对象: $D = \{\text{numerator, denominator} \mid \text{numerator, denominator} \in \mathbb{N} \text{ 且 } \text{denominator} < > 0\}$

数据关系: $R = \{<\text{numerator, denominator}>\}$

基本操作:

Init rational (&n, num, den)

初始条件:den 不为 0

操作结果:有理数 n 被创建并以 num, den 分别赋给 numerator 和 denominator

Destroy rational (&n)

操作结果:有理数 n 被销毁

Getnum(n, &num)

初始条件:有理数 n 存在

操作结果:用 num 返回有理数 n 的分子

Getden(n, &den)

初始条件:有理数 n 存在

操作结果:用 den 返回有理数 n 的分母

Add(&n1, n2, n3)

初始条件:有理数 n3、n2 存在

操作结果:有理数 n1 为有理数 n3 和 n2 的和

Sub(&n1, n2, n3)

初始条件:有理数 n3、n2 存在

操作结果:有理数 n1 为有理数 n2 和 n3 的差

Mul(&n1, n2, n3)

初始条件:有理数 n3、n2 存在

操作结果:有理数 n1 为有理数 n3 和 n2 的积

Div(&n1, n2, n3)

初始条件:有理数 n3、n2 存在,且 n3 不为 0

操作结果:有理数 n1 为有理数 n2 和 n3 的商

}ADT rational

题 1.5 试画出与下列程序等价的框图。

(1) product=1; i=1;

While(i<=n){

product *= i;

```

    i++;
}
(2) i=0;
do{
    i++;
}while(i!=n)&&(a[i]!=x);
(3) switch {

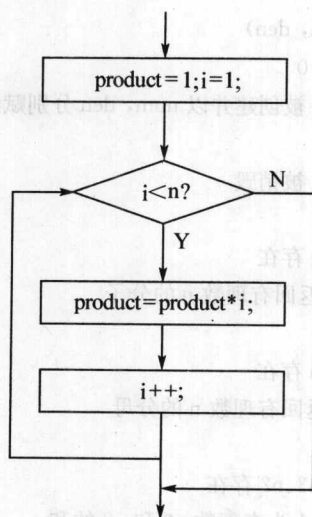
```

case $x < y$: $z = y - x$; break;

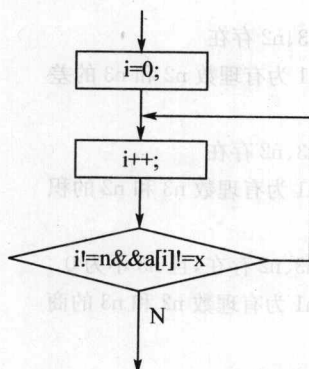
case $x == y$: $z = \text{abs}(x * y)$; break;

default: $z = (x - y) / \text{abs}(x) * \text{abs}(y)$; // $\text{abs}()$ 为取绝对值函数

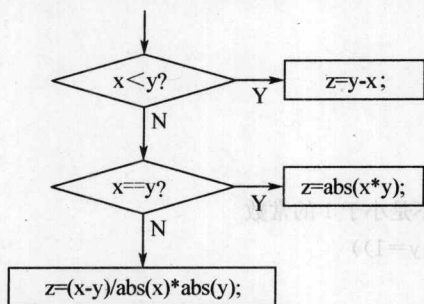
【解】 (1)



(2)



(3)



题 1.6 设 n 为正整数。试确定下列各程序中前置以记号@的语句的频度：

(1) $i=1; k=0;$

while ($i \leq n-1$)

{

@ $k += 10 * i;$

$i++;$

}

(2) $i=1; k=0;$

do{

@ $k += 10 * i;$

$i++;$

}while($i \leq n-1$);

(3) $i=1; k=0;$

while ($i \leq n-1$)

{

$i++;$

@ $k += 10 * i;$

}

(4) $k=0;$

for($i=1; i \leq n; i++$)

{

for($j=i; j \leq n; j++$)

@ $k++;$

}

(5) for($i=1; i \leq n; i++$)

{

for($j=1; j \leq i; j++$)

for($k=1; k \leq j; k++$)

@ $x += \text{delta};$

}

(6) $i=1; j=0;$

```

while(i+j<=n)
{
    @ if(i<j)j++;
      else i++;
}
(7) x=n; y=0;    //n 不是小于 1 的常数
while (x>=(y+1)*(y+1))
{
    @ y++
}
(8) x=91; y=100;
while(y>0)
{
    @ if (x>100){x -=10; y --;}
      else x++
}

```

【解】 (1) $n-1$

(2) $n-2$

(3) $n-1$

(4) $n(n+1)/2$

(5) $n(n+1)(n+2)/6$

(6) $n/2$

(7) $\lfloor \sqrt{n} \rfloor + 1$

(8) 对于每个 y 值 ($y>0$), 语句执行了 11 次。

题 1.7 假设 n 为 2 的乘幂, 并且 $n>2$, 试求下列算法的时间复杂度及变量 count 的值 (以 n 的函数形式表示)

```

int Time (int n)
{
    count=0; x=2;
    while (x<n/2)
    {
        x*=2; count++;
    }
    return (count);
}
//Time

```

【解】 时间复杂度为: $\log_2 n$, $\text{count} = \log_2 n - 1$ 。

题 1.8 按增长率由小至大的顺序排列下列各函数:

2^{100} , $(3/2)^n$, $(2/3)^n$, $(4/3)^n$, n^n , $n^{3/2}$, $n^{2/3}$, $\sqrt{n}$, $n!$, n , $\log_2 n$, $n/\log_2 n$, $(\log_2 n)^2$, $\log_2 (\log_2 n)$, $n \log_2 n$, $n^{\log_2 n}$ 。

【解】 排列顺序为: $(2/3)^n$, 2^{100} , $\log_2(\log_2 n)$, $\log_2 n$, $(\log_2 n)^2$, $\sqrt{n}$, $n^{2/3}$, $n/\log_2 n$, n , $n\log_2 n$, $n^{3/2}$, $(4/3)^n$, $(3/2)^n$, $n^{\log_2 n}$, $n!$, n^n .

题 1.9 已知有实现同一功能的两个算法,其时间复杂度分别为 $O(2^n)$ 和 $O(n^{10})$,假设实现计算机可连续运算的时间为 10^7 秒(100 多天),又每秒可执行基本操作(根据这些操作来估算算法时间复杂度) 10^5 次。试问在此条件下,这两个算法可能解决问题的规模(即 n 值的范围)各为多少? 哪个算法更适宜? 请说明理由。

【解】 可解问题规模分别为: $n \leq \log_2 10^{12}$ 和 $n^{10} \leq 10^{12}$, 可知第一个算法比较合适。

题 1.10 设有以下 3 个函数:

$$f(n) = 21n^4 + n^2 + 1000, \quad g(n) = 15n^4 + 500n^3, \quad h(n) = 5000n^{3.5} + n\log n$$

请判断以下断言正确与否:

(1) $f(n)$ 是 $O(g(n))$

(2) $h(n)$ 是 $O(f(n))$

(3) $g(n)$ 是 $O(h(n))$

(4) $h(n)$ 是 $O(n^{3.5})$

(5) $h(n)$ 是 $O(n\log n)$

【解】 (1)对 (2)错 (3)错 (4)对 (5)错

题 1.11 试设定若干值,比较两函数 n^2 和 $50n\log_2 n$ 的增长趋势,并确定 n 在什么范围内,函数 n^2 的值大于 $50n\log_2 n$ 的值。

【解】 大约 $n > 450$ 时, $n^2 > 50n\log_2 n$ 。

题 1.12 判断下列各对函数 $f(n)$ 和 $g(n)$, 当 $n \rightarrow \infty$ 时,哪个函数增长更快?

(1) $f(n) = 10^2 + \ln(n!) + 10^{n^3}$ $g(n) = 2n^4 + n + 7$

(2) $f(n) = (\ln(n!) + 5)^2$ $g(n) = 13n^{2.5}$

(3) $f(n) = n^{2.1} + \sqrt{n^4 + 1}$ $g(n) = (\ln(n!))^2 + n$

(4) $f(n) = 2^{(n^3)} + (2^n)^2$ $g(n) = n^{(n^2)} + n^5$

三、算法设计题

题 1.13 试写一算法,自大至小依次输出顺序读入的三个整数 X , Y 和 Z 的值。

【解】

```
void Descending()
```

```
{
```

```
    int x, y, z, tmp;
```

```
    scanf("%d, %d, %d", &x, &y, &z);
```

```
    if(x < y)
```

```
    {
```

```
        tmp = x;
```

```
        x = y;
```

```
        y = tmp;
```

```
    }
```

```
if(y<z)
```

```
{
```

```
    tmp=z;
```

```
    z=y;
```

```
    if(x>=tmp) y=tmp;
```

```
    else
```

```
    {
```

```
        y=x;
```

```
        x=tmp;
```

```
    }
```

```
}
```

```
printf("x=%d, y=%d, z=%d", x, y, z);
```

```
//Descending
```

题 1.14 已知 k 阶斐波那契序列的定义为

$$f_0=0, f_1=0, \dots, f_{k-2}=0, f_{k-1}=0;$$

$$f_n=f_{n-1}+f_{n-2}+\dots+f_{n-k}, n=k, k+1, \dots$$

【解】

```
int Fib(int k, int m){
```

```
    int * X;
```

```
    int j=k;
```

```
    int temp=0;
```

```
    if(k==0) return 0; //错误返回 0
```

```
    if(k==1) return 1;
```

```
    if(k>1){
```

```
        X=(int *) malloc(sizeof(int) * k); //动态数组
```

```
        for(int i=0; i<k; i++){
```

```
            X[i]=0;
```

```
        }
```

```
        X[k-1]=1; //初始化序列
```

```
        if(m<k) return X[m];
```

```
        else{
```

```
            while(j<=m){
```

```
                temp=X[0];
```

```
                for(int i=1; i<k; i++){
```

```
                    temp+=X[i]; //前 k 项和
```

```
                    X[i-1]=X[i];
```

```
                }
```

```
                X[k-1]=temp;
```

```
                j++;
```

```

    }
    return X[k-1];
}

}
} //Fib

```

题 1.15 假设有 A,B,C,D,E 五个高等学院校进行田径对抗赛,各院校的单项成绩均已存入计算机,并构成一张表,表中每一行的形式为

项目名称	性别	校名	成绩	得分
------	----	----	----	----

编写算法,处理上述表格,以统计各院校的男、女总分和团体总分,并输出。

【解】

```

typedef struct{
    char * sport;
    enum {male, female} gender;
    int schoolNum;
    char * result;
    int score;
}Result;

```

```

typedef struct{
    int schoolNum;
    int male;
    int female;
    int total;
}Score;

```

```

void Cal(Score s[], int i, Result rec)

```

```

{
    s[i].total += rec.score;
    if(rec.gender == 0) s[i].male += rec.score;
    else s[i].female += rec.score;
} //Cal

```

```

void Summary(Result record[])//原始成绩在 record[]数组中

```

```

{
    Score s[5];
    int i;
    for(i=0; i<5; i++)//初始化
    {
        s[i].schoolNum=i;
        s[i].female=0;

```

```

    s[i].male=0;
    s[i].total=0;
}
i=0;
while(record[i].sport!=NULL) //统计分数
{
    switch(record[i].schoolNum)
    {
        case 0:
            Cal(s, 0, record[i]);
            break;
        case 1:
            Cal(s, 1, record[i]);
            break;
        case 2:
            Cal(s, 2, record[i]);
            break;
        case 3:
            Cal(s, 3, record[i]);
            break;
        case 4:
            Cal(s, 4, record[i]);
            break;
    }
    i++;
}
for(i=0; i<5; i++) //输出分数
{
    printf("School %d:\n", s[i].schoolNum);
    printf("Total score of male:%d\n", s[i].male);
    printf("Total score of female:%d\n", s[i].female);
    printf("Total score of all:%d\n\n", s[i].total);
}
} //Summary

```

题 1.16 试编写算法, 计算 $i! \times 2^{2^i}$ ($i=0, 1, \dots, n-1$) 的值并分别存入数组 $a[\text{arrsize}]$ 的各个分量中。假设计算机中允许的整数最大值为 MAXINT , 则当 $n > \text{arrsize}$ 或对某个 k ($0 \leq k \leq n-1$) 使 $k! \times 2^k > \text{MAXINT}$ 时, 应按出错处理。注意选择你认为较好的输入和输出方法。

【解】

```
int Value(int a[], int n)
```