

软件工程师编程基础丛书

RUANJIAN GONGCHENGSHI BIANCHENG JICHU CONGSHU

数据结构 程序设计题典

李春葆 曾慧 张植民 编著

C 语言程序设计题典
数据结构程序设计题典
C++ 语言程序设计题典
Visual Basic 程序设计题典



清华大学出版社

软件工程师编程基础丛书

数据结构程序设计题典

李春葆 张植民 肖忠付 编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

数据结构是计算机科学及相关专业的重要核心课程。本书通过程序全面讨论了数据结构算法的实现过程。书中收集了大量富于启发性的例题和习题,并全部给出了答案,供读者融会贯通。

全书分为15章,第1章是数据结构的概述。第2章~15章以每章一个专题的形式,分别讨论了顺序表、单链表、双链表、栈、队列、串、广义表、数组、稀疏矩阵、树和二叉树、图、查找、排序和文件等基本类型的数据结构。各章均由本章摘要、例题解析、习题实践和参考答案4部分组成。书中所有的程序均在VC++ 6.0环境下调试通过。

本书可作为计算机科学及相关专业的本、专科学生学习数据结构课程的参考书,对于考研和计算机等级水平考试者而言,本书也具有一定的参考价值。

版权所有,盗版必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

书 名 : 数据结构程序设计题典

作 者 : 李春葆 张植民 肖忠付

出 版 者 : 清华大学出版社(北京清华大学学研大厦.邮编 100084)

<http://www.tup.tsinghua.edu.cn>

责任编辑 : 潘秀燕

印 刷 者 : 北京市朝阳区科普印刷厂印刷

发 行 者 : 新华书店总店北京发行所

开 本 : 787×1092 1/16 印张: 26.5 字数: 644 千字

版 次 : 2002 年 7 月第 1 版 2002 年 7 月第 1 次印刷

书 号 : ISBN 7-302-05688-9/TP·3350

印 数 : 0001~5000

定 价 : 34.00 元

《软件工程师编程基础丛书》序

计算机软件产业是IT行业最前沿和最有前途的产业之一，也最富有竞争性和挑战性。随着我国加入WTO，软件产业的发展越来越受到重视，目前，全国有33所大学设立了教授软件开发的专业学院。

“软件=程序+文档”，程序设计是软件开发的基础。欲成为一个优秀的软件开发人才，必须首先从程序设计开始培养和训练起步。试想一下，如果没有基本的程序设计思想和技术，怎么可能成为一名优秀的软件开发工程师呢？

本丛书是面向培养软件工程师的基础性程序设计教程，它以程序设计为核心，涵盖了一些重要的基础性程序设计课目。本套丛书包括：

1. 《C语言程序设计题典》
2. 《C++语言程序设计题典》
3. 《数据结构程序设计题典》
4. 《Visual Basic程序设计题典》

本丛书具有以下特点：

◆ 强化程序设计的实现过程

要想成为一名优秀的程序设计工程师，必须深入领会程序设计的奥秘，惟有这样才能做到触类旁通。所以本丛书不像一般的软件系统使用手册那样仅列出命令的使用语法，也不像一般的语言教科书那样只注重于抽象的理论方法，而是经过仔细的挑选，从程序实现的角度精心设计出一系列富有代表性的程序设计题目，并给予完整的解析。

◆ 内容叙述力求通俗易懂

本丛书不仅面向计算机专业的本、专科学生，更立足于希望通过基础程序设计训练，继而成为计算机程序设计工程师的有志青年。因此，在文字叙述和内容安排上尽量通俗易懂，力求讲出题目设计思想的来龙去脉，把程序设计的整个“过程”讲透。

◆ 结构布局完整、合理，便于自学

本丛书采用统一的结构，以章为单位，每章讨论一个专题，均由内容要点、例题解析、习题实践和参考答案构成。所有习题都附有参考答案。例题的求解过程详尽，习题参考答案简略，但可以从相应的例题中找到程序设计思路，因而适合通过自学提高程序设计能力。

本丛书的作者都是长期从事高校计算机基础教学工作的教授和副教授，他们集教学和程序设计经验于一体，精心编写了这套丛书，相信会为我国软件人才的培养起到促进作用。

前言

数据结构是计算机科学及相关专业的重要核心课程。对于从事计算机应用领域的开发人员，数据结构基础是必不可少的。本书从程序设计的角度全面讨论了数据结构算法的实现过程。

本书的特点

本书具有以下几方面的特点：

(1) 数据结构课程的目的是设计“好”的程序，但其内容和原理比较抽象，一般是建立在学生有一门程序设计语言（如C语言）基础后才开设该课程，但学生仍很难深入掌握其中奥妙。本书以程序（不仅仅是算法描述）加注释的方式实现了数据结构的算法，便于读者更好地体会算法的思想。

(2) 本书并不是简单地实现了数据结构教程中常用算法，而是收集了大量富于启发性的题目，并全部给予解答。

(3) 不同于数据结构教程，本书每一章一个专题，这样内容更加深入。除了少数章节外，各章的例题1中都给出了该章讨论的数据结构基本运算的实现，并存放在一个文件中，后面的例题或习题中大量使用这样运算，而不必重复编写。体现了一种结构化程序设计思想。

(4) 书中的程序（绝大多数程序给出了执行结果）在Visual C++ 6.0环境下调试通过。而且给出main()主函数，读者可直接上机验证。

对于C语言读者

本书程序在Visual C++ 6.0环境下编写的，读者只需要具备简单的C++基础就能阅读书中的全部程序。为了照顾只熟悉C语言的读者，本书尽量采用C语言的语法。程序中使用到的C++语法说明如下：

(1) 输入输出语句

C语言和C++语言输入输出语句对应如下（仅限本书中使用到的）：

	C语言	C++语言
输入	scanf("%d",&n);与n的类型有关	cin >> n;与n的类型无关
输出	printf("%d",n);与n的类型有关	cout << n; 与n的类型无关
域宽	printf("%3d",n);	cout << setw(3) << x;
头文件	stdio.h	iostream.h(使用域宽还要iomanip.h)
特定符	\n(换一行)	endl(换一行)

(2) 函数参数调用

C++语言有一种引用类型,其符号为“&”,例如,以下C函数:

```
void swap1(int x,int y)
{
    int temp;
    temp=x;x=y;y=temp;
}
```

在调用swap1(a,b)时,这种传值方式不会使实参a、b之值交换。为此可改为传地址方式,即:

```
void swap2(int *x,int *y)
{
    int *temp;
    temp=x;*x=*y;*y=temp;
}
```

这样,在调用swap2(a,b)时,实参a、b之值发生了交换。这两种方式在C++中完全兼容,但使用引用类型时综合了这两种方法的优点,相应的程序如下:

```
void swap3(int &x,int &y)
{
    int temp;
    temp=x;x=y;y=temp;
}
```

这里,在调用swap(a,b)时,实参a、b之值也发生了交换。

对于本书中使用引用类型的地方,读者改为C程序时,采用将swap3()改为等价的swap2()的方式进行即可。

如何阅读本书

阅读本书,要求读者对数据结构的基本内容有所了解。读者可先阅读各章中给出基本运算实现过程的例题1,掌握采用C/C++语言实现基本运算的方法,然后阅读后面的例题,再独立完成后面的相关习题,并可参阅习题答案。相信通过读者的摸索和总结,可以在较短时间内很快地掌握数据结构的内容和方法。

本书可作为计算机科学及相关专业的本、专科学生学习数据结构课程的参考书,也适合考研和计算机等级水平考试者。

由于水平所限,尽管编者不遗余力,仍可能存在错误和不足之处,敬请读者批评指正。

目 录

第1章 概论	1
1.1 本章摘要	2
1.1.1 知识点	2
1.1.2 内容概要	2
1.2 例题解析	4
1.3 习题实践	7
1.4 参考答案	10
第2章 顺序表	14
2.1 本章摘要	15
2.1.1 知识点	15
2.1.2 内容概要	15
2.2 例题解析	16
2.3 习题实践	25
2.4 参考答案	26
第3章 单链表	40
3.1 本章摘要	41
3.1.1 知识点	41
3.1.2 内容概要	41
3.2 例题解析	42
3.3 习题实践	69
3.4 参考答案	72
第4章 双链表	98
4.1 本章摘要	99
4.1.1 知识点	99
4.1.2 内容概要	99
4.2 例题解析	100
4.3 习题实践	117
4.4 参考答案	117
第5章 栈	126
5.1 本章摘要	127
5.1.1 知识点	127

5.1.2 内容概要	127
5.2 例题解析	128
5.3 习题实践	142
5.4 参考答案	143
第6章 队列	154
6.1 本章摘要	155
6.1.1 知识点	155
6.1.2 内容概要	155
6.2 例题解析	156
6.3 习题实践	168
6.4 参考答案	169
第7章 串	182
7.1 本章摘要	183
7.1.1 知识点	183
7.1.2 内容概要	183
7.2 例题解析	184
7.3 习题实践	190
7.4 参考答案	191
第8章 广义表	198
8.1 本章摘要	199
8.1.1 知识点	199
8.1.2 内容概要	199
8.2 例题解析	200
8.3 习题实践	211
8.4 参考答案	212
第9章 数组	223
9.1 本章摘要	224
9.1.1 知识点	224
9.1.2 内容概要	224
9.2 例题解析	227
9.3 习题实践	230
9.4 参考答案	231
第10章 稀疏矩阵	241
10.1 本章摘要	242
10.1.1 知识点	242
10.1.2 内容概要	242

10.2 例题解析	243
10.3 习题实践	247
10.4 参考答案	248
第11章 树和二叉树	257
11.1 本章摘要	258
11.1.1 知识点	258
11.1.2 内容概要	259
11.2 例题解析	262
11.3 习题实践	293
11.4 参考答案	295
第12章 图	318
12.1 本章摘要	319
12.1.1 知识点	319
12.1.2 内容概要	319
12.2 例题解析	322
12.3 习题实践	344
12.4 参考答案	345
第13章 查找	360
13.1 本章摘要	361
13.1.1 知识点	361
13.1.2 内容概要	361
13.2 例题解析	362
13.3 习题实践	370
13.4 参考答案	370
第14章 排序	381
14.1 本章摘要	382
14.1.1 知识点	382
14.1.2 内容概要	382
14.2 例题解析	383
14.3 习题实践	393
14.4 参考答案	394
第15章 文件	400
15.1 本章摘要	401
15.1.1 知识点	401
15.1.2 内容概要	401
15.2 例题解析	402

15.3 习题实践	404
15.4 参考答案	405
参考文献	413

第 1 章

概 论

- 1.1 本章摘要
- 1.2 例题解析
- 1.3 习题实践
- 1.4 参考答案

1.1 本章摘要

1.1.1 知识点

1. 数据结构的概念

数据的逻辑结构、数据的存储结构（物理结构）和数据的运算。

2. 数据的逻辑结构

- 线性结构
- 树形结构
- 图状结构

3. 数据的存储结构

- 顺序存储方式
- 链式存储方式
- 索引存储方式
- 散列存储方式

4. 数据的运算

数据的运算指对数据施加的操作。数据运算的实现是与数据的存储结构有关的。

5. 算法及算法分析

算法是解决问题的方法和步骤。

算法分析是对算法的时间性能和空间性能进行评价（通常指前者）。

1.1.2 内容概要

1. 基本概念

- (1) 数据是信息的载体，它能够被计算机识别、存储和加工处理。
- (2) 数据元素是数据的基本单位。有时一个数据元素包括若干个数据项。

(3) 数据结构指的是数据之间的相互关系, 它一般包括以下三个方面的内容:

- 数据的逻辑结构——数据元素之间的逻辑关系。
- 数据的存储结构——数据元素及其关系在计算机存储器内的表示。
- 数据的运算——对数据施加的操作。

2. 数据的逻辑结构

数据的逻辑结构是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。它有两大类:

(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节点和一个终端节点, 并且所有节点都最多只有一个直接前驱和一个直接后继。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: 一个节点可能有多个直接前驱和直接后继。例如, 树和图都是非线性结构。

3. 数据的存储结构

数据的存储结构是逻辑结构用计算机语言的实现, 即建立数据的机内表示。存储结构主要内容指在存储空间中使用一个存储节点来存储一个数据元素; 在存储空间中建立各存储节点之间的关联来表示数据元素之间的逻辑关系。其中存储节点是指一个数据元素在存储结构中的存储。

数据的存储结构有如下的4种基本方式:

(1) 顺序存储方式

每一个存储节点只含一个数据元素。所有的存储节点相继存储在一个连续的存储区里。用存储节点间的位置关系表示数据元素之间的逻辑关系。

(2) 链接存储方式

每一个存储节点不仅含一个数据元素, 还包括指针。每一个指针指向一个与本节点有逻辑关系的节点, 即用指针表示逻辑关系。

(3) 索引存储方式

每一个存储节点仅含一个数据元素, 所有的存储节点都连续存放。此外, 增设一个索引表。

(4) 散列存储方式

每一个存储节点仅含一个数据元素, 数据元素按散列(Hash)函数确定存储位置。

4. 数据的运算

数据的运算是通过算法描述的, 所以讨论算法是数据结构课程的重要内容之一。选用

算法首先考虑正确性,还要考虑执行算法所耗费的时间和存储空间,同时算法应易于理解、编码、调试等等。常用的运算如下:

- 查找运算——从结构中找出满足某种条件的节点的位置。
- 读取运算——读出结构中指定位置上的内容。
- 插入运算——在结构中的某指定位置上增加一个新节点。
- 删除运算——撤消结构中指定位置上的节点。
- 更新运算——修改结构中某指定节点的内容。

5. 算法分析

算法分析包括从时间和空间方面对算法进行特性分析,从时间方面的分析称为时间复杂度;从空间方面的分析称为空间复杂度。

时间复杂度从好到坏的级别依次是:

常量阶 $O(1)$ 、对数阶 $O(\log_2 n)$ 、线性阶 $O(n)$ 、平方阶 $O(n^2)$ 、指数阶 $O(2^n)$

1.2 例题解析



【例1.1】 以下是两个 n 阶方阵的乘积 $C=A \times B$,给出该算法的时间复杂度。

```
#define max 10
void matmult(int n, float A[max][max], float 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
        }
}
```

【解】 在算法中,语句①的循环控制变量 i 要增加到 $n+1$,测试 $i>n$ 成立才会终止,故

它的频度是 $n+1$ ，但是它的循环体却只能执行 n 次。语句②作为语句①的循环体内的语句应执行 n 次，但语句②本身要执行 $n+1$ 次，所以语句②的频度是 $n(n+1)$ 。同理得到语句③、④、⑤、⑥的频度分别为 n^2 、 $n^2(n+1)$ 、 n^3 、 n^2 。则本算法的时间复杂度 $T(n)$ 为：

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

实际上，我们只需考虑基本语句的时间复杂度即可，上述算法中，语句③⑤⑥是基本语句，它们的时间复杂度为 $O(n^3)$ 。



【例1.2】 设 n 为如下程序段处理的数据个数，求其时间复杂度。

```
for (i=1;i<=n;i=2*i)
    cout << "i=" << i << endl;
```

【解】 其中的基本语句是第2行的cout语句，设它的执行次数为 $f(n)$ ，则有 $2^{f(n)} \leq n$ ，即 $f(n) \leq \log_2 n$ 。

本程序段时间复杂度 $T(n)=f(n) \leq \log_2 n=O(\log_2 n)$ 。

故时间复杂度为 $O(\log_2 n)$ 。



【例1.3】 计算 $n!$ 的递归函数fac(n)如下，试分析它的时间复杂度。

```
int fac(int n)
{
    if (n<=1) return 1;          //①
    else return (n*f(n-1));      //②
}
```

【解】 设 $T(n)$ 为fac(n)的时间开销函数。显然 $T(1)=O(1)$ 。①的时间开销为 $O(1)$ ，递归调用fac(n-1)的时间开销为 $T(n-1)$ ，则②的时间开销为 $O(1)+T(n-1)$ 。则：

$$\begin{aligned} T(n) &= O(1) + T(n-1) = O(1) + O(1) + T(n-2) = 2 \times O(1) + T(n-2) \\ &= \dots = (n-1) \times O(1) + T(1) = n \times O(1) = O(n) \end{aligned}$$



【例1.4】 设计一个算法用不多于 $3n/2$ 的平均比较次数，在数组 $A[1..n]$ 中找出最大和最小值的元素。

【解】 本题的算法思想是：如果在查找出最大和最小值的元素时各遍历一遍所有元素，则至少要比对 $2n$ 次，为此，使用一遍遍历找出最大和最小值的元素。实现本题功能的函数如下：

```
void maxmin(datatype A, int n) //datatype为C语言标准数据类型
{
    int i;
    datatype max, min;
    max=A[1];min=A[1];
    for (i=2;i<=n;i++)
        if (A[i]>max) max=A[i];
        else if (A[i]<min) min=A[i];
    cout << "max=" << max << ", min=" << min << endl;
}
```

在这个函数中，最坏的情况是A中的元素按递减次序排列，这时 $(A[i]>max)$ 条件均不成立，比较的次数为 $n-1$ 。另外每次都要比较 $(A[i]<min)$ ，同样比较的次数为 $n-1$ ，因此，总的比较次数为： $2(n-1)$

最好的情况是A中的元素按递增次序排列，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：

$$\frac{2(n-1)+n-1}{2} = \frac{3n}{2} - \frac{3}{2} \leq \frac{3n}{2}$$

所以本算法的平均比较次数不多于 $3n/2$ 。



【例1.5】 设A是一个线性表 $(a_1, a_2, \dots, a_n)$ ，采用顺序存储结构，则在等概率的前提下，平均插入一个元素需要移动的元素个数是多少？若元素插入在 a_i 与 a_{i+1} 之间 $(0 \leq i \leq n-1)$ 的概率为 $\frac{n-i}{n(n+1)/2}$ ，则平均每插入一个元素所要移动的元素个数又是多少？

【解】 $A=(a_1, a_2, \dots, a_n)$ ，有 $n+1$ 个位置可插入元素，即 a_1 之前， a_1 与 a_2 之间， $\dots$ ， a_{n-1} 与 a_n 之间和 a_n 之后，分别需要移动的元素个数为 $n, n-1, \dots, 1, 0$ ，则平均插入一个元素需要移动的元素个数为：

$$(n + (n-1) + \dots + 1 + 0) \times \frac{1}{n+1} = \frac{n}{2}$$

若元素插入在 a_i 与 a_{i+1} 之间 $(0 \leq i \leq n-1)$ 的概率为 $\frac{n-i}{n(n+1)/2}$ ，其中移动的元素个数为：

$$(n-i) \times \frac{n-i}{n(n+1)/2} = \frac{(n-i)^2}{n(n+1)/2}$$

则平均每插入一个元素所要移动的元素个数为：

$$\sum_{i=1}^{n-1} \frac{(n-i)^2}{n(n+1)/2} = \frac{2n+1}{3}$$

1.3 习题实践

【题1】 分析以下各程序段的时间复杂度。

```
(1) void main()
    {
        int i=1, k=0, n=10;
        while (i<=n-1)
        {
            k+=10*i; i++;
        }
    }

(2) void main()
    {
        int i=1, k=0, n=100;
        do {
            k+=10*i; i++;
        } while (i==n);
    }

(3) void main()
    {
        int i=1, j=0, n=10;
        while (i+j<=n)
            if (i>j) j++;
            else i++;
    }

(4) void main()
    {
        int n=10, x=n, y=0;
        while (x>=(y+1)*(y+1))
            y++;
    }
```