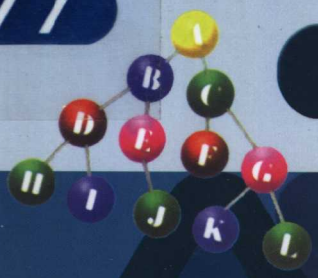

高等院校计算机系列教材

数据结构与算法

廖荣贵　许正宪　王龙发　蔡能聪　编著

数据结构与算法二者息息相关，本书通过浅显易懂的文字与各种运用方式来说明各个主题，并对问题的解决方法与流程进行详尽的图形剖析，同时辅以算法与程序代码的实例，从而增进读者对问题与结构的理解。

清华大学出版社

高等院校计算机系列教材

数据结构与算法

廖荣贵　许正宪　王龙发　蔡能聪　编著

清华大学出版社
北京

内 容 简 介

数据结构与算法息息相关，本书以浅显易懂的文字与各种运用方式来说明各个主题，并对问题的解决方法与流程做详尽的图形剖析，辅以算法与程序代码的实例，从而增进读者对问题与结构的理解。全书共分 13 章，各章的主题分别为数据结构概论、数组、算法、数组结构的算法应用、查找算法、排序算法、堆栈、队列、链表、递归、树、图、散列。

本书非常适合刚学习数据结构课程的学生研读，从书中的内容与顺序的编排来看本书也非常适合大专院校作为教材。

图书在版编目(CIP)数据

数据结构与算法/廖荣贵，许正宪，王龙发，蔡能聪编著.—北京：清华大学出版社，2004.11
(高等院校计算机系列教材)
ISBN 7-302-09731-3

Ⅰ.数…　Ⅱ.①廖…②许…③王…④蔡…　Ⅲ. ①数据结构　②算法分析　Ⅳ.TP311.12

中国版本图书馆 CIP 数据核字(2004)第 105321 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
责任编辑：桑任松　李朋朋
封面设计：陈刘源
印 刷 者：北京市通州大中印刷厂
装 订 者：三河市新茂装订有限公司
发 行 者：新华书店总店北京发行所
开　　本：185×260　印张：25　字数：589 千字
版　　次：2004 年 11 月第 1 版　2004 年 11 月第 1 次印刷
书　　号：ISBN 7-302-09731-3/TP・6728
印　　数：1~5000
定　　价：34.00 元

前　言

在信息科学的领域中，数据结构是一门基础学科，有关人工智能、图像处理、语音识别、并行处理等相关应用的研究，都需要这门学科的辅助；算法是解决问题的方法，要解决问题，只使用数据结构这个工具还稍嫌不足，还需要学习不同的算法，进而在未来的学习道路上，才能不断思考、不断地进步。数据结构与算法息息相关。本书以浅显易懂的文字与各种运用方式来说明各个主题，并对问题的解决方法与流程作详尽的图形剖析，辅以算法与程序代码的实例来，从而增进读者对问题与结构的理解。

全书共分 13 章，各章的主要内容分述如下：

第 1 章“数据结构概论”，介绍数据与结构、数据结构与算法等概念。

第 2 章“数组”，介绍数组的概念、数组类别和计量、数组的遍历和矩阵运算等。

第 3 章“算法”，介绍算法的概念、算法的效率分析及渐进式表示等。

第 4 章“数组结构的算法应用”，介绍数组在多项式的运算、捉大头抽签游戏、魔术方块算法中的应用，对奖算法与数据结构等。

第 5 章“查找算法”，介绍查找算法概述、线性查找法、二分查找法、插补查找法等。

第 6 章“排序算法”，介绍排序算法概述、冒泡排序法、交换排序法、选择排序法、插入排序法、谢尔排序法、基数排序法、快速排序法、归并排序法等。

第 7 章“堆栈”，介绍堆栈概述、堆栈的数据结构和操作、表达式的应用、后缀表达式求值和转换机器码。

第 8 章“队列”，介绍队列概述、队列的数据结构和操作、循环队列、双向队列和特殊队列。

第 9 章“链表”，介绍链表概述、以数组表示单链表和双向链表、以指针和结构表示链表、链表在其他结构中的应用等。

第 10 章“递归”，介绍递归关系以及递归算法在几种数学问题、汉诺塔问题、迷宫问题中的应用等。

第 11 章“树”，介绍树型结构和特性、二叉树的概念、二叉树的数据结构、二叉树的遍历、二叉运算树、堆、二叉查找树等。

第 12 章“图”，介绍图型结构的概念、图的数据结构、图的遍历、生成树和最小成本生成树、最短路径、拓扑排序等。

第 13 章“散列”，介绍散列概述、散列应用与散列函数、溢出处理、散列查找法等。

本书的章节安排由浅入深，目的是培养读者对数据结构与算法的高度兴趣，从最基本的数组结构的使用及其在各种问题上的应用开始谈起；再来介绍算法的分析方法及简单的练习；再进到最常用的搜索算法与排序算法，此为第一阶段，主要使读者与程序设计课程建立联系，读者只要会基本的程序基础，就可利用程序语言编写各种实用的程序。第二阶段介绍利用数组结构扩展到堆栈结构、队列结构、链表结构等，增强读者对线性问题的解决能力。第三阶段介绍递归关系、树型结构和图型结构等较深问题的处理方法，增强读者对空间类问题的解决能力。

本书具有如下特色：

- 每个章节的主题、结构与算法都有详细的图解说明；
- 每个算法都有实际对应的范例程序代码；
- 每个程序都有实际运行结果；
- 公式或运算式有推导过程，详细说明；
- 书中的程序代码以 C 语言为主，可以在各种版本的 C 语言开发环境中编译执行，如 Turbo C、Visual C++、Borland C++ Builder 等；
- 学习评估及范例、练习等除了作者自己编的题材外，另外参考了多种数据结构与算法的典型考题；
- 书中结构层次分明，适合学校教师采用为教材；
- 本书注重训练思考的方法，解决问题的步骤，适合训练程序设计人员的程序设计能力。

期望阅读本书的读者，除了能理解数据结构与算法领域的知识与技能之外，更能从本书的解题思路中得到灵感，可举一反三，将此学科的知识与技能应用在其他信息科技领域中。

目　录

第 1 章　数据结构概论

本章要点：

- 数据与结构
- 数据结构与算法

1.1　数据与结构

1.1.1　数据的演进

在“计算机概论”或“信息科学导论”课程中介绍过计算机是以二进制方式存储数据的，即不管我们人类使用什么类型的数据，要能让计算机工作，要能将数据存储在媒介中，或者要通过通信网络传送数据等，都需要使用二进制的 0 与 1 信号。

0 与 1 的世界是计算机系统的世界，并不是我们人类所熟悉的世界。而且用 0 与 1 表示数据又过于复杂，例如您身上有 100 元钱，用 0 与 1 表示是 01100100，试想如果是 1000 元、10000 元时，其表示方式会更复杂。

于是计算机学家自从发明计算机后，就一直在思索计算机与人类如何沟通的问题，如何将人类所处理的事务或所熟悉的方法通过接口的转换，搭起人类和计算机之间的桥梁。因此逐渐发明了文字编码系统(BCD 码、EBCDIC 码、ASCII 码、BIG5 码、Unicode 码)，另外在数值数据方面的发明则有正负号表示法、浮点数表示法、2 的补码表示法等。这些努力解决了人类用计算机来处理文字和数字的问题，使人类可以利用计算机超快的计算能力和大容量的内存空间处理问题，因而成就了 20 世纪后期人类的第 3 次工业革命——信息革命。

但计算机绝非万能的，计算机也会受本身能力的限制(有限的数据表示方式)而出错，例如 0.1 连加 100 次不会等于 10，你相信吗？为什么会这样呢？

这是因为计算机是以二进制方式存储数值的，而计算机是能力有限的机器设备，无法处理无限的小数循环，所以对于无法表示的数值，它只能迁就数据类型的字节数，只取到一定的位数，即近似值。当你设计程序时，如果要得到精确的数值而不是近似值时，要特别小心这样的问题。

我们先来换算一下十进制值 0.1 存储在计算机中的数值是什么，只要运用十进制转换二进制的方法即可得出，如图 1.1 所示。

由图 1.1 得知$(0.1)_{10}$无法用二进制表示出来，因为二进制会以 0011 四个数字为一组，一直循环下去。既然无法全部表现出来，且$(0.1)_{10}\neq(0.0001100110011)_2$(小数点表示到几位是依据浮点数的格式定义)，所以计算就产生了误差。这个道理与无法用十进制表示 1/3 一样，因为 1/3=0.3333…，也是一个循环小数。

我们已经知道计算机计算出错的原因在于使用浮点数来处理小数部分，而且很多小数数值都像$(0.1)_{10}$一样无法用有限的二进制数字表示，所以程序设计时可以避开小数点的处

理，而是“先以整数取代小数再进行运算”，然后再换算为小数。以“0.1 连加 100 次”的问题为例，先将 0.1×10=1，连加 100 次后得 100，再除以 10，可得答案 10。这样虽然多了一些步骤，但是绝对不会出错，对于要求精确计算的程序而言，例如金融系统的程序，这是必用的方法。

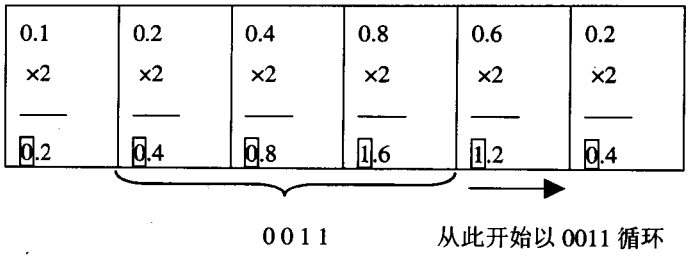

图 1.1 $(0.1)_{10}=(0.0\ 0011\ 0011\cdots)_2$

为了克服这个问题，现在有些程序语言或软件包甚至已经添加了具有这种运算方式的数据类型，例如 Visual Basic 或 Access 中的 Currency 类型(称为“货币类型”)。

以这一段历史来作为本书的开始，目的是要让你知道信息科学不是一天成就的，而是累积了很多经验和需求，由计算机学家或教授不断贡献其研究成果而发展起来的。同样，在程序设计和解决问题方法的领域中，数据结构与算法的演进是每个学习信息科学的学生必须学习的重要学科，本书将介绍此领域的应用和专业知识。

1.1.2 数据与结构

计算机处理的数据都是二进制的 0 与 1，人类所处理的数据则根据问题和工作而有所不同，各种程序语言与应用软件也会定义它能使用的数据类型，例如 C 语言的数据类型如表 1.1 所示。

表 1.1 C 语言的数据类型

类 型	字节数	表示方法	数值范围
整型	2	int	-32 768～32 767
	2	short	-32 768～32 767
	4	long	-2 147 483 648～2 147 483 647
	2	unsigned int	0～65 536
	2	unsigned short	0～65 536
	4	unsigned long	0～4 294 967 295
浮点型	4	float	约 10^{-38}～10^{38}
	8	double	约 10^{-308}～10^{308}
字符型	1	char	0～255

有了数据类型之后，相同类型的数据即可“组织”起来用于运算，这里所谓的组织即本书要介绍的结构(structure)。从建筑的角度来看，数据类似于建材，结构类似于将相同或可结合的建材组织起来变成一道墙、一片天花板或一层地基等。再把“建筑方法”或“建

筑图”加进来才能建造出一座建筑物，而建筑方法和建筑图即类似于程序设计的**算法**(Algorithm)和**流程图**(flow chart)。

所以信息科学的程序设计领域中的数据结构与算法相当于建筑领域中的建筑学或一般领域中的方法论。换句话说数据结构与算法就是程序设计的方法论。

1.2　数据结构及算法

1.2.1　数据结构

1.1 节中介绍了结构用于将数据组织起来，在此可以给**数据结构**(data structure)下一个简单的定义：

【定义】

数据结构探讨的是在计算机中如何有效地存放数据，使其可以方便地被处理。

由定义得知数据结构的重点是有效地存放数据。你或许会问我们用程序语言来设计程序时，都会声明变量数据或数组变量，这不就是数据的安排吗？难道这不是有效地存放数据吗？

但是，我们所作的变量或数组的声明只是数据的基本安排，对于某些较复杂的问题，或者还未见过但已经是数据结构与算法领域的历史遗留问题，则需要将数据进行特定的安排，才能便于被处理。举两个简单的范例来说明，思考一下数据的结构是如何安排的。

范例 1

图 1.2 是美国主要都市的分布图和道路交通的距离，试问数据该如何安排，才能表现出有道路相连的城市之间的距离。

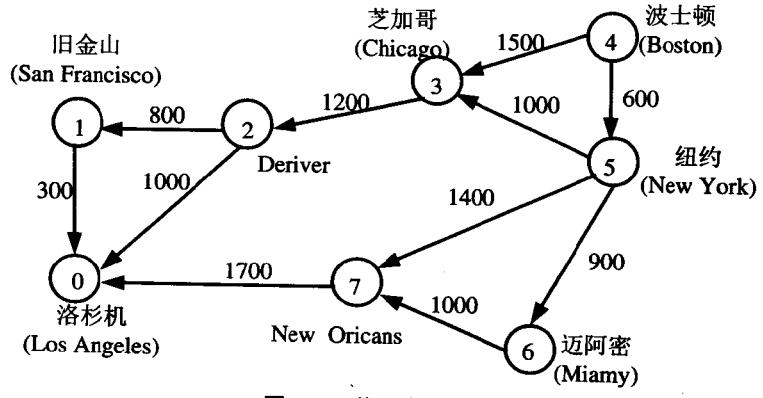

图 1.2　美国交通路网

解答：

用二维数组来表示，数组中有值的元素代表从某起点到某终点的距离。例如波士顿到芝加哥的距离是 1500，则在 P[4][3]中放入 1500。

起点＼终点	0	1	2	3	4	5	6	7
0								
1	300							
2	1000	800						
3			1200					
4				1500		600		
5				1000			900	1400
6								1000
7	1700							

P 数组

范例 2

假设有一组排序后的数据：A[0]＝10，A[1]＝20，A[2]＝30，A[3]＝40，A[4]＝50，如果现在想添加一项数据 35，并存放到 A[5]中，要如何不移动 A 数组的数据存放顺序——但是可额外添加变量或数组，可以由小到大取出数据 10，20，30，35，40，50 呢？

	0	1	2	3	4	5	6
原数据 A	10	20	30	40	50		

	0	1	2	3	4	5	6
添加 35 A	10	20	30	40	50	35	

解答：

请读者开动脑筋想一想！

范例 1 是典型的图形结构问题(第 12 章介绍)，而范例 2 则是典型的链表结构的问题(第 9 章介绍)。数据结构与算法就是介绍各类问题的方法论，是解决很多有趣问题的重要学科。

1.2.2 算法

简单地说，算法就是解决问题的方法，类似于上述建筑例子中的建筑方法和建筑图。算法所要解决的问题除了计算机本身管理的问题(例如内存管理、程序调用和返回、优先队列管理和工作日程安排等)之外，还包括数据处理问题(例如排序和查找)及各种要交由计算机来处理的问题(例如计算最短路径、老鼠走迷宫、项目管理)等。

程序语言是编写程序代码的工具，它提供了语句(程序行)的语法，让你能够利用编辑器写程序，但其重点还是在于解决问题的方法和流程。如果学习过程序设计，编写过程序，其实就已在使用简单问题(例如计算总和、平均值等)的算法了，只是还没有系统地学习各种典型问题的算法罢了。

N.Wirth 曾提出“程序＝数据结构＋算法”，这除了可以说明数据结构、算法和程序之间的关系外，更给出了数据结构和算法的最佳定位。

在此先向你提出两个问题，动动脑筋，看看你面对典型的标准问题是否能想到解决的方法——算法，测试一下你是否有潜力。这两个问题如范例 3 和范例 4 所示。

范例 3

[排序问题]已知有 6 项数据，分别已存储在数组 A 中，请按从小到大排序。

		0	1	2	3	4
原数据	A	10	40	20	50	30

		0	1	2	3	4
排序后	A	10	20	30	40	50

范例 4

[递归问题—河内塔问题]已知有 3 个木桩 A，B，C，目前在 A 木桩上有 3 个铁盘，由小到大叠放在一起，请将这 3 个铁盘搬到 C 木桩，如图 1.3 所示，但需遵守下列规则。

(1) 一次只能搬动一个铁盘。

(2) 小的铁盘只能放在大的铁盘上面。

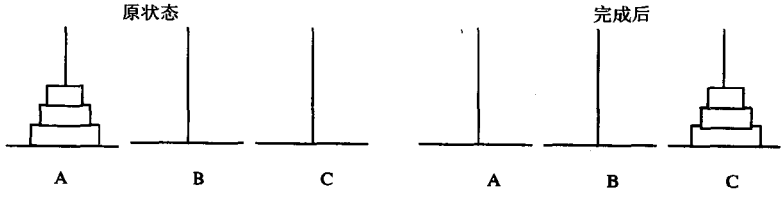

图 1.3　河内塔问题

学习算法除了要学习解决问题的方法之外，还要学习分析和比较算法的方法。算法好坏的衡量标准可分为测量执行次数的时间复杂度和测量所占空间的空间复杂度。第 6 章中专门介绍了历史上有名的各种排序算法，并且比较了它们的优劣。

【重点整理】

1. 自从计算机学家发明计算机后，他们就一直在思索计算机与人类如何沟通的问题，如何将人类所处理的事务或所熟悉的方法，通过接口的转换使之适合用计算机进行处理，从而搭起人类和计算机之间的桥梁。

2. 计算机处理的数据都是二进制的 0 与 1。人类所处理的数据则根据问题和工作而有所不同，各种程序语言与应用软件也会定义它能使用的数据类型。有了数据类型之后，相同类型的数据即可组织起来用于运算，这里所谓的组织就是本书要介绍的结构。数据类似于建材，结构类似于将相同或可结合的建材组织起来。如果再把建筑方法或建筑图加进来才能建造出一座建筑物，那么建筑方法和建筑图就类似于程序设计的算法和流程图。

3. 数据结构的定义：数据结构所探讨的是如何在计算机中有效地存放数据，使其便于被处理。算法简单地说，就是解决问题的方法。

4. N.Wirth 曾提出“程序＝数据结构＋算法”，这除了可以说明数据结构、算法和程序之间的关系外，更给出了数据结构和算法的最佳定位。

5. 学习算法除了要学习解决问题的方法之外，还要学习分析和比较算法的方法，算法好坏的衡量标准可分为测量执行次数的时间复杂度和测量所占空间的空间复杂度。

【学习自测】

选择题

1. 当用计算机存储我们的姓名时，应该使用什么样的数据结构？
   A. 字符(1 个字节)
   B. 整数(2 个字节)
   C. 浮点数(4 个字节)
   D. 字符数组(*n* 个字节)
2. 如果有一个整数 25 000，要将其存储到计算机中，应该使用什么样的数据结构？
   A. 字符(1 个字节)
   B. 整数(2 个字节)
   C. 浮点数(4 个字节)
   D. 字符数组(*n* 个字节)
3. 数据结构的种类通常不包含下列哪一个？
   A. 图形(Graph)　B. 队列(Queue)　C. 数组(Array)　D. 模块(Module)
4. 在 C 语言中声明 unsigned int A 时，A 的数值范围是什么？
   A. 0～255　B. 0～65 536　C. －127～128　D. －32 767～32 768
5. 在 C 语言中声明 float f 时，f 所占的字节数为多少？
   A. 1　B. 2　C. 4　D. 8
6. 在 C 语言中声明 double d 时，d 所占的字节数是多少？
   A. 1　B. 2　C. 4　D. 8

简答题

1. 什么是数据结构？什么是算法？两者有什么关系？
2. 请将以下数字以二进制形式表示
   (a) $(37)_{10}$
   (b) $(A3B9)_{16}$
   (c) $(723)_{8}$
   (d) $(0.001)_{10}$
3. 请解释为什么$(0.1)_{10}$无法用二进制完全表示，只能取得近似值？
4. N.Wirth 曾提出“程序＝_____＋_____”。

第2章 数 组

本章要点：

- 什么是数组
- 数组类型和计量
- 数组的遍历
- 矩阵运算

2.1 什么是数组

2.1.1 数组概论

在数学中有一个阶乘数 $n!$，代表 $1\times2\times3\times\cdots\times n$，以符号表示为：

$$n!=\prod_{i=1}^{n}=1\times2\times3\times\cdots\times n$$

例如 1!=1，2!=1×2=2，3!=1×2×3，以此类推。如果要输出 1!～10!的值，其 C 语言程序如下：

```
void  main( )
{  int  i,fact = 1 ;
   for  ( i = 1 ; i<= 10 ; i++)
   {   fact = fact * i ;
       printf  ( " %d ! = %d ", i , fact);
    }
}
```

假如希望用 10 个变量分别存放 1!，2!，3!，…，10!的值，以利于其他程序的使用，则其 C 程序语言(未使用数组，要由此引出数组的应用)：

```
void main ( )
{  int i,fact1=1,fact2=1,fact3=1,fact4=1,fact5=1;
   int fact6=1,fact7=1,fact8=1,fact9=1,fact10=1;
   for(i=1;i<=1;i++) fact1=fact1*i;
   for(i=1;i<=2;i++) fact2=fact2*i;
   for(i=1;i<=3;i++) fact3=fact3*i;
   for(i=1;i<=4;i++) fact4=fact4*i;
   for(i=1;i<=5;i++) fact5=fact5*i;
   for(i=1;i<=6;i++) fact6=fact6*i;
   for(i=1;i<=7;i++) fact7=fact7*i;
   for(i=1;i<=8;i++) fact8=fact8*i;
   for(i=1;i<=9;i++) fact9=fact9*i;
   // 其他程序可应用 fact1,fact2,...,fact9 等
}
```

由上述程序得知，如果程序语言只提供单一变量的声明和使用，则上述程序中相类似的程序行(如下)要重复写 10 行：

```
for ( i=1;i<=10;i++ )  fact1= fact1 * i;
```

如果要记录 1! ~ 100!的值，则需要 100 行类似的程序行，而且变量声明也要 100 个。这是很不实际的处理方法，因此计算机学家对这些相似的变量给出了一个不同于单一变量的形式，称为**数组**(array)，且提供一个下标(index)用于指定数组中元素的位置，这样就可以像单个变量一样使用各元素。上述程序运用数组声明方式存放数据的 C 语言程序如下：

```
void main ( )
{ int i,fact[] ;
    fact[1] = 1 ;
    for ( i = 2 ;  i < = 10 ; i ++)
      fact[i] = fact[i-1]*i;
}
```

上述程序中的 fact [i] = fact [i – 1] * i；经由 i = 2 ~ 10 的循环控制可得：

fact [2]=fact [1] *2=2=2!
fact [3]=fact [2] *3=6=3!
fact [4]=fact [3] *4=3! *4=4!
• • •
fact [10]=fact [9] *10=9! *10=10!

上述 3 个程序范例只是简单地概述数组的概念，实际上数组是最基本的数据结构形式，它既简单又便于使用。后面介绍的各种结构中，有些也会使用数组为基础构建出其他结构。

2.1.2 数组结构

数组的表示方式类似于数学的代数模型，只是数学代数模型是一种数字代换之间的关系，而数组还可以存放数据。以计算和存放 *n*!的值为例，其关系如图 2.1 所示。

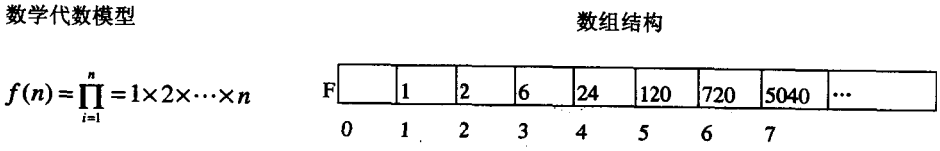

图 2.1 数学代数模型与数组结构

数组不只是数学的代数模型而已。也可以将表格数据视为数组结构。例如图 2.2 所示的表格，即可视为数组结构，也可利用下标方便地存取各个位置的数据。

学生考试成绩数据结构

100
90
80
60
⋮
100

产品价格表

APPLE	100
BOOK	30
CAT	200
DOG	50
⋮	
PEN	5

图 2.2 表格数据是数组结构