中国计算机学会教育专业委员会 推荐
全国高等学校计算机教育研究会 出版
高等学校规划教材

数据结构与算法

熊岳山 刘 越 编著

计算机学科教学计划

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

高等学校规划教材

数据结构与算法

熊岳山　刘　越　编著

電子工業出版社
Publishing House of Electronics Industry
北京・BEIJING

内 容 简 介

数据结构与算法是计算机专业的重要基础课，是该专业的核心课程之一，是一门集技术性、理论性和实践性于一体的课程。本书重点介绍抽象数据类型、基本数据结构、C 语言数据结构描述、数据结构的应用、算法设计与分析以及算法性能评价等内容，进一步使读者理解数据抽象与编程实现的关系，提高用计算机解决实际问题的能力。本书内容包括基本数据类型、抽象数据类型、顺序表、链表、串、树和二叉树、图、递归与分治算法、贪心算法、分支限界和动态规划等内容。

本书结构合理，内容丰富，算法描述清晰，用 C 语言编写的算法代码都已调试通过，便于自学，可作为高等院校计算机专业、军事院校的基础合训专业和其他相关专业的教材和参考书，也可供从事计算机软件开发的科技工作者参考。

图书在版编目（CIP）数据

数据结构与算法 / 熊岳山，刘越编著. —北京：电子工业出版社，2007.8
高等学校规划教材
ISBN 978-7-121-04777-0

Ⅰ. 数… Ⅱ. ①熊… ②刘… Ⅲ. ①数据结构-高等学校-教材 ②算法分析-高等学校-教材 Ⅳ. TP311.12

中国版本图书馆 CIP 数据核字（2007）第 114272 号

策划编辑：马 岚
责任编辑：马 岚　　特约编辑：马爱文
印　　刷：北京东光印刷厂
装　　订：三河市皇庄路通装订厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1092　1/16　印张：17.25　字数：357 千字
印　　次：2007 年 8 月第 1 次印刷
印　　数：5000 册　定价：24.80 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

数据结构与算法是计算机科学与技术一级学科相关专业的重要基础课程之一，是软件开发和维护的基础。计算机的数据处理能力是计算机解决各种实际问题的关键。现实世界中的实际问题经过抽象，得出反映实际事物本质的数据表示后，才有可能被计算机处理。从实际问题抽象出数学模型，得出它的数据表示后，如何用计算机所能接受的形式来描述这些数据（包括数据本身与数据之间的关系），如何将这些数据及其相互之间的关系存储在计算机中，如何用有效的方法处理这些数据，如何在构建的数据结构上设计高效的算法，是数据结构与算法研究的主要问题。

本书是在深入研究国内外同类教材的基础上，结合“数据结构”和“算法设计与分析”课程的多年教学经验编写而成的，书中重点围绕抽象数据类型的 C 语言实现进行了介绍。第 1 章为数据结构概论，主要介绍数据结构概念，其中包括逻辑结构、存储方法、算法复杂度分析、基本数据类型、抽象数据类型与结构描述。第 2 章介绍向量、栈和队列及其应用，其中包括向量、栈和队列的逻辑结构，抽象数据类型向量、栈和队列的描述，Josephus 问题求解、栈与后缀表达式求值、栈与递归、队列与离散事件模拟等应用实例。第 3 章介绍链表及其应用，其中包括动态存储、单链表、循环表、双链表、栈和队列的链接存储。第 4 章介绍串，其中包括串的定义、串的存储以及串的模式匹配算法（BMP 算法）。第 5 章介绍各种排序方法，其中包括排序的基本概念，被排序文件的存储表示，折半插入排序、Shell 排序、起泡排序、快速排序、归并排序和外排序等各种排序方法，各种排序方法的时空效率，算法的实现细节和算法的时空效率等。第 6 章介绍了线性表的查找，其中包括有关查找的概念，顺序查找、折半查找、分块查找和散列查找。第 7 章介绍树和二叉树，其中包括树（树林）和二叉树的概念、树（树林）和二叉树的遍历、抽象数据类型 BinaryTree 与 BinaryTree 结构、二叉树的遍历算法。第 8 章介绍树形结构的应用，其中包括二叉排序树、平衡的二叉排序树、B-树和 B^+-树、键树和 2-3 树、Huffman 最优树、堆排序和判定树。第 9 章介绍图结构，其中包括图的基本概念、图的存储表示、Graph 结构的构造与实现、图的遍历、最小代价生成树、单源最短路径问题、每一对顶点间的最短路径问题、有向无回路图。第 10 章为算法设计与分析，其中包括递归与分治、回溯法、分支限界法、贪心法和动态规划法等。

本书是作为计算机专业、军事院校的基础合训专业和其他相关专业的“数据结构与算法”课程教材编写的，也可供从事计算机软件开发和计算机应用的工程与科技人员参考。具备 C 语言基础的读者即可学习本书。

书中不带“*”的章节可用 60 学时讲授完成，讲授全部内容可在 70~80 学时内完

成。此外，为了配合课堂教学，便于学生理解和掌握所学知识，提高程序设计编程能力，应另外配有 20~30 小时的上机时间。

本书的出版得到了国防科技大学计算机学院、计算机系、603 教研室的大力支持，在此深表谢意。特别感谢陈怀义教授和姚丹霖副教授在《数据结构——C++描述》一书中的辛勤工作与许多富有创造性的思想，感谢殷建平、肖晓强、祝恩等老师在使用本书讲义后提出的宝贵意见，正是因为这些同志的热情帮助，才使得本书能顺利出版。由于时间仓促，加之作者水平有限，书中错误在所难免，敬请广大读者和专家批评指正。

作　者

2007 年 6 月

目　录

第 1 章　数据结构概述

计算机的数据处理能力是计算机解决各种实际问题的基础，但是现实世界中的实际问题必须经过抽象，得出反映实际事物本质的数据表示后，才有可能被计算机处理。如何从实际问题抽象出它的数学模型，得出它的数据表示，这不属于本课程的内容；但如何用计算机所能接受的形式来描述这些数据（包括数据本身及数据与数据之间的关系），如何将这些数据及其相互之间的关系存储在计算机中，如何用有效的方法处理这些数据，则是数据结构课程要研究的主要问题。

1.1　基本概念

1.1.1　数据、数据元素和数据对象

数据是客观事物的符号表示，是对现实世界的事物采用计算机能够识别、存储和处理的形式进行描述的符号的集合。计算机能处理多种形式的数据。例如，科学计算软件处理的是数值数据；文字处理软件处理的是字符数据；多媒体软件处理的是图像、声音等多媒体数据。

数据元素是数据的基本单位。在计算机程序中，数据元素通常是作为一个整体来处理的。一个数据元素又可以由若干个数据项组成。数据项包括两种，一种是初等项，为数据的不可分割的最小单位；一种是组合项，由若干个数据项组成。

例如，表 1-1 所示的学生情况表就是描述学生基本情况的数据。

表 1-1　学生情况表

学号	姓名	性别	年龄	籍贯	班别	成绩			
						数学	物理	化学	外语
6001	张三	女	19	北京	15	82	85	90	92
6002	李四	男	20	上海	15	90	92	91	95
6003	王五	男	18	湖南	15	93	95	91	94
…	…	…	…	…	…	…	…	…	…

每个学生的情况占一行，每行则是一个数据元素。每个数据元素由学号、姓名、性别、年龄、籍贯、班别、成绩等数据项组成。学号、姓名、性别、年龄、籍贯、班别为初等项。而成绩为组合项，又分为数学成绩、物理成绩、化学成绩、外语成绩等初等项，而且学号这个数据项是一个关键字（或关键码），因为它的值能唯一标识一个数据元素。

数据对象是性质相同的数据元素的集合，是数据集合的一个子集。数据元素则是数据对象集合中的数据成员。例如，学生情况表就是一个数据对象。整数集合和复数集合都是数据对象。

1.1.2 数据结构

在任何数据对象中，数据元素都不是孤立存在的，它们相互之间存在一种或多种特定的关系，这种关系称为结构。

数据的结构是指数据的组织形式，由数据对象及该对象中数据元素之间的关系组成。数据结构可以形式地描述为一个二元组：

Data Structure = (D, R)

其中，D 是数据对象，即数据元素的有限集；

R 是该数据对象中所有数据元素之间关系的有限集。

例 1-1 用符号$<k_i, k_j>$表示两数据元素 k_i 和 k_j 的先后次序关系，即 k_i 在 k_j 的前面，则下面两种二元组分别表示两个不同的逻辑结构。

结构 1 = (D, R)

$D = \{k_0, k_1, \cdots, k_{n-1}, k_n\}$, $R = \{r\}$

$R = \{<k_0, k_1>, <k_1, k_2>, \cdots, <k_{n-2}, k_{n-1}>, <k_{n-1}, k_n>\}$

结构 2 = (D, R)

$D = \{k_0, k_1, k_2, k_3, k_4\}$, $R = \{r\}$

$r = \{<k_0, k_1>, <k_0, k_2>, <k_1, k_3>, <k_1, k_4>\}$

以上是数学意义的数据结构概念，是数据结构的逻辑描述，即逻辑结构。尽管在一般情况下，我们所说的数据结构即指数据的逻辑结构，但当这种数据结构要放在计算机中进行处理时，数据结构概念的含义就不仅如此了。涉及计算机的数据结构概念，至今尚未有一个公认的标准定义，但一般认为应包括以下三个方面：

(1) 数据元素及数据元素之间的逻辑关系，又称为数据的逻辑结构；

(2) 数据元素及数据元素之间的关系在计算机中的存储表示，又称为数据的存储结构或物理结构；

(3) 数据的运算，即对数据施加的操作。

数据的逻辑结构是从逻辑关系上描述数据，是根据问题所要实现的功能而建立的。数据的逻辑结构是面向问题的，是独立于计算机的。数据的存储结构是指数据在计算机中的物理表示方式，是根据问题所要求的响应速度、处理时间、存储空间和处理速度等建立的。数据的存储结构是依赖于计算机的。每种逻辑结构都有一个运算的集合，例如最常见的运算有检索、插入和排序等，这些运算在数据的逻辑结构上定义，只规定“做什么”，在数据的存储结构上考虑运算的具体实现，规定“如何做”。

例如，表 1-1 就是一个数据结构，它由若干个数据元素组成，每个学生的情况数据

就是一个数据元素。对于任何一个数据元素，除第一个元素外，其他每个元素都有且仅有一个前驱，第一个元素没有前驱；除最后一个元素外，其他每个元素都有且仅有一个后继，最后一个元素没有后继。这就是数据的一种逻辑结构。

要利用这张表对学生的情况进行计算机管理，首先要把这张表存入计算机。可以把表中的这些数据元素顺序邻接地存储在一片连续的存储单元中，也可以用指针把各自存储的数据元素按表中顺序链接在一起，这种表在计算机中的存储方式就是数据的存储结构。学生情况的管理，必然涉及学生成绩的登记和查询、学生的插班和退学等，这些管理行为就是数据的运算，这里涉及数据元素的查找、插入和删除等操作。这些操作在数据的逻辑结构上定义，在数据的存储结构上实现。

1.2　数据结构的分类

对于一个数据结构而言，每个数据元素可称为一个结点，数据结构中所包含的数据元素之间的关系就是结点之间的关系。如果两个结点 k 和 k'之间存在的关系可用有序对 $<k, k'>$表示，则称 k'是 k 的后继，k 是 k'的前驱，k 和 k'互为相邻结点。如果 k 没有后继，则称 k 为终端结点。如果 k 没有前驱，则称 k 为开始结点。如果 k 既不是终端结点，也不是开始结点，则称 k 为中间结点。

数据的逻辑结构可分为两大类：线性结构和非线性结构。

线性结构中有且仅有一个开始结点和一个终端结点，并且所有结点最多只有一个前驱和一个后继。线性表是典型的线性结构。学生情况表就是一个线性表。

非线性结构中的一个结点可能有多个前驱和后继。如果一个结点最多只有一个前驱，而可以有多个后继，则这种结构就是树。树是最重要的非线性结构之一。如果对结点的前驱和后继的个数不进行限制，则这种结构就是图。图是最一般的非线性结构。数据的这几种逻辑结构可用图 1.1 表示。

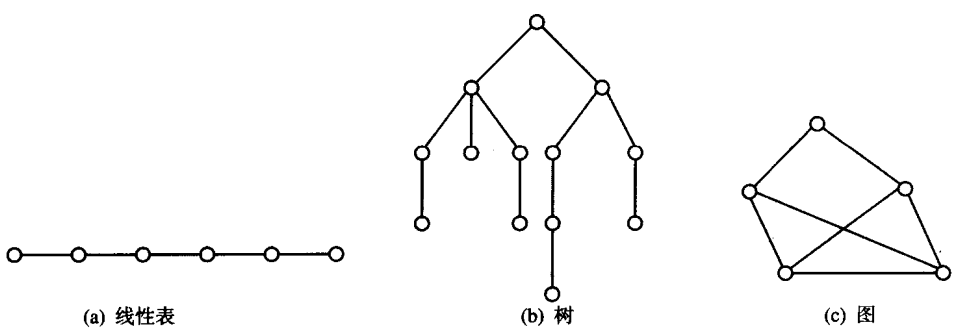

图 1.1　基本的逻辑结构

数据的存储结构取决于 4 种基本存储方法：顺序存储、链接存储、索引存储和散列存储。顺序存储方法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。结点之

间的逻辑关系用存储单元的邻接关系来体现。顺序存储主要用于线性结构，非线性结构也可以通过某种线性化方法实现顺序存储。通常顺序存储是用程序语言的数组来描述的。

对于逻辑上相邻的结点，链接存储方法不要求其在存储物理位置上也相邻，结点之间的逻辑关系由附加的指针来表示。非线性结构常用链接存储，线性结构也可以链接存储。通常链接存储是用程序语言的指针来描述的。

索引存储方法是在存储结点数据的同时，还建立附加的索引表。索引表的每一项称为索引项。一般情况下索引项由关键字（关键字是结点的一个或多个字段的组合，其值能唯一确定数据结构中的一个结点）和地址组成。一个索引项唯一对应于一个结点，其中的关键字是能唯一标识该结点的数据项，地址指示该结点的存储位置。

散列存储方法是根据结点的关键字计算出该结点的存储地址，然后按存储地址存放该关键字对应的数据元素。

这 4 种基本存储方法形成 4 种不同的存储结构，如图 1.2 所示。

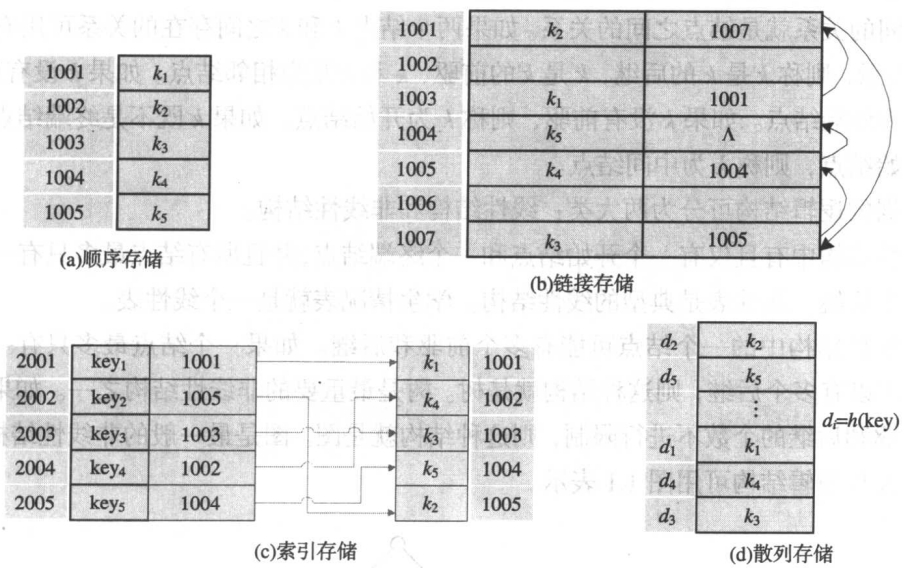

图 1.2 基本存储方法

采用不同的存储方法，同一种逻辑结构可以得到不同的存储结构。一种逻辑结构可采用一种方法存储，也可采用多种方法组合起来进行存储。

存储结构是数据结构概念不可缺少的一个方面，所以常常将同一逻辑结构的不同存储结构用不同的名称标识。例如，线性表的顺序存储称为顺序表，线性表的链接存储称为链表，线性表的散列存储称为散列表。

数据的运算也是数据结构概念不可缺少的一个方面。同一种逻辑结构采用同一种存储方式，如果定义的运算不同，也将以不同的名称标识。例如，若将线性表上的插入和删除操作限制在表的一端进行，则称为栈；若将插入限制在表的一端进行，将删除限制

在表的另一端进行，则称为队列。如果这种线性表顺序存储，则称为顺序栈或顺序队列；如果链接存储，则称为链式栈或链式队列。

在计算机环境下研究数据结构，应该将数据的逻辑结构、数据的存储结构和数据的运算看成是一个整体，只有了解清楚了这三个方面，才能真正了解数据结构。

1.3 数据类型

1.3.1 基本类型和组合类型

在早先的高级程序设计语言中都有数据类型的概念。数据类型是一组性质相同的值的集合以及定义在这个集合上的一组操作的总称。

N.Wirth 曾指出：算法+数据结构=程序。在早先的高级程序设计语言中，数据结构是通过数据类型来描述的。数据类型用于刻画操作对象的特性。每个变量、常量和表达式都属于一个确定的数据类型，例如整型、实型或字符型等，这些数据类型规定了数据可能取值的范围以及允许进行的操作。例如，C 语言程序中的变量 k 定义为整型 int，则它可能取值的范围是{0, +1, –1, +2, –2, …, maxint, minint}，其中 maxint 和 minint 是所用计算机上整型量所能表示的最大整数和最小整数。对它可施行的操作有算术一元运算（+和–）、算术二元运算（+，–，*，/和%）、关系运算（<=，>=，==和!=）以及赋值运算（=）等。

在高级程序设计语言中，数据类型分为两种，一种是基本类型，如整型、实型和字符型等，其取值范围和允许的操作都是由系统预先规定的；另一种是组合类型，它由一些基本类型组合构造而成，如记录、数组和结构等。基本数据类型通常是由程序设计语言直接提供的，而组合类型则由用户借助程序设计语言提供的描述机制自己定义。这些数据类型都可以看成是程序设计语言已实现了的数据结构。

1.3.2 抽象数据类型

抽象是指从特定实例中抽取共同的性质以形成一般化概念的过程。抽象是对某系统的简化描述，即强调该系统中的某些特性，而忽略一部分细节。对系统进行的抽象描述称为对它的规范说明，对抽象的解释称为它的实现。抽象可分为不同的层次，高层次抽象将其低层次抽象作为它的一种实现。抽象是人们在理解复杂现象和求解复杂问题时处理复杂性的主要工具。

数据抽象是一种对数据和操作数据的算法的抽象。数据抽象包含了模块化和信息隐蔽两种抽象。模块化是将一个复杂的系统分解成若干个模块，每个模块与系统某个特定功能有关的信息保持在该模块内。一个模块是对整个系统结构的某一部分的自包含和完整的描述。模块化的优点是便于修改和维护。当系统发现问题后，容易定位问题出在哪

个模块上。信息隐蔽是将一个模块的细节部分对用户隐藏起来，用户只能通过一个受保护的接口来访问某个模块，而不能直接访问一个模块的内部细节。这个接口一般由一些操作组成，这些操作定义了一个模块的行为。这样，错误的影响被限制在一个模块内，增强了系统的可靠性与可维护性。

数据类型已经体现了数据的抽象。例如，在计算机中使用二进制定点数和浮点数实现数据的存储和运算，如果使用汇编语言，程序员可直接使用它们的自然表示，如 15.5，1.35E10 或 10 等，不必考虑其实现细节，这是二进制数据的抽象。在高级语言中，出现了整型、实型和双精度实型等数据类型，给出了更高一级的数据抽象。面向对象程序语言出现后，可以用抽象数据类型定义更高层次的数据抽象，如各种表、树和图，甚至可以定义窗口和管理器等。这种数据抽象的层次为软件设计者提供了有利的手段，使设计者可以从抽象的概念出发，从整体上进行考虑，然后自顶向下逐步展开，最后得到所需的结果。

抽象数据类型（Abstract Data Type，简称 ADT）是指抽象数据的组织和与之相关的操作。抽象数据类型通常由用户定义，用以表示应用问题的数据模型，它可以看成是数据的逻辑结构以及在逻辑结构上定义的操作。ADT 可用如下形式描述：

```
ADT ADT_Name
{
    Data:                          /* 数据说明 */
        数据元素之间逻辑关系的描述
    Operations:                    /* 操作说明 */
        Operation1
        Input:                     对输入数据的说明
        Preconditions:             执行操作前系统应满足的状态
        Process:                   对数据执行的操作
        Output:                    对返回数据的说明
        Postconditions:            执行操作后系统的状态
        Operation2
        …
} /*ADT*/
```

抽象数据类型体现了数据抽象，它将数据和操作封装在一起，使用户程序只能通过在 ADT 里定义的某些操作来访问其中的数据，从而实现了信息隐蔽。ADT 既独立于它的具体实现，又与具体的应用无关，这样就可以使软件设计者把注意力集中在数据及其操作的理想模型的选择上。

下面给出一个矩形抽象数据类型的例子。

矩形由长和宽决定，不同的长和宽构成不同的矩形。与矩形相关的主要操作分别是计算面积和计算周长。因此，数据长和宽及求面积和周长的操作就构成了矩形的抽象数

据类型。

封装的矩形数据和操作的抽象数据类型如下：

```
ADT Rectangle
{
    Data:
        非负实数，给出矩形的长和宽
    Operations:
        InitRectangle
            Input:          矩形的长和宽
            Preconditions:  无
            Process:        建立矩形
            Output:         返回矩形
            Postconditions: 无
        Area
            Input:          无
            Preconditions:  无
            Process:        计算矩形的面积
            Output:         返回面积值
            Postconditions: 无
            Perimeter
            Input:          无
            Preconditions:  无
            Process:        计算矩形的周长
            Output:         返回周长值
            Postconditions: 无
}
```

基于抽象数据类型（矩形）的抽象结果，可得到用 C 语言描述的数据抽象和运算抽象，如下所示。

```
#include <stdio.h>
#include <stdlib.h>

struct rectangle
{
    float length;
    float width;
};
typedef struct rectangle Rectangle;

Rectangle *InitRectangle (float l, float w)          /* 初始化函数 */
{
```