21世纪高等院校计算机系列课程教材

数据结构教程

Data Structure Tutorial

王 唯 主 编

张 杰 沈 楠 贾红宇 刘 洋 杨光祖 陈玉和 副主编

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

21 世纪高等院校计算机系列课程教材

数据结构教程

主　编　王　唯
副主编　张　杰　沈　楠　贾红宇
　　刘　洋　杨光祖　陈玉和

北京理工大学出版社
BEIJING INSTITUTE OF TECHNOLOGY PRESS

内容简介

本书采用项目导向方式，通过应用示例，介绍了数据结构的基本知识。本书内容翔实，系统性强，深入浅出，通俗易懂。示例程序启发性强、有代表性，且全部通过了上机调试，可以直接运行。

本书内容包括：绪论、线性表、栈、队列、串和数组、树、图、查找和排序等。

可以作为高等院校的教材或参考书，也可以作为相关人员的参考书。

图书在版编目(CIP)数据

数据结构教程 / 王唯主编. —北京：北京理工大学出版社，2010.2

ISBN 978-7-5640-3015-5

Ⅰ. ①数… Ⅱ. ①王… Ⅲ. ①数据结构-高等学校-教材 Ⅳ. ①TP311.12

中国版本图书馆 CIP 数据核字(2010)第 015583 号

出版发行 / 北京理工大学出版社

社　　址 / 北京市海淀区中关村南大街 5 号

邮　　编 / 100081

电　　话 / (010)68914775(办公室)　68944990(批销中心)　68911084(读者服务部)

网　　址 / http://www.bitpress.com.cn

经　　销 / 全国各地新华书店

印　　刷 / 保定市中画美凯印刷有限公司

开　　本 / 787 毫米×1092 毫米　1/16

印　　张 / 13.5

字　　数 / 307 千字

版　　次 / 2010 年 2 月第 1 版　2010 年 2 月第 1 次印刷

印　　数 / 1～2000 册　　责任校对 / 陈玉梅

定　　价 / 28.00 元　　责任印制 / 边心超

前　言

数据结构是计算机及相关专业的很重要的专业基础课程。它不仅是计算机程序设计的理论基础，而且是学习计算机操作系统、编译原理、数据库原理等课程的重要基础。

数据结构的主要任务是讨论数据的各种逻辑结构和数据在计算机中的存储表示，以及各种非数值运算的算法的实现。通过数据结构课程的学习，使学生能使用数据结构的基本分析方法来提高编辑程序的能力和应用计算机解决实际问题的能力。

本书是高等院校计算机及相关专业的系列教材之一，编写的理念在于注重学生实际能力的培养，在教材编写体系上，按照“提出任务—训练目的—任务分析—任务实现—相关知识点”的思路，力求在实际可操作性上有所突破。所选内容本着循序渐进、综合提高的原则，既保持知识的系统性，又适当拓宽和加深了知识点，采用项目导向方式，通过应用示例，介绍了数据结构的知识，使学生加深对算法的理解和具体程序设计技巧的掌握。

从体系结构而言，本书是用一个“数据结构实验演示系统”为主线来组织教材的编写的。每一章的主要算法构成一个相对独立的子系统（即子模块），子系统既是各章教学的重点内容，也是上机实验的主要算法。各个子系统可以通过菜单的选择对本章的基本算法进行实验和演示，也可以用它来检验相关习题的正确性；而系统也是开放式的，对于学有余力的同学，可以将数据结构的其他算法扩充到整个实验演示系统中去。

从编写风格而言，本书力求做到简明扼要，条理清楚，并尽量避免抽象的理论论述和复杂的公式推广。本书内容共分9章，第1章绪论，介绍了数据结构与算法的基本概念，并对算法的时间复杂度和空间复杂度做了介绍；第2章到第5章，介绍了线性表、栈、队列，串等线性结构的逻辑特征、存储方法以及常用算法的实现及基本应用；第6章和第7章，介绍了树和图两种非线性数据结构的逻辑特征、存储方法以及相关算法的实现和基本应用；第8章主要介绍了顺序查找、二分查找、分块查找和二叉排序树的查找方法以及散列存储的基本方法；第9章介绍了在计算机中广泛使用的各种排序方法。各章内容相对独立，自成体系。

本书作者都是从事计算机教学和科研的教师，由王唯主编，负责总体构思，并确定章节框架和写作内容，张杰、沈楠、贾红宇、刘洋、杨光祖、陈玉和任副主编，姚策、张鸿宇、慕海涛参与编写。本书在编写过程中自始至终得到了北京理工大学出版社的大力支持，同时也参考了许多学者的研究成果，在此一并表示感谢。

由于时间仓促，作者水平有限，书中难免有不妥之处，恳请读者批评指正。

编者

目　录

第1章 绪 论

【知识点】

数据结构中常用的基本概念和术语；

算法描述和分析方法。

【难点】

算法时间复杂度。

【要求】

了解数据的逻辑结构和物理结构；

了解算法对于程序设计的重要性；

掌握算法时间复杂度的分析方法。

1.1 什么是数据结构

1.1.1 从数据结构实验演示认识数据结构

先来看一个简易的数据结构实验演示系统。

数据结构实验演示系统主菜单

```
* * * * * * * * * * * * * * * * * *
*        1——线 性 表               *
*        2——栈                     *
*        3——队    列               *
*        4——串                     *
*        5——二 叉 树               *
*        6——图                     *
*        7——查    找               *
*        8——排    序               *
*        0——退    出               *
* * * * * * * * * * * * * * * * * *
        请选择菜单号（0—8）：
```

按提示进行选择，例如选择2，即进入栈子系统：

```
                栈 子 系 统
* * * * * * * * * * * * * * * * * * * *
*        1——进    栈                 *
*        2——出    栈                 *
*        3——显    示                 *
```

```
*        4——数制转换              *
*        5——逆波兰式              *
*        0——返    回              *
* * * * * * * * * * * * * * * * * * *
```

请选择菜单号（0—5）；

再按提示选择4，进入二进制—十进制转换的演示……

学过C（或C++）等程序设计的人对结构化程序设计的一些特点应有一定的了解，但是对于数据，特别是数据的结构往往缺乏更深层次的认识。著名的计算机科学家N. Wirth提出“算法+数据结构=程序”的思想，明确地指出了数据结构实际上是程序的主要部分。

数据结构是一门介于数学、计算机硬件和计算机软件三者之间的一门核心课程。在计算机科学中，数据结构不仅是一般非数值计算程序设计的基础，而且是设计和实现汇编语言、编译程序、操作系统、数据库系统，以及其他系统程序和大型应用程序的重要基础. 打好“数据结构”这门课程的扎实基础，将会对程序设计有进一步的认识，使编程能力更上一个台阶，从而提高学习和开发应用软件的能力。

从我国计算机教学现状来看，数据结构不仅仅是计算机专业教学计划中的核心课程之一，而且已经逐步成为非计算机专业的重要选修课程。

1.1.2 数据结构研究的内容

用计算机解决具体问题需要经过的步骤：

（1）从具体问题抽象出适当的数学模型；

（2）设计解数学模型的算法；

（3）编制程序、运行并调试程序，直到解决实际问题。

1.2 数据的逻辑结构

1.2.1 基本概念

1. 数据

数据（Data）是信息的载体，是对客观事物的符号表示。通俗地说，凡是能被计算机识别、存取和加工处理的符号、字符、图形、图像、声音、视频信号等一切信息都可以称为数据。

2. 数据元素

数据元素（Data Element）是对现实世界中某独立个体的数据描述，是数据的基本单位。数据元素也称为结点（Node），在计算机中，常作为一个整体来处理。

3. 数据项

数据项（Data Item）是数据不可分割的、具有独立意义的最小数据单位，是对数据元素属性的描述。数据项也称为域，或字段（Field）。

4. 数据对象

数据对象（Data Object）是性质相同的数据元素的集合，是数据的一个子集。

例如在“学生入学情况表”中，数据对象就是全体学生记录的集合。

5. 数据结构

数据结构（Data Structure）是相互之间存在的一种或多种特定关系的数据元素的集合。

（1）集合——结构中数据元素之间，除了“同属于一个集合”关系之外，别无其他关系。

（2）线性结构——结构中的数据元素之间存在着“一对一”的关系。

（3）树形结构——结构中的数据元素之间存在着“一对多”的关系。

（4）图形结构——结构中的数据元素之间存在着“多对多”的关系。

四类基本数据结构的示意图如图1-1所示。

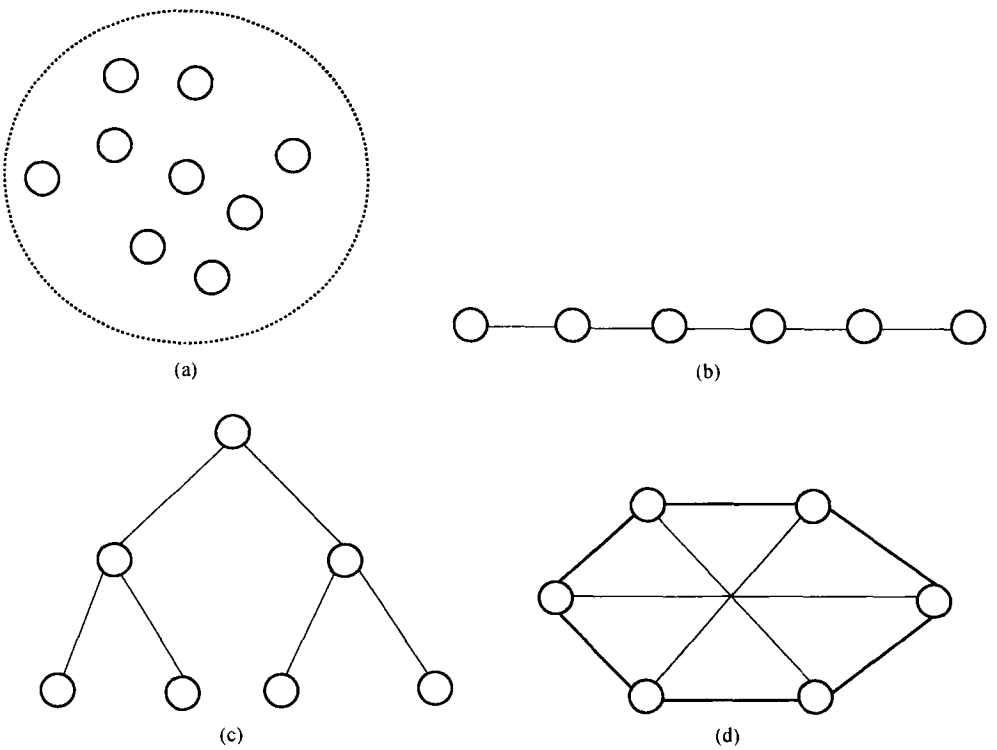

图1-1 四类基本数据结构的示意图

（a）集合结构；（b）线性结构；（c）树形结构；（d）图形结构

1.2.2 逻辑结构的描述

数据元素之间的逻辑关系，称为数据的逻辑结构。

一个数据的逻辑结构可以用二元组来表示：

$$G = (D, R)$$

式中：D为数据元素的集合；R为D上所有数据元素之间关系的有限集合。

【例 1 -1】一种数据结构 Line = （D，R），其中：

D = {01，02，03，04，05，06，07，08，09，10}

R = {r}

r = {<05，01>，<01，03>，<03，08>，<08，02>，<02，07>，<07，04>，<04，06>，<06，09>，<09，10>}

这种数据结构为线性结构，如图 1 -2 所示。

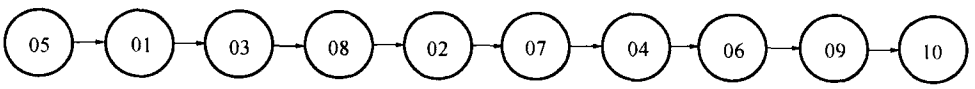

图 1 -2 线性结构

【例 1 -2】一种数据结构 Tree = （D，R），其中：

D = {01，02，03，04，05，06，07，08，09，10}

R = {r}

r = {<01，02>，<01，03>，<01，04>，<02，05>，<02，06>，<02，07>，<03，08>，<03，09>，<04，10>}

这种数据结构为树形结构，如图 1 -3 所示。

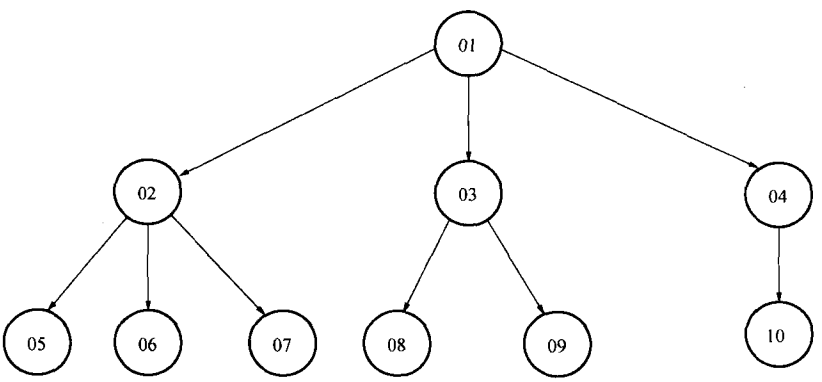

图 1 -3 树形结构

【例 1 -3】一种数据结构 Graph = （D，R），其中：

D = {a，b，c，d，e}

R = {r}

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

圆括号表示的关系集合是无方向的，如（a，b）表示从 a 到 b 之间的边是双向的。尖括号表示的关系集合是有方向的，如 <a，b> 表示从 a 到 b 之间的边是单向的。

其特点是各个结点之间都存在着多对多（*M*：*N*）的关系，即每个结点都可以有多个直接前驱或多个直接后继，如图 1 -4 所示，把具有这种特点的数据结构叫做图形结构，简称图。

1.3 数据的存储结构

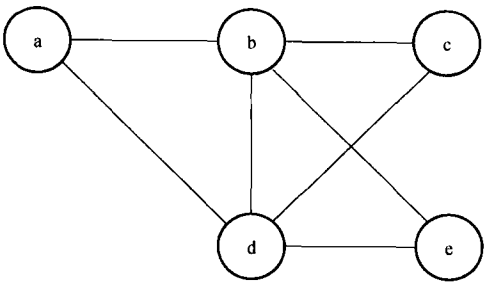

图1-4 图形结构

数据元素及其关系在计算机存储器内的表示，称为数据的存储结构。数据元素在计算机中主要有以下四种不同的存储结构。

1. 顺序存储

顺序存储结构的特点是借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系。例如，一个字母占一个字节，输入A、B、C、D、E，并存储在2000起始的连续的存储单元，如图1-5所示，即为顺序存储结构。

2. 链式存储

借助指示元素存储地址的指针（Pointer）来表示数据元素之间的逻辑关系。例如，图1-6为表示复数z=2.0+4.8i的链式存储结构。其中地址2000存放实部，地址2100存放虚部，实部与虚部的关系用值为“2100”的指针来表示（本例仅仅是为了简化讨论而作的一个引例，实际应用中，像复数这样简单的结构并不需要采用链式存储结构）。

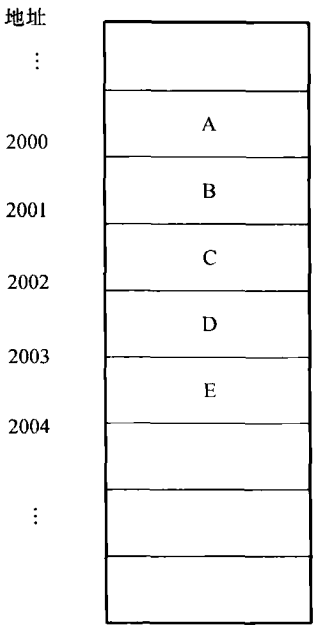

图1-5 顺序存储结构

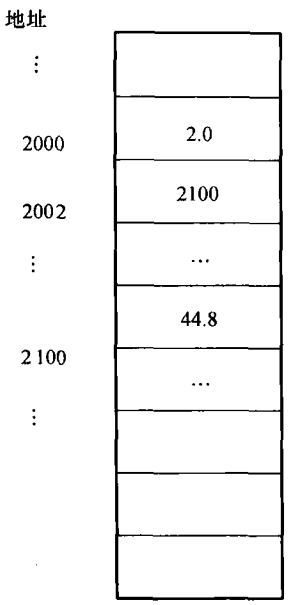

图1-6 链式存储结构

3. 索引存储

索引存储是在原有存储数据结构的基础上，附加建立一个索引表，索引表中的每一项都由关键字（能唯一标识一个结点的数据项）和地址组成。索引表反映了按某一个关键字递增或递减排列的逻辑次序，主要作用是为了提高数据的检索速度。

4. 散列存储

散列存储是通过构造散列函数来确定数据存储地址或查找地址的。

例如：某一地区进行新中国成立后出生人口的统计：用“出生年份 - 1948 = 存储地址”来构造一个函数，这样就能方便地得到一个某地区新中国成立后出生人口的调查表 1 - 1。

表 1 - 1 某地区新中国成立后出生人口调查表

存储地址	01	02	03	…	21	…	54	55
出生年份	1949	1950	1951	…	1969	…	2002	2003
人　数	1000	1200	1500	…	1800	…	1350	1100

1.4 算法和算法分析

算法与数据结构的关系紧密，在算法设计时先要确定相应的数据结构，而在讨论某一种数据结构时也必然会涉及相应的算法。任何一个算法的设计都取决于选定数据间的逻辑结构，而算法的实现则依赖于数据所采用的存储结构。

1.4.1 算法特性

1. 算法

算法（Algorithm）是对特定问题求解步骤的一种描述，是指令的有限序列。其中每一条指令表示一个或多个操作。

2. 算法的特性

（1）有穷性：一个算法必须在有限步骤之后结束，并且每步应该在有限的时间内完成。

（2）确定性：算法的每一条指令必须有确切的定义，而无两义性。

（3）正确性：能按设计要求解决具体问题，并得到正确结果。

（4）输入：一个算法具有 0 个或多个输入。

（5）输出：一个算法具有 1 个或多个输出。

3. 算法与程序的区别

（1）一个算法必须在有穷步之后结束；一个程序不一定满足有穷性。

（2）程序中的指令必须是机器可执行的，而算法中的指令则无此限制。

（3）算法代表了对问题的求解过程，而程序则是算法在计算机上的实现。算法用特定的程序设计语言来描述，就成了程序。

（4）算法与数据结构是相辅相成的。

4. 算法设计要求

（1）正确性：算法应能满足设定的功能和要求。

（2）可读性：思路清晰、层次分明、易读易懂。

（3）健壮性：输入非法数据时应能作适当的反应和处理。

（4）高效性：执行同一问题时时间越短，算法的效率就越高。

（5）低存储量：完成同一功能，占用存储空间应仅可能少。

1.4.2 算法的效率

1. 事后统计法

事后统计法可通过计算机内部计时功能来统计，但缺点是：

（1）必须先运行按照算法编写的程序；

（2）运行时间的统计依赖于计算机硬件和软件的环境，容易掩盖算法本身的优劣。

2. 事先估算法

将一个算法转换成程序并在计算机上运行，其所需要的时间取决于下列因素：

（1）使用何种程序设计语言；

（2）采取怎样的算法策略；

（3）算法涉及的问题的规模；

（4）编译程序产生的目标代码的质量；

（5）机器执行指令的质量。

显然，在各种因素不确定的情况下，使用执行算法的绝对时间来衡量算法的效率是不合适的。在上述各种与计算机相关的软、硬件因素确定以后，那么一个特定算法的运行工作量的大小就只依赖于问题的规模（通常用正整数 n 表示）。

1.4.3 算法效率的评价

算法效率的评价用时间复杂度（所需运算时间）和空间复杂度（所占存储空间）表示，重点是时间复杂度。一个算法所需的运算时间通常与所解决问题的规模大小有关。用 n 表示问题规模的量，把算法运行所需的时间 T 表示为 n 的函数，记为 $T(n)$。不同的 $T(n)$ 算法，当 n 增长时，运算时间增长的快慢很不相同。一个算法所需的执行时间就是该算法中所有语句执行次数之和。

（1）时间复杂性

当 n 逐渐增大时 $T(n)$ 的极限情况，一般简称为时间复杂度。时间复杂度常用数量级的形式来表示，记作 $T(n)=O(f(n))$。其中，大写字母 O 为 Order（数量级）的字头，$f(n)$ 为函数形式，如 $T(n)=O(n^2)$。当 $T(n)$ 为多项式时，可只取其最高次幂项，且它的系数也可略去不写。一般地，对于足够大的 n，常用的时间复杂性存在以下顺序：

$$O(1) < O(\lg n) < O(n) < O(n\lg n) < O(n^2) < (n^3) < \cdots < O(2^n)$$

其中，$O(1)$ 为常数数量级，即算法的时间复杂性与输入规模 n 无关。

【例1-4】 交换 A 和 B 内容程序段的时间复杂度。

```
T = A;
A = B;
B = T;
```

解 以上三条单个语句均执行 1 次，该程序段的执行时间是一个与问题 n 无关的常数，因此，算法的时间复杂度为常数阶，记作 $T(n)=O(1)$。

【例 1-5】计算下面求累加和程序段的时间复杂度。

```
① x=0; y=0                          // 执行2次
② for (k=1; k<=n; k++)
③ x++;                              // 执行n次
④ for (i=1; i<=n; i++)
⑤ for (j=1; j<=n; j++)
⑥ y++;                              // 执行n^2次
```

解 $T(n) = n^2 + n + 2$ $\quad T(n) = O(n^2)$

（2）空间复杂度（Spase Complexity）：$S(n) = O(f(n))$

小 结

（1）数据结构就是研究数据的逻辑结构、存储结构和运算方法的学科。

（2）数据的逻辑结构包括：集合、线性结构、树形结构、图形结构四种类型。

（3）集合中不存在数据之间的关系；线性结构元素之间存在一对一的关系；树形结构元素之间存在一对多的关系；图形结构元素之间存在多对多的关系。具有一对多和多对多关系的结构又称为非线性结构。

（4）数据的存储结构包括：顺序存储、链式存储、索引存储、散列存储四种。

（5）顺序存储可以采用一维数组来存储；链式存储可以采用链表结构来存储；索引存储则在原有存储数据结构的基础上，附加建立一个索引表来实现，主要作用是为了提高数据的检索速度；而散列存储则是通过构造散列函数来确定数据存储地址或查找地址的。

（6）算法是对特定问题求解步骤的一种描述，是指令的有限序列。算法具有：有穷性、确定性、正确性、输入、输出等特性。

（7）算法的设计要求包括：正确性、可读性、健壮性、高效性和低存储量等。

（8）算法的效率通常用时间复杂度与空间复杂度来评价，应该逐步掌握其基本分析方法。

（9）通常把算法中包含简单操作次数的多少叫做算法的时间复杂度。一般只要大致计算出相应的数量级即可；一个程序的空间复杂度是指程序运行从开始到结束所需的存储量。

（10）一个算法的时间和空间复杂度越好，则算法的效率就越高。

实验 1

1. 实验目的

（1）复习 C（或 C++）语言指针的用法；

（2）复习 C（或 C++）语言结构体的用法；

（3）理解算法时间复杂度分析的基本方法；

（4）通过实验程序，分析它们的时间复杂度。

2. 实验内容

（1）用指针方式编写程序：从键盘输入 10 个整型数据，并存入数组，要求将 10 个数中

最大的数与第1个输入的数交换；将10个数中最小的数与最后1个输入的数交换。

（2）有5个学生，每个学生的数据包括学号、姓名、三门课的成绩、平均分。

要求：从键盘依次输入5个学生的学号、姓名、三门课成绩，自动计算三门课成绩的平均分，并将5个学生的数据在屏幕上输出。

（3）编写程序，并分析时间复杂度。

① 键盘输入三个整数，并按从小到大的次序输出。

② 将1～10存入数组 a [10]，并将其逆序输出。

③ 编写程序，打印如图1-7所示图形。

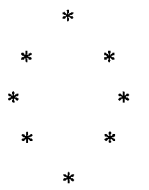

图1-7 打印的图形