普通高等教育“十一五”国家级规划教材 计算机系列教材

北京市高等教育精品教材立项项目

实用数据结构

林小茶 编著

清华大学出版社

普通高等教育“十一五”国家级规划教材 计算机系列教材

北京市高等教育精品教材立项项目

林小茶 编著

实用数据结构

清华大学出版社
北京

内 容 简 介

本书为与计算机应用相关的专业量身定做，保留了经典数据结构的主要内容，但是做了一些必要的删减，以适应相对较少的课时安排；同时，还选择了一些实用性比较强的实例作为案例。在讲解数据的存储结构时，使用了大量的图表，有助于学生对数据结构及相关算法的理解。

本书的主要内容包括概述、栈与队列、线性表、线性表的链式存储、哈希表与索引表、内排序、树与二叉树和图。在各章内容的安排上不求大而全，力求少而精，讲解透彻，重点突出。

本书既可以作为计算机相关专业本科学生学习数据结构的教材，也可作为自学者的教材或参考书。

图书在版编目(CIP)数据

实用数据结构/林小茶编著. —北京：清华大学出版社，2013
计算机系列教材
ISBN 978-7-302-33828-4

Ⅰ. ①实… Ⅱ. ①林… 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(2013)第 215278 号

责任编辑：张　民
封面设计：常雪影
责任校对：梁　毅
责任印制：何　芊

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
课件下载：http://www.tup.com.cn，010-62795954
印 装 者：三河市李旗庄少明印装厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　张**：17.75　　**字　数**：447 千字
版　　次：2013 年 12 月第 1 版　　**印　次**：2013 年 12 月第1 次印刷
印　　数：1～3000
定　　价：29.00 元

产品编号：055260-01

《实用数据结构》 前言

数据结构是高等学校计算机和信息类专业的重点课程，也是对本科生来说比较难掌握的一门课程，经常被学生们称为“杀手”。原因是多方面的，例如，最初程序设计的课程未让学生对程序设计感兴趣，反而使学生惧怕程序设计，而数据结构是以程序设计为基础的；又例如，按照计算机学科的要求，学习数据结构之前应该先学习离散数学，由于各种原因，很多学校并未严格按照要求排课。本书尝试简化传统数据结构的内容，突出重点，强调应用，希望学生通过学习数据结构课程能够提高程序设计的能力，使学生们在认真学习了数据结构课程之后，能够编写一些实用的程序。

对于现阶段的人才培养应该着重对能力的培养，而不是简单地掌握理论。因此本书在编写的过程中，遵循谭浩强老师提出的“提出问题—解决问题—归纳分析”的新三部曲，强调从实践中获取知识。本书给出了能够解决实际问题的大量算法，希望学生们在阅读和总结这些算法的基础上提高自己程序设计的水平。因此，本书的大部分算法只要经过简单的修改，就能上机运行，具有很好的实用价值，也给学习者带来方便。

本书将经典的数据结构内容做了一些重新整合，去掉了一些笔者认为只是为了应付考试而没有实用价值的内容。例如，本书并没有将数组和(字符)串的内容作为单独的章节来讨论，而是将这部分内容融入其他章节，作为解决具体问题的一种方法，从而避免了只讨论抽象而简单的理论，而不注重实用的讲解方法。本书在讲解数据的存储结构时，使用了大量的图表，帮助学生对数据结构的理解。

考虑到在数据结构的学习中，教师需要在课堂上对大量的算法进行讲解，而学生们应该在此基础上大量阅读并理解数据结构的经典算法，因此本书对所有的算法都做了详细的注释，对一些难度比较大的算法，在用 C 语言描述之前，还使用伪语言对算法进行了描述。

全书共分为 8 章：第 1 章概述、第 2 章栈与队列、第 3 章线性表、第 4 章线性表的链式存储、第 5 章哈希表与索引表、第 6 章内排序、第 7 章树与二叉树和第 8 章图。

注意：本书尝试了一种不同于其他数据结构教材的排列顺序，主要是考虑学生的接受能力，由于线性表的链式存储相对复杂，很多学生在学习中感到很困难，而栈和队列的基本操作比较简单，便于理解，因此将栈和队列内容放在了第 2 章。但是，有关栈和队列的应用还是有一定难度，可以灵活安排案例讲解的内容和时间，不要求一定按照教材的顺序讲课。

本书在编写时，尽量使内容通俗易懂，适于自学，由浅入深，便于理解。在各章内容的安排上不求大而全，力求少而精，讲解透彻，重点突出。

编者水平有限，错误在所难免，请广大读者批评指正。

作者电子邮件地址：xiaocl@bistu.edu.cn。

作　者

2013 年 11 月于北京

FOREWORD

《实用数据结构》目录

第1章 概 述

数据是计算机处理的对象，为了计算机能够更方便、高效地处理数据，如何将要处理的数据进行有效和系统的安排和存储，是数据结构要研究的内容。换言之，将所搜集的数据做有效和系统的安排，建立数据与数据彼此间的关系，称为数据结构(Data Structure)。

1.1 什么是数据结构

由于计算机所解决的问题都是从生活中抽象出来的，因此如何将生活中的问题用计算机来表达是程序设计者必须研究的课题，生活中既有计算 1+2 这样简单的问题，也有走迷宫、下棋这样的复杂问题。计算 1+2 这种简单的数学问题不是数据结构的研究重点，数据结构主要研究迷宫、下棋这样的相对复杂的非数值运算的问题。在计算机内部，1 和 2 是用二进制 00000001 和 00000010 来表示的，那么，迷宫、棋盘在计算机内部如何表示呢？在此基础上又如何求解迷宫问题和解决人机对弈问题呢？这都是数据结构的研究内容。

数据结构是信息的组织方式。要计算机完成一项任务，用不同的数据结构表示要处理的数据时将产生不同的执行效率。这就有必要研究各种不同的数据结构表示相同的数据其效率差异和适用场合。

首先我们来看家庭图书的管理。

一般的家庭，图书的数量是比较少的，一个大书柜就能放下几乎所有的书，假设玲玲属于不太勤快的那种人，她也许会将书随手放在书柜的某个位置；而丹丹会将书按自己的办法分门别类地存放，小说放第一层，经济类图书放第二层，计算机类图书放第三层，其他书放第四层。那么，拿书的时候谁会比较方便呢？当然是丹丹。如果玲玲想找一本书，除非她有超于常人的记忆力，否则她有可能需要翻遍所有的书，才能找到自己想要的书，也许运气好的时候(或者是刚刚放进去还记得位置)能少翻几本书就找到了。但是丹丹就可以效率比较高地找到想要的书，想看小说就在第一层找，想学学计算机知识就在第三层找。

从上面的描述中，可发现以下几点：

(1) 同样是书的存放，由于存放方式的不同，取书的方法也不同。

(2) 两种方法的适用性是不同的。第一种方法简单易行，不用考虑，随便放就行了，但是取书的时候比较麻烦，如果书比较少，尚可接受；第二种方法比第一种方法要复杂一些，但是由于将书按一定的规则进行了分类存放，在书比较多的情况下能更方便地找到自己想要的书。

下面用 C 程序来模拟第一种方法存书和取书的过程。

例 1.1 模拟将少量的书按顺序放在书柜里的方法，使用一维数组存储图书的信息。

```
#include "stdio.h"
#include "string.h"
#define SIZE 80
struct Book
{  char name[20];
   /*关于书的其他描述可以在此说明*/
};
int store(struct Book a[]);
void lookfor(struct Book a[],int amount);
int main()
{  int amount;
   struct Book book[SIZE];                    /*定义存放书的一维数组*/
   amount=store(book);                        /*调用函数输入若干本书的信息*/
   lookfor( book,amount);                     /*调用函数查询某本书*/
   return 0;
}
int store(struct Book a[])
{  int amount,i;                              /*定义变量*/
   printf("请输入书的总数: ");                 /*提示用户输入书的总数*/
   scanf("%d",&amount);
   for (i=0;i<amount;i++)                     /*循环接收 amount 本书*/
   {  printf("请输入一个书名: ",amount);       /*存入数组 a 中*/
      scanf("%s", a[i].name);
   }
   return amount;                             /*返回书的总数*/
}
void lookfor(struct Book a[],int amount)
{  int i;
   char book[20];                             /*定义变量*/
   printf("请输入你要找的书名: ");             /*提示用户输入要找的书名*/
   scanf("%s",book);
   for (i=0;i<amount;i++)                     /*开始查询*/
       if (strcmp(book,a[i].name)==0)         /*找到书*/
       {  printf("请在第 %d 个位置上取书\n",i+1);
          break;
       }
   if (i==amount)                             /*没有找到*/
       printf("抱歉!没有找到对应的书!");
}
```

本例中使用结构数组 book 来存放书。程序的主函数中调用了两个自定义函数，store 函数负责存储所有的图书信息，lookfor 函数则负责查找一本书。

假设有“数据结构”、“C 语言程序设计”，“诛仙”、“问对问题赚对钱”、“衰退的时代”、“桥牌”、“红楼梦”和“离散数学”8 本书需要存储，那么 store 函数执行以后，数组 book 的存储图如图 1-1 所示。

数组下标	数组book内容
0	数据结构
1	C语言程序设计
2	诛仙
3	问对问题赚对钱
4	衰退的时代
5	桥牌
6	红楼梦
7	离散数学
8	
9	
10	
⋮	

图 1.1 数组 book 中存储的图书信息

amount 中存放的是书的总数，最初为用户输入，并作为 store 函数的返回值，然后传递给函数 lookfor，作为控制循环的终值。

这是一个非常简单的 C 程序，希望读者能够读懂这个程序，如果不能读懂本程序，建议重新学习 C 语言程序设计，然后再继续学习本书。

由于程序有些复杂，第二种存储方式用图 1.2 来说明一下。

依据第二种存储方式，我们只需要在 6 本书中寻找是否有自己需要的书籍，因为已经将图书进行了分类，假设每一类中最多有 6 本。

0	诛仙	问对问题赚对钱	数据结构	桥牌
1	红楼梦	衰退的时代	C语言程序设计	
2			离散数学	
3				
4				
5				

图 1.2 分类存储图书信息

显然，第二种方法在程序处理上会要比例 1.1 的复杂一些，但程序的效率有所提高，尤其是加快了查询的速度。假设，要查询“衰退的时代”，需要在第 2 列进行查找，比较两次就找到了。而按照第一种方法，必须逐一与一维数组中的每个元素进行比较，直到与最后一个数据（长度范围内的）进行比较以后才找到，此时，已经做了 6 次比较的动作。

两种方法的程序效率是不相同的，首先因为两个程序存储书籍信息的方式不同，而存储方式的不同又导致了程序必须执行不同的操作才能完成存储、查询等任务。经过更深入的分析，我们会发现，任意存放的书籍也是有一定的联系的，除了第一本书和最后一本书，对于其他任何一本书来说，它一定要么左右，要么上下与另一本书挨着。同样，第二种存放方式书之间的关系更为密切，要按照一定的规则排列，不能随意。

通过上述的分析，在解决书籍存放问题时，需要研究三点：书籍之间的逻辑关系（书籍间的联系）、书籍信息在计算机中的存储方式和对书籍信息的操作（存储、查询等）。而这正是数据结构要研究的三个问题。

数据结构研究的是非数值问题中数据之间的逻辑关系、具有逻辑关系的数据在计算

机内的表示方式(存储方式)以及对数据的操作。

以研究数据结构的观点,上述两种存放书籍的方法中,书籍之间的逻辑关系是不相同的。挨着放的书籍之间的逻辑关系是“线性关系”,而分类存储书籍之间的关系是“非线性关系”。

对于随意存放的书来说,每本书左边的以及右边的两本书有逻辑关系,一本书左边的书在逻辑关系上是该书的“直接前驱”,而一本书右边的书逻辑关系上是该书的“直接后继”,显然,第一本书没有“直接前驱”,最后一本书没有“直接后继”。这就是线性关系的特征,而直接前驱和直接后继都是数据结构的术语(假设从概念上将四层书看成是一体的)。

图 1.3 描述了随意存放的书之间的逻辑关系。

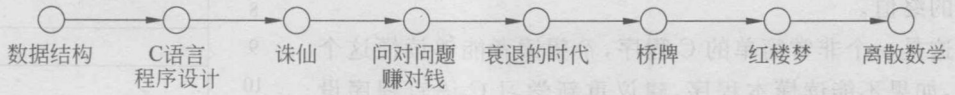

图 1.3 随意存放的书之间的逻辑关系

对于分类存储的书来说,书与书之间的逻辑关系要比随意放置的书复杂一些,可用数学上的集合来描述。首先,所有的书属于一个大的集合,这个集合可以分为 4 个集合:小说属于一个集合,经济类图书属于第二个集合……,图 1.4 描述了这种逻辑关系。

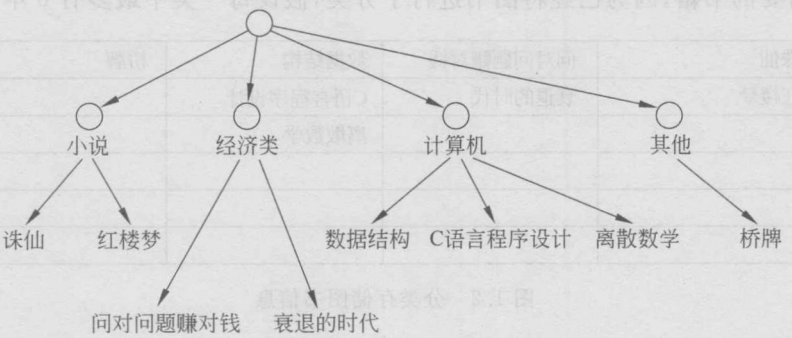

图 1.4 分类存放的书之间的逻辑关系

分类存放的书之间的关系是一种非线性的关系,我们称之为“树”型关系。

具有逻辑关系的数据在计算机内存储时,所采取的存储方式应该能够描述数据之间的逻辑关系。例 1.1 中,书的信息存储在一维数组中,除了 book[0]和 book[7],对于其他 book[i],book[i−1]中存储的是 book[i]的直接前驱,book[i+1]中存储的是 book[i]的直接后继。这种存储方式称为顺序表。

对于一种逻辑关系,还可以有不同的物理存储方式。仍以随意存放的书为例,存储的另一种方法如图 1.5 所示,这种存储结构称为单链表。数据之间的逻辑关系通过每个结点的指针表示,《数据结构》的后继是《C 语言程序设计》,《C 语言程序设计》的后继是《诛仙》,依此类推。

图 1.4 表示的数据的逻辑关系,不仅可以用图 1.5 表示的单链表来存储,还可以使用其他存储方式,如索引存储、散列表存储等。这些物理存储的方式有各自不同的特点,是数据结构要研究的重要内容。

图 1.5 单链表存储的随意存放的图书信息

在研究了图书数据的逻辑关系和物理存储以后，最后就要研究如何对数据进行操作了。如果数据之间具有相同的逻辑关系，但是以不同的物理方式存储，那么，对数据的操作方式一般也是不同的。可以想像，建立单链表存储图书信息和建立顺序表存储图书信息，操作方式肯定是不同的。操作方法的不同将使得操作效率存在差异，所以，在研究数据操作的同时，还必须研究这些操作的效率，研究结果将作为解决实际问题时选择数据结构的依据。

图书问题是本书经常要讨论的案例，通过该案例的学习，可以帮助我们理解数据结构的相关概念。

1.2 数据结构的相关概念和术语

数据(Data)是能够输入到计算机中，并被计算机识别、存储和处理的符号集合。数据可以分为数值数据和非数值数据。数值数据包括整数、实数或复数；非数值数据包括字符、文字、图形、图像和声音等。

在 1.1 节讨论图书问题时，我们将一本图书的信息存储在一个结构体中。如果需要的话，还可以存储有关该书的其他信息。

数据元素(Data Element)是组成数据的基本单位，每个数据元素是具有独立意义的个体，在程序设计时，将其作为一个整体来对待。数据元素又可称为元素、结点、顶点、记录等。

对于每本书而言，可能需要用书名、作者、出版社和出版地等信息来描述，对一本书的描述信息就是一个数据元素。例如，{数据结构(第二版)、严蔚敏、清华大学出版社、北京}描述的就是一本书，所以是一个数据元素。

又例如，表 1.1 描述的是一所大学集中管理的教室钥匙的信息，对每把钥匙的描述信息包括钥匙的材质、颜色、长度、是否有备份以及被谁借走了等。表 1.1 列出了部分钥匙的信息，每一行是一个数据元素。

表 1.1 描述钥匙的数据表

教室号	材质	长度	是否备份	借者
1101	钢	10	是	杨言
1105	铜	12	否	罗珊
1202	镍白铜	12	否	夏青
…				
4305	镍白铜	15	是	林灵

一个数据元素

数据项(Data Item)是数据结构中讨论的最小单位,一般将其看作不能再分解的数据。一个数据元素可由若干个数据项组成。例如,描述书的时候,一个数据元素描述了一本书的情况,其中包括书的书名、作者、出版社和出版地;而描述钥匙的数据表中,一个数据元素描述了一把钥匙的情况,其中包括钥匙的教室号、材质、长度、是否备份和借者等数据项。

数据对象(Data Object)是一组具有相同性质的数据集合。例如:

数字(Digit)={0,1,2,3,4,5,6,7,8,9,…}

字母(Letter)={A,B,C,…,Z,a,b,…,z}

整数(Integer)={0,±1,±2,±3,…}

上面列举的数据对象都是可以使用C语言的标准数据类型(Data Type)来表示的。下面的数据对象需要用C语言的构造类型来描述:

字符串(String)= {"a", "b",…, "aa", "ab", "ac",…}

教室钥匙={{1101,"钢",10, "是", "杨言"},{1105,"铜",12, "否", "罗珊"},…}

短信息={{12:10:30,"13126809***","13126808***","你好","已发送"},

{12:10:35,"13226808***","13126807***","下午见", "发送失败"},…}

上面的例子中短信息的数据含义:12:10:30是发送时间,13126809***是源手机号,13126808***是目标手机号,"你好"是发送的内容,"已发送"是发送的状态。

0,1,2,…,9都是数字的实例。数据对象的每个实例要么是一个原子(不能再进一步分解),要么是由其他数据对象的实例复合而成。按照这种观点,整数132可以看作数字1、3和2按顺序组成。

不难看出,一个数据项的所有数据就是一个数据对象,而一组数据元素也是一个数据对象。

数据对象的实例以及构成实例的元素通常都有某种联系,例如,自然数0是最小的自然数,1是仅比0大的自然数,而2是1之后的下一个自然数等。另外,与数据对象相关的还有一组操作。这些操作可以把对象的某个实例转换成该对象的另外一个实例,或转换成另一个数据对象的实例,或者同时进行上述两种转换。例如,整数相加、整数相减都是与数据对象Integer相关的操作。在C语言中,加和减都是系统提供的运算符,而整数的幂运算则需要使用函数调用来完成。

因此,可以这样说,高级程序设计语言对于通用的数据对象已经提供了操作的方法,而对于那些高级程序设计语言中没有涉及的数据对象来说,对其的操作进行研究就是数据结构的任务。

数据结构研究各数据元素之间存在的关系。数据结构包括数据的逻辑结构、存储结构(物理结构)和数据的运算三方面的内容。

1. 数据的逻辑结构

数据元素之间的逻辑关系称为数据的逻辑结构。从数学的角度观察,逻辑结构可以形式地定义为(K,R)(或(D,S)),其中,K是数据元素的集合,R是K上的关系的集合,这两个集合都是有限集。