HZ BOOKS
华章教育

面向CS2013计算机专业规划教材

数据结构

C语言描述

第2版

殷人昆 编著

Data Structure in C

Second Edition

机械工业出版社
China Machine Press

面向CS2013计算机专业规划教材

数据结构

C语言描述

第2版

殷人昆 编著

Data Structure in C

Second Edition

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

数据结构：C 语言描述 / 殷人昆编著 . —2 版 . —北京：机械工业出版社，2017.2
（面向 CS2013 计算机专业规划教材）

ISBN 978-7-111-55982-5

I. 数… II. 殷… III. ①数据结构 – 高等学校 – 教材 ②C 语言 – 程序设计 – 高等学校 – 教材 IV. ① TP311.12 ② TP312.8

中国版本图书馆 CIP 数据核字（2017）第 025870 号

本书以专业基础能力培养为目标，承续计算机程序设计基础课程，遵照 ACM CS2013 规范和教育部计算机科学与技术教学指导委员会关于《高等学校计算机专业人才专业能力构成与培养》的要求编写而成，旨在培养学生的基本计算思维能力，提高学生的算法设计和程序实现能力，并为学生提高系统开发能力打下良好的基础。

全书共分 10 章，主要介绍了数据结构基本概念与基本知识、线性表及其基本操作、栈和队列、字符串、数组和广义表、树与二叉树的概念和应用、图、查找和排序。

本书可以作为计算机科学与技术及相关专业本科生的教材，也可以作为计算机专业考研（硕士、工程硕士、博士）的复习教材，还可以供使用计算机进行系统开发的人员学习使用。

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）
责任编辑：佘　洁　　责任校对：董纪丽
印　　刷：北京诚信伟业印刷有限公司　　版　　次：2017 年 3 月第 2 版第 1 次印刷
开　　本：185mm × 260mm　1/16　　印　　张：25
书　　号：ISBN 978-7-111-55982-5　　定　　价：55.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88378991　88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

前　言

2013 年，ACM、IEEE 发布了 Computer Science Curricula 2013（简称 CS2013），对新时期计算机学科的核心知识体系做了梳理。在此之前，为适应计算机科学与技术专业人才培养的需求，教育部高等学校计算机科学与技术教学指导委员会颁发了《高等学校计算机科学与技术专业公共核心知识体系与课程》规范，从问题空间、知识取向、能力要求等方面，给出了教育改革和专业能力培养的要求。按照知识取向的不同，把计算机科学与技术学科划分为计算机科学（CS）、计算机工程（CE）、软件工程（SE）和信息技术（IT）4 个方向。从问题空间划分，计算机科学属于科学型，计算机工程和软件工程属于工程型，信息技术属于应用型。

综合上述两个规范，数据结构课程涵盖的知识单元有 5 个：基本算法（AL3）、算法与问题求解（PF2）、基本数据结构（PF3）、递归（PF4）和分布式算法（AL4）。教学大纲包括：1）算法、算法的时间复杂度和空间复杂度、最坏和平均时间复杂度等概念；2）算法描述；3）常用算法设计方法，包括迭代法、穷举搜索法、递推法、算法的递归描述技术；4）数据结构的基本概念和术语，包括数据结构、数据类型、抽象数据类型、信息隐藏；5）线性表与线性表的存储结构，包括顺序表、链接表；6）栈、队列；7）串；8）多维数组和广义表；9）树形结构及其应用，包括树、森林、二叉树、线索二叉树、Huffman 树；10）图及其应用，包括图的基本概念、图的存储结构、图的遍历、生成树和最小生成树、最短路径、拓扑排序；11）常用排序算法，包括插入排序、交换排序、选择排序；12）常用查找技术，包括线性表上的查找、树的查找、散列技术；13）文件，包括顺序文件、索引文件、索引顺序文件、散列文件。

不同的学科方向对以上知识单元的侧重点有所不同，见下表。

学科方向	涵盖知识点	学时 （课堂教学 + 上机实习（大作业））
计算机科学	算法分析基础、算法策略、基本算法、分布式算法、可计算性理论基础、算法与问题求解、基本数据结构、递归	48 + 16
计算机工程	基本算法分析、算法策略、计算算法、分布式算法、算法复杂性、算法与问题求解、基本数据结构、递归	48 + 16
软件工程	计算机科学基础［程序设计基础、算法及数据结构/表示、抽象（使用和支持）程序设计语言基础］、测试（异常处理）	48 + 16
信息技术	基本数据结构、算法和问题解决、递归	48 + 16

作者从 1987 年开始，在清华大学多年配合严蔚敏老师讲授数据结构课程，从 1996 年开始参与研究生（硕、博）入学考试的命题并批改试卷。多年的授课实践，积累了一些心

得与体会，深感现有的许多教材虽然看上去叙述清晰、思路严密，但往往比较表面化，忽略了许多知识点深层隐藏的东西，使得学生陷入“一学就会，一用就错，一考就糊”的尴尬境地。有鉴于此，作者决心基于多年的教学经验和知识积累，编写一本适合的教材，以便广大学生学好数据结构课程。

本书以专业基础能力培养为目标，承续计算机程序设计基础课程，遵照 CS2013 和教育部计算机科学与技术教学指导委员会关于《高等学校计算机专业人才专业能力构成与培养》的要求编写而成，旨在培养学生的基本计算思维能力，提高学生的算法设计和程序实现能力，并为学生提高系统开发能力打下良好的基础。也就是说，本书的宗旨是培养学生从对问题的理解入手，运用已掌握的算法和数据结构知识寻找解决途径，构建适当的算法和基本程序，以求得问题的解决。在从简单到复杂讨论数据结构的过程中逐步引入常用的算法设计策略，通过编程上机实验到工程实践，使学生初步掌握分析问题、解决问题的方法。

全书共分 10 章，各章内容如下：

第 1 章介绍了数据、数据结构及算法等基本概念与基本知识，学习数据结构所需要掌握的 C 语言程序设计预备知识，简单的算法性能分析，包括最坏和平均时间复杂度估计。

第 2 章介绍了线性表及其基本操作内容，并给出基于数组与链表表示的线性表基本操作函数的实现方法。在介绍线性表的应用时，讨论了集合与多项式的实现。

第 3 章介绍了栈、队列、双端队列、优先队列的概念和常用存储方法，以及栈和队列的应用举例，还讨论了递归、分治、回溯等算法设计的策略。

第 4 章介绍了字符串的基本概念，字符串的顺序、堆式、链式存储表示，字符串的模式匹配算法，主要包括 BF 和 KMP 算法。

第 5 章介绍了数组和广义表的概念，包括一维数组的存储表示和多维数组的存储表示，数组元素存储位置的计算，特殊矩阵，包括对称矩阵、三对角矩阵和 w 对角矩阵的压缩存储方法，稀疏矩阵的压缩存储及其矩阵转置、相加、相乘的实现，广义表的特性、存储表示及广义表的递归算法。

第 6 章介绍了树与二叉树的概念、表示方法及其常用操作的实现，还介绍了线索二叉树等特殊的二叉树结构，树与森林的常用表示方法及其基本操作的实现方法。

第 7 章介绍了树与二叉树的应用，包括 Huffman 树、二叉查找（排序）树、AVL 树（高度平衡的二叉查找树）、堆、并查集等。特别地，本章借讨论八皇后问题，进一步讨论了回溯法，并引入了状态树和剪枝方法。

第 8 章介绍了图的概念、图的基本表示方法及其常用操作的实现，还介绍了图的遍历以及连通图、双连通图，此外还介绍了图的应用，包括生成树和最小生成树、最短路径、拓扑排序、关键路径等，多次涉及回溯法、贪心法、动态规划等算法设计策略。

第 9 章介绍了多种常用的查找算法，包括顺序查找、折半查找及其扩展、索引顺序查找（分块查找）、多路查找树、B 树、B^+ 树、键树、散列表等。

第 10 章介绍了多种常用的排序算法。

全书采用 C 语言作为数据结构及算法的描述工具，并采用 C ++ 的引用型参数以简化

程序。算法描述力求结构化，注重编程风格，每个算法基本保持在100行之内，可读性强。与第1版相比，第2版的所有程序都有完整的用VC++语言编辑器调试通过的源代码，读者可以通过华章网站（www. hzbook. com）或直接向作者本人索取。

各章所附习题不包括选择题，但精选了大量综合应用题，这些习题的参考解答请参看配套教材《数据结构习题精析与考研辅导》。

由于作者的水平有限，可能在某些方面考虑不周，书中难免存在疏漏或错误，恳请读者提出宝贵意见。

作者邮箱：yinrk@ tsinghua. edu. cn 或 yinrk@ sohu. com。

编者

2016年12月于清华园荷清苑

教学建议

周次	教学内容	学时	程序上机实习/大作业（参考）
1	数据结构的概念，算法及其特性，算法设计过程和设计策略，算法的简单分析与时间、空间复杂度	3	①统计执行频度 ②出错处理：报错方式
2	线性表，顺序表及其操作的实现，单链表及其操作的实现，单链表的应用	3	用有序链表实现集合及其常用操作
3	循环单链表，循环双链表，多项式及其运算的实现	3	用链表实现多项式的存储及插入、删除、相加、相乘、输出操作
4	栈的概念与实现，队列的概念与实现，双端队列的概念，递归与递归栈	3	火车调度车厢排列 大作业：离散事件模拟——银行队列
5	汉诺塔与分治，迷宫与回溯，字符串的定义与存储，字符串的模式匹配	3	字符串的堆式存储及基本操作，以及查找、插入、删除、替换操作
6	一维与多维数组的存储，特殊矩阵压缩存储，稀疏矩阵的三元组表示与转置、相加运算的实现	3	实现迷宫算法，并对递归与非递归算法进行比较
7	广义表的概念和存储表示，树与森林的概念，二叉树性质，二叉树的存储表示	3	①递归建立二叉树 ②递归打印二叉树
8	二叉树遍历，二叉树计数，线索二叉树	3	不用栈非递归建立、遍历和输出二叉树（增加双亲指针）
9	树与森林的存储表示，树与森林遍历，Huffman 树与应用	3	大作业：Huffman 算法——编码、译码
10	二叉查找树，平衡二叉树，堆与优先队列的概念，堆的建立、插入和删除	3	实现小根堆的建立、插入与删除，要求堆元素是结构型
11	并查集及其操作，图的概念，图的存储表示，图的遍历算法	3	①建立图的邻接表 ②层次序遍历
12	图的连通性，最小生成树，最短路径	3	①用避圈法构建最小生成树 ②用破圈法构建最小生成树
13	拓扑排序，关键路径，查找的概念	3	大作业：建立求解欧拉回路问题的标准函数
14	顺序查找，折半查找，索引顺序查找，B 树，B^+ 树	3	①实现斐波那契查找算法 ②实现插值查找算法
15	散列表，排序的概述，直接插入排序，希尔排序，起泡排序，快速排序，简单选择排序	3	①测定希尔排序性能 ②测定快速排序性能
16	堆排序，归并排序，基数排序，排序方法的下界，排序方法的比较，外部排序简介	3	①测定归并排序性能 ②测定堆排序性能

说明：1）建议课堂教学 48 学时，并全部在多媒体教室内进行。

2）建议上机和工程实践（大作业）总学时为 16，如果不够，可在课外追加。学生尽可能使用自备计算机。

3）建议上 3 次习题课，可安排在第 4 周、第 10 周和第 15 周，总结学生在作业和上机过程中遇到的问题，总结解题的类型及解题的方法。

目　录

CHAPTER 1

第 1 章 绪　论

【本章学习的要点】

理解：数据、数据元素、数据的逻辑结构和存储结构、数据类型的概念。

理解：算法的定义及算法的特性。

理解：算法设计的方法和策略，算法设计的过程，评判算法优劣的标准。

掌握：使用 C 语言描述数据结构和算法的方法。

掌握：算法的性能分析，算法的性能标准，算法的后期测试，算法的事前估计，空间复杂度度量，时间复杂度度量，时间复杂度的渐近表示法，渐近的空间复杂度。

1.1　数据结构的概念及分类

开发一个计算机系统，最基本的工作就是编程。没有程序，没有数据，再好的计算机硬件也是没有用的。譬如修建高速铁路，没有好的运行计划程序、调度程序、网络通信程序和机车控制程序，不可能让整个线路运转起来。因此，一个计算机系统离不开程序和数据。算法和数据结构是程序的核心，是支持问题解决所采取的数据组织方式。

当前，算法和数据结构已成为计算机科学与技术学科和软件工程学科一门重要的专业基础课程，也是许多后续课程（如操作系统、计算机系统结构、计算机网络、编译原理、数据库、人工智能等）的先修课程。它不仅是计算机和软件工程专业的必修课，也是许多非计算机专业学生的选修课和从事系统开发人员的培训课程。

1.1.1　为什么要学习数据结构

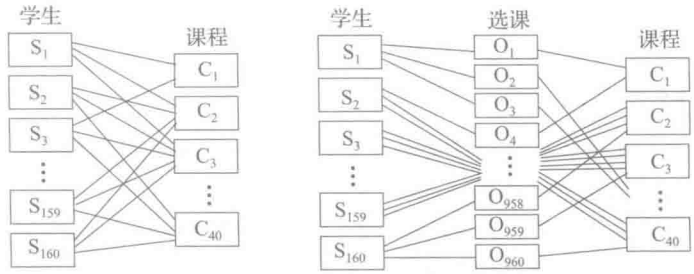
当今世界是一个互联网普及、信息大爆炸的世界，各种 APP（应用程序）已经为人们的生活提供了种种方便。不同领域的人们对各种 APP 的需求越来越广泛，然而，这都离不开计算机程序的开发人员。如何开发出性能优良、可靠性高的程序，是摆在计算机从业人员面前的首要问题，这些需求形成了一门基础性学科——程序设计方法学，而算法和数据结构正是程序设计方法学的核心。

【例 1-1】 开发大学选课系统。假设某学期为计算机系的 160 名本科生设置了 40 门可选课程，限定每位学生一个学期只能选 6 门课程。这样，学生和课程之间出现了多对多的关系，如图 1-1a 所示。如果引入一个交互实体“选课”，学生和课程之间的关系就转化为两个一对多的关系，如图 1-1b 所示。

图 1-1 给出了问题所涉及数据之间的关系，这些关系包括一对一、一对多和多对多的关系。数据及其他们之间关系的选择得当，可以大大简化问题，设计出良好的解决方案，降低解决问题的难度，甚至可以大幅降低最终程序的运行时间或节省大量的存储空间。

事实上，问题的最终解决取决于程序，而程序的质量又取决于算法的选择和数据的组织方

式。著名的瑞士科学家、图灵奖获得者沃思（Niklaus Wirth）在其经典著作“Algorithms + Data Structures = Programs”中强调“程序的构成与数据结构是两个不可分割的、联系在一起的问题。”他又引用霍尔（C. A. R. Hoare）在“Notes on Data Structuring”中的名言“不了解施加于数据上的算法就无法决定如何构造数据，反之，算法的结构和选择却常常在很大程度上依赖于作为基础的数据结构”，从而给出一个权威性的定义：程序就是在数据的某些特定的表示方式和结构的基础上对抽象算法的具体表述。



a）一个多对多的关系　　b）两个一对多的关系

图 1-1　课程与学生之间的两种联系

因此，我们学习数据结构的意义在于编写高质量的程序。因为人们的直观概念总是数据先于算法——你总得先有对象才有施加于它的算法。

1.1.2　与数据结构相关的基本术语

1）数据（data）我们在日常生活中会遇到各种信息，如用语言交流的思想，银行与商店的商业交易，在战争中用于传递命令的旗语等。这些信息必须转换成数据才能在计算机中进行处理。因此，数据的定义是：数据是信息在计算机程序中的表示形式，是描述客观事物的数、字符，以及所有能输入到计算机中并被计算机程序识别和处理的符号的集合。

数据大致可分为两类：一类是数值数据，包括整数、浮点数、复数、双精度数等，主要用于工程和科学计算，以及商业事务处理；另一类是非数值数据，主要包括字符和字符串，以及文字、图形、图像、语音等。

2）数据元素（data element）数据的基本单位是数据元素，它是计算机处理或访问的基本单位。例如，一个学生名册中的每个学生记录，一个字符串中的每一个字符，一个数组的每一个数组元素。不同场合下数据元素可以有别名，如元素、记录、结点、表项等。

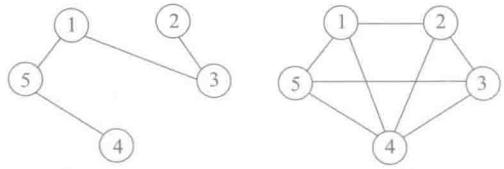
3）数据项（data item）数据元素可以是单个元素（称为原子），如整数 1、2 等，也可以由数据项构成。数据项又称为属性、字段、域，用以描述该元素的特征。数据元素中的数据项可以分为两种，一种叫做初等项，如学生的性别、籍贯等，这些数据项是在数据处理时不能再分割的最小单位；另一种叫做组合项，如学生的成绩，它可以再划分为物理、化学等更小的项。

4）数据对象（data object）从狭义的观点把数据对象定义为具有一定关系的相同性质的数据元素的集合；从广义的观点把数据对象定义为一组值的集合，且数据对象应是数据抽象和处理抽象的封装体，即数据对象的声明中不但要包含属性还要包含可用的操作。

5）数据结构（data structure）在数据处理中所涉及的数据元素之间都不是孤立的，在它们之间存在着某种关系，这种数据元素之间的关系称为结构。例如，招生考试时把所有考生按考试成绩从高到低排队，所有考生记录都将处在一种有序的序列中。又例如，在 n 个网站之间建立通信网络，要求以最小的代价将 n 个网站连通，如图 1-2a 所示，这样，在所有网站之间形

成一种树状关系；反之，要求网络中任一网站出现故障，整个网络仍然保持畅通，这样，在所有网站之间形成一种网状关系，如图 1-2b 所示。

由此可以引出数据结构的定义：数据结构是由与特定问题相关的某一数据元素的集合和该集合中数据元素之间的关系组成的。



a）网站之间的树状关系　　b）网站之间的网状关系

图 1-2　n 个网站之间的连通关系

6）数据类型（data type）从程序设计角度来看，数据类型与数据结构的概念是相通的，主要用于刻画程序中操作对象的特性。数据类型明确地或隐含地规定了在程序执行期间变量、表达式或函数可能取值的范围，以及作用在这些取值上的操作。因此，数据类型是一个值的集合和定义在这个值集合上的一组操作的总称。例如，整数类型是由在 -2^{15} ~ $2^{15}-1$ 之间的整数构成的，并包括对这些整数的加、减、乘、除和取模运算等。

数据类型按照其值的复杂程度不同，可分为**基本数据类型**和**结构数据类型**两种。基本数据类型中的每个数据元素都是无法再分割的整体，如一个整数、浮点数、字符、指针、枚举量等，故称为原子类型。结构数据类型由基本数据类型或子结构类型按照一定的规则构造而成，例如，一个学生的学籍卡片是一个结构数据类型，除了包括姓名、性别、年龄等基本类型外，还包括如家庭成员等子结构类型。

【例 1-2】 在 C 语言中对一个数据表（Data List）的数据类型定义如程序 1-1 所示。

程序 1-1　数据表的构造型类型定义

```
#define maxSize 100                    //表空间的大小,可根据实际情况决定
typedef int ElemType;                  //表中元素的数据类型,假定为 int
typedef struct {
    ElemType elem[maxSize];            //存放表元素的向量
    int n;                             //当前的表长度
} DataList;
```

数据类型和数据结构又是什么关系呢？一般认为，数据结构只考虑数据元素和数据元素之间的关系，而数据类型不但考虑数据结构，还要考虑数据元素的内容，即元素的值。

数据对象和数据结构又是什么关系呢？从对象技术的意义上讲，数据对象不但包括了数据结构，还包括了数据结构的存储表示和施加于其上的运算。可以说，数据对象包含了数据结构所涉及的所有层面。

辨析　*有关数据结构的讨论主要涉及数据元素和数据元素之间的关系，不涉及数据元素本身的内容。关于数据元素的内容，在系统开发时考虑。*

7）抽象数据类型（Abstract Data Type，ADT）在软件设计时，常常提到“抽象”和“信息隐蔽”。那么什么是抽象呢？抽象的本质就是抽取反映问题本质的东西，忽略非本质的细节。对于数据的抽象，可以用一个例子说明。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，而在汇编语言中则给出了各种数据的自然表示，如 15.5，1.3E10，10 等，它们是二进制数据的抽象，编程人员在编写程序时可以直接使用它们，不必考虑实现的细节。到了高级语言，给出了更高一级的数据抽象，出现了整型、实型、字符型、双精度型等。待到抽象数据类型出现，可以进一步定义出更高级的数据抽象，如各种表、队列、图，甚至窗口、管理器等。这种数据抽象的层次为设计者提供了有力的手段，使得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。

抽象数据类型通常是指由用户定义、用以表示应用问题的数据模型，又称数据抽象。抽象数据类型不像C语言中的构造（struct）类型那样，分别定义数据结构和相关操作，而是把数据成分和一组相关的操作封装在一起，从而实现更高一级的抽象。

【例1-3】 程序1-2给出了自然数（Natural Number）的抽象数据类型定义。

程序1-2 自然数的抽象数据类型

```
ADT NaturalNumber IS
/* objects:自然数是整数的有序子集合,它开始于0,结束于机器能表示的最大整数 MAXINT*/
{
    function:对于所有的 x,y∈NaturalNumber, +、-、<、==、= 等都是可用的服务。
    Zero ():NaturalNumber                  //返回 0;
    IsZero (x):Boolean                     //if ( x == 0 ) 返回 True; else 返回 False;
    Add (x,y):NaturalNumber                //if ( x+y <= MAXINT ) 返回 x+y;
                                           //else 返回 MAXINT;
    Equal (x,y):Boolean                    //if ( x == y ) 返回 1; else 返回 0;
    Successor (x):NaturalNumber            //if ( x == MAXINT ) 返回 x;
                                           //else 返回 x+1;
    Subtract (x,y):NaturalNumber           //if ( x < y ) 返回 0; else 返回 x-y;
}                                          //NaturalNumber
```

程序1-2给出“NaturalNumber”的抽象数据类型描述，objects简要说明数据对象是什么，有什么特点，接下来是一系列可用操作的列表，在操作的说明中要给出前置条件（IF）和后置条件（…THEN…ELSE）。前者给出操作正确执行所需的先决条件，后者表明在前置条件确定后应得到的结果，目的是保证抽象数据类型中每一个操作的正确性。

抽象数据类型最大的特点是把使用与实现分离，实行封装和信息隐蔽。就是说，抽象数据类型有两个视图：外部视图和内部视图。外部视图包括抽象数据类型名称、包含数据对象的简要说明和一组可提供使用者使用的操作，这是为了给使用者从外部了解和使用抽象数据类型所必需的；内部视图包括数据对象的存储结构定义和基于这种存储表示的各种操作的实现，包括了一切实现的细节。

现代程序设计学一个重要的观点是基于对象建立系统，而不是基于功能建立系统。因为系统的功能随着时间的推移必须适应新的情况和要求而不断更新，系统的对象相对比较稳定。每次修改系统功能不会导致系统全局修改，只需做局部修改即可。在这种情况下，要求构成系统的每一构件一定要满足“封装”和“信息隐蔽”，不允许直接存取构件内部的数据和调用构件内部不对外公开的操作，只能通过构件提供的允许使用者调用的操作来存取构件内保存的数据。这种构件的组织恰恰就是抽象数据类型。

抽象数据类型是属于概念层次的模型，它的实现就是面向对象系统中的“类”，所以，类和对象是抽象数据类型的实现层次的表示。

C++和Java语言都可以完全按照抽象数据类型来定义类。所以很多数据结构教材采用C++或Java语言描述。C语言没有可以实现抽象数据类型的相应机制，所以只有先用typedef struct…来定义结构类型，再分别定义相关的操作。不过，从教学观点来看，用C语言描述比较简洁，初学的学生容易上道，掌握了相关知识后将来还可以再学习。

1.1.3 数据结构的分类

1. 分解和抽象

数据结构的核心技术是分解和抽象。首先是数据的分解和抽象，通过对问题的分解划分出

数据的层次（数据－数据元素－数据项）；再通过抽象，舍弃数据元素与实现相关的细节，就得到数据的逻辑结构。其次是处理的分解和抽象，通过分解将处理需求划分成各种功能，再通过抽象舍弃功能的实现细节，就得到算法的定义。上述两个方面的结合将问题变换为数据结构和算法。这是一个从具体（即具体问题）到抽象（即数据结构和算法）的过程。

通过增加对实现细节的考虑进一步得到存储结构和实现运算，从而完成程序设计的任务。这是一个从抽象（即数据结构）到具体（即具体实现）的过程。

熟练地掌握这两个过程是数据结构课程在专业技能培养方面的基本目标。

2. 逻辑结构与存储结构

数据元素间的逻辑关系是指数据元素间抛弃了实现细节后所形成的关系，这种关系只是反映了问题的需求而不考虑如何实现。数据的逻辑结构在其数据元素间仅表现了这种逻辑关系。例如，在商店销售系统中的交易文件、商品文件，在图书管理系统中的图书文件等。数据的存储结构则是指数据逻辑结构的存储方式，是属于实现层面的，亦称为数据结构的存储映像。

数据的逻辑结构根据问题所要实现的功能建立，数据的存储结构则根据问题解决所要求的响应速度、处理时间、修改时间、存储空间和单位时间的处理量等来实现数据的逻辑结构。它们之间的关系如图 1-3 所示。从用户角度所能看到的只是数据的逻辑结构，所以通常所称的数据结构只是数据的逻辑结构。按照现代程序设计的思想，用户最好不要直接访问数据的逻辑结构所包含的数据内容，而应通过一组用户可见的操作来访问这些数据内容。逻辑结构可以通过一组特定的操作映射到存储结构。

图 1-3　逻辑结构与存储结构间的关系

辨析　作为数据结构的使用者，可以不关心数据结构的具体实现，只需了解数据的逻辑结构及其相关的可用操作，能够用好它就可以了。对于数据结构的设计者或学习者，需要理解数据的逻辑结构、存储结构和相关操作的实现。

3. 逻辑结构的分类

在解决特定问题时，依据可能遇到的数据元素之间关系的不同，数据的逻辑结构（在本书下文中简称为数据结构）如图 1-4 所示。

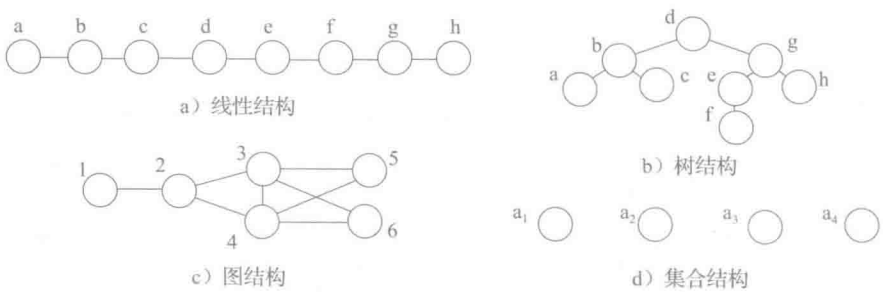


图 1-4　不同数据结构中各数据元素间的联系

1）线性结构。在这种结构中所有数据元素都按某种次序排列在一个序列中，如图 1-4a 所示。它的元素之间的关系是一对一的，如线性表、向量、栈、队列、优先队列、字典等。

2）非线性结构。在非线性结构中各个数据元素不再保持在一个线性序列中，每个数据元素可能与零个或多个其他数据元素发生联系。根据关系的不同，可分为树结构和图结构。树结