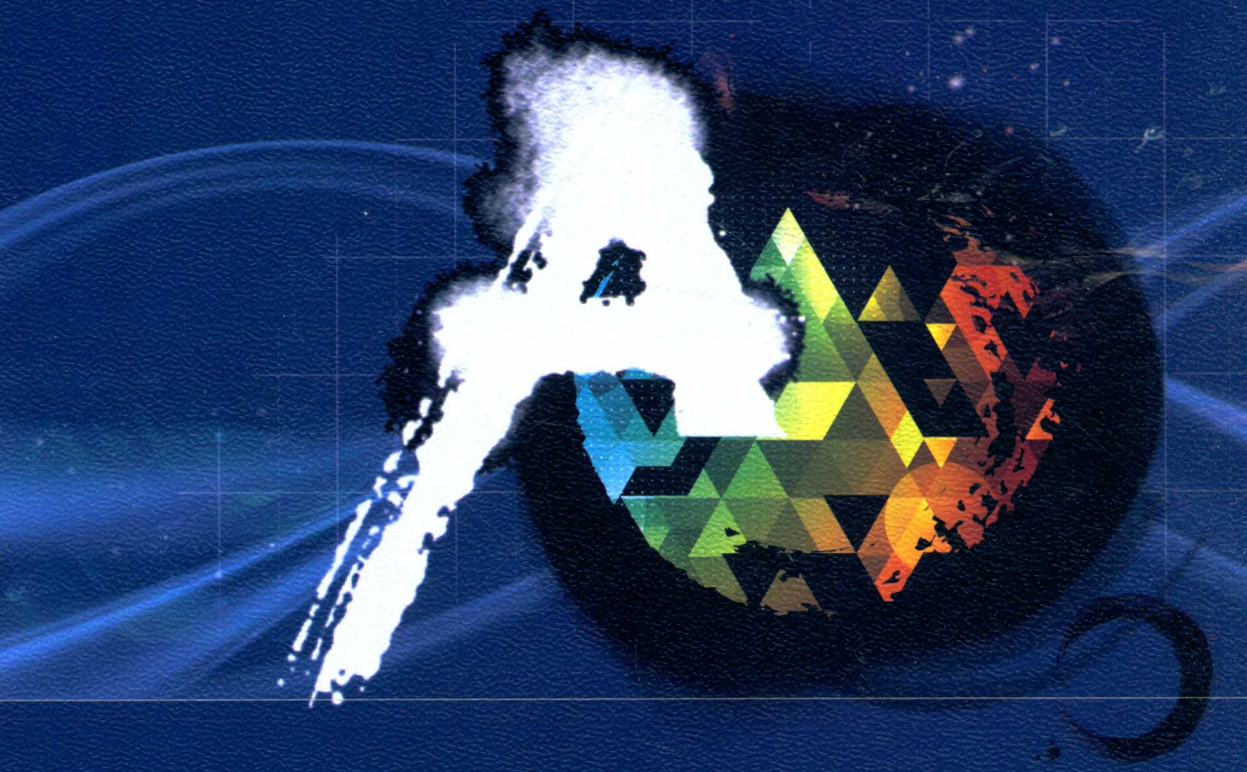
普通高等教育“十二五”规划教材



数据结构(Java语言描述)

◎ 丁海军　编著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

普通高等教育“十二五”规划教材

数据结构
(Java 语言描述)

丁海军　编著

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

内 容 简 介

本书以作者多年数据结构课程教学经验为基础编写而成。全书共 9 章，第 1 章介绍了数据结构的基本概念及算法复杂度分析的详细框架和步骤；第 2～5 章是对线性结构的详细介绍，这一部分是整个数据结构的基础，包括顺序表、链表、栈、队列、稀疏矩阵以及线性表的查找和排序等内容；第 6～8 章主要研究树结构，第 6 章介绍了二叉树及树的性质、遍历算法及其应用，第 7 章研究了查找二叉树及相关算法，第 8 章介绍了堆结构及其应用；第 9 章介绍了图结构及关于图的几个基础算法。

本书以 Java 语言作为数据结构及算法的描述语言，以 Java 环境的集合框架为参照组织教学内容，便于读者更好地将课程内容运用到实际的软件开发过程中。本书配套有 PPT、习题解答等。

本书可作为普通高等院校计算机科学与技术、软件工程、信息管理与信息系统、信息与计算科学、电子信息等专业的算法与数据结构等课程的教材，也可作为工程技术和自学数据结构人员的参考读物。

图书在版编目（CIP）数据

数据结构：Java 语言描述/丁海军编著. —北京：电子工业出版社，2015.12

ISBN 978-7-121-27530-2

Ⅰ. ①数… Ⅱ. ①丁… Ⅲ. ①数据结构－高等学校－教材 ②JAVA 语言－程序设计－高等学校－教材

Ⅳ. ①TP311.12 ②TP312

中国版本图书馆 CIP 数据核字（2015）第 265386 号

策划编辑：任欢欢

责任编辑：任欢欢

印　　刷：三河市华成印务有限公司

装　　订：三河市华成印务有限公司

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：19　字数：486.4 千字

版　　次：2015 年 12 月第 1 版

印　　次：2015 年 12 月第 1 次印刷

定　　价：39.80 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

Preface

数据结构是计算机科学与技术及相关专业的基础课程，该课程是软件设计与开发的重要理论与实践基础。对于计算机相关专业学生来说，数据结构课程是最有价值的课程之一。数据结构课程是一门理论与实践并重的课程，要求学生既要掌握数据结构的基本概念、基础理论，又要掌握程序的设计、编写、调试、测试等技能；另外，数据结构课程概念多，许多内容理论深奥，学习难度比较大，一直是计算机专业比较困难的课程之一。

对于计算机专业学生来说，区别于其他专业的核心能力有两个方面：其一是算法设计能力；其二是软件工程能力。由于历史原因，目前数据结构课程偏重于算法设计能力的学习和训练，而对软件工程能力的培养考虑则普遍不足。

本书精选课程内容，突出重点。希望在训练学生计算机专业核心能力方面起到一定作用。概括起来说，本书主要特色包括以下几个方面。

1．内容相对全面，注重基础，突出重点和难点

（1）本书包含了传统数据结构课程的主要内容，删除已经很少使用的线索二叉树、外排序等内容。对广义表只做概念性介绍，对于字符串只介绍匹配算法。增加了一般教材涉及较少但应用广泛的并查集、堆、优先队列等内容。

（2）对重点和难点内容花费更多的篇幅进行解释。对递归的深入理解历来是学习难点，我们将二叉树、栈与递归融合起来讲解。对哈夫曼编码解码算法、排序二叉树的删除算法、AVL 平衡二叉树的平衡调节算法等传统的难点内容给出了更加详细的讲解和程序实现，以便于深入理解算法思想。

（3）将稀疏矩阵和图的存储统一起来。将图的各种存储表示方法看成稀疏矩阵的表示，这样既贯通了知识点之间的联系，学生也更容易理解。

2．采用 Java 语言描述

现在绝大多数数据结构教材都采用 C 语言描述，我们选择 Java 作为课程描述语言主要基于以下几点考虑：

（1）希望学生在大学四年的学习中一以贯之地以某一种语言长期进行编程实践。在教学中我们发现，大学低年级学生一般使用 C 语言作为课程作业和实践的编程语言，但是到高年级进入到软件开发时，更多地使用 Java、C#等这类表达能力更强的面向对象语言工具，这样往往会出现学生对程序设计语言学习了很多，但是精通的很少，基础教学与实践应用脱节的现象。因此，我们希望从低年级开始在就使用 Java 作为编程语言和平台，从而为学生未来进行软件开发提供足够的训练。

（2）Java 语言是一门工业级的面向对象的语言。其完善的面向对象特性对于描述各种复杂的数据结构特别适用。Java 语言具有更好的抽象描述能力，描述抽象数据类型比较方便。Java 语言是一种完善的但是比 C++容易理解的面向对象的语言，面向对象语言比 C 语言这

类单纯面向过程语言更便于描述各种复杂的数据结构。

（3）Java语言更加规范，适合尽早训练学生的工程意识。

3. 注重算法和软件工程的平衡

传统的数据结构教材更多地从算法这个视角描述数据结构，这当然没有问题，毕竟“程序 = 数据结构+算法”，但是很多时候有意或无意地忽视了数据结构本身的描述和组织，不能从更为全局或者抽象的角度描述数据的结构和组织方式，忽视软件单元的重用、接口标准化等这样一些软件工程基本要素。从软件工程的角度看，如果我们在课程的学习过程中能潜移默化地培养学生有关软件工程的思维，将抽象、标准化、软件重用、设计模式等软件工程要素融合到数据结构的学习过程，可以更好地培养学生的工程意识。

本书中我们给出了一些主要数据结构的完整的程序代码，使学生在不冲淡算法学习这个主题的情况下，能学到从头设计基础类库的思维过程及需要考虑的一些因素。

我们给出的主要数据结构Java类尽可能与Java标准集合框架（Collection Frame）相对应，学生在学完本课程之后，对Java集合框架的内部实现细节会有一个基本的了解。同时也学会了如何逐渐设计一个复杂的基础类库。

程序设计能力的训练有其自身的规律，不可能一蹴而就，也没有捷径可走。计算机专业学生必须具有该专业的基本素质，熟练掌握一门程序设计语言，在此基础上掌握数据结构、算法的重要设计思想和一些常用算法。同时还要掌握一些重要的软件工程方法，只有这样才能为未来的职业生涯打好坚实基础。

本书除了算法的伪代码外，所有程序都在Java 7 和Java 8环境下调试通过。

本书由丁海军编著，在编写过程中，得到教育部“卓越工程师培养计划”和教育部“专业综合改革试点”项目的支持。

感谢电子工业出版社任欢欢编辑的支持鼓励。

丁海军

2015年10月

目录

CONTENTS

第 1 章　绪论 …… 1

1.1　数据结构的概念 …… 1
1.1.1　为什么要学习数据结构 …… 2
1.1.2　有关概念和术语 …… 4
1.1.3　数据结构的三要素 …… 5
1.2　抽象数据类型 …… 6
1.2.1　数据类型 …… 6
1.2.2　抽象数据类型 …… 7
1.3　算法概念及算法设计的问题 …… 8
1.3.1　什么是算法 …… 8
1.3.2　算法特性 …… 10
1.3.3　算法的结构和表示方法 …… 10
1.3.4　算法设计原则 …… 11
1.3.5　几种基本的算法设计方法和策略 …… 12
1.3.6　编程解决问题的一般步骤 …… 12
1.4　算法分析 …… 13
1.4.1　时间复杂度分析的两类方法 …… 13
1.4.2　时间复杂度分析的理论框架 …… 14
1.4.3　非递归算法时间复杂度分析步骤 …… 19
1.4.4　典型非递归算法的时间复杂度类型 …… 20
1.4.5　递归算法时间复杂度分析步骤 …… 22
1.4.6　空间复杂度 …… 23
1.5　数据结构课程的内容 …… 23
1.6　习题 …… 24

第 2 章　线性表 …… 27

2.1　线性表的逻辑结构 …… 27
2.2　顺序表概念及存储特点 …… 29
2.2.1　顺序表的逻辑特点 …… 29
2.2.2　顺序表面向对象描述 …… 29
2.3　顺序表的重要算法及实现 …… 31
2.3.1　初始化 …… 31
2.3.2　顺序表容量管理 …… 31
2.3.3　数据存取 …… 32

2.3.4 向顺序表中插入元素……33
2.3.5 删除顺序表中的元素……34
2.3.6 查找元素……35
2.3.7 顺序表中元素的有序插入与排序……36
2.3.8 顺序表转换为数组……37
2.3.9 顺序表转换为字符串……38
2.4 单链表概念及类定义……38
2.4.1 单链表基本概念……38
2.4.2 链表面向对象描述……41
2.5 单链表重要算法实现……44
2.5.1 数据存取……44
2.5.2 向链表中插入元素……44
2.5.3 删除链表节点……47
2.5.4 查找节点……49
2.5.5 向链表中有序插入节点……50
2.5.6 链表排序……52
2.5.7 链表转换为字符串和数组……53
2.6* 链表迭代器……54
2.6.1 迭代器的概念……54
2.6.2 与迭代器有关的 Java 语言特性……55
2.6.3 链表类 LinkList 迭代器的实现……56
2.7 循环链表与双向链表……58
2.7.1 循环链表……58
2.7.2 双向链表……59
2.8* 顺序表和链表的比较……60
2.9 习题……61
第 3 章 特殊的线性结构……64
3.1 栈……64
3.1.1 基本概念……64
3.1.2 链栈——栈的链表实现……65
3.1.3 顺序栈——栈的数组实现……66
3.1.4 表达式求值……68
3.2 队列……72
3.2.1 队列概念……72
3.2.2 链式队列……73
3.2.3 顺序队列……74
3.2.4 循环队列……75
3.2.5 队列应用……76
3.3 特殊矩阵……78
3.3.1 矩阵存储方式……78

3.3.2 对称矩阵和三角矩阵……79
3.3.3 对角矩阵……80
3.4 稀疏矩阵……81
3.4.1 三元组……81
3.4.2 矩阵抽象数据类型……82
3.4.3 稀疏矩阵的三元组顺序表表示……83
3.4.4 稀疏矩阵的行链表表示……88
3.4.5 稀疏矩阵的十字链表表示……91
3.5 广义表……92
3.5.1 广义表的定义和基本运算……93
3.5.2 广义表的存储结构……95
3.6 习题……96
第 4 章 线性查找算法……99
4.1 查找的基本概念……99
4.1.1 什么是查找及查找结构……99
4.1.2 查找结构的分类……100
4.1.3 平均查找长度……100
4.2 线性查找表……100
4.2.1 顺序查找……100
4.2.2 二分查找……100
4.2.3 分块查找……102
4.2.4 顺序表三种查找方法的比较……103
4.3 哈希查找……103
4.3.1 哈希表……103
4.3.2 哈希函数的构造方法……104
4.3.3 处理冲突的方法……106
4.3.4 哈希查找算法性能分析……109
4.4 哈希映射……109
4.4.1 映射（Map）概念……109
4.4.2 哈希表实现映射……110
4.5 串匹配……112
4.5.1 简单的模式匹配算法……112
4.5.2 KMP 模式匹配算法……114
4.6 习题……118
第 5 章 排序算法……122
5.1 基本概念……122
5.2 插入排序……122
5.2.1 算法设计……123
5.2.2 时间复杂度分析……124

5.3 选择排序 …… 124
5.3.1 算法设计 …… 124
5.3.2 时间复杂度分析 …… 125
5.4 冒泡排序 …… 126
5.4.1 算法设计 …… 126
5.4.2 时间复杂度分析 …… 127
5.5 快速排序 …… 127
5.5.1 快速排序的思想 …… 127
5.5.2 快速排序算法实现 …… 128
5.5.3 时间复杂度分析 …… 129
5.6 归并排序 …… 130
5.6.1 有序表的合并算法 …… 130
5.6.2 递归归并排序 …… 131
5.6.3 迭代归并排序 …… 132
5.6.4 时间复杂度分析 …… 133
5.7 基数排序 …… 134
5.7.1 桶排序 …… 134
5.7.2 基数排序应用举例 …… 134
5.8* 希尔排序 …… 135
5.8.1 排序过程与算法设计 …… 135
5.8.2 时间复杂度分析 …… 137
5.9 习题 …… 138

第 6 章 二叉树与树 …… 145

6.1 树与二叉树一般概念 …… 145
6.1.1 树 …… 145
6.1.2 二叉树 …… 146
6.2 二叉树的性质 …… 148
6.3 二叉树存储结构 …… 150
6.3.1 顺序存储 …… 150
6.3.2 链式存储 …… 151
6.4 二叉树遍历 …… 152
6.4.1 二叉树遍历概念 …… 152
6.4.2 先序遍历（DLR） …… 153
6.4.3 中序遍历（LDR） …… 154
6.4.4 后序遍历（LRD） …… 154
6.4.5 层次遍历 …… 155
6.4.6 遍历的应用 …… 156
6.4.7 根据已知的遍历序列恢复二叉树 …… 157
6.5 哈夫曼（Huffman）树及其应用 …… 159
6.5.1 最优二叉树（Huffman 树）概念 …… 159

6.5.2 Huffman 树的构造方法 …… 160
6.5.3 Huffman 编码 …… 162
6.5.4 Huffman 编码、解码算法实现 …… 164
6.6 递归 …… 172
6.6.1 递归调用树 …… 172
6.6.2 递归栈 …… 176
6.6.3 二叉树遍历的非递归算法 …… 179
6.7 树和森林 …… 182
6.7.1 基本概念 …… 182
6.7.2 树的存储结构 …… 182
6.7.3 树的遍历 …… 185
6.7.4 树的深度优先搜索与回溯法 …… 186
6.7.5 树与并查集 …… 190
6.8 习题 …… 192
第 7 章 查找树 …… 198
7.1 二叉查找树概念 …… 198
7.1.1 二叉查找树定义 …… 198
7.1.2 二叉查找树的面向对象描述 …… 199
7.2 二叉查找树主要算法 …… 202
7.2.1 求最大、最小值算法 …… 202
7.2.2 查找算法 …… 202
7.2.3 插入算法 …… 204
7.2.4 删除算法 …… 206
7.2.5 时间复杂度分析 …… 209
7.3 AVL 平衡二叉树 …… 210
7.3.1 平衡二叉树的概念 …… 211
7.3.2 平衡调整 …… 214
7.3.3 AVL 树的查找算法 …… 218
7.3.4 AVL 树的插入算法 …… 218
7.3.5 AVL 树删除算法 …… 219
7.3.6 AVL 树时间复杂度分析 …… 220
7.4 B-树 …… 222
7.4.1 分块索引查找 …… 222
7.4.2 B-树定义 …… 222
7.4.3 B-树查找 …… 223
7.4.4 B-树的插入 …… 225
7.4.5 B-树删除 …… 226
7.5 B+树 …… 228
7.6 习题 …… 229

第 8 章　堆与优先队列及堆排序 ····· 233

8.1　堆 ····· 233
8.1.1　堆的基本算法 ····· 234
8.1.2　堆的实现 ····· 237
8.2　优先队列 ····· 239
8.3　堆排序 ····· 240
8.4　习题 ····· 241

第 9 章　图结构及相关算法 ····· 244

9.1　图的概念 ····· 244
9.1.1　什么是图 ····· 244
9.1.2　图论的基本概念 ····· 245
9.1.3　树和森林 ····· 248
9.2　图的基本操作与图的存储结构 ····· 248
9.2.1　图的基本操作 ····· 249
9.2.2　图的存储结构概述 ····· 252
9.2.3　图的邻接矩阵存储 ····· 252
9.2.4　图的邻接表存储 ····· 257
9.2.5　图的边集数组存储 ····· 262
9.3　图的遍历 ····· 263
9.3.1　深度优先遍历 ····· 263
9.3.2　广度优先遍历 ····· 265
9.3.3　利用图遍历算法研究图的连通性 ····· 266
9.4　最小生成树问题 ····· 268
9.4.1　图的生成树和生成森林 ····· 268
9.4.2　图的最小生成树概念 ····· 269
9.4.3　构造最小生成树的 Prim 算法 ····· 270
9.4.4　构造最小生成树的 Kruskal 算法 ····· 273
9.5　最短路径问题 ····· 275
9.5.1　从一个源点到其他各点的最短路径 ····· 276
9.5.2　所有点对之间的最短路径 ····· 280
9.6　拓扑排序问题 ····· 282
9.6.1　偏序关系 ····· 282
9.6.2　拓扑排序 ····· 282
9.6.3　拓扑排序应用 ····· 283
9.7*　关键路径问题 ····· 285
9.7.1　AOE 网（Activity On Edge network） ····· 285
9.7.2　关键路径 ····· 286
9.8　习题 ····· 289

参考文献 ····· 294

第 1 章

绪　论

计算机科学是一门研究数据表示和数据处理的科学。数据是计算机化的信息，它是计算机可以直接处理的最基本和最重要的对象。无论是进行科学计算、数据处理、过程控制，还是对文件进行存储和检索及应用数据库技术，在这些计算机应用领域中，它们都是对数据进行加工处理的过程。因此，要设计出一个结构好、效率高的程序，必须研究数据的特性及数据间的相互关系及其对应的存储表示，并利用这些特性和关系设计出相应的算法和程序。

数据结构也称为信息结构。著名计算机科学家 P.Wegner 认为：计算机科学就是“一种关于信息结构转换的科学”。“结构”一词可以当成动词看待，解释为“把某些成分（或称原子、元素、成员等）按一定规律组织在一起的过程和方式”；也可以作为名词，理解为“把某些成分按一定方式组织起来而形成的实体”。对于复杂程序，需要处理的数据往往很多，这些数据之间往往又有错综复杂的相互联系，因此要有效地存储这些数据及其相互关系，如何使它们能够在程序中有效地使用，以及如何把它们组织起来，这是我们要解决的一个最重要的问题。数据结构课程讨论的就是数据的组织问题。实际上，数据结构问题不仅与程序设计有关，而且与计算机科学的许多其他领域也都有密切的关系。学好这门课程对于深入地理解计算机科学，继续学习计算机科学涉及的其他课程是非常重要的。要搞好计算机方面的工作，无论是实际的应用开发工作，还是理论研究工作，本课程的内容都是必不可少的知识基础。

学习任何课程都必须了解它与本学科其他课程的联系，明确其在更广泛的学科领域整体中的位置，只有这样才能抓住学习的要领，理解课程内容的实质，对于数据结构也不例外。图 1-1 形象地显示了数据结构知识在应用计算机解决问题过程中的地位，以及它与计算机科学技术领域中其他课程之间的关系。

作为数据结构的基础，有算法分析与设计的理论和方法，以及关于数据模型的理论。实现数据结构需要采用某种描述语言（通常是某种程序设计语言），并遵循一定的设计方法。学习和研究数据结构的目的是解决问题。

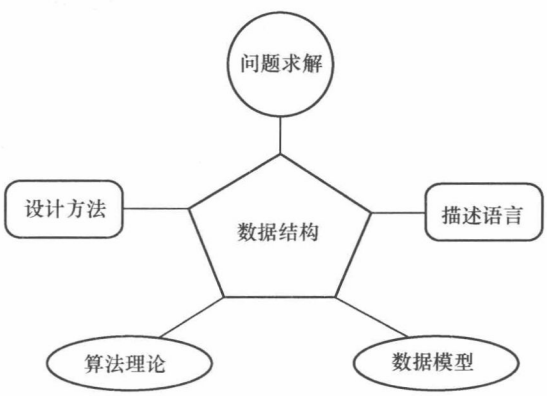


图 1-1　数据结构与其他课程的关系

1.1　数据结构的概念

数据结构是计算机科学与技术专业的基础课，是十分重要的核心课程。所有的计算机系

统软件和应用软件都要用到各种类型的数据结构。因此，要想更好地运用计算机来解决实际问题，仅掌握几种计算机程序设计语言是难以应付众多复杂的问题的。若要有效地使用计算机，充分发挥计算机的性能，还必须学习和掌握好数据结构的有关知识。打好“数据结构”这门课程的扎实基础，对于学习计算机专业的其他课程，如操作系统、编译原理、数据库管理系统、软件工程、人工智能等都是十分有益的。

1.1.1 为什么要学习数据结构

用计算机解决问题常常可以看成对实际问题的一种模拟，从这种观点出发，工作目标就是在计算机中建立一个与实际问题有着密切对应关系的模型，在这个模型中，计算机内存的数据表示了需要被处理的实际对象，包括其内在的性质和关系，处理这些数据的程序则模拟对象领域中的实际过程。最后，通过将计算机程序的运行结果在实际领域中给予解释，便得到了实际问题的解。

在计算机发展的初期，人们使用计算机的目的主要是处理数值计算问题。当我们使用计算机来解决一个具体问题时，一般需要经过下列几个步骤：首先要从该具体问题抽象出一个适当的数学模型，然后设计或选择一个解此数学模型的算法，最后编写出程序进行调试、测试，直至得到最终的解答。例如，求解梁架结构中应力的数学模型的线性方程组，该方程组可以使用迭代算法来求解。

由于当时所涉及的运算对象是简单的整型、实型或布尔类型数据，所以程序设计者的主要精力是集中于程序设计的技巧上，而无须重视数据结构。随着计算机应用领域的扩大和软、硬件的发展，非数值计算问题显得越来越重要。据统计，当今处理非数值计算性问题占用了90%以上的机器时间。这类问题涉及的数据结构更为复杂，数据元素之间的相互关系一般无法用数学方程式加以描述。因此，解决这类问题的关键不再是数学分析和计算方法，而是要设计出合适的数据结构，才能有效地解决问题。

以下列举的就是这一类的具体问题。

【例 1-1】学生信息检索系统。当需要查找某个学生的有关情况的时候，或者想查询某个专业或年级的学生的有关情况的时候，只要我们建立了相关的数据结构，按照某种算法编写相关程序，就可以实现计算机自动检索。因此，可以在学生信息检索系统中建立一张按学号顺序排列的学生信息表和分别按姓名、专业、年级顺序排列的索引表。由这4张表构成的文件便是学生信息检索的数学模型，计算机的主要操作便是按照某个特定要求（如给定姓名）对学生信息文件进行查询。

诸如此类的还有电话自动查号系统、考试查分系统、仓库库存管理系统等。在这类文档管理的数学模型中，计算机处理的对象之间通常存在着一种简单的线性关系，这类数学模型可称为**线性的数据结构**。

【例 1-2】八皇后问题。在八皇后问题中，处理过程不是根据某种确定的计算法则，而是利用试探和回溯的探索技术求解。为了求得合理布局，在计算机中要存储布局的当前状态。从最初的布局状态开始，一步步地进行试探，每试探一步形成一个新的状态，整个试探过程形成了一棵隐含的状态树，如图1-2所示（为了描述方便，将八皇后问题简化为四皇后问题）。回溯法求解过程实质上就是一个遍历状态树的过程。在这个问题中所出现的树也是一种数据

结构，它可以应用在许多非数值计算的问题中。

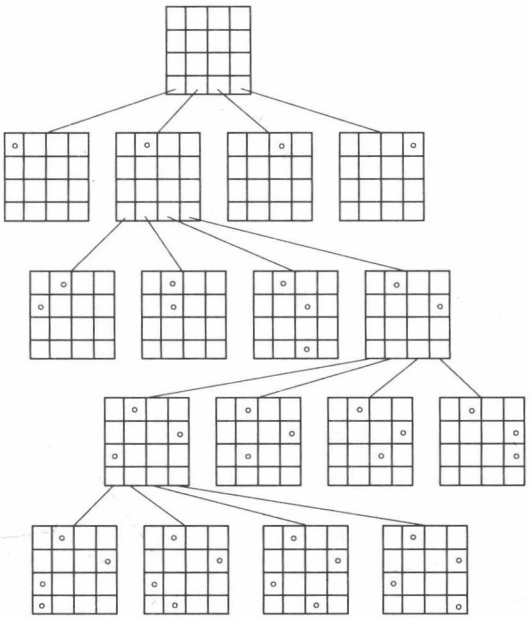


图 1-2　四皇后问题求解形成的树结构

【例 1-3】教学计划编排问题。一个教学计划包含许多课程，在教学计划包含的许多课程之间，有些必须按规定的先后次序进行，有些则没有次序要求（见表 1-1），即有些课程之间有先修和后修的关系，有些课程可以任意安排次序。这种各个课程之间的次序关系可用一个称为图的数据结构来表示，如图 1-3 所示。有向图中的每个顶点表示一门课程，如果从顶点 v_i 到 v_j 之间存在有向边<v_i，v_j>，则表示课程 i 必须先于课程 j 进行。

由以上三个例子可见，描述这类非数值计算问题的数学模型不再是数学方程，而是诸如表、树、图之类的数据结构。因此，可以说数据结构课程主要是研究非数值计算的程序设计问题中所出现的计算机操作对象以及它们之间的关系和操作的学科。

学习数据结构的目的是为了了解计算机处理对象的特性，将实际问题中所涉及的处理对象在计算机中表示出来并对它们进行处理。与此同时，通过算法训练来提高学生的思维能力，通过程序设计的技能训练来促进学生的综合应用能力和专业素质的提高。

表 1-1　计算机专业课程先修关系

课程编号	课程名称	先修课程
C_1	计算机导论	无
C_2	数据结构	C_1，C_4
C_3	汇编语言	C_1
C_4	C 程序设计语言	C_1
C_5	计算机图形学	C_2，C_3，C_4
C_6	接口技术	C_3
C_7	数据库原理	C_2，C_9
C_8	编译原理	C_4
C_9	操作系统	C_2

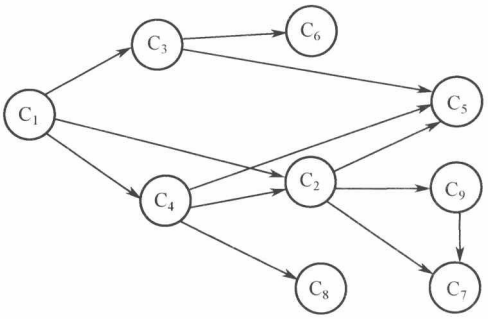


图 1-3　教学计划编排问题的数据结构

1.1.2　有关概念和术语

在系统地学习数据结构知识之前，先对一些基本概念和术语赋予确切的含义。

数据（Data）是信息的载体，它能够被计算机识别、存储和加工处理。它是计算机程序加工的原料，应用程序处理各种各样的数据。计算机科学中，所谓数据就是计算机加工处理的对象，它可以是数值数据，也可以是非数值数据。数值数据是一些整数、实数或复数，主要用于工程计算、科学计算和商务处理等；非数值数据包括字符、文字、图形、图像、语音等。

数据元素（Data Element）是数据的基本单位。在不同的条件下，数据元素又可称为元素、节点、顶点、记录等。例如，学生信息检索系统中学生信息表中的一个记录、八皇后问题中状态树的一个状态、教学计划编排问题中的一个顶点等，都被称为一个数据元素。

有时，一个数据元素可由若干个**数据项（Data Item）**组成，例如，学籍管理系统中学生信息表的每一个数据元素就是一个学生记录。它包括学生的学号、姓名、性别、籍贯、出生年月、成绩等数据项。这些数据项可以分为两种：一种称为初等项，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；另一种称为组合项，如学生的成绩，它可以再划分为数学、物理、化学等更小的项。通常，在解决实际应用问题时把每个学生记录当成一个基本单位进行处理。

数据对象（Data Object）或**数据元素类（Data Element Class）**是具有相同性质的数据元素的集合。在某个具体问题中，数据元素都具有相同的性质（元素值不一定相等），属于同一数据对象（数据元素类），数据元素是数据元素类的一个实例。例如，在交通咨询系统的交通网中，所有的顶点是一个数据元素类，顶点 A 和顶点 B 各自代表一个城市，是该数据元素类中的两个实例，其数据元素的值分别为 A 和 B。

结构（Structure）：为了理解数据结构这个概念，首先应该对“结构（Structure）”这个词汇的含义进行理解，结构这个词一般具有如下几个含义：

① 许多部件组成的事物整体；

② 事物的构造方式及组成部分的排列方式。

那么数据结构是以数据元素作为组成“部件”的一种复杂的数据组织方式。

因此，**数据结构（Data Structure）**是指数据元素及数据元素之间的关系，是复杂数据的一种组织方式。在实际问题中，数据元素之间都不会是孤立的，在它们之间都存在着这样或那样的关系。

1.1.3　数据结构的三要素

程序或算法所处理的数据一定是有某种内在联系的。这种数据之间的内在联系就是数据之间的关系。数据之间的关系可以分成两个方面来看待，即数据的逻辑关系和数据的存储关系。

研究数据结构时，主要从 3 个方面入手，这 3 个方面称为**数据结构的三要素**，即**数据的逻辑结构、数据存储结构、数据操作**（也称为算法）。

1．数据的逻辑结构

数据的逻辑关系指的是客观上数据对象之间所具有的关系，这往往与具体的应用需求有关。比如说，一个班的学生，如果要求将所有学生按学号排列，那么学号由小到大就规定了一种逻辑关系；如果要求按拼音顺序排列，那么每个学生姓氏的拼音就规定了一种逻辑关系。

再比如，一元二次方程求根算法中，对于 3 个系数 a, b, c，尽管在算法或程序中，可以分别用 3 个独立变量表示，但它们在逻辑上仍然是有关系的，即它们同属于一个一元二次方程的 3 个系数。

数据的逻辑结构可以看成从具体问题抽象出来的数学模型，它与数据的存储无关，独立于计算机，它是从具体问题抽象出来的数学模型。

经过人们长期的研究和实践，将客观世界数据对象之间的关系归纳为以下 4 种普遍的逻辑结构。

（1）集合结构：数据元素间除“同属于一个集合”外，无其他关系。

（2）线性结构：数据可以按某种规则排列成线性表的形式，如一个班的学生名单就是一个线性表，这个班的学生之间就是一种线性关系。线性表一般表示成一个线性序列的形式：

$$(a_1, a_2, \cdots, a_i, \cdots, a_n)$$

（3）树形结构：数据之间呈现倒立的树形结构，每个元素有一个双亲，有 0 个或多个孩子。元素之间呈现一对多的关系。

（4）网状结构（图结构）：这是一种最复杂的关系，每个数据元素都可能有多个相邻的数据元素，数据元素之间呈现一种多对多的关系。

图 1-4 表示了上述 4 类基本数据结构。

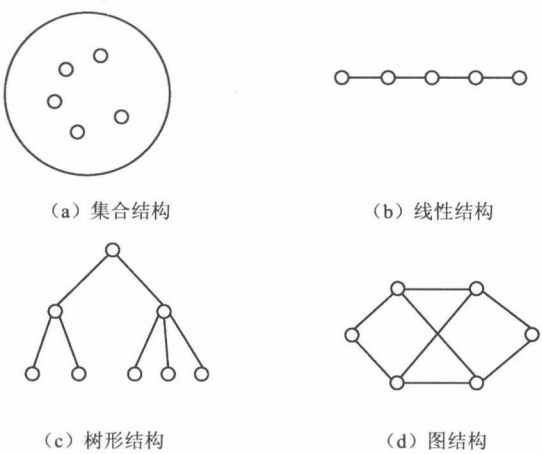


图 1-4　4 类基本数据结构的示意图

从上面所介绍的数据逻辑结构的概念中可以知道，一个数据结构（逻辑结构）有两个要素：一个要素是数据元素的集合，另一个要素是关系的集合。在形式上，数据逻辑结构通常可以采用一个二元组来表示：

Data_Structure ＝（D，R）

其中，D 是数据元素的有限集，R 是 D 上关系的有限集。

数据的逻辑结构可以看成集合论中关系理论的扩展。

2. 存储结构

我们研究数据结构的目的是为了在计算机中实现对它的操作，为此还需要研究如何在计算机中表示一个数据结构。数据结构在计算机中的表示称为数据的物理结构或存储结构。存储结构所研究的是数据结构在计算机中的实现方法，包括数据结构中元素的表示及元素间关系的表示。

算法或程序所要处理的数据最终总是要在计算机存储器中存储。对于少量的数据，我们可以用单个变量表示每一个数据，每个变量在计算机中是如何存储的，我们可以不必关心。但是对于大批量的数据，为了设计一个高效的算法，必须自己定义数据的存储关系。因此除了要研究数据之间的逻辑结构外，更重要的是要研究数据的存储结构。

简单地说，数据的存储结构指的是一批数据在计算机存储器中的存储位置和存储方式，它所研究的是数据的逻辑结构在计算机中的实现方法，包括逻辑数据结构中数据元素的存储及数据元素之间关系的存储。

对存储结构的基本要求是：存储结构必须能够反映数据元素本身及数据元素之间的逻辑关系。

数据的存储结构可采用顺序存储或链式存储的方法。

目前，使用最为广泛的存储结构有以下几种：

（1）**顺序存储结构**：顺序存储方法是把逻辑上相邻的元素存储在物理位置相邻的存储单元中，由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法，顺序存储结构的典型代表是我们已经学过的数组。其基本特点是数据元素一个紧挨着一个地存放。对于逻辑上的线性表（线性结构），采用顺序存储方式时，就称为顺序表。

（2）**链式存储结构**：链式存储对数据元素在存储器中存放位置不做特殊要求，数据元素在存储器中可以随机存放。为了保持数据元素之间的逻辑关系，使用“指针”将每个数据元素联系起来。对于逻辑上的线性表，采用链式存储时，就称为链表。

除了通常采用的顺序存储方法和链式存储方法外，有时为了查找方便还采用索引存储方法和散列存储方法。

同一种逻辑结构可采用不同的存储方法，主要考虑的是运算方便及算法的时空要求。

3. 作用于数据结构上的算法

作用于数据结构上的算法也称为对数据结构的操作。一般来说，不同的数据结构，其算法或操作是不一样的。但是也有几个基本的算法是大多数数据结构操作中经常遇到的，如查找、更新、插入、删除等。

总之，数据的逻辑结构与存储结构密切相关，我们在设计算法思路和框架时，更多地考虑数据的逻辑结构，而在算法实现时，则主要考虑数据的存储结构。

1.2 抽象数据类型

首先来回顾一下在程序设计语言中出现的各种数据类型。

1.2.1 数据类型

数据类型是和数据结构密切相关的一个概念。它最早出现在高级程序设计语言中，用以刻画