面向21世纪课程教材

普通高等教育“十一五”国家级规划教材

数据结构（第2版）

刘大有　虞强源　杨博　王生生　姜丽　等　编著

高等教育出版社
HIGHER EDUCATION PRESS

面向21世纪课程教材

普通高等教育“十一五”国家级规划教材

数据结构(第2版)

Shuju Jiegou

刘大有　虞强源　杨　博　王生生　姜　丽　等　编著

高等教育出版社·北京

HIGHER EDUCATION PRESS　BEIJING

内容提要

本书是国家精品课程“数据结构”的研究成果之一，是面向21世纪课程教材和普通高等教育“十一五”国家级规划教材。本书系统介绍了数据结构的概念、原理与技术，主要内容包括绪论，基本数据结构，排序、查找与内存管理，相关工具和文件等。其中，第一章绪论主要对算法描述语言(ADL)、算法书写规范、数据结构与算法基本概念、算法分析基础和算法正确性证明等进行了介绍；第二至五章是基本数据结构部分，主要涉及线性表、堆栈和队列，数组和字符串，树与二叉树，图结构等内容；第七至九章从算法的视角讨论了排序、查找和内存管理等方面的内容，给出了若干典型算法的描述、时间复杂性分析和相关算法的比较等；第六章和十一章分别对递归和随机数两种主要工具进行了讲解，其中随机数是数据结构的新内容；文件这种复杂的数据结构则在第十章中阐明。

本书的附录主要包括书中ADL算法的C++程序、一些基本数据结构的C++类实现以及习题答案或解题思路。本书配套教学资源(网址：http://computer.cncourse.com)中包括电子教案、ADL算法的C++程序、较难习题答案的C++代码以及相关的测试和运行支持程序，可供读者自学和上机使用。

本书可作为高等学校计算机相关专业的教材和教学参考书，也可供相关专业的工程技术人员参考使用。

图书在版编目(CIP)数据

数据结构/刘大有等编著. —2版. —北京：高等教育出版社，2010.9

ISBN 978-7-04-030213-4

Ⅰ. ①数… Ⅱ. ①刘… 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144478号

策划编辑 倪文慧　　**责任编辑**

版式设计 张　岚　　**责任校对**

出版发行	高等教育出版社	**购书热线**	010-58581118
社　　址	北京市西城区德外大街4号	**咨询电话**	400-810-0598
邮政编码	100120	**网　　址**	http://www.hep.edu.cn http://www.hep.com.cn
经　　销	蓝色畅想图书发行有限公司	**网上订购**	http://www.landraco.com http://www.landraco.com.cn
印　　刷	保定市中画美凯印刷有限公司	**畅想教育**	http://www.widedu.com
开　　本	787×1092　1/16	**版　　次**	1999年6月第1版 2010年9月第2版
印　　张	41.75	**印　　次**	2010年9月第1次印刷
字　　数	950000	**定　　价**	56.00元

物料号　30213-00

郑重声明

前　言

数据结构与算法紧密相关。算法是求解问题的步骤，它必须是确定的（无歧义）、可行的、有限的。此外，它可以有零个或多个输入，但必须有输出。注意不要将算法和计算机程序等同起来，后者可作为描述前者的手段之一。除了使用算法描述语言描述算法之外，还可用流程图、形式语言、程序设计语言，甚至自然语言等对算法进行描述。但不管怎么样，“算法是贯穿在所有计算机程序设计中的一个基本概念”（D. E. Knuth，1997）、“算法是计算机科学的灵魂”（Umesh Vazirani，2008）等观点，却是计算机界众多学者所认同的。

算法学是系统研究算法的一门科学。通常，它主要包括算法设计、算法正确性证明及算法分析三个部分。算法设计是创建算法的过程，它研究良好的创建方法以及算法与数据结构之间的关系和作用；算法或其关键步骤正确性证明的基本途径是数学归纳法；算法时间复杂性分析是算法分析的重要内容之一。在算法的时间复杂性分析方面，有时只得到复杂性的阶是不够的，要给出更精确的分析结果，还要知道复杂性的阶前面的系数。

自 20 世纪 60 年代初开始，计算机越发频繁地用于解决一些非数值分析问题。解决这些问题主要使用计算机的逻辑判断能力而非其算术运算能力。当然，“非数值分析”实际上是关于这一研究领域的一个极其负面的称谓，最好应有一个正面的名称。“信息处理”太过宽泛，而“程序设计技术”又显得过窄，称其为“计算机算法分析”似乎恰当一些。

正是由于数据结构与算法密不可分，当人们说算法非常重要的时候，实际上也就是在说数据结构非常重要。数据结构是计算机算法设计的基础，它在计算机科学中占有十分重要的地位。正如霍尔（C. A. R. Hoare）在 1972 年所阐明的：不了解施加于数据上的算法就不知道怎样去构造数据结构，反之，若不深入研究作为其基础的数据结构，则算法的优美结构与设计将无从谈起。

实际上，数据结构和算法均与数学有密切关系。下面用一个问题求解的过程来简要说明数据结构知识及相关知识的结合应用，以及对数学进行严格训练的必要性。同时，我们还希望通过数据结构的学习使读者对科学研究方法有一个初步的了解。

一个问题求解的大致过程如下。

（1）运用数学知识（如数学分析、代数、离散数学、组合数学、运筹学、概率论等）对待求解问题进行数学建模。

（2）运用数据结构、算法设计等知识设计一个好的计算机算法。

（3）运用数学知识（如离散数学知识等）对算法或算法的关键步骤的正确性进行证明。用算法分析知识（包括许多数学知识）等对算法的时空复杂性等进行分析，进而对算法进行优化。

（4）运用程序设计与程序测试知识完成算法的编程与测试，对测试结果给出分析，以便对改

进算法、重新选择数据结构或构造新的数据结构等给出反馈信息。

本书的编写思路和主要特点如下。

1. 用算法描述语言(ADL)和C++程序设计语言描述算法

用面向对象的程序设计语言描述施加于数据结构之上的算法,不仅有利于面向对象技术与数据结构知识的结合,而且也为上机实践提供了方便。但若换一个角度思考,这一做法也有不足之处——用高级程序设计语言描述算法不仅要占用较多篇幅,而且还会涉及较多的程序设计细节。因此,算法的要点有可能湮没于过多的细节之中,不利于读者将其注意力集中于数据结构与算法的关键部分。基于这些考虑,在正文中我们采用一种抽象算法描述语言(ADL)来描述算法,而将算法的C++实现放到附录中。

2. 突出启发式教学与因材施教

我们认为实现启发式教学和因材施教的关键是着力设计启发式教学和因材施教的内容。为此,针对一些较难的知识点,书中设计了启发式教学案例,在相关算法的衔接处阐明选择相关算法的理由及相关算法的特点,在展开算法描述之前给出算法的关键思想的刻画,在每章的结尾处给出相关算法的比较。

下面给出一个关于有序表查找的启发式教学的例子。

(1) 若实现从顺序查找算法到对半查找算法B的"跳跃",必须实现:① 从只考虑$K=K_i$和$K\neq K_i$两种情况到考虑$K=K_i$、$K>K_i$和$K<K_i$三种情况;② 从每次比较下一个关键词到每次比较被查找的子文件的中点。

(2) 算法B运行时间的阶是多少?对于成功查找算法B最多需要多少次比较?这些问题的答案可通过启发性的提示由读者来给出。例如,每经过一次比较,待查找文件之长度将减半。

(3) 若实现从算法B到一致对半查找算法U的"跳跃",需要实现一个思想转弯,即从使用l、u、i这3个指针到仅使用i和m两个指针($m\leftarrow\lfloor m/2\rfloor$,下次被查找子文件长度折半)。

(4) 若实现从算法U到一致对半查找算法C的"跳跃",需要想到用一张表存放m值。这样,算法运行时省去计算m的时间,是用空间换时间。

(5) 当对半查找算法的改进到了"山重水复疑无路"的时候,能启发读者跳出对半二叉查找树的局限转而去探索新的二叉查找树——斐波那契(Fibonacci)二叉查找树,即斐波那契查找算法F。

3. 重视算法时间复杂性分析

对经典算法或给出了时空复杂性分析,或给出了分析结果。特别是,对时间复杂性阶相同的一批相关算法,不但给出了算法复杂性的阶,而且还给出了阶前面的系数。

例如,在第八章中,对经典算法给出了严格时间复杂性分析。为了对一些相关算法进行比较,给出了时间复杂性分析的精确结果,如下表所示。

	对于一次成功的查找	对于一次不成功的查找
算法B的近似平均运行时间	$(18\log_2 N-16)u$	$(18\log_2 N+12)u$

续表

	对于一次成功的查找	对于一次不成功的查找
算法 C 的近似平均运行时间	$(8.5\log_2 N-6)u$	$(8.5\log_2 N+12)u$
算法 F 的近似平均运行时间	$(7.05\log_2 N+1.08)u$	$(7.05\log_2 N+5.23)u$

由上表可看出，如果只考虑算法复杂性的阶，则难以分辨算法 B、C、F 的优劣。

4. 将科研成果转化为教材内容

我们在第五章中增加了两方面内容。其一是复杂网络。客观世界中的很多系统或待求解的问题都可抽象成一个图结构。由于这些系统和问题本身具有很高的复杂性，因此将由它们抽象成的图结构称为“复杂网络”，如社会网络、生物网络以及人们熟知的电网、交通网、电信网和万维网等技术网络。科学家已经发现，尽管客观世界中复杂系统各不相同，但它们对应的复杂网络结构却惊人地相似。这方面的内容将启发读者将图结构与相关的新兴研究方向联系起来。其二是基于图的信息搜索算法 PageRank(谷歌搜索引擎的核心算法)。它是近年来图结构最成功的应用实例之一，并被普遍认为是信息检索和数据挖掘领域中最具代表性的算法之一。这部分内容有助于读者深入了解图结构的重要性，激发他们学习图结构的兴趣，并启发他们面向应用设计出更有效的图算法。

5. 增加了“随机数”一章

增加的主要理由有两个。其一是，早在 40 年之前，著名计算机科学家 D. E. Knuth 就出版了随机数方面的专著(《The Art of Computer Programming》第二卷《半数值算法》，由两章组成，“随机数”是第一章)，详细阐述了随机数与半数值算法、计算机程序设计的紧密关系；其二是，近年来随机数在诸如密码学、仿真、抽样、数值分析、计算机程序设计、决策、机器学习、数据挖掘、贝叶斯网、进化计算、美学和娱乐等领域都得到了广泛应用。

6. 背景、历史与参考文献

为使读者能全面了解和掌握每一章的内容，本书在很多章都增加了与重要知识点相关的背景、历史和参考文献，描述了知识点的重要应用，给出了相关的应用实例。例如，在本书第五章中增加了图的出现、发展与应用的内容。第八章中，在各节的末尾增加了本节所述内容的历史与文献。

7. 总结和对比

每章的结尾都对重要知识点、算法等进行了总结，或对本章的相关算法进行了分析、对比。例如，第九章的小结中分析了本书描述的动态存储分配算法，介绍了模拟实验结果，并与其他相关算法进行了对比，希望让读者从多个视角观察问题，以便对动态存储分配算法及其研究有全面、深入的了解。

8. 强调数学上的严格

本书重视内容的严谨性，并试图使读者受到严格的数学训练。具体做法包括：对书中与某些算法的正确性问题以及与算法复杂性分析或数据结构概念相关的重要定理、引理等都给出了严

格的数学证明，对主要概念都给出严格的形式化定义。

例如，一致对半查找算法C的正确性依赖于每次用来找中点的m的序列的严格推证，m的序列如下：

$$\lfloor (N+2^0)/2^1 \rfloor, \lfloor (N+2^0)/2^1 \rfloor +/- \lfloor (N+2^1)/2^2 \rfloor,$$
$$\lfloor (N+2^0)/2^1 \rfloor +/- \lfloor (N+2^1)/2^2 \rfloor +/- \lfloor (N+2^2)/2^3 \rfloor, \cdots$$

9. 习题与因材施教

每章之后都附加了精选的习题，并给出了每道习题的难度等级：【5】级是最高的难度级别；对于【3】级以上难度的习题，还给出了解答习题所需的平均时间；对用到高等数学知识的习题进行标记；对于【3】级以上难度的习题，一般给出了提示或分级提示（给出分级提示的习题通常是有相当难度的题目，如果读了第一级提示后仍然没有解题思路，才去读第二级提示，以此类推）。

同时，对大多数较难的习题均在附录中给出了答案或解题思路。为方便读者上机实践，对较难的习题都编写了C++程序，并将其放入本书的配套教学资源中，读者可从 http://computer.cncourse.com 下载。

本书由刘大有主笔，虞强源和杨博参加了部分章节的统稿。吉林大学“数据结构”课程教学团队中的其他一些主要成员，包括王生生、刘亚波、孙成敏、孙舒杨、李嘉菲、欧阳继红、姜丽和谢琦等（按姓氏笔画为序），在“数据结构”课程建设以及本书撰写，习题选择、解答与相关C++程序设计等方面做了大量的工作，在此对他们的辛勤劳动表示诚挚的谢意。

鉴于计算机科学技术的飞速发展以及时间和水平所限，书中错误及不足在所难免，敬请专家和读者批评指正。编者联系方式为 liudy@jlu.edu.cn。

刘大有

2010年7月

目　录

第一章　绪论 …… 1
1.1　为什么要学习数据结构 …… 1
1.2　数据结构概念 …… 2
1.2.1　数据的逻辑结构 …… 3
1.2.2　数据的存储结构 …… 4
1.2.3　对数据结构的操作 …… 6
1.2.4　数据结构示例 …… 6
1.3　算法 …… 6
1.3.1　算法及其特性 …… 6
1.3.2　算法的描述 …… 7
1.3.3　算法的评价准则 …… 9
1.4　算法的正确性证明 …… 10
1.5　算法分析基础 …… 13
1.5.1　算法时间复杂性的分析方法 …… 13
1.5.2　复杂性函数的渐近表示 …… 16
1.5.3　算法时间与空间分析 …… 18
1.5.4　计算复杂性和算法的效率 …… 19
小结 …… 20
参考文献与推荐读物 …… 21
习题 …… 22
第二章　线性表、堆栈和队列 …… 24
2.1　线性表的定义和基本操作 …… 24
2.2　线性表的顺序存储结构 …… 24
2.3　线性表的链接存储结构 …… 27
2.3.1　单链表 …… 27
2.3.2　循环链表 …… 32
2.3.3　双向链表 …… 33
2.4　复杂性分析 …… 36
2.5　堆栈 …… 36
2.5.1　堆栈的定义和基本操作 …… 36
2.5.2　顺序栈 …… 37
2.5.3　链式栈 …… 39
2.5.4　顺序栈与链式栈的比较 …… 40
2.5.5　堆栈应用——括号匹配 …… 41
2.6　队列 …… 42
2.6.1　队列的定义和基本操作 …… 42
2.6.2　顺序队列 …… 43
2.6.3　链式队列 …… 45
2.6.4　顺序队列与链式队列的比较 …… 47
2.6.5　队列与堆栈的扩展 …… 47
小结 …… 48
参考文献与推荐读物 …… 48
习题 …… 49
第三章　数组和字符串 …… 52
3.1　数组 …… 52
3.1.1　数组的存储和寻址 …… 52
3.1.2　一维数组类 …… 54
3.2　矩阵 …… 55
3.2.1　矩阵类 …… 55
3.2.2　特殊矩阵 …… 57
3.2.3　三元组表 …… 59
3.2.4　十字链表 …… 61
3.3　字符串 …… 64
3.3.1　字符串的定义与字符串类 …… 64
3.3.2　模式匹配算法 …… 67
小结 …… 71
参考文献与推荐读物 …… 72
习题 …… 73
第四章　树 …… 76
4.1　树的基本概念 …… 76

4.1.1 树的定义 …… 76
4.1.2 树的相关术语 …… 77
4.2 二叉树 …… 79
4.2.1 二叉树定义和主要性质 …… 79
4.2.2 二叉树顺序存储 …… 83
4.2.3 二叉树链接存储 …… 84
4.2.4 二叉树遍历 …… 86
4.2.5 创建二叉树 …… 92
4.2.6 复制二叉树 …… 93
4.3 线索二叉树 …… 94
4.3.1 线索二叉树定义 …… 94
4.3.2 线索二叉树存储 …… 95
4.3.3 线索二叉树基本算法 …… 97
4.4 树和森林 …… 104
4.4.1 树与二叉树的转换 …… 104
4.4.2 树的顺序存储 …… 108
4.4.3 树的链接存储 …… 110
4.4.4 树和森林的遍历 …… 115
4.5 压缩与哈夫曼树 …… 117
4.5.1 文件编码 …… 117
4.5.2 扩充二叉树 …… 118
4.5.3 哈夫曼树和哈夫曼编码 …… 119
4.6 应用 …… 122
4.6.1 表达式求值 …… 122
4.6.2 分类与决策树 …… 124
小结 …… 128
参考文献与推荐读物 …… 128
习题 …… 129
第五章 图 …… 131
5.1 图的基本概念 …… 132
5.2 图的存储结构与类定义 …… 134
5.2.1 存储结构 …… 134
5.2.2 Graph 类 …… 137
5.3 图的遍历算法 …… 140
5.3.1 深度优先遍历 …… 140
5.3.2 广度优先遍历 …… 142
5.4 拓扑排序 …… 143
5.5 关键路径 …… 147
5.6 最短路径问题 …… 152
5.6.1 无权最短路径问题 …… 153
5.6.2 正权最短路径问题 …… 154
5.6.3 每对顶点之间的最短路径 …… 158
5.7 最小支撑树 …… 160
5.7.1 普里姆算法 …… 161
5.7.2 克鲁斯卡尔算法 …… 164
5.8 图的应用 …… 165
5.8.1 可及性与 Warshall 算法 …… 165
5.8.2 连通分量 …… 166
5.8.3 图在网络分析和信息检索中的应用 …… 167
小结 …… 171
参考文献与推荐读物 …… 172
习题 …… 173
第六章 递归 …… 177
6.1 递归的定义 …… 177
6.2 基本递归过程 …… 180
6.3 递归过程实现与堆栈 …… 182
6.4 递归法求解问题 …… 186
6.4.1 委员会问题 …… 186
6.4.2 回溯 …… 188
6.5 递归的效率 …… 191
小结 …… 193
参考文献与推荐读物 …… 194
习题 …… 194
第七章 排序 …… 198
7.1 排序问题的基本概念 …… 198
7.2 插入排序 …… 200
7.2.1 直接插入排序 …… 200
7.2.2 Shell 排序 …… 202
7.3 交换排序 …… 204

7.3.1 冒泡排序 …… 204
7.3.2 快速排序 …… 207
7.4 选择排序 …… 215
7.4.1 直接选择排序 …… 215
7.4.2 堆排序 …… 215
7.5 合并排序 …… 220
7.6 基于关键词比较的排序算法分析 …… 223
7.6.1 平方阶排序算法及改进算法 …… 223
7.6.2 线性对数阶排序算法 …… 224
7.6.3 分治排序的一般方法 …… 225
7.6.4 基于关键词比较的排序算法下界 …… 226
7.7 分布排序 …… 227
7.7.1 基数分布 …… 228
7.7.2 值分布 …… 230
7.8 外排序 …… 231
7.8.1 外存储器 …… 231
7.8.2 磁带排序 …… 232
7.8.3 磁盘排序 …… 241
小结 …… 245
参考文献与推荐读物 …… 246
习题 …… 247
第八章 查找 …… 253
8.1 顺序查找 …… 254
8.1.1 无序表的顺序查找 …… 254
8.1.2 有序表的顺序查找 …… 256
8.2 基于关键词比较的查找 …… 256
8.2.1 对半查找 …… 257
8.2.2 一致对半查找 …… 260
8.2.3 斐波那契查找 …… 263
8.2.4 插值查找 …… 266
8.3 二叉查找树 …… 268
8.3.1 基本概念和性质 …… 268
8.3.2 查找、插入和删除 …… 269
8.3.3 平均情况时间分析 …… 273
8.4 最优二叉查找树 …… 274
8.4.1 访问频率 …… 274
8.4.2 最优二叉查找树 …… 275
8.4.3 近似最优树的构造 …… 281
8.5 平衡树 …… 284
8.5.1 高度平衡树 …… 285
8.5.2 重量平衡树 …… 294
8.6 红黑树 …… 300
8.6.1 红黑树的性质 …… 300
8.6.2 旋转 …… 301
8.6.3 插入 …… 303
8.6.4 删除 …… 306
8.7 B树及其变形树 …… 311
8.7.1 多叉树 …… 311
8.7.2 B树 …… 312
8.7.3 B树变形树 …… 318
8.8 数字查找 …… 323
8.8.1 检索结构查找 …… 323
8.8.2 数字树查找 …… 330
8.9 散列 …… 332
8.9.1 散列函数 …… 333
8.9.2 冲突调节 …… 337
8.9.3 删除 …… 346
8.9.4 重量平衡树的应用——按位置查找 …… 347
小结 …… 347
参考文献与推荐读物 …… 349
习题 …… 353
第九章 内存管理 …… 357
9.1 概述 …… 357
9.2 均匀大小记录的分配和回收算法 …… 359
9.2.1 记录分配算法 …… 359
9.2.2 访问计数器法 …… 360

9.2.3 废料收集方法 …… 362
9.3 不同大小记录的分配和回收算法 …… 367
9.3.1 查找分配策略 …… 368
9.3.2 边界标识法 …… 371
9.3.3 压缩分配 …… 374
9.4 伙伴系统 …… 376
9.4.1 伙伴系统概述 …… 376
9.4.2 分配记录和释放记录算法 …… 379
小结 …… 381
参考文献与推荐读物 …… 383
习题 …… 385
第十章 文件 …… 388
10.1 文件的基本概念 …… 388
10.1.1 文件及其分类 …… 388
10.1.2 文件的逻辑结构与存储结构 …… 390
10.2 顺序文件 …… 391
10.2.1 顺序无序文件 …… 393
10.2.2 顺序有序文件 …… 396
10.2.3 增补文件 …… 397
10.3 散列文件 …… 398
10.3.1 散列文件 …… 398
10.3.2 可扩充的散列文件 …… 399
10.4 索引文件 …… 403
10.4.1 动态索引结构和静态索引结构 … 404
10.4.2 ISAM 文件 …… 405
10.4.3 VSAM 文件 …… 409
10.5 多关键字文件 …… 412
10.5.1 多重链表文件 …… 413
10.5.2 倒排文件 …… 416
小结 …… 416
参考文献与推荐读物 …… 417
习题 …… 417
第十一章 随机数 …… 419
11.1 生成随机数 …… 419
11.1.1 均匀分布随机数 …… 420
11.1.2 其他分布随机数 …… 427
11.2 随机数检验 …… 429
11.2.1 一般检验方法 …… 430
11.2.2 经验检验方法 …… 434
11.3 随机排列与随机组合 …… 436
11.3.1 随机排列 …… 436
11.3.2 随机组合 …… 437
11.4 应用 …… 439
11.4.1 随机算法 …… 439
11.4.2 使用随机数的快速排序算法 …… 441
小结 …… 441
参考文献与推荐读物 …… 441
习题 …… 442
附录 …… 444
附录 1 各章算法的 C++实现 …… 444
附录 2 习题参考答案或解题思路 …… 579

1 第一章 绪 论

1.1 为什么要学习数据结构

“数据结构”是计算机科学技术学科的一门重要的专业基础课程，它可为学习后续的专业课程提供必要的知识和技能准备。例如，计算机科学技术学科的后续专业课程“编译技术”中要使用堆栈(Stack)、散列表(Hash Table)和语法树等，“操作系统”中会用到队列(Queue)、存储管理和目录树等，“数据库系统”中将涉及线性表(Linear Table)、多链表和索引树等基本数据结构及相关算法。除上述数据结构之外，排序、查找、图、递归和随机数等在问题求解中都有重要应用。在计算机科学中，排序或许是研究得最多的运算，排序实际上是程序员应当掌握的应用广泛的一项基本工具。从大的方面观察，可将查找看作“信息的存储和检索”。每个数据或媒体中心，检索和存储无时无刻不在进行。从小的方面着手，可将查找看作“查表”。查找是许多程序中最耗费时间的部分，用好的查找算法替代差的查找算法可显著提高程序的运行速度。许多引起人们关注的问题都可以用图来建模，最短路径计算成了计算机科学中的基本应用，近来的复杂网络研究非常活跃，它与图关系密切。递归是功能强大的问题求解工具，许多算法使用递归公式可获得最简单的表达，而且对许多问题来说最有效的解决方案就是以这种自然递归公式为基础的。随机数有许多应用，如算法的实验研究、现实系统的模拟以及设计用概率回避最坏情况的算法(如仿生算法)等。

“数据结构”课程的目的就是从对问题抽象和求解的角度来介绍常用的数据结构，阐明其内在的逻辑关系、在计算机中的存储表示，以及刻画施加于其上之各种操作的算法。学习上述理论知识，综合运用相关知识，再结合上机实践，将使读者具备求解比较复杂的问题的能力，即掌握问题求解建模，选择恰当数据结构，或构造新的数据结构，设计较优算法，证明算法(或算法关键步骤)正确性，分析算法的时空复杂性，对算法编程、调试等方面的能力。

本章将讨论关于数据结构的预备知识，如数据和数据结构的概念、算法的概念、算法描述语言、算法分析基础以及算法的正确性证明等。

1.2 数据结构概念

为了使读者对数据结构的发展过程有一个基本的了解，下面对数据结构的发展历史做简要介绍[1,2]。

20 世纪 40 年代，电子计算机刚刚诞生，当时的计算机只能处理数值计算问题，涉及的数据非常简单，数据量小且形式统一，尚不存在数据结构方面的问题。

20 世纪 50 年代末，计算机开始大量应用于解决非数值计算问题，其加工的数据也从简单的数值发展到复杂数据。这个时期，随着各种高级程序设计语言的出现，程序设计语言支持的数据类型逐渐增多，产生了数组、记录、串和层次表结构等新的数据结构。

20 世纪 60 年代，美国计算机界提出了“信息结构”的概念，它可认为是数据结构概念的雏形。1968 年，美国计算机学会(ACM，Association for Computing Machinery)发布了建议性的计算机教学计划，首次将数据结构列为一门独立的课程。尽管 ACM 把数据结构规定为一门课程，但对课程的范围却没有作明确规定。当时的数据结构几乎是表、树、图论的同义语，甚至还包括了集合代数论、格、关系等离散数学的内容。那时的数据结构还未成为一门系统和相对独立的课程。直到著名计算机科学家克努斯(D. E. Knuth)陆续出版了包括《基本算法》、《半数值算法》和《排序和查找》等三卷在内的旷世之作《计算机程序设计技巧》后，人们才第一次对数据结构有了全面、深刻的认识。该书系统地论述了数据的逻辑结构和存储结构，并给出了各种典型算法和算法复杂性分析等，为数据结构的发展奠定了理论基础。

此后的工作进一步阐明了程序、算法以及数据结构之间的密切关系，其中的代表性工作有：1972 年，著名计算机科学家霍尔(C. A. R. Hoare)在其论文《数据结构札记》中，澄清了关于数据结构术语和概念等方面的杂乱局面，并深刻论述了算法与数据结构密不可分的关系；1976 年，著名计算机科学家沃思(N. Wirth)出版了名为《算法＋数据结构＝程序》的专著，不仅形象地描述了数据结构、算法与程序之间的关系，还明确地提出数据结构和算法对程序设计的重要性。

数据结构的内容随着计算机科学技术的发展在不断更新和完善。

20 世纪 70 年代后期，随着数据库技术的成功应用，数据结构相应地增加了文件组织、存储和管理等方面的内容。

20 世纪 80 年代，随着面向对象概念和面向对象技术的兴起，数据结构增加了抽象数据类型等概念，产生了一种关于数据结构的新看法。这种看法认为数据结构是对数据组织形式的模拟，将数据集视为一种用户可访问的抽象工具，构造这种抽象工具的研究可能使人们从传统的数据结构过渡到面向对象的范型(Object-Oriented Paradigm)。

下面给出有关的基本概念和术语。

定义 1.1 **数据**是指对象的表示，即按照适合于通信、解释或处理(借助人或自动装置)的方式所形成的关于事实、概念或指令的表示；数据只是表示，而无含义[3]。

数据是计算机科学技术中最基本的概念，它是计算机程序要处理的“原料”，是所有被计算机

识别、存储和加工处理的符号的总称[4,5]。数据不仅指整数和实数，计算机可以处理的字符串、图像、声音等都被称为数据。例如，一个简单的数值计算程序所使用的数据是一些实数或整数，一个编译程序所使用和加工的数据是源程序，而一个能修改自身的计算机程序所使用和加工的数据(对象)就是其自身。

数据元素(或称数据成分)还可被称为元素、结点或顶点等。数据元素可大可小，大到一篇文稿、一本书，小到一个字符，甚至是计算机二进制数中的一位。数据元素可由若干个数据项(也称为域或字段)组成。

例 1.1 在如表 1.1 所示的通讯录表中，行表示一个结点(数据元素)，每个结点由“姓名”、“区号”、“办公电话”和“手机”4 个域(数据项)组成。

表 1.1 通讯录表

姓　名	区　号	办公电话	手　机
赵一	010	53644587	13911001234
钱二	020	89634159	13809771333
孙三	021	45976528	13916586666
李四	024	63427541	13804076111

定义 1.2 数据结构[3]是指由若干数据成分按照一定方式构成的复合数据以及作用于其上的函数或运算。数据成分及其间的数据约束关系合称为数据结构的逻辑结构。数据结构在数学上可用适当的数学结构以及其上的函数变换统一地定义。

迄今为止，数据结构的定义在计算机科学技术界还未取得完全认同。有些学者认为数据结构应由数据的逻辑结构、数据的存储结构及其运算三部分组成。

1.2.1 数据的逻辑结构

数据的逻辑结构从比较抽象的角度刻画数据结构所具有的数学性质[3]，将数据元素抽象为结点，数据元素之间的关系作为连接结点的边。

一个逻辑结构可形式定义为一个二元组[5] $L=(N,R)$。其中，N 是结点的有限集合，R 是定义在集合 N 上的二元关系 r 的集合。

设 $L=(N,R)$是一个逻辑结构。$r_1 \in R$ 是与线性结构、树结构和二叉树结构对应的一种关系。若 $a,b \in N$，且$(a,b) \in r_1$，则称 a 是 b 的前驱结点，b 是 a 的后继结点，a 和 b 是相邻结点；若不存在 $a \in N$ 使得$(a,b) \in r_1$，则称 b 为始结点；若不存在 $b \in N$ 使得$(a,b) \in r_1$，则称 a 为终结点。$r_2 \in R$ 是与图结构对应的一种关系。若 $a,b \in N$，且关系$(a,b) \in r_2$，则称 a 和 b 是相邻结点；对于有向图结构，若存在$(a,b) \in r_2$，则称 a 是 b 的前驱结点，b 是 a 的后继结点。

讨论逻辑结构的分类一般以关系集合 R 中关系 r 的分类为主。数据的逻辑结构可分为两大类[5]。

(1) 线性结构

其特点是:若线性结构的结点数为 1,则该结点既是始结点又是终结点;若线性结构的结点数为 2,则有且仅有一个始结点和一个终结点,始结点有一个后继结点(简称后继),终结点有一个前驱结点(简称前驱),中间结点(非始结点、非终结点)有且仅有一个前驱结点和一个后继结点。

该结构的关系 r 是一种线性关系,也称为前后关系、大小关系等。

(2) 非线性结构

其特点是:结构中的结点可能有多个前驱结点和多个后继结点(在数据结构的含义下,多个是指有限多个)。

树(这里是从层次结构的角度考虑的)和图是两种主要的非线性结构。

树也称为树形结构、树结构等。树中有且仅有一个没有前驱结点的结点,称为根结点。其他结点都仅有一个前驱结点,但允许有多个后继结点,从根结点到任一非根结点,有且仅有一条路径。树中的关系 r 也称为层次关系、父子关系等。

图也称为图结构、网络结构等。图中任意结点的前驱和后继结点的个数可以是 0 个或多个。图中的关系 r 也称为相邻关系。

有些书也将数据的逻辑结构分为集合、线性结构、树和图 4 种类型[2,7],如图 1.1 所示。其中,集合的特点是结构中的数据元素之间除了"同属于一个集合"的关系外,别无其他关系。

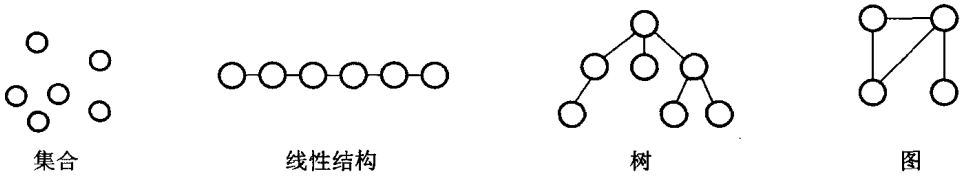

图 1.1 4 种基本数据结构

遵循由简单到复杂、由易到难的原则,在后续第二、四、五章中分别对线性结构、树和图进行具体介绍,并给出相应的算法和应用。

1.2.2 数据的存储结构

在计算机存储器中,人们希望能利用高级程序设计语言通过数据结构的存储结构来实现其逻辑结构。

数据的存储结构(或称物理结构)是指数据的逻辑结构在计算机中所需的存储空间、空间的构成结构以及对该存储结构的访问方式等的总称[3]。

数据的存储结构建立一种由逻辑结构到存储结构的映射[6]:对于逻辑结构 $L=(N,R)$,$r\in R$,建立结点集合 N 到存储区域 M 的映射 $N\to M$。其中,每个结点 $j\in N$ 都对应唯一的连续存储单元 $c\in M$。对于每一个关系元组 $(a,b)\in r$,映射为存储单元的地址顺序关系(或指针的地址指

向关系)。

一般,基本存储映射方法有顺序、链接、索引和散列等 4 种。

1. 顺序存储

顺序存储将一组结点存放在地址相邻的存储单元内,结点间的逻辑关系由存储单元的自然顺序关系来表达,即用一块无空隙的存储区域存储结点数据。例如,用数组存储线性数据结构,数组的元素是数据结点,数据结点按照顺序存储方法存储,结点之间的线性关系用地址单元的顺序关系来自然表达。

顺序存储为使用整数编码访问数据结点提供了便利。当结点等长时,顺序存储方式可以通过地址计算随机存取每个结点。顺序存储通常用于线性数据结构的存储,借助高级程序设计语言中的数组来实现。顺序存储结构被称为紧凑存储结构,紧凑性是指除了存储有用的数据外,不存储其他的附加信息。紧凑性有时可用存储密度来度量。在第二章将详细介绍顺序存储。

2. 链接存储

链接存储通过在结点的存储结构中附加指针字段来存储结点间的逻辑关系,任意的逻辑关系 r 都可以用指针地址来表达。其中,数据结点一般由数据字段和指针字段两部分组成。数据字段存放结点本身的数据,指针字段存放指向后继结点的指针。

链接存储灵活性很大,适用于那些需要经常动态变化(插入、删除等操作)的数据结构,通常借助高级程序设计语言中的指针类型来实现。在第二章将详细介绍链接存储。

3. 索引存储

索引存储可以看作顺序存储方法的一种推广,通过定义一个由整数域 Z 映射到存储地址域的函数,把整数索引值映射到结点的存储地址,从而形成一个存储一串指针的索引表,每个指针指向存储区域的一个数据结点。

索引方法是程序设计中经常使用的方法。将大小不等的数据结点顺序存储后,用索引方法仍可通过整数下标来间接访问数据结点的位置。对于非顺序存储结构来说,使用索引表是快速地经由整数索引值找到其对应的数据结点的唯一方法。在 8.7 节、10.4 节将详细介绍索引存储的相关内容。

4. 散列存储

散列法利用散列函数(Hash Functions)进行索引值的计算,然后通过索引表求出结点的指针地址,是索引方法的一种延伸和扩展。

散列存储对应于高速检索的结构。散列存储的关键问题是如何恰当地选择散列函数,构建散列表,并解决在构建散列表时的"冲突"问题。在 8.9 节、10.3 节将详细介绍散列存储的相关内容。

以上介绍的 4 种存储方法既可单独使用,又可组合起来对数据结构进行存储。例如,在图的邻接表(顶点表—边表)表示法中,顶点表一般选用顺序存储方式,而边表则常用链接存储方式。算法既取决于所选定的逻辑结构,又依赖于所采用的存储结构。同一个逻辑结构采用不同的存储方式,可以得到不同的存储结构。在选择存储结构时,一定要考虑是否能正确反映其对应的逻辑结构,是否能更好地满足相应算法的时空复杂性要求等。