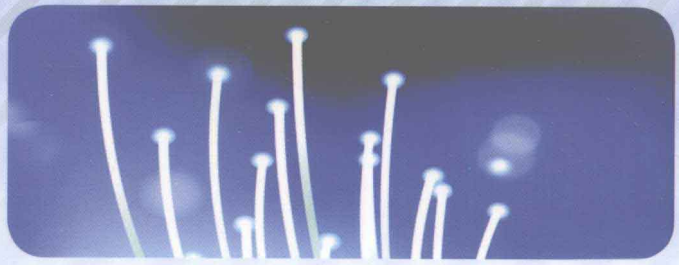
全国教育科学“十一五”规划课题研究成果

数据结构

Data Structure

主　编　魏振钢
副主编　汪桂兰



高等教育出版社
HIGHER EDUCATION PRESS

全国教育科学“十一五”规划课题研究成果

数据结构

Shuju Jiegou

主　编　魏振钢
副主编　汪桂兰

高等教育出版社·北京
HIGHER EDUCATION PRESS　BEIJING

内容提要

“数据结构”是计算机专业的一门重要的基础课程，它对于学生程序设计能力的培养起着非常重要的作用。

本书是全国教育科学“十一五”规划课题研究成果，针对应用型本科计算机及相关专业而编写，以基本数据结构和算法设计为知识单元，系统地介绍了数据结构的知识与应用、算法的设计与分析方法。本书的主要内容包括线性表、串、栈与队列、数组与广义表、树、图、查找、内部排序和文件等，每章均配有大量的习题和模拟题，并安排相关的案例和实验题目，可供学生实习使用。

本书注重理论与实践相结合，内容深入浅出，可以作为计算机类或信息类相关专业的本科或专科教材，也可供从事计算机工程与应用工作的科技工作者参考。

图书在版编目(CIP)数据

数据结构/魏振钢主编. —北京：高等教育出版社，
2011.2
ISBN 978-7-04-020932-7

Ⅰ. ①数… Ⅱ. ①魏… 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(2011)第 011574 号

策划编辑	刘 茜	**责任编辑**	康兆华	**封面设计**	张雨微	**责任绘图**	尹 莉
版式设计	余 杨	**责任校对**	刘 莉	**责任印制**	张泽业		

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
		网上订购	http://www.landraco.com http://www.landraco.com.cn
经 销	蓝色畅想图书发行有限公司	**畅想教育**	http://www.widedu.com
印 刷	三河市春园印刷有限公司		
开 本	787×1092 1/16	**版 次**	2011 年 2 月第 1 版
印 张	22.5	**印 次**	2011 年 2 月第 1 次印刷
字 数	550 000	**定 价**	32.80 元

物料号 20932-00

前　言

数据结构是程序设计的重要理论基础，是计算机学科的核心专业课程。它对于学生程序设计能力的培养具有举足轻重的作用，因此，我们编写这本《数据结构》教材。

“数据结构”课程的教学要求是：学会分析计算机所加工的数据结构的特性，以便为所涉及的数据选择适当的逻辑结构和存储结构，并设计解决相应问题的算法，初步掌握算法运行所需要的时间和空间的分析方法。另一方面，本课程的学习过程也是复杂程序设计的训练过程，要求学生所编写的程序结构清晰、正确和易读，符合软件工程的规范。如果说高级语言的程序设计课程对学生所进行的是结构化程序设计的初步训练，那么“数据结构”课程就要培养他们的数学建模能力、解决复杂问题的能力以及理论联系实际的能力。

本书的特色是面向计算机专业的应用型本科人才培养，既保留了传统的内容，又引入了前沿的技术要素，内容全面，深入浅出，易于学习。

首先，本书在每一章的开头部分都配有内容导读，介绍本章的主要知识点、重点和难点，并通过知识结构图使读者对于本章的学习有一个总体的把握，在学习的时候能够做到有的放矢。

其次，本书的大多数章节都配有案例，这些案例均来自实际问题。例如，在“线性表”一章，我们引入约瑟夫环的问题，这些案例不仅具有现实意义，还富有趣味性，能够帮助学生培养学习的兴趣，同时锻炼学生理论联系实际的能力。这必将有助于应用型人才的培养。

第三，在本书中引入了计算机技术发展的最新要素。虽然以往的教材版本繁多，但是内容几乎相同。数据结构本身也应该是一门发展中的学科，理应有新知识点的扩充。本书中主要引入多核与并行计算领域中的相关知识，既能扩宽学生的视野，又能激励人们用发展的眼光去认识和研究“数据结构”这门课程。

第四，本书中的算法均在 Visual C + + 环境下调试通过，便于学生上机调试，加深对内容的理解。

第五，习题及思考题丰富。本书每一章后面都配有大量的习题，这些习题紧扣知识要点，能够帮助学生课后复习和提高。在附录中还提供了 4 套期末考试模拟试卷以及最近两年计算机专业全国硕士研究生入学考试统考试题“数据结构”部分的内容。同时，还为每一章都提供了若干实验题目。

本书中的章节安排如下。

第 1 章“绪论”：作为全书的开篇，本章讲述了数据结构的研究内容及学习数据结构的意义，还介绍了数据结构中的一些基本术语、算法的概念及分析方法。对目前比较先进的多核技术与并行算法设计作了适当的描述，并对 C 语言中的一些常见语句和主要知识点进行总结和归纳。

第 2 章“线性表”：介绍线性表的定义、基本运算、线性表的顺序存储结构和链式存储结构、相应算法的实现以及性能分析，并对顺序存储结构和链式存储结构进行了比较。同时，引入多线程链表的概念及其应用，最后通过实例阐述运用线性表解决实际问题的思路与方法。

第 3 章“串”：介绍串的基本概念、基本运算、串的 3 种存储结构及其基本运算的算法实现，

对串的模式匹配算法作了详细的介绍，最后结合实例讲述串的应用。

第4章“栈与队列”：从栈与队列的定义、栈与队列的顺序表示与实现、栈与队列的链式表示与实现、栈与队列的应用等方面进行介绍，并对递归的概念、递归程序设计作了相应的介绍。

第5章“数组与广义表”：主要介绍数组的两种存储表示方法、稀疏矩阵的压缩存储方法及适用范围、特殊矩阵的压缩存储方法、以三元组表示稀疏矩阵时进行矩阵运算所采用的处理方法、广义表的定义、运算及算法实现。

第6章“树”：首先介绍树和二叉树的概念和性质，研究它们的多种存储结构及相应的操作、二叉树的遍历方法及线索二叉树，还介绍了树与二叉树的相互转换，最后详细讲述哈夫曼树的构建和哈夫曼编码。

第7章“图”：首先讲述图的定义及术语，介绍图的存储结构、图的遍历，并讨论图的连通性问题、最短路径及有向无环图，最后通过实例对图的应用作了介绍。

第8章“查找”：首先介绍查找的基本概念，然后介绍基于线性表的查找法、基于树的查找法以及散列技术，最后通过应用实例讲述查找技术的应用。

第9章“内部排序”：主要介绍选择排序、插入排序、交换排序、归并排序和基数排序等排序方法的基本思想、排序过程和算法实现，并对各种排序算法进行简单的比较和分析。

第10章“文件”：首先简要介绍文件的基本概念、逻辑结构、操作和存储结构等内容，然后详细介绍几种常见的文件组织形式，讨论如何有效地组织数据、高效地利用数据。

本书可以作为普通高等学校计算机类或信息类相关专业的本科“数据结构”课程教材，也可以作为信息类相关专业的选修课教材。教师可以根据学时、专业和学生的实际情况，选讲或不讲程度较深的章节。

建议学时分配如下：第1章2学时，第2章6学时，第3章6学时，第4章2学时，第5章2学时，第6章10学时，第7章8学时，第8章6学时，第9章6学时，第10章6学时，共计54学时。教师可以根据实际情况进行适当的调整。

本书第1、2章及附录由汪桂兰撰写，第4、5章由孙月江撰写，第3、8、9章由辛琰撰写，第6、7章由高云撰写，第10章由杨研研撰写，王小华协作撰写多核技术部分，孙喜洲负责大部分程序的调试工作。全书由魏振钢进行修改并负责统稿。

本书中的程序的调试环境在附录中有所介绍。本书的配套习题答案及课件将在中国高校计算机课程网站上提供，网址 http://computer.cncourse.com。

在此感谢读者选择使用本书，欢迎广大读者对本书的内容提出意见和建议，我们将不胜感激，作者联系方式 guilan168@yahoo.com.cn。

编 者

2010年10月

目　录

第 1 章　绪论

本章导读

本章首先讲述数据结构的研究内容、学习数据结构的意义，并介绍了数据结构中的基本概念。然后对算法的概念及算法分析方法作了简单的介绍，对目前比较先进的多核技术及并行算法设计作了适当的描述，并对 C 语言中的一些常见语句和主要知识点进行总结和归纳。

本章的重点是数据结构及算法的基本概念、算法的分析方法及问题求解的一般步骤。

本章的难点是评价算法的标准及分析方法。

本章的知识结构如图 1.1 所示。

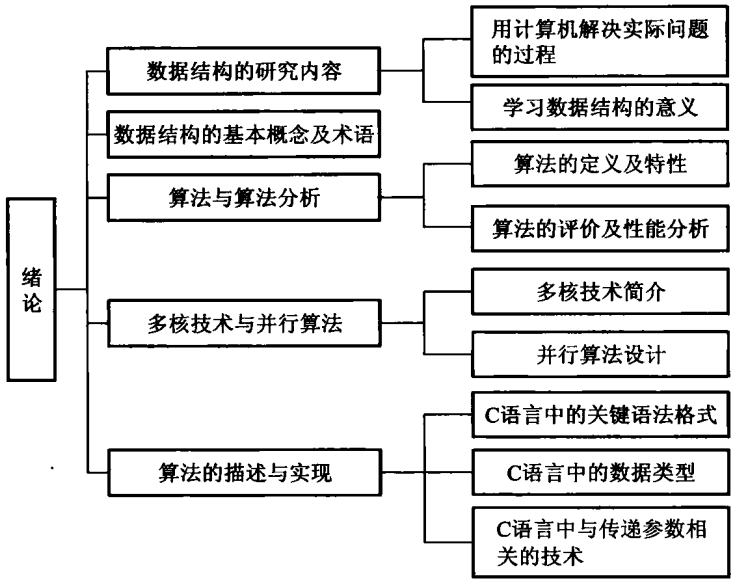


图 1.1　第 1 章的知识结构

1.1　数据结构的研究内容

1.1.1　用计算机解决实际问题的过程

通常用计算机解决一个具体的问题时，大致需要经过以下几个步骤：首先要从具体的问题中抽象出一个适当的数学模型，然后设计一个求解此数学模型的算法，最后编制程序，并进行测试、调整直至得到最终的解答。

1. 用计算机解决实际问题的步骤

(1) 建立数学模型

建立数学模型的实质就是分析问题。首先从实际问题中抽取操作对象,并找出这些操作对象之间的关系,然后用数学语言加以描述。例如,求解桥梁结构中的应力的数学模型为线性方程组,预报人口增长情况的数学模型为微分方程。

数值计算类问题的数学模型一般可以用数学方程或数学公式来描述。然而,对于非数值计算问题,如图书资料的检索、职工档案的管理、博弈等问题,它们的数学模型是无法用数学方程或数学公式来描述的。在这类问题的处理对象中,各个分量不再是单纯的数值型数据,而更多的是字符、字符串以及用编码表示的其他信息。因此,首要的问题是去掉某些细节,转而寻找它们的共性,这个被抽取出来的共性就是一种数学模型。目前已经有很多成熟的数学模型,当人们解决实际问题时,可以将一个具体的问题转换为熟悉的模型,然后借助这一模型来实现求解。"数据结构"、"离散数学"等课程中都介绍了不少模型。例如,要描述一个群体中的个体之间的关系时,可以采用"数据结构"和"离散数学"中所介绍的图的结构;要描述一个工程中的关系或进展情况时,可以采用"数据结构"中所介绍的 AOV 网或 AOE 网等。即使所建立的模型没有现成的求解方法,借助于已有模型的适当组合,也比较易于构造求解算法。

(2) 构造求解算法

在建立了模型之后,一个具体的问题就被转变成一个用模型来描述的抽象问题。借助于这一模型以及已有的知识,就可以相对容易地描述出原问题的求解方法,即算法。从某种意义上说,该算法不仅能够实现原有问题的求解,而且还可以实现许多类似问题的求解,尽管这些问题的产生背景及其描述形式可能存在一定程度的差异。

(3) 选择合适的存储结构

在构造出求解算法之后,就要考虑如何在计算机上实现求解了。首先要选择合适的存储结构,以便将问题所涉及的数据(其中包括数据中的基本对象及对象之间的关系)存储到计算机中。通过本书后续章节的讲解可知,不同的存储形式对问题的求解会产生较大的影响,所占用的存储空间也可能会存在较大的差异。

(4) 编写程序并测试

在确定了存储结构之后,就可以编写程序了。存储结构和问题要求决定了编写程序的方法。程序编写并调试完毕后,经过测试就可以交付使用了。

2. "数据结构"课程的研究内容与上述建模求解过程的关系

(1) 与建模求解过程的关系

"数据结构"课程中介绍了许多基本的数据结构模型及其运算,如线性表、栈和队列、树和二叉树、图、二叉排序树、堆等。读者通过学习,不仅应该掌握这些基本模型及其应用方法,还应该能够根据实际问题选择合适的模型。

(2) 与算法设计的关系

"数据结构"课程对每种数据结构都讨论了相应的基本运算及其实现方法,其中有些算法是非常经典的。掌握这些基本的算法将有助于进行更为复杂的算法的设计。

(3) 与存储结构的关系

"数据结构"课程对每种数据结构都讨论了其具体的存储结构及这种结构对算法的影响。

例如,在对顺序表执行插入和删除操作时,平均需要移动表中的一半元素,而采用链式存储结构则不需要移动任何元素。通过对这些内容进行学习,可以使学生在解决问题时能够根据问题的要求熟练地选择和设计合理的存储结构。

(4) 与编程的关系

在实现各种数据结构的算法中涉及许多具有代表性的算法,通过对这些算法进行学习,有助于提高编程技术。

综上所述,"数据结构"课程的研究内容与用计算机解决实际问题的步骤密切相关,如图 1.2 所示。正因为如此,这一课程在计算机专业课程中占有极其重要的地位,是考核学生计算机专业能力的重要内容之一。

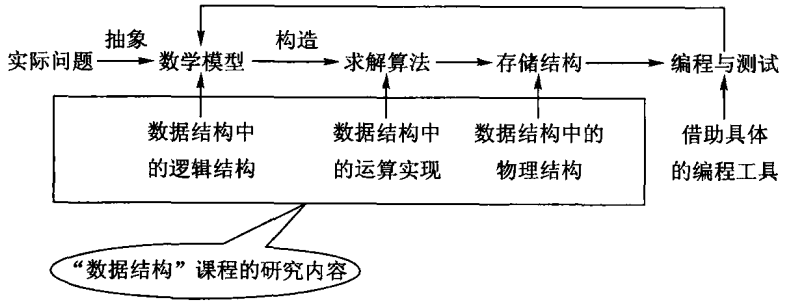


图 1.2 问题求解过程与数据结构

1.1.2 学习数据结构的意义

数据结构是计算机学科各个专业的核心课程之一,它的研究不仅涉及计算机硬件(特别是机器编码理论和存储方法等),而且和计算机软件的研究有着密切的关系,无论编译程序还是操作系统都涉及如何组织数据从而使数据的检索和存取更为方便的问题。因此,"数据结构"不仅是程序设计的基础,而且是设计和实现编译程序、操作系统、数据库系统及其他系统程序和大型应用程序的重要基础。

"数据结构"作为一门独立的课程,在美国是从 1968 年开始讲授的。在此之前,它的某些内容曾经在其他课程中有所涉及。1968 年,在美国一些大学计算机系的教学计划中,开始把"数据结构"规定为一门课程,但是对其内容并未作明确规定。随后,"数据结构"这个概念被扩充到计算机网络、集合代数、网格和关系等理论中,从而演变为现在的"离散结构"的内容。然而,由于数据必须在计算机中被处理,因此,不仅要考虑数据本身的数学特性,而且还要考虑数据的存储结构,这样就进一步扩大了数据结构的内容。

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。由于当时所涉及的运算对象是简单的整型、实型和布尔型数据,所以程序设计者的主要精力就集中在程序设计的技巧上,而无须重视数据结构。随着计算机应用领域的不断扩大和计算机软、硬件的逐步发展,"非数值型问题"显得越来越重要。据统计,当今处理非数值型问题占用了 90% 以上的机器时间,这类问题所涉及的数据结构更为复杂,数据元素之间的关系一般无法用数学方程加以描述。因此,解决此类问题的关键已经不再是分析数学模型和计算方法,而是要设计出合适的数据结构,只有这样才能有效地解决问题。

著名的瑞士计算机科学家沃思(N. Wirth)教授曾经提出“算法 + 数据结构 = 程序”。这里的“数据结构”是指数据的逻辑结构和存储结构,而算法则是对数据运算过程的描述。由此可见,程序设计的实质是对实际问题选择一种好的数据结构,再加上设计一个好的算法,而好的算法在很大程度上则取决于描述实际问题的数据结构。下面列举几个例子。

例 1.1 电话号码查询问题。

假设要编写一个计算机程序以查询某城市或某单位内部的私人电话,要求对给定的任意姓名进行判断。若该人装有电话,则要求迅速查找到其电话号码;否则,指出该人未安装电话。为了解决这一问题,首先要构造一张电话号码登记表,每人均在表中登记两项信息:姓名和电话号码。在将众多的登记项合并成一个数据表时,有多种不同的组织形式,数据查找速度取决于表的结构及存储方式。

最简单的方式是把表中的数据按照某种次序(如登记次序)依次存储在计算机内部一组连续的存储单元中。若用高级语言进行描述,就是把整个表作为一个数组,个人信息(即个人的姓名和电话号码)组成数组中的一个元素。查找数据时从表的第一项开始,依次查对姓名,直到找到指定的姓名或者确定表中不存在要查找的姓名为止。这种查找方法对于一个规模不大的单位或许是可行的,但是对于一个有几十万乃至上百万私人电话的城市就不太适用了。因此,常用的做法是把这张表按姓氏拼音或姓氏笔画排列,并另外构造一张姓名索引表,这类似于汉语词典的组织形式。对这样的表的查找可以先在索引表中查对姓氏,然后根据索引表中的地址再到登记表中核查姓名,这样查找登记表时就无须查找其他姓氏了。因此,在这种新的结构上所产生的查找方法就会更加有效。这两张表便是人们为解决电话号码查询问题而建立的索引存储数学模型,如图 1.3 所示。这种模型的主要操作是按照某个特定的要求(如给定的姓名)去对登记表进行查询。诸如此类的实例还有人事档案管理和图书资料管理等。在这类文档管理的数学模型中,被计算机处理的对象之间通常存在某种简单的线性关系,故称这类数学模型具有线性数据结构。

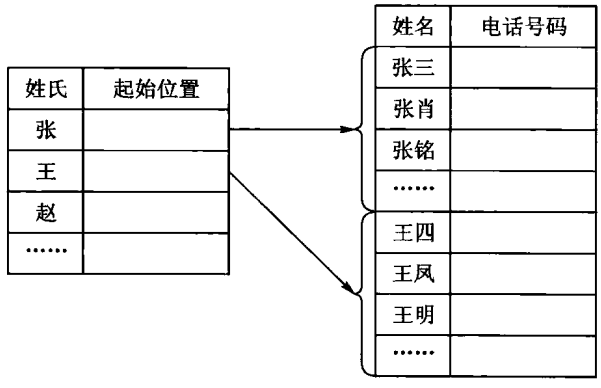


图 1.3 电话号码查询问题的索引存储数学模型

例 1.2 田径选拔赛项目的时间安排问题。

假设某校的田径选拔赛共设置 6 个比赛项目,即跳高、跳远、标枪、铅球、100 米(m)和 200 米短跑,规定每名选手至多参加 3 个项目的比赛。现在有 5 名选手报名参赛,他们所选择的项目如表 1.1 所示。

表 1.1 参赛选手报名项目表

姓 名	项 目 1	项 目 2	项 目 3
丁一	跳高	跳远	100 米
马二	标枪	铅球	—
张三	标枪	100 米	200 米
李四	铅球	200 米	跳高
王五	跳远	200 米	—

现在要求设计一个竞赛日程安排表,以便在尽可能短的时间内完成比赛。为了能够较好地解决这个问题,首先应该选择一个合适的数据结构来表示它。为此,可以设计这样的一张图,图中的顶点代表竞赛项目,在所有不能同时进行比赛的两个项目之间连上一条边。显然,同一个选手所选择的多个项目是不能在同一时间内开展比赛的,因此该选手所选择的项目中应该两两有边相连。由此可得到如图 1.4 所示的数据结构模型,其中 A、B、……、F 分别对应于 6 个项目。例如,丁一选择的 3 个项目分别是 A、B 和 E,因而 A 与 B 之间、A 与 E 之间以及 B 与 E 之间均有边相连。

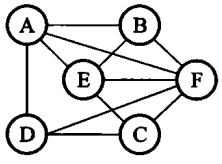


图 1.4 安排项目的数据结构模型

上述由顶点和边所组成的无向图是数据结构中非线性数据结构中的一类。田径选拔赛项目的时间安排问题可以抽象为对该无向图进行“着色”操作,即使用尽可能少的颜色去给图中的每个顶点着色,使得任意两个有边相连的相邻顶点都被着上不同的颜色。每一种颜色表示一个比赛时间,着上同一种颜色的顶点就是可以安排在同一个时间段内竞赛的项目。例如,顶点 A 和 C 不相邻,可以选取颜色 1 为它们着色;同理,顶点 B 和 D 可以着同一种颜色 2;顶点 E 和 F 之间有边相连,它们是相邻的,应该分别着上颜色 3 和颜色 4。也就是说,只要安排 4 个不同的时间段展开竞赛即可。时间段 1 内可以比赛跳高(顶点 A)和标枪(顶点 C),时间段 2 内可以比赛跳远(顶点 B)和铅球(顶点 D),时间段 3 和时间段 4 内则可以分别比赛 100 米(顶点 E)和 200 米(顶点 F)。

从上述例子不难看出,解决问题的一个关键步骤是选取合适的数据结构来表示问题,只有这样才能写出有效的算法。

1.2 数据结构的基本概念及术语

本节将对数据结构的一些概念和术语赋予确定的含义,以便与读者取得“共同的语言”。这些概念和术语将在后续章节中多次出现。

数据是对客观事物的符号表示,在计算机科学中是指所有能够被输入计算机中并被计算机程序所处理的符号。它是计算机程序加工的“原料”。例如,一个利用数值分析方法求解代数方程的程序,其处理对象是整数和实数;一个编译程序或文字处理程序的处理对象是字符串;在电

话号码查询系统中,被处理的对象则是个人信息,其中包括姓名和电话号码。对计算机科学而言,数据的内涵极为宽泛,如图像和声音等都可以通过编码而归入数据的范畴。

数据元素是指数据中具有独立意义的个体,它是数据的基本单位,在计算机程序中通常被作为一个整体进行考虑和处理,如图 1.3 中的个人电话号码信息、图 1.4 中的每个顶点。在某些情况下,数据元素也称为元素、结点、顶点、记录。有时候,一个数据元素可以由若干个数据项(又称为字段、域、属性)组成。**数据项**是具有独立含义的最小标识单位,如电话号码查询系统中的一个数据元素由姓名和电话号码这两个数据项构成。

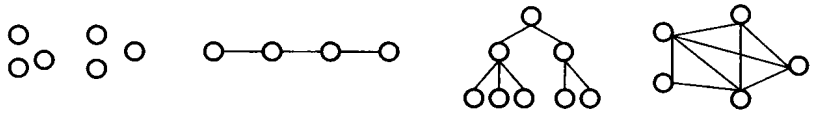
数据结构是指组成数据的元素之间的结构关系,即数据的组织形式。它一般包括以下 3 个方面的内容。

① 数据元素之间的逻辑关系,又称数据的**逻辑结构**。

② 数据元素及其相互之间的关系在计算机内部的表示,又称数据的**存储结构**。

③ 数据的运算,即对数据所施加的操作。

数据的逻辑结构是从逻辑关系上来描述数据,它与数据的存储结构无关,它是独立于计算机而存在的。因此,数据的逻辑结构可以被看作从具体问题抽象出来的数学模型。根据数据元素之间的关系的不同特性,通常有下列 4 种基本结构,其关系如图 1.5 所示。



(a) 集合结构 (b) 线性结构 (c) 树状结构 (d) 图形结构(网状结构)

图 1.5 4 种基本结构的关系

① 集合结构:结构中的数据元素之间除了具有同属于一个集合的关系外,便再无其他关系。

② 线性结构:结构中的数据元素之间存在着“一对一”的线性关系。

③ 树状结构:结构中的数据元素之间存在着“一对多”的层次关系。

④ 图形结构或网状结构:结构中的数据元素之间存在着“多对多”的任意关系。

数据的存储结构是数据的逻辑结构用计算机语言的实现,也称为物理映像,它依赖于特定的计算机语言。对机器语言而言,存储结构是具体的,但是只在高级语言的层面上来讨论存储结构。数据的存储结构可用以下 4 种基本存储方法得到。

1. 顺序存储方法

该方法是把逻辑上相邻的结点存储在物理位置上相邻的存储单元里,结点之间的逻辑关系由存储单元的邻接关系来体现。由此得到的存储表示称为顺序存储结构。通常顺序存储结构是借助于数组来描述的。

2. 链式存储方法

该方法不要求逻辑上相邻的结点在物理位置上亦相邻,结点之间的逻辑关系是由附加的指针字段来表示的。由此得到的存储表示称为链式存储结构。通常链式存储结构是借助于指针来描述的。

3. 索引存储方法

该方法是在存储结点数据的同时,还建立附加的索引表。索引表中的每一项都称为索引项,其一般形式是“(关键字,地址)”,关键字是能够唯一标识一个结点的数据项。若每个结点在索

引表中都有一个索引项，则该索引表称为稠密索引。若一组结点在索引表中只对应一个索引项，则该索引表称为稀疏索引。稠密索引中的索引项的地址用于指示结点所在的存储位置，而稀疏索引中的索引项的地址则用于指示一组结点的起始存储位置。

4. 散列存储方法

该方法的基本思想是根据结点的关键字直接计算出该结点的存储地址。

上述 4 种基本存储方法既可以单独使用，也可以组合起来对数据结构进行存储映像。对同一种逻辑结构采用不同的存储方法，可以得到不同的存储结构。选择何种存储结构来表示相应的逻辑结构，需要视具体的要求而定，主要考虑的是运算的便利性及算法的时空要求。

由于数据的运算也是数据结构中不可分割的一个方面，在给定了数据的逻辑结构之后，根据所定义的运算集合及其运算性质的不同，也有可能导致完全不同的数据结构。例如，若对线性表的插入和删除运算限制在表的一端进行，则该线性表称为栈；若对插入运算限制在表的一端进行，而对删除运算限制在表的另一端进行，则该线性表称为队列。更进一步地，若线性表采用顺序表或链表作为存储结构，则对插入和删除运算做出上述限制之后，可以分别得到顺序栈或链栈、顺序队列或链队列。

数据的运算被定义在数据的逻辑结构上，每种逻辑结构都有一个运算集合。例如，最常用的运算有检索、插入、删除、更新和排序等。实际上，这些运算是在抽象的数据上所施加的一系列抽象的操作。所谓抽象的操作，是指人们只知道这些操作“做什么”，而无须考虑“如何做”。只有在确定了存储结构之后，才能考虑如何实现这些运算。本书所讨论的数据运算均以 C 语言描述的算法来实现。

“数据结构”课程的研究内容如图 1.6 所示。

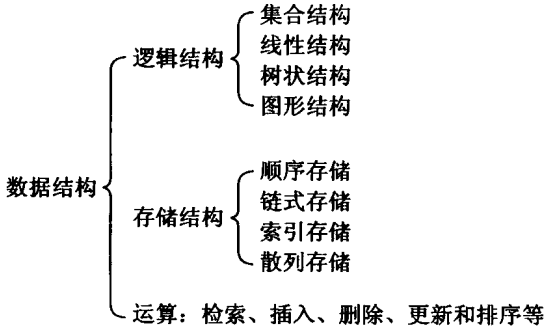


图 1.6 “数据结构”课程的研究内容

由此可见，对于一种数据结构，需要涉及其逻辑结构、存储结构和运算这 3 个方面。也就是说，对于每种数据结构都要注意这 3 个方面的联系。

1.3 算法与算法分析

1.3.1 算法的定义及特性

算法是对特定问题的求解步骤的一种描述，它是指令的有限序列，其中的每一条指令都表示一个或多个操作。此外，一个算法还具有以下 5 个重要的特性。

(1) 有穷性

对于任意一组合法的输入值,在执行有穷的步骤之后一定能够结束,即算法中的操作步骤有限,且每个步骤都能够在有限的时间内完成。

(2) 确定性

算法中的每一条指令都必须有确切的含义,读者在理解时不会产生歧义。在任何条件下,算法只有唯一的执行路径,即对于相同的输入只能得到相同的输出。

(3) 有输入

作为算法加工对象的量值,通常都体现为算法中的一组变量。有些输入量需要在算法执行的过程中输入,而有些算法表面上没有输入量,实际上输入量已经被嵌入算法之中。

(4) 有输出

它是一组与"输入"有确定关系的量值,是算法进行数据加工后得到的结果。这种确定的关系即为算法的功能。

(5) 可行性

算法中所描述的操作都是可以通过已经实现的基本运算执行有限次来实现的。

1.3.2 算法的评价及性能分析

通常在设计一个算法时应该考虑到以下目标。

① 正确性:算法应当能够正确地求解问题。

② 可读性:算法应当具有良好的可读性,便于人们阅读与理解。

③ 稳健性:当用户输入非法数据时,算法能够适当地作出反应或进行处理,而不会产生莫名其妙的输出结果。

④ 对执行效率与低存储量的要求:执行效率是指算法的运行时间,低存储量的要求是指算法在运行过程中所需要的最大存储空间应该尽可能地小。

一个算法所消耗的时间应该是该算法中的每条语句的执行时间之和,而每条语句的执行时间则是该语句的执行次数(又称频度)与该语句执行一次所需要的时间的乘积。但是,当算法被转换为程序之后,每条语句执行一次所需要的时间取决于机器指令的性能、速度以及编译所产生的代码质量,这是很难确定的。假设每条语句执行一次所需要的时间均为单位时间,一个算法消耗的时间就是该算法中的所有语句的频率函数值之和。于是,人们就可以独立于计算机软、硬件系统来分析算法所耗费的时间,即 T(时间)与 f(频率)成正比。

一般地,将算法求解问题的输入量称为问题的规模,并用整数 n 来表示。例如,矩阵乘积问题的规模是矩阵的阶数,而图论问题的规模则是图中的顶点数或边数。算法的**时间复杂度**则是该算法消耗的时间,它是该算法所求解问题的规模 n 的函数。当问题的规模 n 趋于无穷大时,就把时间复杂度 $T(n)$ 的数量级称为算法的渐进时间复杂度。将其记为:

$$T(n) = O(f(n))$$

例 1.3 语句{ ++x;s=0;}的频度(频率)为 1,时间复杂度为常量阶,可以表示为 $T(n) = O(1)$。

```
for(i=1;i<=n; ++i)
    ++x; /* 语句的频度为 n,时间复杂度为线性阶,可以表示为 T(n) = O(n) */
for(i=1;i<=n;i++)
```