

华东高校计算机基础教育研究会推荐教材

数据结构

——用C语言描述

宁正元 主编



中国水利水电出版社
上海交通大学出版社
东南大学出版社

华东高校计算机基础教育研究会推荐教材

数据结构——用 C 语言描述

宁正元 主 编

张健 舒坚 邓桂英 朱穗颖 易金聪 编著

康宝生 主 审

中国水利水电出版社
上海交通大学出版社
东南大学出版社

内 容 提 要

本书是根据高校计算机教材编委会讨论制定的《数据结构》课程教学大纲编写的教材。系统地介绍了各种类型的数据结构和检索排序的各种方法。对每一种数据结构从其逻辑结构、抽象运算、存储结构以及在相应存储结构上的运算实现都进行了详细的描述，并尽可能地列举了各种数据结构的不同的应用实例。对每一个算法都用 C 语言描述其实现，并对算法的性能给出定量或定性的分析比较。最后一章讨论了文件的各种组织方法。

本书内容丰富、概念清楚、叙述严谨、可读性强。每章配有丰富的例题、习题和上机实习题，并配套出版与之相对应的《数据结构习题解析与上机指导》，便于教学和学生自学。

本书以知识单元为基本构件，便于拆卸和重组，以适应不同院校的不同培养层次的教学需要。可作为大专院校计算机科学与技术专业的教材，更适合作为各校非计算机专业辅修计算机专业课程的教材，也可作为一切从事计算机科学与技术工作的科技人员的自学参考书。

图书在版编目 (CIP) 数据

数据结构——用C语言描述/宁正元主编. —北京: 中国水利水电出版社, 2000.5

华东高校计算机基础教育研究会推荐教材

ISBN 7-5084-0363-0

I. 数… II. ①宁… III. 数据结构-高等学校-教材 IV. TP311.12

中国版本图书馆CIP数据核字 (2000) 第60611号

书 名	数据结构——用 C 语言描述
作 者	宁正元 主编
审 校	康宝生
出版、发行	中国水利水电出版社 (北京市三里河路 6 号 100044) 网址: www.waterpub.com.cn E-mail: sale@waterpub.com.cn 电话: (010)63202266(总机)、68331835(发行部)
经 售	全国各地新华书店
排 版	北京万水电子信息有限公司
印 刷	北京牛山世兴印刷厂
规 格	787×1092 毫米 16 开本 16.25 印张 390 千字
版 次	2000 年 6 月第一版 2001 年 2 月北京第二次印刷
印 数	5001 — 10000 册
定 价	20.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社发行部负责调换

版权所有·侵权必究

华东高校计算机基础教育研究会

教材编委会名单

名誉主任：潘云鹤 顾冠群

主任：张森

副主任：李文忠

秘书长：赵民德

编委（以姓氏笔画为序）：

石冰	石文俊	宁正元	东鲁红
冉榴红	朱敏	吕刚	李正凡
张钧良	张煦	陈轩	陈凤兰
杭必政	周金辉	俞俊甫	徐安东
秦军	詹国华		

前 言

《数据结构》是计算机科学与技术专业教学计划中的一门核心课程，是计算机科学与技术专业和其他有志从事计算机与技术工作的人员的一门重要的专业基础课程。计算机科学与技术学科各领域都要用到各种数据结构，而且要从从事计算机科学与技术工作，尤其是计算机应用领域的开发研制工作，必须具备良好的数据结构基础。如语言编译要使用栈、散列表、语法树等；操作系统中要使用队、存储管理表、目录树等；数据库系统要使用线性表、链表、索引树等；人工智能中要使用广义表、检索树、有向图等；软件工程、面向对象的程序设计、计算机图形学、多媒体技术、计算机辅助设计等都要用到许多数据结构。

《数据结构》课程的教学要求是，学会分析研究计算机加工的数据对象的特性，以便在实际应用中选择适当的数据结构、存储结构和相应的算法，初步掌握算法的时间与空间性能分析技巧，得到复杂程序设计的训练。本书旨在阐明常用数据结构内在的逻辑关系，讨论它们在机内的存储表示，并结合实例说明它们在各种运算操作时的动态性质及执行算法。这样，不仅为读者学习后续课程提供了必要的知识准备，而且更重要的是进一步提高了软件设计与编程的能力和水平。

本书内容共分十章。第一章引出数据结构与算法的一些基本概念与术语，并对算法描述及算法分析作了简单说明；第二章到第五章由浅如深地介绍了线性结构中的线性表、栈、队、串、数组及广义表的基本定义、基本算法和基本应用；第六章、第七章介绍了非线性结构的树、二叉树和图；第八章、第九章介绍了在实际应用中非常广泛的检索和排序的基本算法，并进行了简单的时间、空间性能分析；第十章讨论了文件及其组织。全书以知识单元为基本构件、各知识单元内容相对独立，具有可拆卸性与可重组性，便于不同院校的不同培养层次按需要组织教学。全书侧重应用性，力求内容与应用实例相结合，有利于学生加深对内容的理解和掌握。

为了便于自学，本书在文字描述上力求通俗易懂；在算法描述上力求结构清楚，正确易懂；在实例列举上力求恰当，并附有相应地练习题和上机实习题。作为高校教材，建议讲授学时为 60 学时，上机实习学时为 20 学时。应该鼓励学生分析和改进算法，提出新的算法，特别是实际编程实现各种算法。强调多做习题和上机实习题的重要性。

本书是受华东高校计算机教材编委会的委托，由华东地区多所高校在第一线讲授数据结构课程的具有丰富教学经验的教师共同编写而成的。本书的第一、二、五、八、九章由宁正元执笔，第三章由张健执笔，第四章由舒坚执笔，第六章由邓桂英执笔，第七、十章由朱穗颖执笔，附录由易金聪整理。全书由宁正元主编、统稿、修改、定稿，西北大学康宝生博士后主审。由于时间仓促和作者水平所限，本书一定还存在不少问题，敬请广大读者批评指正。感谢华东地区计算机教材编委会和中国水电出版社、上海交通大学出版社、东南大学出版社对本书的出版给予的支持和鼓励；感谢各有关高校给予作者的大力支持。

作者

2000 年 2 月

目 录

前言

1 绪论	1
1.1 数据结构在计算机科学与技术中所处的地位.....	1
1.1.1 什么是数据结构.....	1
1.1.2 《数据结构》课程的地位.....	3
1.2 基本概念和术语.....	4
1.3 数据类型和抽象数据类型.....	6
1.4 算法描述与算法评价.....	8
1.4.1 算法描述.....	8
1.4.2 算法的设计要求.....	11
1.4.3 算法的评价.....	12
习题一.....	15
2 线性表	16
2.1 线性表的基本概念.....	16
2.1.1 线性表的逻辑结构.....	16
2.1.2 线性表的运算.....	17
2.2 线性表的顺序存储.....	17
2.2.1 顺序表.....	17
2.2.2 顺序表上的基本运算.....	19
2.3 线性表的链式存储结构.....	22
2.3.1 单链表及其单链表上的基本运算.....	23
2.3.2 循环链表.....	31
2.3.3 双向链表.....	32
2.4 线性表顺序存储结构和链式存储结构.....	35
2.5 线性表的应用举例.....	36
习题二.....	39
上机实习题一.....	40
3 栈和队列	42
3.1 栈.....	42
3.1.1 栈的定义和操作.....	42

3.1.2	栈的顺序存储结构.....	43
3.1.3	栈的链式存储结构.....	48
3.2	栈的应用举例	49
3.2.1	表达式求值	49
3.2.2	递归的实现	52
3.3	队列	55
3.3.1	队列的定义和操作.....	55
3.3.2	队列的顺序存储结构	56
3.3.3	队列的链式存储结构	60
3.4	队列的应用举例	62
	习题三	64
	上机实习题二.....	66
4	串	67
4.1	串的基本概念和存储结构	67
4.1.1	串的基本概念	67
4.1.2	串的存储结构.....	69
4.1.3	串变量的存储	71
4.2	串的基本运算	71
4.2.1	串的联接	73
4.2.2	求子串.....	74
4.2.3	子串的插入和删除.....	74
4.2.4	串的置换	75
4.3	串的模式匹配算法.....	76
4.3.1	求子串位置的定位函数	76
4.3.2	模式匹配的一种改进算法.....	79
4.4	汉字串	84
	习题四	87
5	数组和广义表.....	88
5.1	数组及其运算	88
5.2	数组的顺序存储结构.....	89
5.3	矩阵的压缩存储	91
5.4	广义表.....	104
5.4.1	广义表的定义.....	104
5.4.2	广义表的存储结构.....	105
5.4.3	广义表的运算.....	107
5.5	m元多项式的表示.....	108
	习题五	110

上机实习题三.....	110
6 树.....	112
6.1 树的基本概念和术语.....	112
6.2 二叉树.....	115
6.2.1 二叉树的定义.....	115
6.2.2 二叉树的性质.....	116
6.2.3 二叉树的存储结构.....	117
6.2.4 二叉树的应用——二叉搜索树.....	119
6.3 遍历二叉树.....	121
6.3.1 遍历二叉树的定义及递归算法.....	121
6.3.2 按前序序列建二叉树.....	122
6.3.3 遍历二叉树的非递归算法.....	123
6.3.4 二叉树算法举例.....	125
6.4 线索二叉树.....	128
6.5 树和森林.....	131
6.5.1 树的存储结构.....	131
6.5.2 森林与二叉树的转换.....	133
6.5.3 树与森林的遍历.....	134
6.6 哈夫曼树.....	135
6.6.1 基本术语.....	135
6.6.2 构造哈夫曼树.....	136
6.6.3 哈夫曼编码.....	136
6.6.4 哈夫曼算法的实现.....	137
6.7 回溯法与树的遍历.....	140
习题六.....	142
上机实习题四.....	143
7 图.....	144
7.1 图的概念及术语.....	144
7.2 图的存储结构.....	146
7.2.1 邻接矩阵.....	146
7.2.2 邻接表.....	148
7.2.3 邻接多重表.....	150
7.3 图的遍历.....	151
7.3.1 深度优先搜索遍历.....	152
7.3.2 广度优先搜索遍历.....	153
7.4 最小生成树.....	154
7.4.1 生成树和最小生成树.....	154

7.4.2	普里姆算法	155
7.4.3	克鲁斯卡尔算法	157
7.5	最短路径	158
7.5.1	求从一个顶点到其他各顶点的最短路径	158
7.5.2	求每一对顶点之间的最短路径	160
7.6	拓扑排序	162
	习题七	165
	上机实习题五	166
8	检索	168
8.1	检索的基本概念	168
8.2	线性表的检索	169
8.2.1	顺序检索	169
8.2.2	折半检索	170
8.2.3	分块检索	174
8.3	树表的检索	176
8.3.1	二叉排序树	177
8.3.2	平衡的二叉检索树	183
8.4	B 树	185
8.5	Hash 检索技术	189
8.5.1	Hash 表技术	189
8.6	哈希表的检索及分析	193
	习题八	195
	上机实习题六	196
9	排序	198
9.1	排序的基本概念	198
9.2	插入排序	199
9.2.1	直接插入排序	199
9.2.2	希尔排序	202
9.2.3	其他插入排序	205
9.3	交换排序	208
9.3.1	冒泡排序	208
9.3.2	快速排序	210
9.4	选择排序	214
9.4.1	直接选择排序	214
9.4.2	堆排序	216
9.5	归并排序	220
9.6	基数排序	223

9.7 内部顺序方法的比较和选择.....	226
9.8 外排序简介.....	226
习题九.....	227
上机实习题七.....	228
10 文件.....	230
10.1 文件的基本概念.....	230
10.2 顺序文件.....	231
10.3 索引文件.....	231
10.3.1 ISAM 文件.....	232
10.3.2 VSAM 文件.....	234
10.4 散列文件.....	235
10.5 多关键字文件.....	236
10.5.1 多重表文件.....	236
10.5.2 倒排文件.....	237
习题十.....	238
附录一 关键词索引.....	239
附录二 算法索引.....	243
参考文献.....	246

1 绪 论

随着现代电子计算机技术的发展,计算机的应用领域从最初的简单科学计算逐步发展到人类活动的各个领域;计算机处理的对象从最初的简单的数值或字符到现在带有不同复杂结构的各种数据,这就给程序设计带来一些新的问题。为了要设计出一个好的算法或程序,除了要掌握所用计算机的语言外,还要研究各种数据的特性和数据之间存在的关系。这就要求加强对数据结构这门学科的学习,从而更有效地使用计算机。

本章将介绍数据结构研究的对象、基本概念和术语、算法的概念及描述方法(C语言描述)、数据类型以及抽象数据类型,并概述数据结构的发展概况及其在计算机科学中的地位。

1.1 数据结构在计算机科学与技术中所处的地位

1.1.1 什么是数据结构

一般来说,用计算机解决一个实际问题时,一般是先对具体问题抽象,建立起实际问题的求解模型,然后设计出相应的算法,编写程序并上机调试,最终使实际问题得到解决。当人们用计算机处理的是数值计算问题时,所用的求解模型可以用数学方程描述,所涉及的运算对象一般是整型、实型或逻辑型等一些简单的数据类型。因此,程序设计者的主要精力是集中于程序设计的技巧上,而不是数据的组织和存储上。但随着计算机应用领域的不断扩大,计算机处理的对象更多的是非数值计算问题,如图书资料查询、电话号码自动管理、交通道路规划、博弈游戏等问题,它们的数学模型无法用数学方程来进行描述,此时就必须建立相应的数据结构来进行描述;分析问题中所用到的数据是如何组织的,研究数据之间的关系如何,进而为解决这些问题设计出合适的数据结构。

【例1】职工信息管理

一个单位要对所有职工的基本情况进行管理,必须要经常了解职工的各种信息,包括经常查询职工的编号、姓名、性别、以及月收入等情况,并对整个单位的情况作统计、汇总等工作。如何利用计算机来辅助管理这些信息呢?显然,应先考虑如何对整个单位每个职工的信息进行有效的组织,并考虑如何将这些信息存入计算机中,利用计算机进行有效的处理。

一般可以建立一个花名册对每位职工的信息进行编排,即组织这些信息如图 1.1 所示。花名册中每个职工的信息可由编号、姓名、性别、年龄、月收入等项目组成,占表的一行。这时候整个花名册就构成了一个数据结构,或者称整个花名册就是一个数据结构,而把表中的每一行看作一个结点或一个元素、一个记录,它由编号、姓名、性别、年龄、月收入等项目组成,从而整张表构成了一个关于职工花名册的数学模型,此时的花名册就是一张所谓的

线性表，此时计算机就可以按某个特定要求对其进行操作。表中的结点和结点之间是一种简单的线性关系。表中有且仅有一个结点为表头（第一个结点），有且仅有一个结点为表尾（最后一个结点）；对表中任一结点，与它相邻且在它前面的结点最多只有一个，这就是上述花名册表的逻辑结构。当考虑上述花名册表中的数据用计算机进行计算或处理时，就先要涉及到这些结点如何在计算机中的存储表示的问题，也就是存储结构的问题。对于这样的花名册表，有可能是调入一些职工必须增加新的结点，而某些职工调离时又必须将这些相应结点从表中删除掉。究竟如何进行插入、删除、修改、查找，这就是数据的运算问题，只有弄清这些问题后，才能有效地使用花名册表这个数据结构，有效地解决该单位职工基本信息的计算机辅助管理的问题。

编号	姓名	性别	年龄	月收入
1	李泉	男	51	980
2	王怡	女	47	945
3	张三	男	35	870
4	马小丁	男	27	840
...

图 1.1 职工花名册表

【例 2】旅游交通网络图

假若要建立一个旅游交通网络咨询系统，用来回答游客提出的各种问题。这时必须对各个城市的情况进行研究，对这些城市所代表的信息进行分析、组织。我们此时就可采用一种称之为图的结构来表示实际的交通网络，如图 1.2 所示，图中顶点表示城市，边表示城市间交通联系，这个交通网络图此时就表示一个数据结构。

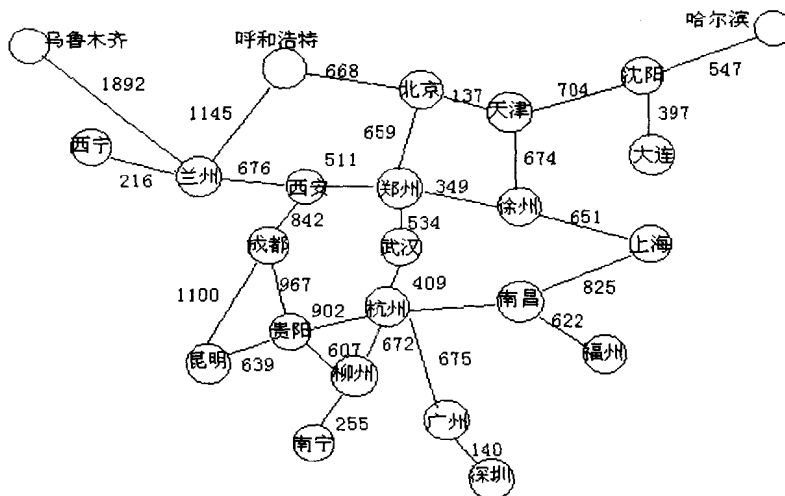


图 1.2 一个表示交通网的例图

在这个数据结构中，结点之间的关系可以是任意的，图中任意两个结点之间都可以相关。

构造出这张交通图后，要利用计算机进行处理，还得先讨论如何将这张图放入计算机中，这就必然要涉及到所谓图的存储结构问题。也就是说要对图中结点本身的值以及每个结点之间的可能的关系都存入计算机，然后再利用计算机进行操作处理，这就是所谓对图的各种运算。如一位旅客要从 A 城到 B 城，他希望选择一条旅途中转次数最少的路线，假设图中每一站都需要换车，这个问题反映到图上就是要找一条从顶点 A 到顶点 B 包含边的数目最少的路径，也就是所谓利用图进行的最短路径的运算问题。还有一些诸如关键路径等问题都可认为是対图这种数据结构所作的运算。

从以上两个例子可见，要描述这类的非数值计算问题的数学模型，单靠数学方程是无法满足的，它涉及到一些诸如表、图还有树之类的数据结构。这也是数据结构的研究对象，所以数据结构课程实际上是一门研究非数值计算问题的程序设计中计算机的操作对象以及它们之间的关系和运算操作等的一门学科。它和高级程序设计语言课程不同，高级程序设计语言课程所讨论的是简单程序设计问题，而数据结构课程所讨论的是在数据结构组织的基础上的复杂程序设计问题。

1.1.2 《数据结构》课程的地位

《数据结构》作为一门独立的课程在国外是从 1968 年才开始设立的。1968 年在美国一些大学的计算机系的教学计划中，把《数据结构》规定为一门课程，但对课程的范围没有作明确规定。当时，数据结构几乎是图论，特别是表、树的理论的同义词。随后，数据结构这个概念被扩充到包括网络、集合代数论、格、关系等方面，从而变成了现在称之为《离散结构》的内容。然而，由于数据必须在计算机中进行处理，因此，不仅考虑数据本身的数学性质，而且还必须考虑数据的存储结构，这就进一步扩大了数据结构的内容。

从 60 年代末到 70 年代初，出现了大型程序，软件也相对独立，结构程序设计成为程序设计方法学的主要内容，人们就越来越重视数据结构，认为程序设计的实质是对确定的问题选择一种好的结构，加上设计一种好的算法，正如瑞士著名的计算机科学家 N.Wirth 所提出的那个著名的公式“程序=算法+数据结构”所阐述的那样。从 70 年代中期到 80 年代初，各种版本的数据结构著作相继出现。

目前我国，《数据结构》也已经不仅仅是计算机科学技术专业的教学计划中的核心课程之一，而且是其他非计算机专业的主要选修课程之一。

《数据结构》在计算机科学中是一门综合性的专业基础课。数据结构的研究不仅涉及到计算机硬件（特别是编码理论、存储装置和存取方法等）的研究范围，而且和计算机软件的研究有着密切的关系，无论是编译程序还是操作系统，都涉及到数据元素在存储器中的分配问题。在研究信息检索时也必须考虑如何组织数据，以便查找和存取数据元素更为方便。因此，可以认为数据结构是介于数学、计算机硬件和计算机软件三者之间的一门核心课程。在计算机科学中，数据结构不仅是一般程序设计（特别是非数值计算的程序设计）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。

随着计算机科学技术的发展，计算机应用的普及，仅仅掌握计算机语言和程序设计的技巧和方法，而缺乏有关数据结构的知识，几乎很难应付复杂的课题，是不能有效地利用计算机的。

1.2 基本概念和术语

数据 (Data) 是指信息的载体, 是对客观事物的符号表示, 它能够被计算机识别、存储和加工处理, 也就是说数据是对有效地输入到计算机中并能被计算机程序处理的符号的总称。例如, 解代数方程程序中所用到的整数和实数, 文本编辑中所用到的函数和字符串等, 都是程序加工处理的对象, 客观事物中的声音、图形、图像, 经过某些处理也可使计算机识别、存储和处理, 所以它们也可属于数据的范畴。

数据元素 (Data element) 是数据的基本单位, 有时也称之为结点、元素、顶点、记录等。有时一个数据元素可由若干个数据项 (Data item) 组成, 如上面职工花名册中的每个人的信息就是一个数据元素, 而每个人的信息包括编号、姓名、性别、年龄、月收入等, 这每一个项目就属于数据项。数据项是数据的不可分割的最小单位。

数据对象 (Data object) 是性质相同的数据元素的集合, 是数据的一个子集。例如整数数据对象是集合 $N=\{0, +1, -1, +2, -2, \dots\}$, 字母字符数据对象是集合 $C=\{ 'A', 'B', 'C' \dots\dots 'Z' \}$ 。

数据类型 (Data type) 是对在计算机中表示的同一数据对象及其在该数据对象上的一组操作的总称。如整数, 在计算机中它是 $[\text{minint} \dots \text{maxint}]$ 上的整数, 在这个集合上可以进行加、减、乘、整除、求模等一些基本操作 (maxint 称作最大整数, 在不同的计算机中它的值不同)。

数据类型有时还分为原子数据类型和结构数据类型。原子数据类型是由计算机语言系统提供的一些简单类型, 其值不可分解, 如整型、实型、字符型; 而结构数据类型是借用计算机语言中类型定义的有关语法机制来定义的, 或可看作是由原子数据类型用某种方式 (可以简单也可以复杂) 组合而成的, 因而其值是可以分解的, 其成分可以是原子的, 也可以是结构的, 不同的组合方式形成不同的结构类型, 例如 C 语言中的结构体、数组、文件、共同体结构等。

数据结构 (Data structure) 指的是数据及数据之间的相互关系。由上节例子可以发现任何问题中, 数据元素都不是孤立的, 在它们之间具有某种关系, 这种数据元素相互之间的关系有时也称为结构。一般来说, 数据结构包括以下三个方面的内容:

(1) 数据元素之间的逻辑关系, 有时也称为数据的逻辑结构 (Logical structure)。

(2) 数据元素及其关系在计算机内存中的表示 (又称为映像), 将其称之为数据的物理结构或数据的存储结构。它包括数据元素的表示和数据元素之间关系的表示。

(3) 数据的运算及实现, 即对数据元素可以施加的操作及其这些操作在相应的存储结构上的实现。

对于数据结构三个部分的理解, 我们必须知道数据的逻辑结构是从逻辑关系上描述数据, 与数据的存储无关, 是独立于计算机的。因此, 数据的逻辑结构可以看作是从具体问题抽象出来的数学模型。如例 1 中的职工花名册表, 它仅表示的是每个职工信息 (数据元素) 之间的关系而已, 只是在逻辑关系上说明每个职工信息之间的相互关系。根据数据元素之间的不同特性, 通常有下列四类基本逻辑结构:

(1) 集合: 结构中的数据元素之间除了“同属于一个集合”的相互关系外, 别无其他关系。

(2) 线性结构：结构中的数据元素存在一对一的关系，即所谓的线性关系，如例 1 中的职工花名册就可看作是一个典型的线性结构，所以又称为线性表。这个表中的数据元素之间的逻辑关系是：对表中任何一个结点，与其相邻且在它之前的结点（有时称作直接前趋）最多只有一个，与表中任一结点相邻且在其后的结点（又称为直接后继）也最多只有一个。且在这个表中只有第一个结点没有前驱，又称为开始结点，同样也只有最后一个结点无后继，称为终端结点，其他结点之间都是相邻的，且一个跟着一个。也就是说这个表的数据元素之间的关系具有线性结构的逻辑特征，有且仅有一个开始结点和一个终端结点，并且所有结点最多只有一个直接前趋和一个直接后继。

(3) 树型结构：结构中的数据元素之间存在一个对多个的关系。树型结构实际上是结点之间有分支、并具有层次关系的结构，它非常类似于自然界中的树。例如：我们平常所说的家谱，行政组织机构等都可用树形象地表示，也就是说可将其抽象成为树形结构来处理。在这种结构中，每个结点允许有多个直接后继。

(4) 图状或网状结构：结构中的数据元素之间的关系存在多个对多个的关系，如例 2 中所提到的交通网络图就是一个图状结构。对这个图而言，其结点（图中常称为顶点）的前趋和后继个数都不加限制，结点之间的关系则是任意的，图中任意两个结点之间都可以相关。有时我们也可将树形结构、集合和网状结构归纳为非线性结构，此时数据逻辑结构就可分为线性结构和非线性结构两类。

数据的存储结构是指逻辑结构用计算机语言的实现，也就是数据元素及其关系如何存放放入计算机内存的问题。如在例 1 中，我们要在机内存储职工花名册表，就必须考虑如何存储每个职工的信息本身，同时还必须考虑如何将每个职工信息之间的相互联系在机内体现出来，也就是必须存储它们之间的相互关系。一般来说，数据的存储结构可按以下四种基本存储方法而得到：

(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中，结点间的逻辑关系由存储单元的邻接关系而体现。由此得到的存储表示称为顺序存储结构（Sequential storage structure），通常可借助于计算机语言的数组来描述。该方法可用于线性的数据结构的存储，如例 1 中的职工花名册就可利用一个一维数组来存储，对于一些非线性结构也可以通过某种线性化处理后实现顺序存储。

(2) 链状存储方式：该方法不要求逻辑上相邻的结点在物理位置上也相邻，也就是说数据元素可以存储在任意位置。显然，为了实现数据元素之间逻辑关系的存储，就必须通过一些附加的手段来存储这种相互关系，一般可以用指针（Pointer）来实现，由此得到的存储表示称为链状存储结构（Linked storage structure）。通常可借助计算机程序语言的指针类型或者游标来描述。

(3) 索引存储方法：该方法通常是在存储结点信息的同时，建立附加的索引表，索引表中由很多个索引项组成，索引项的一般形式为关键字、地址，其中关键字是指可唯一标识一个结点的数据项。索引表一般有稠密索引（Dense index）和稀疏索引（Sparse index）两种，如果每个结点在索引表中都有一个索引项，该索引表称为稠密索引，稠密索引中索引项的地址可指出结点所在的存储位置，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，稀疏索引中的索引项可指示一组结点的起始存储位置。

(4) 散列存储方法：该方法是依据结点的关键字直接计算出该结点的存储地址，而后

将结点按某种方式存入该地址的一种存储方法。

一般来说，以上四种基本的存储方法，既可单独使用，也可以组合起来对数据结构进行存储。同一种逻辑结构采用不同的存储方法，可得到不同的存储结构。选择何种存储结构来存储表示相应的逻辑结构就要视具体问题的要求而定，也可依据运算是否方便和算法的时间效率和空间要求而定。显然，它也是依赖于所用的计算机语言的，我们一般可借助高级程序语言来描述数据的存储结构。

数据的运算是指定义在数据的逻辑结构上的一组操作的集合。每种逻辑结构都允许有某些运算。例如，一些常用的如检索、插入、删除、定位、修改、排序等。这些运算实际上是在逻辑结构上对抽象数据所施加的一系列抽象的操作，也就是说，对于这些操作，我们知道这些操作可以“做什么”而不需考虑它们是“如何实现”，只有对某种数据结构选定了存储结构之后，此时才去考虑如何将这组运算具体地实现，也就是运算的实现。运算的实现依赖于所选取的存储结构，依赖于不同的计算机程序设计语言。本书所讨论的运算，均以 C 语言描述的算法来实现。

1.3 数据类型和抽象数据类型

数据类型是具有相同性质的计算机数据的集合及在这个数据上的一组运算，是和数据结构密切相关的概念。在用高级程序语言编写的程序中，每个变量、常量和表达式都有一个它所属的数据类型。类型明显或隐含地规定了在程序执行期间变量或表达式所有可能的取值范围，以及在数值范围上的所允许进行的操作。例如，整型数，它是 $[-\text{minint} \dots \text{maxint}]$ 区间上的整数（ maxint 即最大整数，依据所使用的不同计算机及程序设计语言，它具有不同的值）。在这个整数集上就可以进行加、减、乘、整除、取模等操作。在高级程序语言中，数据类型按照它的结构是否可以分解，可以分为原子类型和结构类型。原子类型即值不可分解的类型，如 C 语言中的整型、字符型、实型、指针类型等；而结构类型则指的是其值可由若干成分按某种结构组成的类型，是可以分解的，它所包含的各成分可以是原子类型也可以是结构类型。如 C 语言中提供的结构体，它相当于其他高级程序语言中的记录，可由许多成员（或称分量）组成。每一个成员称为结构体中的域，可以是整型或字符型，也可以是结构体或数组等。

在分析一些复杂的情况或处理复杂的事物时，我们一般采用抽象思维的方法，即舍去复杂系统中非本质的成分，只对其中一些本质的可以反映系统重要特性的东西归纳提炼出来，设计出系统的模型，然后深入研究这些模型，得出一般的规律性的东西，解决实际问题。这种抽象思维的方法同样适用于软件系统的设计和研究。在复杂的软件系统设计中同样可以应用抽象思维的方法，对软件系统中所包括的数据和操作进行抽象分析，从而更有效地进行软件系统的设计，将数据抽象和运算抽象紧密地结合在一起就构成所谓抽象数据类型的概念。

抽象数据类型（Abstract Data Type 简称 ADT）是指一个数据模型及定义在该数据模型上的一组操作。显然，抽象数据模型的定义仅取决于它的一组逻辑特性，与其在计算机内部如何表示和实现无关。换句话说，不论它的内部结构如何变化，只要它的数学特性不变，都不影响它的外部的使用。

抽象数据类型是算法设计和程序设计中的重要概念。严格地说，它是算法的一个数据模型连同定义在该模型上的作为该算法构件的一组运算，从而明确地把数据模型与作用在该模

型上的运算紧密地联系起来。事实上的确如此，数据模型的运算依赖于数据模型的具体表示，因为数据模型的运算是以数据模型中的数据变量作为运算对象、运算结果，或二者兼而有之。另一方面，有了数据模型的具体表示，有了在数据模型上运算的具体实现，运算的效率也就随之确定。于是就有了这样一个问题，如何选择数据模型的具体表示使该模型的各种运算的效率尽可能提高？很明显，对于不同的运算组，为使组中所有运算的效率都尽可能的高，其相应的数据模型具体表示的选择将是不同的。在这个意义下，数据模型的具体表示又反过来依赖于数据模型上定义的那些运算，数据模型与定义在该模型上的运算之间存在着这种密不可分的关系，是抽象数据模型的概念产生的背景与依据。

应该指出，抽象数据类型的概念并不是全新的概念，它实际上是我们熟悉的基本数据类型概念的引伸和发展。由计算机高级程序语言知识可知，基本数据类型已隐含着数据模型和定义在该模型上的运算的统一，只是当时还没有形成抽象数据类型的概念罢了。如整型就是整数值数据模型与加、减、乘、除、取模等几种运算的统一体。每一种基本数据类型都连带着一组基本运算，只是这些基本数据类型中的数据模型的具体表示和基本运算的具体实现都很规范，都可以通过内置而隐藏起来，使人们看不到它们的封装。我们习惯于在实际设计中用基本数据类型名和相关的运算名，没有意识到抽象数据类型的概念实际上已经孕育在基本数据类型的概念之中。那又如何定义抽象数据类型呢？

依据抽象数据类型的概念，对抽象数据类型进行定义就是约定抽象数据类型的名字，同时，约定在该类型上定义的一组运算的各个运算的名字，明确各个运算分别要有多少个参数。这些参数的含义和顺序以及运算的功能。一旦定义清楚，就可十分简便地引用抽象数据类型。如可定义线性表的抽象数据类型如下：其中数据关系可以是在对数据对象分析后建立起来的一种数据模型。

ADT LIST

{ 数据对象: $D = \{ a_i, i \in \text{elemset}, i=1, 2, \dots, n; n \geq 0 \}$

/*定义线性表，它是具有 n 个数据元素的有限序列*/

数据关系: $R = \{ \langle a_i, a_{i+1} \rangle, a_i \in D, i=1, 2, \dots, n \}$

/* 说明线性表数据元素之间的相互关系.*/

基本运算:

SETNULL(L)

初始条件: 线性表 L 存在;

运算结果: 将线性表 L 置成空表;

LENGTH(L)

运算结果返回 L 线性表的数据元素个数即表长;

GET(L, i)

初始条件: 线性表 L 存在, 且 $1 \leq i \leq \text{LENGTH}(L)$,

运算结果: 返回表 L 的第 i 个结点;

LOCATE(L, x)

初始条件: 线性表 L 中存在一个值为 x 的结点;

运算结果: 返回该结点的位置。若表 L 中存在多个值为 x 的结点, 则是 L 中首次找到的结点位置, 当表中不存在值为 x