

21 世纪高等院校规划教材 · 计算机类

SHUJU JIEGOU C YUYAN BAN

数据结构 (C语言版)

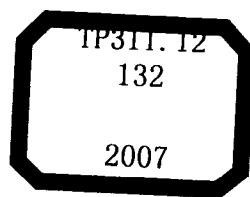
主 编 秦 锋

副主编 汤文兵 章曙光 汪 军

杨学兵 陈学进 袁志祥

中国科学技术大学出版社

21 世纪高等院校规划教材 · 计算机类



数 据 结 构

(C 语 言 版)

主 编 秦 锋

副主编 汤文兵 章曙光 汪 军

杨学兵 陈学进 袁志祥

中国科学技术大学出版社

2007 · 合肥

内 容 简 介

本书全面系统地介绍了线性表、队列、堆栈、树、图等基本数据结构,以及这些数据结构在计算机中的存储及算法实现,系统地介绍了各种查找及排序算法的实现和效率分析,最后一章给出了数据结构综合应用实例。书中各种算法采用C语言描述,注重程序设计风格。

本书有配套教材《数据结构(C语言版)例题详解与课程设计指导》,书中有各知识点的归纳与总结,有例题详解及习题解答以及课程设计指导。

有关教学参考资料的电子文档可到 <http://jpkc.ahut.edu.cn/sjgg/> 网址下载。

本书语言流畅、内容通俗易懂,可作为计算机类及信息类专业教材,也可供广大研究开发人员自学提高时使用。

图书在版编目(CIP)数据

数据结构: C语言版/秦锋主编. —合肥: 中国科学技术大学出版社, 2007. 2
ISBN978-7-312-02041-4

I. 数… II. 秦… 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.
TP311.12 TP312

中国版本图书馆CIP数据核字(2007)第014281号

中国科学技术大学出版社 出版发行

(安徽省合肥市金寨路96号, 邮政编码: 230026)

合肥学苑印务有限公司印刷

全国新华书店经销

开本: 787×1092/16 印张: 19.25 字数: 480 千

2007年2月第1版 2007年2月第1次印刷

印数: 1—4000册

ISBN978-7-312-02041-4

定价: 28.00元

前言

以计算机科学技术为核心的信息技术正在深刻地改变着人们的工作方式、生活方式和思维方式。如果说计算机是由硬件和软件两部分组成,显然软件应该是计算机的灵魂、程序设计应该是计算机科学技术最重要的基础。写出高质量的程序是每个软件开发者所追求的目标,要达到这个目标仅靠学习几门高级语言是远远不够的,正如一个人仅靠认识几个汉字是决不可以写出好文章一样。数据结构这门课程正是开启程序设计知识宝库的金钥匙,其主要目的是培养学生将现实世界抽象为数据和数据模型的能力以及利用计算机进行数据存储和数据加工的能力。学好数据结构,可以使你掌握更多的程序设计技巧,为以后学习计算机专业课程及走上工作岗位从事计算机大型软件开发打下良好的基础。目前,数据结构是我国高校计算机类专业(含计算机科学与技术、软件工程、网络工程、信息安全等)的核心课程之一,也是其他信息类专业(含信息管理与信息系统、通信工程、信息与计算科学等)的必修课程之一。正因为它在计算机培育计划中的重要地位,大多数高校计算机类专业研究生入学考试都将数据结构作为必考课程之一。

本书内容共 10 章,第 1 章重点介绍了数据结构与算法的基本概念,介绍了评价算法优劣的主要指标以及衡量算法效率的时间复杂度和空间复杂度;第 2 章到第 4 章,重点介绍了线性表、栈、队列和字符串等线性结构的逻辑特性、存储结构,以及基本操作算法;第 5 章到第 7 章,重点介绍了多维数组、广义表、树、二叉树、图等非线性结构的逻辑特征、存储表示及基本操作算法的实现和具体应用;第 8 章到第 9 章,介绍了在软件开发中广泛运用的两种操作:排序和查找,对一些常用的查找、排序算法进行了详细描述及效率分析。第 10 章将线性结构和树、图等非线性结构归纳在一起,运用树、图的遍历算法设计思想去解决实际问题,让读者理论联系实际,以加深对各种数据结构的理解。

书中算法采用 C 语言描述,每段代码都保持良好的设计风格,配有丰富的例题和习题。

数据结构是一门实践性很强的课程,读者在进行理论学习的同时,需要多动手编写程序上机调试,以加深对所学知识的理解,提高编程能力。

本书可作为高等院校计算机类或信息类相关专业数据结构课程教材,建议理论课时为 50 至 70 学时,上机及课程设计等实践课时为 20 至 30 学时。各院校可根据本校的专业特点和具体情况适当增删教学内容。

本书有配套教材《数据结构(C 语言版)例题详解与课程设计指导》,书中有各知识点的归纳与总结,有例题详解及习题解答以及课程设计指导。

本书由秦锋教授任主编,由汤文兵、章曙光、汪军、杨学兵、陈学进、袁志祥任副主编。

第 1、6 章由秦锋编写，第 2、3 章由汪军编写，第 4 章由陈学进编写，第 5、8 章由章曙光编写，第 7 章由汤文兵编写，第 9 章由袁志祥编写，第 10 章由杨学兵编写，全书由秦锋负责修改并统稿。因编者水平有限，书中难免有不足甚至错误之处，敬请广大读者批评指正！

为方便教学，授课教师可到 <http://jpkc.ahut.edu.cn/sjjg/> 网址下载电子教案等相关教学参考资料。

建议或者意见请来函到：fqin@ahut.edu.cn。

编 者

2006.10

目 录

第1章 绪论	(1)
1.1 什么是数据结构	(1)
1.2 基本概念和术语	(5)
1.3 算法和算法描述语言	(8)
1.4 算法分析	(9)
本章小结	(18)
习题	(18)
第2章 线性表	(22)
2.1 线性表的逻辑结构	(22)
2.2 线性表的顺序存储及运算实现	(24)
2.3 顺序表应用举例	(31)
2.4 线性表的链式存储和运算实现	(33)
2.5 顺序表和链表的比较	(46)
本章小结	(47)
习题	(48)
第3章 栈和队列	(51)
3.1 栈	(51)
3.2 栈的应用举例	(59)
3.3 栈与递归	(69)
3.4 队列	(76)
3.5 队列应用举例	(85)
本章小结	(89)
习题	(89)
第4章 串	(93)
4.1 串及其基本运算	(93)
4.2 串的顺序存储及基本运算	(95)
4.3 串的堆存储结构	(104)
4.4 串的链式存储结构	(108)
4.5 文本编辑——串操作应用	(110)
本章小结	(112)
习题	(112)

第 5 章 数组和广义表	(115)
5.1 数组.....	(115)
5.2 特殊矩阵的压缩存储.....	(117)
5.3 广义表.....	(124)
本章小结.....	(127)
习题.....	(128)
第 6 章 树和二叉树	(131)
6.1 树的基本概念.....	(131)
6.2 二叉树.....	(134)
6.3 遍历二叉树.....	(138)
6.4 线索二叉树.....	(147)
6.5 树与森林.....	(154)
6.6 哈夫曼树.....	(159)
本章小结.....	(168)
习题.....	(168)
第 7 章 图	(173)
7.1 图的基本概念.....	(173)
7.2 图的存储结构.....	(177)
7.3 图的遍历.....	(186)
7.4 最小生成树.....	(190)
7.5 最短路径.....	(196)
7.6 有向无环图及其应用.....	(201)
本章小结.....	(211)
习题.....	(212)
第 8 章 查找	(216)
8.1 基本概念.....	(216)
8.2 线性表的查找.....	(218)
8.3 树表查找.....	(222)
8.4 哈希表查找(杂凑法).....	(240)
本章小结.....	(248)
习题.....	(248)
第 9 章 排序	(253)
9.1 基本概念.....	(253)
9.2 插入排序.....	(255)
9.3 交换排序.....	(261)
9.4 选择排序.....	(266)
9.5 归并排序(Merging Sort)	(270)

本章小结·····	(272)
习题·····	(273)
第 10 章 数据结构综合应用·····	(277)
10.1 各种结构类型之间的关系概述·····	(277)
10.2 二叉树与分治策略·····	(281)
10.3 图的深度优先遍历及其应用·····	(285)
本章小结·····	(295)
习题·····	(296)
参考文献·····	(297)

第1章 绪 论

数据结构是计算机、信息管理、信息与计算科学等信息类专业最重要的专业基础课程,掌握好数据结构的知识将直接关系到后续专业课程的学习。数据结构主要研究四个方面的问题:1)数据的逻辑结构,即数据之间的逻辑关系;2)数据的物理结构,即数据在计算机内的存储方式;3)对数据的加工,即基于某种存储方式的操作算法;4)算法的分析,即评价算法的优劣。本章重点介绍数据结构研究问题所涉及的基本知识和概念。

[本章学习要求]

了解:研究数据结构的目的是意义。

了解:数据结构基本概念和相关术语。

掌握:算法基本概念和算法评价依据。

掌握:算法的时间复杂度和空间复杂度。

1.1 什么是数据结构

程序设计是计算机学科各个领域的基础。在计算机发展的早期,程序设计所处理的数据都是整型、实型等简单数据,绝大多数的应用软件都是用于数值计算。随着信息技术的发展,计算机逐渐进入到金融、商业、管理、通讯以及制造业等各个行业,广泛地应用于数据处理和过程控制,计算机加工处理的对象也由纯粹的数值型数据发展到字符、表格和图像等各种具有一定结构的数据,这就给程序设计带来一些新的问题,数据结构的概念就是在这种背景下产生的。

1.1.1 数据结构的定义

计算机解决一个具体问题一般需要经过下列几个步骤:首先要从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,最后编出程序进行调试、测试,直至得到最终的解答。显然如何抽象出数学模型并设计解此数学模型的算法就成为问题的关键,由于早期计算机所涉及的运算对象是简单的整型、实型或布尔类型数据,数学模型或者比较简单或者就是某个数学公式,程序设计者的主要精力可以集中于程序编码的设计上,无须重视数据的逻辑关系和存储结构。随着计算机应用领域的扩大和软、硬件技术的发展,非数值计算问题显得越来越重要。据统计,当今用计算机处理的非数值计算性问题约占90%以上的机器时间。如图书资料的检索、职工档案管理、博弈游戏等,这类问题涉及到的数据结构相当复杂,数据元素之间的相互关系无法用数学方程或数学公式来描述。这类问题的处理对象中的各分量不再是单纯的数值型数据,更多的是字符、字符串或其他编码表示的信息。对这类复杂问题的处理,设计者必须把重点放在如何将加工对象的各种信息按其逻辑特性组

【例 1.1】 人口信息查询

身份证号	姓 名	
340105750209001	王成明	
100202861021007	陈海龙	
220104800923002	王利芳	
.....		
340504560511039	丁凯非	
100104730816011	李达明	

(a) 无序的人口信息表

身份证号	姓 名	
100104730816011	李达明	
100202861021007	陈海龙	
220104800923002	王利芳	
.....		
340105750209001	王成明	
340504560511039	丁凯非	

(b) 有序的人口信息表

			地址	身份证号	姓 名	...
北京	10	1	→ 1	100104730816011	李达明	...
.....				100202861021007	陈海虎	...
吉林	22	11567	→			...
.....			→ 11567	220104800923002	王利芳	...
安徽	34	35689	→			...
			→ 35689	340105750209001	王成明	...
				340504560511039	丁凯非	...

(c) 人口信息索引表

图 1.1 人口信息表

这种查找速度太慢, 对于一个人口数量不多的地区或许是可行的, 但对于全国范围或者一个有几千万甚至几亿人口的地区就不行了。一种常用的做法是把登记表按照身份证号从小到大排序(见图 1.1 (b)), 并存储在计算机内一组连续的存储单元中, 在查找时可以采用折半查找算法(第 9 章有详细介绍), 查找速度将大大提高。我们还可以根据身份证前两位数字代表省份这一特征将此表建成索引表的形式, 见图 1.1(c), 查找时根据身份证前两个数字首先找到所在的地区或省份, 然后再按照某种方法继续查找下去, 这种查找速度也比较快。

从这个例子可以看出, 人口信息登记表如何构造、如何存储在计算机内存中将直接影响查找算法的设计以及算法的执行效率。图 1.1 所示的表便是我们为解决问题而建立的数学模型。这类模型的主要操作是查询、修改、插入及删除等。诸如此类的还有库存管理、图书资料管理等。在这种数学模型中, 计算机处理的对象之间通常存在一种简单的线性关系, 故这类数学模型可称为线性的数据结构。

【例 1.2】 人机对弈问题

1996 年 IBM “深蓝” 计算机与卡斯帕罗夫进行国际象棋人机大战, 结果“深蓝”取得胜利, 震惊了整个世界。十年之后的 2006 年浪潮“天梭”战胜了由柳大华等五位中国象棋大师组成的大师团队。计算机之所以能和对局并取胜是因为有人将对局的策略事先存入计算机, 通过其超强的逻辑判断和计算能力寻找最佳走法。计算机通常采用穷举法, 即通过对所有可能的招法的演化结果进行比较, 最后选择出一个最好的招法。在某个“局面”中, 如果有 38 种可能的棋子走法, 无论选择其中哪一种, 都会再生成一个新局面。如果将所有这些变化列起来, 就会形成一棵“博弈树”, 如图 1.2 (b) 就是井字棋“博弈树”的一个局部。计算机要做的, 就是比较最后一层结果, 选择其中一个最好的, 然后逆推, 找到本局面最好的一个走法。棋子走法越多“博弈树”越大, 对计算机存储和计算能力的要求越高, 如浪潮“天梭”每秒最多可计算 42 亿步棋。在对弈问题中, 计算机操作的对象是对弈过程中可能出现的棋盘格局或棋盘状态。图 1.2(a) 就是井字棋的一个格局, 而格局之间的关系是由比赛规则决定的。显然这种关系不是线性的, 因为从一个棋盘格局可以派生出许多格局。“博弈树”的树根是对弈开始之前的棋盘状态, 而所谓的叶子就是可能出现的结局, 对弈的过程就是从树根到某个叶子的过程。“博弈树”就是解决这类问题的数学模型, 它也是一种数据结构。

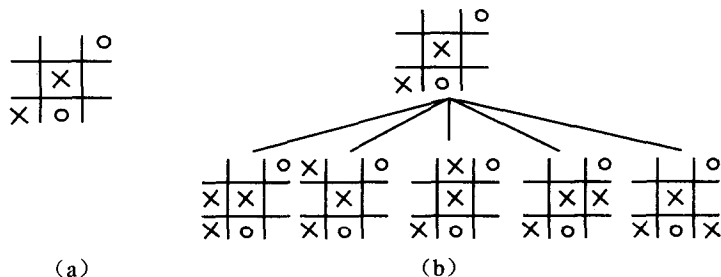


图 1.2 井字棋对弈树

【例 1.3】 最小代价问题

几个村庄之间要架设输电线路, 根据电能的可传递性, 并不需要在每对村庄之间架设线

路。如何用最小代价解决这个问题。

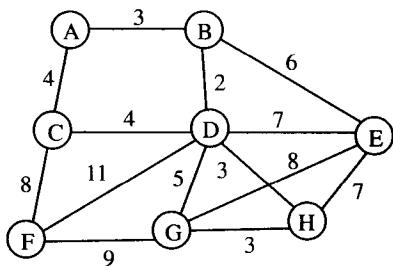


图 1.3 输电线路示意图

处理此类问题就需要用到图这种数学模型。用顶点代表村庄，每对顶点之间的边代表村庄之间的线路，边的权值就是线路的建设费用。如图 1.3 所示。只要把这个图的有关信息存储在计算机中，利用图论中最小生成树算法，就可以解决最小代价问题。显然在图这种数学模型中，数据之间的关系更为复杂，但它能更准确的描述客观世界。图也是常见的数据结构之一。

从上述三个例子可见，描述非数值计算问题的数学模型不再是数学方程或数学公式，而是诸如表、树、图之类的数据结构。因此，简单说来，数据结构是一门研究非数值计算的程序设计问题中计算机的操作对象及相互间的关系和运算的学科。

1.1.2 学习数据结构的意义

数据结构是计算机科学与技术专业的核心基础课程。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课题的。要想有效地使用计算机、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。学好“数据结构”这门课程，将为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、人工智能等打下良好的基础。

数据结构作为一门独立的课程最早是在美国开设的，1968 年美国唐·欧·克努特教授开创了数据结构的最初体系。他所著的《计算机程序设计技巧》第一卷《基本算法》，是第一本较系统地阐述数据的逻辑结构和存储结构及其操作的著作，从 20 世纪 60 年末到 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，数据结构的地位显得更为重要，人们认为程序设计的实质是对确定的问题选一个好的结构，加上一种好的算法。当时，数据结构几乎和图论，特别是表、树的理论为同义语。随后，数据结构这个概念被扩充到包括网络、集合代数、格、关系等方面，从而变成现在称之为“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，这就进一步扩大了数据结构的内容。

瑞士计算机科学家沃斯 (N.Wirth) 曾以“算法+数据结构=程序”作为他的一本著作的名称。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，并设计一个好的算法。因此，仅仅掌握几种计算机语言和程序设计方法，而缺乏数据结构知识，则难以解决众多复杂的问题。

数据结构的研究不仅涉及计算机硬件（特别是编码理论、存储装置和存取方法等）的研究范围，而且与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，都涉及如何组织数据，使检索和存取数据更为方便。因此，可以认为，数据结构是介于数学、计算

机硬件和软件三者之间的一门核心课程。

目前,数据结构是我国高校计算机专业的核心课程之一,也是其他信息类专业如信息管理、通信工程、信息与计算科学等必修课程之一。

1.2 基本概念和术语

本节中将对一些基本概念和术语加以定义和解释,这些概念和术语将在以后的章节中多次出现。

1.2.1 数据与数据元素

数据(Data)是对客观事物的符号表示,它能被计算机识别、存储和加工处理。它是计算机程序加工的“原料”。例如,一个代数方程求解程序所用到的数据是整数和实数;一个编译程序处理的对象是字符串(源程序)。在计算机科学中,数据的含义相当广泛,如客观世界中的声音、图像等,经过某些处理也可被计算机识别、存储和处理,因而它们也属于数据的范畴。

数据元素(Data Element)是数据的基本单位,有时也称为元素、结点、顶点或记录。一个数据元素可能由若干数据项(Data Item)组成。例如,在【例 1.1】人口信息查询系统中身份证号、姓名等数据项组成了一个数据元素,即一个数据元素包含了二个以上的数据项。数据项是最小标识单位,有时也称为字段、域或属性。数据元素也可以仅有一个数据项。

数据结构(Data Structure)是指数据元素之间的相互关系,即数据的组织形式。它一般包括以下三个方面的内容:

- (1) 数据元素之间的逻辑关系,也称为数据的逻辑结构(Logical Structure)。它独立于计算机,是数据本身所固有的。
- (2) 数据元素及逻辑关系在计算机存储器内的表示方式,称为数据的存储结构(Storage Structure)。它是逻辑结构在计算机存储器中的映射,必须依赖于计算机。
- (3) 数据运算,即对数据施加的操作。运算的定义直接依赖于逻辑结构,但运算的实现必依赖于存储结构。

1.2.2 数据的逻辑结构与存储结构

数据的逻辑结构是从逻辑关系上描述数据,不涉及数据在计算机中的存储,是独立于计算机的。可以说,数据的逻辑结构是程序员根据具体问题抽象出来的数学模型。数据中元素通常有下列几种形式的逻辑关系:

- (1) 集合:任何两个元素之间都没有逻辑关系,每个元素都是孤立的。
- (2) 线性结构:结构中的元素之间存在一对一的关系,即所谓的线性关系。例【例 1.1】中的人口信息表就是一个线性结构,在这个结构中,各元素(由一个人的身份证号、姓名和其他信息组成)排列成一个表,第一个元素之后紧跟着第二个元素,第二个元素之后紧跟着

第三个元素，依次类推，整个结构就像一条“链”，故有“线性结构”之称。

(3) 树形结构：结构中的数据元素之间存在一对多的关系。这种结构像自然界中的倒长的“树”一样，呈分支、层次状态。在这种结构中，元素之间的逻辑关系通常称做双亲与子女关系。例如，家谱、行政组织结构等都可用树形结构来表示。

(4) 图状结构：结构中的元素之间存在多对多的关系，也就是说元素间的逻辑关系可以是任意的。在这种结构中，元素间的逻辑关系也称做邻接关系。

通常将树形结构、图状结构归纳为非线性结构。因此，数据的逻辑结构可分为两大类，即线性结构和非线性结构。

数据的**存储结构**（也称物理结构）是指数据在计算机内的表示方法，是逻辑结构的具体实现。因此，存储结构应包含两个方面的内容，即数据元素本身的表示与数据元素间逻辑关系的表示。数据的存储结构有下列四种基本方式：

(1) 顺序存储：其方法是把数据元素依次存储在一组地址连续的存储单元中，元素间的逻辑关系由存储单元的位置直接体现，由此得到的存储表示称为顺序存储结构（**Sequential Storage Structure**）。高级语言中，常用一维数组来实现顺序存储结构。该方法主要用于线性结构，非线性结构也可通过某种线性化的处理，实现顺序存储。

(2) 链式存储：将数据元素存储在一组任意的存储单元当中，而用附加的指针域表示元素之间的逻辑关系，由此得到的存储表示称为链式存储（**Linked Storage Structure**）。使用这种存储结构时，往往把一个数据元素及附加的指针放在一个结构体中作为一个结点。高级语言中，常用指针变量来实现链式存储。

(3) 索引存储：该方法的特点是在存储数据元素的同时，还建立附加的索引表。图 1.1(c) 就是一个索引表，索引表中每一项称为索引项。索引项的一般形式是：（关键字，地址）。关键字是指能唯一标识数据元素的数据项。若每个数据元素在索引表中均有一个索引项，则该索引表称为稠密索引（**Dense Index**）。若一个索引项对应一组数据元素，则该索引表称为稀疏索引（**Sparse Index**）。

(4) 散列存储：该方法是依据数据元素的关键字，用一个事先设计好的函数计算出该数据元素的存储地址，然后把它存入该地址中。这种函数称为散列函数，由散列函数计算出的地址称为散列地址。

上述四种存储方式，既可单独使用，也可组合使用。逻辑结构确定后，采取何种存储结构，要根据具体问题而定，主要的考虑因素是运算方便、算法效率与空间的要求。

存储结构的描述与程序设计语言有关。通常用高级语言中的类型说明来描述存储结构，不必涉及计算机的内存地址。

1.2.3 数据运算

数据运算是对于数据施加的操作。每种逻辑结构都有一个基本运算的集合，最常用的基本运算有：检索(查找)、插入、删除、更新、排序等。因为这些运算是在逻辑结构上施加的操作，因此它们同逻辑结构一样也是抽象的，只规定“做什么”，无须考虑“如何做”。只有确定了存储结构后，才能考虑“如何做”。简言之，运算是在逻辑结构上定义，在存储结构上实现。

必须注意, 数据结构包含逻辑结构、存储结构和运算三方面的内容。同一逻辑结构采用不同存储结构, 得到的是不同的数据结构, 我们常用不同的数据结构名来标识它们。例如, 线性结构采用顺序存储, 我们称之为顺序表; 采用链式存储时则称为链表。同样, 同一逻辑结构定义不同的运算也会导致不同的数据结构。例如, 若限制线性结构的插入、删除在一端进行, 则该结构称为栈; 若限制插入在一端进行, 而删除在另一端进行, 则称为队列。更进一步, 若栈采用顺序存储结构, 则称为顺序栈; 若栈采用链式存储, 则称为链栈。顺序栈与链栈也是两种不同的数据结构。

1.2.4 数据类型与抽象数据类型

数据类型(Data Type)是和数据结构密切相关的一个概念, 几乎所有高级语言都提供这一概念。数据类型是一个值的集合和在这个集合上定义的一组操作的总称。例如, C 语言中的整型变量, 其值集为某个区间上的整数(区间大小依赖于不同的机器), 定义在其上的操作为: 加、减、乘、除和取模等运算。

按“值”可否分解, 可把数据类型分为两类:

(1) 原子类型: 其值不可分解, 如 C 语言的基本类型(整型、字符型、实型、枚举型)、指针类型和空类型。

(2) 结构类型: 其值可分解成若干成分(或称分量), 如 C 语言的数组类型、结构类型等。结构类型的成分可以是原子类型, 也可以是某种结构类型。我们可以把数据类型看做程序设计语言已实现的数据结构。

引入数据类型的目的, 从硬件角度考虑, 是作为解释计算机内存中信息含义的一种手段; 对用户来说, 实现了信息的隐蔽, 即将一切用户不必了解的细节都封装在类型中。例如, 用户在使用整数类型时, 既不需要了解整数在计算机内如何表示, 也不必了解其操作(比如两个整数相加)硬件是如何实现的。

抽象数据类型(Abstract Data Type, 简称 ADT)是指一个数学模型以及定义在该模型上的一组操作。抽象数据类型的定义取决于它的一组逻辑特性, 而与其在计算机内部如何表示和实现无关。即不论其内部结构如何变化, 只要它的数学特性不变, 都不影响其外部的使用。

抽象数据类型和数据类型实质上是一个概念。例如, 整数类型是一个 ADT, 其数据对象是指能容纳的整数, 基本操作有加、减、乘、除和取模等。尽管它们在不同处理器上的实现方法可以不同, 但由于其定义的数学特性相同, 在用户看来都是相同的。因此, “抽象”的意义在于数据类型的数学抽象特性。

但在另一方面, 抽象数据类型的范畴更广, 它不再局限于前述各处理器中已定义并实现的数据类型, 还包括用户在设计软件系统时自己定义的数据类型。为了提高软件的重用性, 在近代程序设计方法学中, 要求在构成软件系统的每个相对独立的模块上, 定义一组数据和施于这些数据上的一组操作, 并在模块的内部给出这些数据的表示及其操作的细节, 而在模块的外部使用的只是抽象的数据及抽象的操作。这也就是面向对象的程序设计方法。

抽象数据类型的定义可以由一种数据结构和定义在其上的一组操作组成,而数据结构又包括数据元素间的关系,因此抽象数据类型一般可以由元素、关系及操作三种要素来定义。

1.3 算法和算法描述语言

数据运算是通过算法来描述的,因此讨论算法是数据结构课程的重要内容之一。

算法(Algorithm)是对特定问题求解步骤的描述,是指令的有限序列,其中每条指令表示一个或多个操作。对于实际问题不仅要选择合适的数据结构,还要有好的算法,只有这样才能更好地求解问题。

一个算法必须具备下列五个特性:

(1) 有穷性: 一个算法对于任何合法的输入必须在执行有限步骤之后结束,且每步都可在有限时间内完成。

(2) 确定性: 算法的每条指令必须有确切含义,不能有二义性。在任何条件下,算法只有唯一的一条执行路径,即对相同的输入只能得出相同的结果。

(3) 可行性: 算法是可行的,即算法中描述的操作均可通过已经实现的基本运算的有限次执行来实现。

(4) 输入: 一个算法有零个或多个输入,这些输入取自算法加工对象的集合。

(5) 输出: 一个算法有一个或多个输出,这些输出应是算法对输入加工后合乎逻辑的结果。

程序与算法十分相似,但程序不一定要满足有穷性。例如,操作系统启动后,即使没有作业要处理,它仍然会处于等待循环中,以等待新的作业进入,所以不满足有穷性

算法可以使用各种不同的方法来描述。

最简单的方法是使用自然语言,用自然语言来描述算法的优点是简单且便于人们对算法的阅读,缺点是不够严谨,容易产生二义性。

可以使用程序流程图、N-S图等算法描述工具。其特点是描述过程简洁、明了。

用以上两种方法描述的算法不能够直接在计算机上执行,若要将它转换成可执行的程序还有一个编程的问题。

可以直接使用某种程序设计语言来描述算法,不过直接使用程序设计语言并不容易,而且不太直观,常常需要借助于注释才能使人看明白。

为了解决理解与执行之间的矛盾,人们常常使用一种称为伪码语言的描述方法来进行算法描述。伪码语言介于高级程序设计语言和自然语言之间,它忽略高级程序设计语言中一些严格的语法规则与描述细节,因此它比程序设计语言更容易描述和被人理解,而比自然语言更接近程序设计语言。它虽然不能直接执行但很容易被转换成高级语言。本书采用C语言作为描述数据结构和算法的工具,但在具体描述时有所简化,如常常将类型定义和变量定义省略。书中所有的算法都是以函数形式表达出来,算法的书写格式如下:

```
<函数类型> <函数名> ([<参数表>])
```

```
/*函数功能的简单说明*/
```