



普通高等教育“十二五”重点规划教材·计算机系列  
中国科学院教材建设专家委员会“十二五”规划教材

严丽丽 主编

# 数据结构

DATA STRUCTURE



科学出版社

普通高等教育“十二五”重点规划教材·计算机系列  
中国科学院教材建设专家委员会“十二五”规划教材

# 数 据 结 构

严丽丽 主编

王 成 邓奉先 龙著乾 副主编

魏应彬 主审

## 内 容 简 介

本书介绍了各种类型的数据结构,包括逻辑结构、物理结构和数据操作。力求以通俗易懂的讲解配以图示方法,使读者能对抽象的内容进一步理解。

本书共分8章,叙述了几种不同数据结构和查找、排序技术,阐述了线性表、栈、队列、串、数组、二叉树、树、图等各种基本数据结构的概念;从物理角度讲解了每种逻辑结构的不同存储结构,以及相应操作的实现和结构特点分析;从算法的角度详细介绍了不同的排序和查找。

本书内容翔实,图文并茂,各章后都有习题和实训题。实训代码均在TurboC上调试通过,对理解数据结构有一定帮助。

本书旨在为计算机及相关专业的应用型本科及高职高专学生教学使用,也可以作为从事计算机软件开发人员和自学人员的参考书。

### 图书在版编目(CIP)数据

数据结构/严丽丽主编. —北京:科学出版社,2011  
ISBN 978-7-03-032027-8

I. ①数… II. ①严… III. ①数据结构-高等职业教育-教材  
IV. ①TP311.12

中国版本图书馆 CIP 数据核字(2011)第 162223 号

责任编辑:赵丽欣 郭丽娜 / 责任校对:刘玉靖  
责任印制:吕春珉 / 封面设计:子时文化

科学出版社出版

北京东黄城根北街16号  
邮政编码:100717

<http://www.sciencep.com>

铭浩彩色印装有限公司印刷

科学出版社发行 各地新华书店经销

\*

2011年8月第 一 版 开本:787×1092 1/16

2011年8月第一次印刷 印张:17

印数:1—3 000 字数:399 000

定价:32.00 元

(如有印装质量问题,我社负责调换(骏杰))

销售部电话 010-62142126 编辑部电话 010-62134021

版权所有,侵权必究

举报电话:010-64030229;010-64034315;13501151303

# 普通高等教育“十二五”重点规划教材 计算机系列专业基础课类学术编审委员会

主    任：杜先能    吴晓蓓    单启成    杨静宇    曹进德  
副 主 任：韩忠愿    许  勇    严云洋    周鸣争    方厚加  
          李敬兆    秦  锋    庄玉良    朱贵喜    戴仕明

编委成员：（排名不分先后，按姓氏笔画为序）

|      |      |      |      |      |
|------|------|------|------|------|
| 卜红宝  | 于春玲  | 尹  静 | 方瑞芬  | 毛岷林  |
| 王建农  | 王忠群  | 王维民  | 王靖国  | 史国川  |
| 史春联  | 乔正洪  | 刘传领  | 刘家琪  | 安  进 |
| 朱正礼  | 江家宝  | 汤静芳  | 严  峥 | 严丽丽  |
| 吴德琴  | 宋正虹  | 张居晓  | 张洪斌  | 张裕荣  |
| 李  胜 | 杨  枢 | 邵  杰 | 陈  鹏 | 陈海燕  |
| 周志刚  | 周明争  | 於文刚  | 林  莉 | 姜  华 |
| 宦集体  | 施永香  | 赵树宇  | 凌海云  | 徐卫军  |
| 钱  峰 | 梁  明 | 黄海生  | 葛武滇  | 潘子宇  |

# 前 言

随着科学技术的迅猛发展,计算机在非数值计算领域的应用越来越广泛,而数据结构方面的知识是解决此类问题的基础,因此,在计算机与信息技术类专业的教学中数据结构具有突出的地位和作用。

数据结构是计算机与信息技术类专业的一门重要的专业基础课程,也是该专业的一门核心课程。在计算机与信息科学中,数据结构不仅是一般程序设计的基础,还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的重要基础,本课程将为后续课程的学习、程序设计效率的提高奠定良好的基础。

本书共分8章。第1章是概述,主要介绍了数据结构的相关概念和术语,以及算法的特点和评价标准等内容;第2章介绍了线性表的有关概念和基本操作;第3章对栈和队列的知识进行了重点介绍;第4章为数组和字符串,对数组的基本表示和实现进行了介绍,还介绍了串的存储标记及基本操作;第5章为树,包括二叉树、树、森林的基本概念、表示与实现,以及哈夫曼树的应用等内容;第6章介绍了图的基本理论,并对最短路径、最小生成树、拓扑排序等应用做了重点介绍;第7章主要介绍了常用的排序方法;第8章主要介绍了几种传统的查找方法,并对散列查找等现代查找方法进行了详细介绍。

本书特点如下:

1. 本书以应用型本科高职高专学生的认知水平为出发点,设置技能训练内容以及相关知识点,充分体现“够用、实用”原则,降低教学难度。
2. 内容翔实、示例丰富、讲解细致,操作性较强。
3. 课后习题丰富,并都配有相应的参考答案,便于学生参考。

本书旨在为计算机及相关专业的应用型本科及高职高专学生教学,也可以作为从事计算机软件开发人员和自学人员的参考书。

本书由严丽丽任主编,王成、邓奉先和龙著乾任副主编。严丽丽编写了第1、2、3、4、8章,王成编写了第6、7章,邓奉先编写了第5章和前3章参考答案,龙著乾编写了后5章参考答案。全书由严丽丽负责统稿和定稿,魏应彬教授审稿。

本书在编写过程中,得到海南大学信息学院钟声教授、海南中橡电子商务有限公司赵颂章高级工程师等专家的指点和帮助。海南软件职业技术学院的桂吉吉教授、陈鹤年、王弗雄、季文天、何军余、白晓波、周任云、黎剑等教师为本书的编写也付出了辛勤的努力,在此一并表示衷心的感谢。

由于时间紧迫以及作者的水平有限,书中难免有不当之处,恳请广大读者和同行专家批评指正!

# 目 录

|                            |    |
|----------------------------|----|
| 第 1 章 概述                   | 1  |
| 1.1 什么是数据结构                | 1  |
| 1.2 基本概念和术语                | 1  |
| 1.3 算法描述和算法分析              | 3  |
| 1.3.1 算法的概念                | 3  |
| 1.3.2 算法设计的要求              | 4  |
| 1.3.3 算法的描述                | 5  |
| 1.3.4 算法性能的评价              | 5  |
| 1.4 本课程学习指导                | 6  |
| 1.5 本章小结                   | 7  |
| 1.6 习题                     | 7  |
| 第 2 章 线性表                  | 8  |
| 2.1 什么是线性表                 | 8  |
| 2.2 线性表的顺序存储结构及其算法         | 9  |
| 2.2.1 线性表的顺序存储结构           | 9  |
| 2.2.2 顺序表的运算               | 10 |
| 2.2.3 顺序表应用——班级考勤统计        | 13 |
| 2.3 线性表的链式存储结构             | 15 |
| 2.3.1 动态内存分配及其管理           | 15 |
| 2.3.2 线性链表                 | 17 |
| 2.3.3 循环链表                 | 26 |
| 2.3.4 双向链表                 | 27 |
| 2.3.5 静态链表                 | 29 |
| 2.4 线性链表的应用——一元多项式的表示及加法运算 | 30 |
| 2.5 本章小结                   | 33 |
| 2.6 习题                     | 33 |
| 2.7 实训题                    | 35 |
| 实训一 学生基本信息                 | 35 |
| 实训二 线性链表的基本操作              | 35 |
| 第 3 章 栈和队列                 | 36 |
| 3.1 栈                      | 36 |
| 3.1.1 栈的定义                 | 36 |
| 3.1.2 栈的存储结构及其基本运算         | 37 |

|                              |           |
|------------------------------|-----------|
| 3.1.3 栈的应用 .....             | 41        |
| 3.2 队列 .....                 | 44        |
| 3.2.1 队列的定义 .....            | 44        |
| 3.2.2 队列的存储结构及其基本运算的实现 ..... | 45        |
| 3.2.3 队列的应用 .....            | 52        |
| 3.3 本章小结 .....               | 55        |
| 3.4 习题 .....                 | 56        |
| 3.5 实训题 .....                | 56        |
| 实训一 表达式求值 .....              | 56        |
| 实训二 商品货架管理 .....             | 57        |
| <b>第4章 数组和字符串</b> .....      | <b>58</b> |
| 4.1 数组 .....                 | 58        |
| 4.1.1 数组的定义和操作 .....         | 58        |
| 4.1.2 数组的顺序存储和访问 .....       | 59        |
| 4.1.3 数组的类型的实现 .....         | 61        |
| 4.1.4 特殊矩阵的压缩存储 .....        | 62        |
| 4.2 串 .....                  | 66        |
| 4.2.1 字符串的基本操作 .....         | 67        |
| 4.2.2 定长字符串的实现 .....         | 68        |
| 4.2.3 可变长字符串的实现 .....        | 73        |
| 4.2.4 字符串的模式匹配 .....         | 77        |
| 4.2.5 字符串应用举例 .....          | 79        |
| 4.3 本章小结 .....               | 81        |
| 4.4 习题 .....                 | 81        |
| 4.5 实训题 .....                | 83        |
| 实训一 字符串操作 .....              | 83        |
| 实训二 稀疏矩阵转置 .....             | 83        |
| <b>第5章 树</b> .....           | <b>84</b> |
| 5.1 树 .....                  | 84        |
| 5.1.1 树的基本概念 .....           | 84        |
| 5.1.2 树的基本术语 .....           | 85        |
| 5.1.3 树的基本运算 .....           | 86        |
| 5.2 二叉树 .....                | 86        |
| 5.2.1 二叉树的概念 .....           | 86        |
| 5.2.2 二叉树的性质 .....           | 87        |
| 5.2.3 二叉树的存储结构 .....         | 89        |
| 5.2.4 遍历二叉树 .....            | 92        |
| 5.2.5 哈夫曼树和哈夫曼编码 .....       | 94        |

|                         |     |
|-------------------------|-----|
| 5.2.6 应用实例 .....        | 98  |
| 5.3 树和森林 .....          | 103 |
| 5.3.1 树的存储结构 .....      | 104 |
| 5.3.2 树、森林与二叉树的转换 ..... | 106 |
| 5.3.3 树和森林的遍历 .....     | 109 |
| 5.4 本章小结 .....          | 110 |
| 5.5 习题 .....            | 111 |
| 5.6 实训题 .....           | 112 |
| 实训 二叉树的应用 .....         | 112 |
| <b>第6章 图</b> .....      | 113 |
| 6.1 图的定义和基本术语 .....     | 113 |
| 6.1.1 图的定义 .....        | 113 |
| 6.1.2 图的基本术语 .....      | 114 |
| 6.2 图的存储结构 .....        | 117 |
| 6.2.1 邻接矩阵 .....        | 117 |
| 6.2.2 邻接表 .....         | 118 |
| 6.3 图的遍历 .....          | 120 |
| 6.3.1 深度优先搜索(DFS) ..... | 120 |
| 6.3.2 广度优先搜索(BFS) ..... | 122 |
| 6.4 图的应用 .....          | 125 |
| 6.4.1 最小生成树 .....       | 125 |
| 6.4.2 最短路径 .....        | 128 |
| 6.5 拓扑排序 .....          | 135 |
| 6.5.1 AOV 网 .....       | 135 |
| 6.5.2 拓扑排序 .....        | 136 |
| 6.6 本章小结 .....          | 139 |
| 6.7 习题 .....            | 140 |
| 6.8 实训题 .....           | 141 |
| 实训 图的存储和遍历 .....        | 141 |
| <b>第7章 排序</b> .....     | 143 |
| 7.1 基本概念 .....          | 143 |
| 7.2 插入排序 .....          | 144 |
| 7.2.1 直接插入排序 .....      | 144 |
| 7.2.2 希尔排序 .....        | 146 |
| 7.3 交换排序 .....          | 148 |
| 7.3.1 冒泡排序 .....        | 148 |
| 7.3.2 快速排序 .....        | 149 |
| 7.4 选择排序 .....          | 151 |



|                     |     |
|---------------------|-----|
| 7.4.1 简单选择排序        | 152 |
| 7.4.2 堆排序           | 153 |
| 7.5 归并排序            | 156 |
| 7.6 基数排序            | 159 |
| 7.6.1 多关键字排序        | 159 |
| 7.6.2 链式基数排序        | 160 |
| 7.7 排序方法的比较         | 164 |
| 7.8 本章小结            | 165 |
| 7.9 习题              | 166 |
| 7.10 实训题            | 166 |
| 实训 排序算法的实现          | 166 |
| <b>第8章 查找</b>       | 168 |
| 8.1 查找的基本概念         | 168 |
| 8.2 基于线性表的查找方法      | 169 |
| 8.2.1 顺序查找法         | 169 |
| 8.2.2 折半查找法         | 170 |
| 8.2.3 分块查找法——索引顺序查找 | 172 |
| 8.3 树表查找法           | 173 |
| 8.3.1 二叉排序树         | 173 |
| 8.3.2 平衡二叉树         | 178 |
| 8.4 哈希表查找           | 185 |
| 8.4.1 哈希表与哈希查找      | 185 |
| 8.4.2 构造哈希函数的方法     | 186 |
| 8.4.3 处理冲突的方法       | 188 |
| 8.4.4 哈希表的查找分析      | 194 |
| 8.5 本章小结            | 196 |
| 8.6 习题              | 196 |
| 8.7 实训题             | 199 |
| 实训 查找的实现            | 199 |
| <b>附录</b>           | 200 |
| <b>参考文献</b>         | 260 |

# 第 1 章 概 述

## 学习目标

- ◆ 了解数据结构的发展、基本概念。
- ◆ 理解数据类型、数据的逻辑结果和存储结构等概念。
- ◆ 了解算法概念。
- ◆ 了解对算法性能评价的重要指标。

## 重点和难点

- ◆ 数据结构相关术语及概念。
- ◆ 栈和队列的基本操作实现算法及应用。

毋庸置疑,如今的计算机已深入到人类社会的各个领域。计算机的应用不再局限于科学计算,而更多地用于管理、控制和数据处理等非数值计算的处理工作。与此相应,计算机加工处理的对象由纯粹的数值发展到字符、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。为了编写出一个“好”的程序,必须分析待处理的对象的特性以及各处理对象之间存在的关系。这就是“数据结构”这门课程形成和发展的背景。

## 1.1 什么是数据结构

数据结构的起源,在于程序设计经验的积累,通过对程序设计方法学的研究,程序设计已经从一种技巧发展成为一门学问。1986 年美国 T. O. Knuth(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 年代末到 70 年代初,出现了大型程序,软件也相对独立,结构化程序设计成为程序设计方法学的主要内容,人们逐渐重视数据结构,认为程序设计的实质是对确定问题选择一种好的结构,加上设计一种好的算法。从 70 年代中期到 80 年代初期,各种版本的数据结构著作相继问世。

数据结构的研究不仅涉及计算机硬件的研究范围,而且和计算机软件的研究有着更加密切的联系,无论是操作系统还是编译系统,都涉及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,以使查找和存取数据元素更加方便。在计算机科学中,数据结构不仅是一般程序设计的基础,而且是设计和实现编译程序、操作系统、数据库系统以及其他系统程序和大型应用程序的重要基础。

## 1.2 基本概念和术语

在本节中,我们将介绍数据结构相关的基本概念和术语。这些概念和术语将在以后

的章节中多次出现。

### 1. 数据

数据(Data)是对客观事物的符号表示,是所有能输入到计算机中并被计算机程序处理的符号的总称,包括数值、字符、声音、图像等。它是计算机程序加工的“材料”。比如,一个用简单运算(加、减、乘、除)方法解决学生成绩统计问题的程序,其处理对象是整数或实数类的数值,类似记事本这类文字处理程序的处理对象是字符等。

### 2. 数据元素的数据项

数据元素(Data Element)是组成数据的基本单位,在计算机程序中往往将数据元素作为一个整体进行考虑和处理。

数据项(Data Item)是数据元素的基本组成部分,是数据元素中不可再分的数据单元。

一般来说,数据项在程序设计语言中是可以标准类型来描述的,而数据元素则常常需要由自定义类型进行描述。比如,在学生管理系统中,描述一个学生基本信息的数据可以称为数据元素,而描述学生信息中的“学号”、“姓名”、“年龄”、“性别”、“入学成绩”、“班级”、“籍贯”等则为数据项。

### 3. 数据对象

数据对象(Data Object)是指性质相同的数据元素的集合,是数据的一个子集。比如,实数集、整数集和字符集等。

### 4. 数据结构

数据结构(Data Structure)是指相互之间存在一种或多种特定关系的数据元素的集合,即各元素相互间存在一种“结构”的关系。

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。数据元素之间的相互关系包括三个方面:数据的逻辑结构、存储结构和操作集合。

### 5. 逻辑结构

数据元素间的逻辑关系称为逻辑结构。依据数据元素间关系的不同特性,有四种基本结构。

① 集合 表示结构元素之间只存在“同属于一个集合”这一单一的关系。

② 线性结构 表示结构元素之间存在一对一的关系。

③ 树形结构 表示结构中数据元素之间的关系存在一对多的关系。比如,家族中的家谱编排、单位里的上下级关系等。

④ 图状结构或网状结构 表示结构中的数据元素之间存在多对多的关系。比如,现实生活中的关系网等。

图 1-1 为上述四类基本结构的示意图。

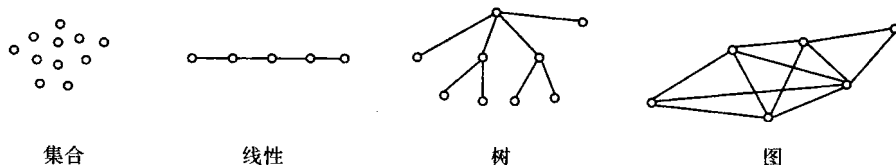


图 1-1 四类基本结构关系示意图

数据的逻辑结构包括两大类:线性结构,包括线性表、队列、栈、字符串、数组、广义表;非线性结构,包括树和图。

#### 6. 存储结构

数据在计算机中的表示(又称映像)称为数据的物理结构,也叫存储结构。它是逻辑结构在计算机中的实现,包括数据元素的表示和关系的表示。数据元素的关系在计算机中有两种不等的不同的表示方法:顺序映像存储结构和非顺序映像。由此得到两种不同的存储结构:顺序存储结构和非顺序存储结构(也叫链式存储结构)。初学者要注意理解逻辑结构和物理结构的差异。逻辑结构表示数据元素之间的逻辑关系,是抽象的;而存储结构是逻辑结构在计算机中的实现,是面向计算机、面向现实世界的。在后面将要介绍的算法中可以看到,任何一个算法的设计都取决于所选定的数据逻辑结构,而算法的实现则依赖于所采用的存储结构。

#### 7. 操作集合

在结构上的操作是数据结构这门课程要研究的重要内容。数据结构是研究一类数据的表示及其相关运算的操作。一般来讲,数据的操作主要有以下几种。

- ① 添加 在数据结构中指定的位置上新增数据元素。
- ② 删除 删去数据结构中某个指定的数据元素。
- ③ 修改 改变数据结构中某个位置的数据元素值。
- ④ 查询 在数据结构中查找某个给定要求的数据元素。
- ⑤ 排序 对数据结构中的元素重新排列顺序,使数据元素按某种顺序排列。

#### 8. 数据类型

数据类型(Data Type)和数据结构关系非常密切,在早期的高级程序设计语言中,数据类型用以刻画操作对象(程序)的特性。在用高级语言编写的程序中,每个变量、常量和表达式都具有一个确定的数据类型。比如,C语言中的短整型(int)、字符型(char)等。类型显式或隐式地规定了程序执行期间变量或表达式所有可能的取值范围,以及在这些数据上所允许的操作。因此,学者们也将数据类型定义为一个值的集合和定义在这个集合之上的一组操作的总称。比如,在C语言中定义一个实型变量,则定义在其上的操作为加、减、乘、除等算术运算。

按照数据元素取值的不同特性,高级语言中一般都包括两种数据类型:原子类型和结构类型。原子类型的值是不可再分的,像C语言中的int类型、char类型、指针类型等;结构类型的值是由若干组成部分按某种结构组成的,可以再次分解,且各组成部分可以是原子类型,也可以是结构类型。

## 1.3 算法描述和算法分析

### 1.3.1 算法的概念

算法(Algorithm)是对特定问题求解过程的一种描述,是指令的有限集合,其中,每一条指令表示一个或多个操作。算法具有以下五个特性。

### 1. 有穷性

表示一个算法总是在执行有穷步之后结束,不能无限循环下去。

### 2. 确定性

算法中每一条指令应有确切的含义,无二义性。

### 3. 可行性

算法中所描述的操作都是可以通过已实现的基本运算执行有穷次来实现的。

### 4. 输入

一个算法可以有 0 个或多个输入。

### 5. 输出

一个算法有一个或多个输出,不同的输出取决于不同的输入。

算法的上述五种特性中,有穷性、确定性和可行性是最重要的特性。描述算法的工具具有符号语言、自然语言及计算机高级语言。读者在学习过程中,需要特别注意算法与程序的区别。两者的关系是:程序是采用某种计算机语言并遵照这种计算机语言的语法来描述的算法。

## 1.3.2 算法设计的要求

一般来说,用计算机解决一个具体问题时,必须设计出一个解决问题的算法,最后编成程序、进行测试、调整直至得到最终解答。一个好的算法应考虑以下目标。

### 1. 正确性

算法的正确性表示算法应该满足具体问题的需求。设计或选择的算法应当能正确反映这种需求,否则,算法的正确与否的衡量标准就不存在了。其中“正确”之意大致可以分为四个层次。

(1) 程序不含语法错误。

(2) 程序对于几组输入数据能够得出符合要求的结果。

(3) 程序对于精心选择的典型、苛刻而带有刁难性的输入数据能够得到符合要求的结果。

(4) 程序对于一切合法的输入数据都能产生符合规格说明要求的结果。

在以上四层含义中,达到第四层意义下的正确性是极为困难的,因为不同的输入数据可能是非常巨大的一个数量,逐一验证是不现实的。一般对于大型软件来说,要进行专业测试,通常是以第三层意义的正确性来作为衡量程序是否合格的标准。

### 2. 可读性

算法首先是提供给人阅读和交流,其次才是被机器执行。可读性好的算法有助于他人对算法的正确理解和对程序的维护,难懂晦涩的程序容易隐藏错误且难以调试和修改。

### 3. 健壮性

一个好的算法不仅能处理正确的数据也能应对非法数据的输入。对于用户输入的非法数据,好的算法应能适当做出正确的反应或进行相应的处理,而不会产生莫名其妙的输出结果。比如,用户登录自己的邮箱时,输入错误的账号或密码时,系统能及时把错误信息反馈给用户,而不是无反应或结束登录操作。

#### 4. 效率与存储量需求

一般来说,算法的效率是指算法的执行时间。同一问题可能存在多个不同的算法,在多个算法中执行时间短的算法效率高。存储量需求是指算法在执行过程中所需要的最大存储空间。效率和存储量需求都与所求解问题的规模有关。

### 1.3.3 算法的描述

算法描述了数据对象的元素之间的关系,描述算法的工具可以是高级程序设计语言、流程图甚至自然语言,只要能准确描述出解决问题的步骤即可。

程序是算法在计算机中的实现,与所用计算机及所用语言有关,必须符合该语言的语法,并考虑语言的特点。

一般描述求解问题的算法需要经历以下几步。

- (1) 找出与求解有关的数据元素之间的关系。
- (2) 确定在某一数据对象上所应用的运算。
- (3) 考虑数据元素的存储表示。
- (4) 设计实现求解的算法,并用语言加以描述。

用高级语言描述算法具有严格准确的优点,但也有语言细节过多的弱点,因此,一般常用“类C语言”、“类PASCAL语言”、“类C++语言”或“类Java语言”来描述,描述时可以忽略语言的语法细节。为了便于学生掌握算法,本书所有算法一律采用C语言来描述。

### 1.3.4 算法性能的评价

评价一个算法,除了考虑正确性、可读性和健壮性这三个最基本的特性之外,还要考虑算法的效率问题。效率的评价标准往往是时间和空间两个方面。时间反映算法在计算机上执行时所消耗的时间,同一求解问题执行时间短的算法要优于执行时间长的算法。空间是指算法执行过程中所耗费的存储空间。

#### 1. 算法的时间效率

一个算法的执行时间大致等于其所有语句执行时间的总和,而语句的执行时间是该条语句执行次数和执行每次所需时间的乘积。语句执行的绝对时间要综合考虑机器的软、硬件环境,所以用绝对时间来衡量算法的时间是不太现实的。因此,算法分析时一般是忽略计算机软、硬件因素,只关注算法本身,可以认为一个特定算法执行时间的大小,只依赖于问题的规模(用 $n$ 表示)。一般情况下,从算法中选取一种操作作为基本操作,并以该操作的重复执行次数作为算法的时间度量,通常表示为问题规模 $n$ 的函数 $f(n)$ ,则算法的时间度量可记为

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

它表示随着问题规模 $n$ 的增长,算法执行时间的增长率与 $f(n)$ 的增长率相一致,称为算法的渐进时间复杂度,简称时间复杂度。时间复杂度是评价算法时间效率的指标。

一般情况下,对一个问题或一类算法只需选择一种基本的操作来讨论时间复杂度,有时需要考虑多种操作。

比如,下面的三个程序段中,黑体字部分可以选作基本的操作。

```
(1) sum=1000;
    for(i=1;i<=n;i++) sum -=i;
(2) for(i=1;i<n;i++)
    { k=r[j];
      for(j=i+1;j<n+1;j++)
        if (r[j].key>r[k].key) k=j;
      if(k!=i)
        r[i]<-->r[j];
    }
(3) {sum++;s=100;}
```

程序段(1)的时间复杂度为  $O(n)$ ;程序段(2)的基本操作有两个,内层比较执行次数为  $n^2$ ,交换操作的执行次数为  $n-1$ ,则该程序段的时间复杂度为  $O(n^2)$ ;程序段(3)的语句只执行 1 次,因此,  $T(n)=O(1)$ 。

下面列出了几种常见的时间复杂度数据量级。

$O(1)$ :常量级

$O(n)$ :线性级

$O(\lg n)$ :对数级(较好)

$O(n^2)$ :平方级

$O(2^n)$ :指数级(最差)

## 2. 算法空间效率

空间复杂度作为算法所需存储空间的度量,记作:  $S(n)=O(f(n))$ ,  $n$  为问题规模。

一般情况下,一个程序在计算机上执行时,除了需要存储空间本身所用的指令、常数和输入数据之外,还需要一些对数据进行操作辅助存储空间。其中,对于输入数据所占的存储量只取决于问题本身,与算法无关,因此,我们只需要分析算法在实现时所需要的辅助空间就可以了。若算法执行时所需要的辅助空间相对于输入数据量而言是个常数,则称这个算法为原地工作,辅助空间为  $O(1)$ 。

## 1.4 本课程学习指导

数据结构课程的学习不单纯是知识性学习,更重要的是专业能力的培养。在学习中应注重算法设计与程序设计的训练,通过实践形成良好的程序设计风格,并培养进行复杂程序设计的能力,这是开设数据结构课程的根本目的。

在学习方法上,对于基本信息结构这部分内容,注重的是“结构”及其实现,淡化算法效率分析的概念,对于线性表、栈与队列、串、数组、树、图等典型的数据结构,应理解逻辑结构的定义特点,并掌握其不同存储结构下基本操作的实现。对于每一种典型算法,除了掌握 C 程序描述,更重要的是掌握算法的基本思想,以便使用不同的语言加以实现,并了解每一种算法的时间复杂度和空间复杂度,建立算法效率的思想。

## 1.5 本章小结

本章重点介绍了以下概念。

- 数据类型、数据的逻辑结构和存储结构等概念。
- 算法、算法效率度量以及算法分析知识。

通过对本章的学习,对数据结构课程的研究内容和学习目的有了初步了解,为学习后面内容做好准备。

## 1.6 习 题

### 1. 填空题

(1) 数据元素之间的逻辑关系又称为\_\_\_\_\_,根据数据元素之间的不同关系,有线性结构、\_\_\_\_\_、\_\_\_\_\_、集合结构。

(2) 评价算法性能优劣的指标有两个,它们是\_\_\_\_\_和\_\_\_\_\_。

2. 分析下列程序段的时间复杂度。以黑体标识的语句为基本操作。

```
(1) x=50;
    y=0;
    while(x>y)  y++;
(2) x=100;y=200 ;
    while(y>0)
        if(x>10) {x=x-10 ;y--;}
        else x++;
(3) for(i=1;i<=n;i++)
        for(j=1;j<=n;j++)
            sum++;
(4) for(i=1;i<=n;i++)
        for(j=i;j<=n;j++)
            sum--;
```



## 第2章 线性表

### 学习目标

- ◆ 理解线性表的逻辑定义和线性表的特征。
- ◆ 掌握线性表的顺序存储结构及其算法的实现。
- ◆ 掌握线性链表的描述及其算法的实现。
- ◆ 理解循环链表和双向循环链表的描述。

### 重点和难点

- ◆ 线性表的顺序存储结构及其基本操作。
- ◆ 线性链表的存储结构及其基本操作。
- ◆ 线性表的应用。

## 2.1 什么是线性表

线性表是一种最简单、最基础的数据结构,它的使用非常广泛。这种数据结构的数据元素之间是一对一的关系,故称为线性表。本节讨论线性表的基本概念,包括线性表的定义以及线性表的基本操作。

线性表(Linear List)是由  $n$  个数据元素组成的有限序列,其中  $n$  为数据元素的个数,即线性表的长度。当  $n=0$  时,称为空表;当  $n>0$  时,线性表记为  $(a_1, a_2, \dots, a_i, \dots, a_n)$ 。

线性表可以用一个标识符来命名,如果用  $A$  来表示线性表,则  $A=(a_1, a_2, \dots, a_i, \dots, a_n)$ 。线性表中元素的类型可以是高级语言提供的基本数据类型或由用户自定义的数据类型,如整型、实型、字符型和结构体等。为了方便起见,在本书的算法、程序以及例题中,大多数数据元素类型都用整型数据来表示。

在线性表中,元素之间存在着线性的逻辑关系:表中有且仅有一个开始结点,或称表头结点  $a_1$ ;有且仅有一个终端结点,或称表尾结点  $a_n$ ;除开始结点外,表中的每个结点  $a_i(1<i\leq n)$  均只有一个前驱结点;除终端结点外,表中的每个结点  $a_i(1\leq i<n)$  均只有一个后继结点。数据元素在线性表中位置只取决于它的序号,如

$$\begin{array}{lcl} \text{线性表} & (a_1, a_2, \dots, a_i, \dots, a_{n-1}, a_n) \\ \text{序号} & 1 & 2 \quad \dots \quad i \quad \dots \quad n-1 \quad n \end{array}$$

在日常生活中有许多线性表的例子。例如,26个英文字母(A,B,C,...,Z)就是一个线性表,表中的每一个字母是一个数据元素。再如,学生考勤表、图书馆图书目录等也是线性表。

下面请看一个学生考勤表的例子,如表2-1所示。