

 高职高专“十一五”规划示范教材




徐翠霞 主编

# 数据结构简明教程 (C语言描述)

 北京航空航天大学出版社



 高职高专“十一五”规划示范教材

# 数据结构简明教程(C语言描述)

徐翠霞 主编

北京航空航天大学出版社

## 内 容 简 介

本书系统地介绍了各种常用的数据结构,内容丰富,概念讲解清楚,叙述严谨流畅,逻辑性强。书中配备了大量的案例,每个案例都经过精心的考虑,既能帮助读者理解知识,又具有启发性。本书对给出的每一种算法,均先描述了它的基本思路和要点,使得算法清晰易读,便于学生理解和掌握。

本书共分9章,内容包括绪论,线性表,栈和队列,串、多维数组和广义表,树和二叉树,图,查找,排序和综合实训。

本书可作为全国高等学校计算机类专业或信息类专业的教材,也可供广大从事计算机工程与应用的科技工作者参考。

### 图书在版编目(CIP)数据

数据结构简明教程: C语言描述/徐翠霞主编. —北京:

北京航空航天大学出版社, 2009.9

ISBN 978-7-81124-879-1

I. 数… II. 徐… 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(2009)第142827号

### 数据结构简明教程(C语言描述)

徐翠霞 主编

责任编辑 魏军艳

\*

北京航空航天大学出版社出版发行

北京市海淀区学院路37号(100191) 发行部电话:010-82317024 传真:010-82328026

<http://www.buaapress.com.cn> E-mail: [bhpress@263.net](mailto:bhpress@263.net)

北京市媛明印刷厂印装 各地书店经销

\*

开本:787×960 1/16 印张:18 字数:403千字

2009年9月第1版 2009年9月第1次印刷 印数:3000册

ISBN 978-7-81124-879-1 定价:35.00元(含光盘1张)

# 前 言

在计算机及其应用的各个领域,都会用到各种各样的数据结构,学会分析、研究计算机加工对象的特性,选择合适的数据结构和存储表示,以及编制相应的实现算法,是计算机工作者不可缺少的技能。因此,“数据结构”是高等院校计算机专业教学中的一门重要的专业基础课。在我国当前的计算机专业教学计划中,它是核心课程之一。

目前,市面上大多数有关“数据结构”的教材都采用传统编排模式,先是总体介绍,然后是基础知识,再是算法体系等。这种方法以知识点为主线,拘泥于语法细节,而忽略了程序本身的重要性。

基于案例教学过程的实践和思考,更是为了培养读者的编程能力,编者提出了这样一个问题:能不能打破传统教材和讲授模式,先以案例入手,提出解决问题的方法和思路,分析问题需要的知识点,然后根据需要讲解知识点,再解决提出的问题,最后举一反三,并以应用实例加以提升和巩固知识点,实现综合应用的目的。适合这样讲授的教材或参考书目市面上很少,编者经过不断探讨并通过多年的案例教学经验积累,最终形成了本书。

案例教学是计算机语言教学最有效的方法之一。好的案例对学生理解知识、掌握如何应用知识十分重要。本书以指导案例教学为目的,围绕教学内容组织案例,对学生的知识和能力训练具有很强的针对性,主要特色如下:

- (1) 以知识线索设计案例,分解知识点,有明确的目的和要求,针对性强。
- (2) 选择有代表性的案例,突出重点知识的掌握和应用。
- (3) 将技术指导、代码与分析、应用提高、相关知识有机结合。
- (4) 注意新方法、新技术的应用。
- (5) 处理好具体实例与思想方法、局部知识应用与综合应用的关系。
- (6) 强调实用性,培养应用能力。

本书中每一个案例的结构模式为“案例说明→案例目的→技术要点→代码及分析→相关知识及注意事项”。每一章均包含多个案例,并配有相应的习题。通过强化案例和实训教学,加深学生对理论知识的理解。

本书在内容的组织上,本着由浅入深、循序渐进的原则,注重基本知识和基本概念的介绍,结合案例重点介绍实用性较强的内容,对难度过大的内容只作少量介绍,使学生有的放矢,掌握所学内容。

本书注重培养学生的实际应用技能和综合解决问题的能力,使学生能熟练掌握和运用理论知识解决实际问题,达到学以致用为目的,能真正地为培养新世纪的适用型人才出一份力。

本书配有教学光盘。其内容兼顾了随教学进度安排的特色实验,也考虑到全部教学内容完成后的课程设计和模拟试题,因此教学光盘是本书的最佳教学辅助资料。

本书由潍坊学院的徐翠霞担任主编。参与本书编写工作的有:潍坊学院徐翠霞,山东潍坊科技学院谭娟,山东潍坊职业学院姜伟强和广东省惠州学院计算机系赵义霞。其中,第3章、第5章、第6章和整个教学光盘由徐翠霞编写;第2章和第7章由谭娟编写;第8章和第9章由姜伟强编写;第1章和第4章由赵义霞编写。全书由徐翠霞负责统稿。

由于时间仓促,加之作者水平有限,若书中有疏漏和不足,恳请广大读者批评指正,以使本书得以改进和完善。

编 者  
2009年6月

# 目 录

## 第 1 章 绪 论

1.1 基本概念和术语 .....	1
案例 1.1 数据模型的确定 .....	1
1.2 算法和算法评价 .....	4
案例 1.2 矩阵乘法算法的时间复杂度分析 .....	4
本章小结 .....	7
习题 1 .....	8

## 第 2 章 线性表

2.1 线性表的顺序存储 .....	9
案例 2.1 模拟集合的基本运算 .....	9
2.2 线性表的链式存储 .....	18
案例 2.2 一元多项式的加法运算 .....	18
案例 2.3 采用循环链表模拟约瑟夫(Josephus)问题 .....	32
本章小结 .....	38
习题 2 .....	38

## 第 3 章 栈和队列

3.1 栈 .....	40
案例 3.1 数制转换(十进制数转换为二、八、十六进制数) .....	40
案例 3.2 后缀表达式的求值 .....	50
3.2 栈与递归 .....	57
案例 3.3 迷宫问题 .....	57
3.3 队 列 .....	65
案例 3.4 农夫过河问题 .....	65
本章小结 .....	79
习题 3 .....	79

## 第 4 章 串、多维数组和广义表

4.1 串 .....	80
案例 4.1 模拟“天书密码”的生成和破解 .....	80
案例 4.2 无回溯的模式匹配 .....	88

4.2 多维数组和广义表	97
案例 4.3 稀疏矩阵的转置	97
本章小结	109
习题 4	109
<b>第 5 章 树和二叉树</b>	
5.1 树和二叉树的存储表示	111
案例 5.1 完全二叉树的基本运算	111
5.2 树和二叉树的遍历	123
案例 5.2 表达式树的构造及输出	123
5.3 线索二叉树	135
案例 5.3 中序线索二叉树及其遍历	135
5.4 哈夫曼树及其应用	141
案例 5.4 哈夫曼编码器	141
本章小结	148
习题 5	148
<b>第 6 章 图</b>	
6.1 图的存储和遍历	150
案例 6.1 图的连通性判定	150
6.2 最小生成树和最短路径	165
案例 6.2 局域网络布线问题	165
6.3 有向无环图及其应用	178
案例 6.3 教学计划的编制	178
本章小结	187
习题 6	187
<b>第 7 章 查找</b>	
7.1 线性表的查找	189
案例 7.1 学生信息表的分块查找	189
7.2 树表的查找	200
案例 7.2 借助二叉排序树实现单词统计	200
7.3 散列表的查找	215
案例 7.3 散列表的双散列探测查找	215
本章小结	226
习题 7	226

---

**第 8 章 排 序**

8.1 插入排序 .....	227
案例 8.1 希尔排序 .....	227
8.2 交换排序 .....	233
案例 8.2 双向冒泡排序 .....	233
8.3 选择排序 .....	238
案例 8.3 堆排序 .....	238
8.4 归并排序和基数排序 .....	245
案例 8.4 链式基数排序 .....	245
本章小结 .....	253
习题 8 .....	253

**第 9 章 综合实训**

实训 1 停车场管理 .....	255
实训 2 五叉路口交通灯的管理系统 .....	264
本章小结 .....	270
习题 9 .....	271

**附录 常用数据结构的类型定义****参考文献**



# 第 1 章 绪 论

**教学目标与要求:** 本章主要介绍数据结构中常用的基本概念和术语以及学习数据结构的意义。通过本章的学习,要求:

- ◇ 熟悉各名词、术语的含义,掌握基本概念,特别是数据的逻辑结构和存储结构之间的关系。分清哪些是逻辑结构的性质,哪些是存储结构的性质;掌握集合、线性结构、树和图等常用的数据结构。
- ◇ 掌握算法的特性和算法描述的方法。
- ◇ 掌握评价算法的一般规则,算法的时间复杂度的定义和数量级表示,时间复杂度在最好、平均、最坏 3 种情况下的含义,算法的空间复杂度的定义和数量级表示。
- ◇ 掌握计算语句频度和估算算法时间复杂度的方法。

**教学重点与难点:** 本章重点是了解数据结构的逻辑结构、存储结构及数据的运算 3 方面的概念及相互关系,难点是算法复杂度的分析方法。

## 1.1 基本概念和术语

### 案例 1.1 数据模型的确定

#### 【案例说明】

在程序设计课程的学习中,计算机解题的基本方法是:分析问题,确定数据模型;设计相应的算法;编写程序;反复调试程序直至得到正确的结果。有些问题的数据模型可以用具体的代数方程、矩阵等来表示。然而,更多的实际问题是无法用数学方程来表示的。

对于下面的 3 个问题,试分别设计出合适的数据模型或数据结构。

- (1) 图书馆的书目检索系统自动化问题。
- (2) 计算机和人对弈问题。
- (3) 多叉路口交通灯的管理问题。

#### 【案例目的】

- (1) 了解数据结构的基本概念和术语,能够根据需求为实际问题确定合适的数据模型。
- (2) 掌握线性表、树和图 3 种数据结构的基本特点。

## 【技术要点】

### (1) 图书馆的书目检索系统自动化问题

在图书馆内有各种名目的卡片：有按书名编排的，有按作者编排的，还有按分类编排的，等等。若实现自动检索，则计算机处理的对象便是这些卡片上的书目信息。

列在一张卡片上的一本书的书目信息可由登录号、书名、作者名、分类号、出版单位和出版时间等若干项组成，每一本书都有唯一的一个登录号，但不同的书目之间可能有相同的书名、相同的作者名或者相同的分类号。

在书目自动检索系统中，可以建立一张按登录号顺序排列的书目文件和3张分别按书名、作者名和分类号顺序排列的索引表。

由这4张表构成的文件便是书目自动检索的数据结构，可谓线性结构。

### (2) 计算机和人对弈问题

棋盘为 $3 \times 3$ 的方格，当一方的3个棋子占同一行、同一列或同一对角线时便为胜方。

在对弈问题中，计算机操作的对象是对奕过程中可能出现的棋盘状态——格局。从一个棋盘格局可以派生出几个格局，因此，若将从对弈开始到结束的过程中所有可能出现的格局都画在一张图上，则可得到一棵倒长的“树”。“树根”是对弈开始之前的棋盘格局，而所有的“叶子”就是可能出现的结局，对弈的过程就是从树根沿树枝到某个叶子的过程。

树是该问题的数据模型，它也是一种数据结构。

### (3) 多叉路口交通灯的管理问题

在多叉路口需设几种颜色的交通灯才能既使车辆相互之间不碰撞，又能达到车辆的最大流通。假设多岔路口既有单行道也有双行道。在路口有若干条可行的通路，其中有的可以同时通行，而有的不能同时通行。那么，在路口应如何设置交通灯进行车辆的管理呢？

通常，这类道路交通问题的数据模型是一种称为图的数据结构。例如，在此例的问题中，可用图中一个顶点表示一条通路，而通路之间互相矛盾的关系以两个顶点之间的连线表示。

## 【相关知识及注意事项】

### 1. 数据

数据(Data)是客观事物的数字、字符以及所有能输入计算机中并被计算机程序处理的符号的集合，它是计算机程序加工的“原料”。常见的数据可分为数值数据和非数值数据。数值数据包括整数、实数，主要用于工程计算、科学计算和商务处理等；非数值数据包括字符、文字、图形、图像和语音等。

### 2. 数据元素

数据元素(Data Element)是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。一个数据元素可以由一个或多个数据项(Data Item)组成，其中能够唯一标识一个

数据元素的数据项称为关键字。数据项是数据不可分割的最小单位。

例如,学籍管理系统中学生信息表的每一个学生记录就是一个数据元素。它包括学生的学号、姓名、性别、籍贯、出生年月及成绩等数据项。通常,在解决实际问题时是把每个学生记录当做一个基本单位进行访问和处理的。

### 3. 数据结构

数据结构(Data Structure)是指按照某种关系组织起来的一组数据,按照一定的存储表示方式把它存储在计算机中,并在这些数据上定义了一个运算的集合。数据结构一般包括以下3个方面的内容:数据的逻辑结构、数据的存储结构以及数据的运算。

#### (1) 数据的逻辑结构

对数据之间关系的描述称为数据的逻辑结构。在任何问题中,数据元素之间都不会是孤立的,在它们之间都存在着这样或那样的关系,根据数据元素之间关系的不同特性,通常有下列4类基本结构:

##### ① 集合

在集合中,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。

##### ② 线性结构

该结构中的数据元素之间存在着一对一的关系。

##### ③ 树型结构

该结构的数据元素之间存在着一对多的关系。

##### ④ 图形结构或网状结构

该结构的数据元素之间存在着多对多的关系。

图 1.1 所示为表示上述 4 类基本结构的示意。

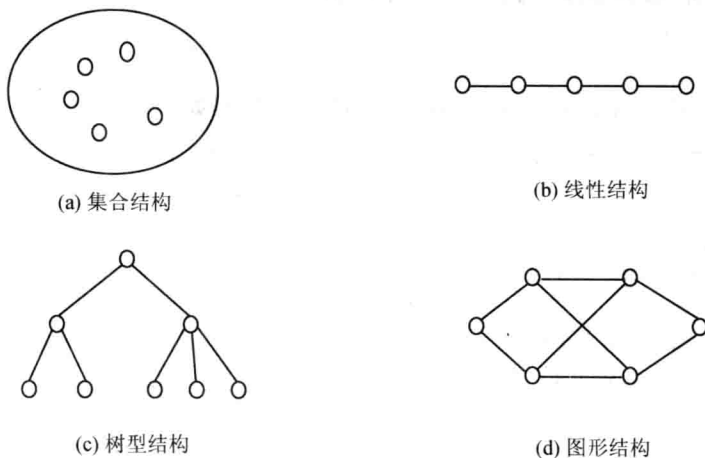


图 1.1 4 类基本结构

## (2) 数据的存储结构

数据的逻辑结构是从逻辑关系上来描述数据,它与数据的存储无关,是独立于计算机的。而研究数据结构的目的是为了在计算机中实现对它的运算,为此还需要研究如何在计算机中表示一个数据结构。

数据结构在计算机中的表示称为数据的物理结构,或称存储结构。它所研究的是数据结构在计算机中的实现方法,包括数据结构中元素的表示及元素间关系的表示。

数据的存储结构可以采用顺序存储方法和链式存储方法。顺序存储方法是把逻辑上相邻的元素存储在物理位置相邻的存储单元中,由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法,通常借助于程序设计语言中的数组来实现。链式存储方法对逻辑上相邻的元素不要求其物理位置相邻,元素间的逻辑关系通过附设的指针来表示,由此得到的存储表示称为链式存储结构,链式存储结构通常借助于程序设计语言中的指针类型来实现。

数据的存储结构是逻辑结构在计算机中的实现,它是依赖于计算机的。

## (3) 数据的运算

数据的运算是在数据上所施加的一系列操作。数据的运算是定义在逻辑结构上的,而运算的具体实现要在存储结构上进行。数据的各种逻辑结构有相应的各种运算,每个逻辑结构都有一个运算的集合。例如,最常用的运算有检索(查找)、插入、删除、更新及排序等。

# 1.2 算法和算法评价

## 案例 1.2 矩阵乘法算法的时间复杂度分析

### 【案例说明】

已知两个  $n$  阶方阵  $A$  和  $B$ ,乘积为  $C=A \times B$ ,算法代码如下:

```
void Mult(int a[][n],int b[][n],int c[][n]){ /* C=A×B */
    int i,j,k;
    for(i=1; i<=n; ++i)
        for(j=1; j<=n; ++j) {
            c[i][j]=0;
            for(k=1; k<=n; ++k)
                c[i][j]+=a[i][k] * b[k][j];
        }
}
```

试分析该算法的时间复杂度。

**【案例目的】**

- (1) 了解算法的特性和算法描述的方法。
- (2) 掌握计算语句频度和估算算法时间复杂度的方法。

**【技术要点】**

该算法中,频度最大的语句是“ $c[i][j] += a[i][k] * b[k][j];$ ”,其频度为  $f(n) = n^3$ , 所以该算法的时间复杂度为  $T(n) = O(n^3)$ 。

由此可见,当有若干个循环语句时,算法的时间复杂度是由嵌套层数最多的循环语句中最内层语句的频度决定的。

**【相关知识及注意事项】****1. 算法及其特性**

算法是对特定问题求解步骤的一种描述。通俗地讲,一个算法就是一种解题方法。更严格地说,算法是由若干条指令组成的有限序列,其中每一条指令表示一个或多个操作。

一个算法应该具有下列特性:

**(1) 有穷性**

一个算法必须总是(对任何合法的输入值)在执行有穷步之后结束,且每一步都可在有穷时间内完成。

**(2) 确定性**

算法中的每一步必须有确切的含义,并且在任何条件下算法只有唯一的一条执行路径,即对于相同的输入只能得出相同的输出。

**(3) 可行性**

一个算法是可行的,即算法中描述的运算都可以通过已经实现的基本运算的有限次执行得以实现。

**(4) 输入**

一个算法具有零个或多个输入,它们是在算法开始前赋予算法最初的量。

**(5) 输出**

一个算法具有一个或多个输出,这些输出是同输入之间存在某种特定关系的量。

**2. 算法描述**

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。

算法还可以使用程序流程图、N-S图等算法描述工具来描述,其特点是描述过程简洁、明了。

算法也可以直接使用某种程序设计语言来描述,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这两者之间的矛盾,人们常常使用一种称为伪代码语言的描述方法来进行算法描述。伪代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。

### 3. 算法评价

究竟如何评价一个算法的好坏呢?显然,算法首先应该是正确的。此外,还要考虑以下3个因素:执行算法所耗费的时间;执行算法所耗费的存储空间,其中主要考虑辅助存储空间;算法应易于理解,易于编码,易于调试等。一个好的算法应有效使用存储空间和有较高的时间效率。

#### (1) 时间复杂度

一个算法所耗费的时间,是指该算法中每条语句的执行时间总和,而每条语句的执行时间是该语句的执行次数(也称为语句频度)与该语句执行一次所需时间的乘积。设执行每条语句所需的时间均为单位时间,则一个算法的时间耗费就是该算法中所有语句的频度之和。

一般情况下,算法中频度最大的语句执行次数是问题规模  $n$  的某个函数  $f(n)$ ,算法的时间量度记作  $T(n)=O(f(n))$ ,它表示当问题规模  $n$  趋向无穷大时,算法执行时间的增长率和  $f(n)$  的增长率相同,称为算法的渐近时间复杂度,简称时间复杂度。

例如,一个算法中频度最大的语句执行次数为  $f(n)=n^2+8n+5$ ,则  $T(n)=O(n^2)$ 。

有的情况下,很多算法的时间复杂度不仅仅是问题规模  $n$  的函数,还与它所处理的数据集的状态有关。这时,通常是根据数据集可能出现的最好情况和最坏情况,估计出算法的最好时间复杂度和最坏时间复杂度。有时,对数据集的分布做出某种假设(如等概率),并讨论算法的平均时间复杂度。

例如,设有一组数据,对其进行冒泡排序,排序结果使得该组数据由小到大排列。

```
void Bubble_sort(int a[],int n){
    /* 将 a 数组中的整数序列按自小到大的顺序排列 */
    int i,j;
    int change,temp;
    for(i=n-1,change=1;i>=1&&change;--i){
        change=0;
        for(j=0;j<i;++j)
            if(a[j]>a[j+1]){
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
                change=1;
            }
    }
}
```

该算法的基本操作是比较或交换序列中相邻的两个整数,当  $a$  中初始序列为自小至大有序时,排序过程中的比较次数为  $n-1$ ,交换次数为 0;当初始序列为自大至小有序时,排序过程中的比较次数为  $n(n-1)/2$ ,交换次数为  $3n(n-1)/2$ 。因此,冒泡排序算法的最好时间复杂度为  $O(n)$ ,最坏时间复杂度为  $O(n^2)$ 。除特别指明外,本书中涉及的时间复杂度均指最坏时间复杂度。

另外,当一个算法的执行时间是一个与问题规模  $n$  无关的常数时,该算法的时间复杂度为常数阶,记作  $T(n)=O(1)$ 。

常见的时间复杂度,按数量级递增排列为

$$O(1) < O(\log_2 n) < O(n) < O(n \log_2 n) < O(n^2) < O(n^3)$$

## (2) 空间复杂度

一个算法的空间复杂度,记作  $S(n)$ ,是指该算法在运行过程中所耗费的辅助存储空间,它也是问题规模  $n$  的函数。若一个算法所耗费的辅助存储空间与问题规模  $n$  无关,记作  $S(n)=O(1)$ 。

算法所耗费的存储空间包括 3 部分,即算法本身所占用的存储空间、算法的输入输出所占用的存储空间和算法在运行过程中临时占用的辅助存储空间。其中,算法的输入输出所占用的存储空间是由算法要解决的问题规模决定的,它不随算法的不同而改变。算法本身所占用的存储空间与实现算法的程序源代码长短有关,要压缩这部分的存储空间,就必须编写较短的算法。算法在运行过程中临时占用的辅助存储空间随算法的不同而异,有的算法只需要占用少量的临时工作单元,而且不随问题规模的大小而改变;有的算法需要占用的临时工作单元数与解决问题的规模  $n$  有关,它随着  $n$  的增大而增大。

例如,前面讨论的冒泡排序算法,在运行过程中临时占用的工作单元数与解决问题的规模  $n$  无关,即该算法的空间复杂度为  $O(1)$ ;而二路归并排序算法的空间复杂度为  $O(n)$ 。

另外,若一个算法是递归算法,其空间复杂度与递归所需的栈空间的大小有关。例如,快速排序算法的最好空间复杂度为  $O(\log_2 n)$ ,最坏空间复杂度为  $O(n)$ 。

## 本章小结

本章简要地介绍了数据、数据结构等基本概念;阐述了数据结构所包含的 3 个方面的内容,即数据的逻辑结构、数据的存储结构和数据的运算;讨论了线性结构和非线性结构的逻辑特征,以及数据存储的 4 种基本方法。希望读者学习这些内容和例子后,能对数据结构的基本概念,具有初步的认识和理解。

算法和数据结构密切相关,不可分割,本章引进了算法、算法的时间复杂度等概念,以及分

析时间复杂度的简易方法。

## 习题 1

1. 数据元素之间的关系在计算机中有几种表示方法? 各有什么特点?
2. 评价一个好的算法, 是从哪几方面来考虑的?
3. 解释和比较以下概念:

算法的时间复杂度 算法 语句频度

4. 对于一个数据结构, 一般包括哪 3 个方面的讨论?
5. 在编制管理通讯录的程序时, 什么样的数据结构较为合适? 为什么?
6. 若有 100 个学生, 每个学生有学号、姓名、平均成绩, 若一般无增删操作, 采用什么样的数据结构最方便? 写出这些结构。
7. 已知如下程序段:

```
for(i=n;i>0;i--)          {语句 1}
{
    x=x+1;                 {语句 2}
    for(j=n;j>=i;j--)      {语句 3}
        y=y+1;             {语句 4}
}
```

语句 1、语句 2、语句 3 和语句 4 的执行次数分别是多少?



## 第 2 章 线性表

**教学目标与要求：**本章主要介绍线性表的逻辑结构、各种存储表示方法、定义在逻辑结构上的各种基本运算，以及在存储结构上如何实现这些基本运算。通过本章的学习，要求：

- ◇ 了解线性表中数据元素之间的逻辑关系以及在计算机中表示这种关系的两种不同的存储结构，即顺序存储结构和链式存储结构。
- ◇ 熟练掌握这两种存储结构的描述方法，如链表中的头结点、头指针和首元结点、尾元结点的区别及循环链表、双向链表的特点等。
- ◇ 熟练掌握在顺序存储结构上实现线性表基本运算的方法，特别是插入、删除与定位等运算。
- ◇ 熟练掌握在各种链表上实现线性表基本运算的方法，能在实际应用中选用适当的链表结构。
- ◇ 能够从时间和空间复杂度的角度比较线性表两种存储结构的不同特点及使用场合。

**教学重点与难点：**本章重点是熟练掌握顺序表和单链表上实现的各种基本算法及相关的时间性能分析；难点是能够使用本章所学到的基本知识设计有效算法解决与线性表相关的应用问题。

### 2.1 线性表的顺序存储

#### 案例 2.1 模拟集合的基本运算

##### 【案例说明】

假设以两个线性表 A 和 B 分别表示两个集合（即同一表中的元素值各不相同），现要求另辟空间构成集合 C、D 和 E，其元素分别为集合 A 和 B 中元素的交集、并集和差集。试在顺序表上编写求集合 C、D 和 E 的算法。例如， $A=(5,11,3,8)$ ， $B=(8,6,11,20,15,2,9)$ 。程序运行结果如图 2.1 所示。

##### 【案例目的】

- (1) 掌握线性表的顺序存储结构。
- (2) 掌握顺序表的基本运算并能灵活应用。