



高等教育“十三五”应用型规划教材

数据结构

(JAVA版)

主编 陈 磊 李玉霞



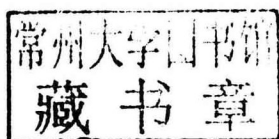
电子科技大学出版社

高等教育“十三五”应用型规划教材·计算机系列

数据结构 (Java 版)

主 编 陈 磊 李玉霞

副主编 董改芳 王来丽 陈冬梅



电子科技大学出版社

图书在版编目(CIP)数据

数据结构:Java 版/陈磊, 李玉霞主编. —成都:电子科技大学出版社,
2016. 11
ISBN 978-7-5647-3942-3

I. ①数… II. ①陈…②李… III. ①数据结构②JAVA 语言-程序
设计 IV. ①TP311.12②TP312.8

中国版本图书馆 CIP 数据核字(2016)第 228525 号

内 容 简 介

本书是根据社会发展及教学改革的新形势,基于培养社会需求的高素质应用型人才的目的,依托高等工科院校计算机类专业相关课程的基本要求而编写的基础教材。本书不但知识点详尽清晰,而且实例丰富有趣,项目的设置注重实用性、可行性和科学性,采用任务驱动方式完成项目开发,基础理论版块和实战训练版块相辅相成,充分调动学生学习的积极性和主动性,培养学生的实际开发能力,使学生形成软件开发思维。

本书可作为计算机相关专业的教材,也可作为从事计算机技术方面工作人员的参考书,对于计算机开发的爱好者来说,它也是一本较实用的自学读物。

数据结构(Java 版)

SHUJU JIEGOU JAVA BAN

主编 陈 磊 李玉霞

出 版:电子科技大学出版社(成都市一环路东一段 159 号电子信息产业大厦 邮编:610051)

策划编辑:刘 愚

责任编辑:刘 愚

主 页:www.uestep.com.cn

电子邮箱:uestep@uestep.com.cn

发 行:新华书店经销

印 刷:北京航天伟业印刷有限公司

成品尺寸:185mm×260mm 印张:13.75 字数:312 千字

版 次:2016 年 11 月第一版

印 次:2016 年 11 月第一次印刷

书 号:ISBN 978-7-5647-3942-3

定 价:32.00 元

■ 版权所有 侵权必究 ■

◆ 本社发行部电话:028-83202463;本社邮购电话:028-83201495。

◆ 本书如有缺页、破损、装订错误,请寄回印刷厂调换。

前 言

数据结构是计算机专业教学计划中的核心课程,也是计算机及相关专业考研和水平等级考试的必考科目。要从事和计算机科学与技术相关的工作,尤其是计算机应用领域的开发和研制工作,必须具备坚实的数据结构基础,许多课程(如操作系统、编译原理、数据库和人工智能等)也要用到数据结构的知识。数据结构课程的主要内容是研究和解决非数值计算的问题,讨论如何合理地组织、存储和处理数据的方法与技术;使读者掌握数据组织、存储和处理的常用方法和常用的算法思想及在实际中的应用技巧,为今后学习后续专业课和进行软件开发打下良好的基础。

本书详细解释了数据结构的基本概念,同时介绍了线性表、串、栈、队列和数组、树的结构和图的结构以及查找和排序等数据结构的相关知识,并给出相应的实例具体说明设计过程。书中的算法都是用 Java 语言编写,并给出了详细的注解。本书概念清楚,选材精练,叙述深入浅出,用了大量的实例、程序和图表来说明基本概念和方法,直观易懂。本书具有以下特色:

(1)采用 Java 语言,注重面向对象的编程,强调算法的逻辑性、严谨性、技巧性以及时间复杂度等。

(2)本着“加强基础,注重算法,循序渐进,因材施教,项目驱动”的教育理念,在算法的设计上做到简单明了,为了更好地理解理论知识,提倡从实用性和实践性的角度学习数据结构。

(3)通俗易懂、图文并茂,详略得当,也便于从事计算机工作的人员自学。

本书选用“项目导引”→“理论基础讲解”→“项目实现”→“实战训练”的架构编写。选用有应用价值、难度适中的项目案例,引入教学内容,激发读者兴趣,然后展开理论知识的学习,继而通过学习的理论知识完成相应的项目案例,最后提供有趣的实战训练项目案例,用于课后巩固理论基础,拓展教学深度和广度。这样的安排既提升了读者的学习兴趣,加强了实践训练,又能强调数据结构的原理和方法。

本书由重庆航天职业技术学院陈磊、包头职业技术学院李玉霞担任主编,内蒙古农业大学董改芳,黑龙江农业职业技术学院王来丽,宁夏大学陈冬梅担任副主编,全书由陈磊、李玉霞统编定稿。具体编写分工如下:项目三、项目四、项目五由陈磊编写,项目六、项目九由李玉霞编写,项目二由董改芳编写,项目七由王来丽编写,项目一、项目八由陈冬梅编写。本书在编写过程中,得到了宁夏回族自治区十三五电气信息类重点建设专业群项目的支持和帮助,是该项目的研究成果之一。

由于编者水平有限,书中错误和不妥之处在所难免,敬请读者批评指正。

编 者
2016 年 6 月

目 录

项目一 走进数据结构的世界	1
基础理论版块	1
任务1 初识数据结构	1
任务2 建立数据结构	2
任务3 解密算法与性能分析	5
实战训练版块	7
任务1 求简单算法的时间复杂度	7
任务2 求矩阵相乘算法的时间复杂度	8
任务3 求简单选择排序算法的时间复杂度	8
项目二 线性表	10
基础理论版块	10
任务1 认识线性表	10
任务2 定义顺序表	12
任务3 顺序表的基本运算	13
任务4 定义单链表	20
任务5 单链表的基本运算	22
任务6 顺序表与单链表的比较	30
实战训练版块	30
任务1 一元多项式相加	30
任务2 员工信息管理	37
项目三 栈与队列	42
基础理论版块	42
任务1 认识栈	42
任务2 认识顺序栈	43
任务3 顺序栈的基本运算	44
任务4 认识链栈	49
任务5 链栈的基本运算	50
任务6 认识队列	55
任务7 认识顺序队列	56
任务8 顺序循环队列的基本运算	58
任务9 认识链队列	63
任务10 链队列的基本运算	64

任务 11 栈与队列的比较	70
实战训练版块	70
任务 1 数制转换	70
任务 2 判断回文	71
项目四 字符串	73
基础理论版块	73
任务 1 认识字符串	73
任务 2 字符串存储结构	74
任务 3 顺序串的基本运算	74
任务 4 顺序串的模式匹配	79
实战训练版块	85
任务 1 密文风云	85
任务 2 宝宝作文快乐改	88
项目五 数组	93
基础理论版块	93
任务 1 认识数组	93
任务 2 数组的顺序存储结构	94
任务 3 特殊矩阵的压缩存储	95
任务 4 稀疏矩阵的压缩存储	96
实战训练版块	101
任务 1 奇幻的魔方	101
任务 2 矩阵相加	102
项目六 树与二叉树	106
基础理论版块	106
任务 1 认识树	106
任务 2 认识二叉树	108
任务 3 二叉树的遍历	111
任务 4 哈夫曼树	118
实战训练版块	124
任务 1 判断二叉树是否相等	124
任务 2 最佳判定树	125
项目七 图	128
基础理论版块	128
任务 1 认识图	128
任务 2 图的邻接矩阵存储结构	131
任务 3 图的邻接表存储结构	136
任务 4 图的遍历	141

任务 5 最小生成树	144
任务 6 最短路径	150
任务 7 拓扑排序	154
实战训练版块	155
任务 1 彩虹村之间路线图	155
任务 2 神秘的名单	157
项目八 查找	160
基础理论版块	160
任务 1 认识查找	160
任务 2 顺序查找	161
任务 3 折半查找	162
任务 4 分块查找	165
任务 5 二叉排序树查找	166
任务 6 哈希表查找	170
实战训练版块	175
任务 学生信息查询系统	175
项目九 排序	183
基础理论版块	183
任务 1 认识排序	183
任务 2 插入排序	184
任务 3 交换排序	188
任务 4 选择排序	192
任务 5 基数排序	201
实战训练版块	204
任务 1 美得冒泡啦	204
任务 2 排序功力琅琊榜	206
参考文献	209

项目一 走进数据结构的世界

本次项目首先了解数据结构的意义、数据结构的研究内容和基本概念,然后简单认识建立数据结构的常用方法,以及算法性能的分析,最后通过一些简单实例训练,加深对本项目学习内容的理解。

基础理论版块

任务1 初识数据结构



任务描述

了解数据结构的意义、数据结构的研究内容和基本概念。



任务实现

1. 数据结构的意义

随着计算机科学与技术的不断发展,计算机的应用领域已不再局限于科学计算,而更多地应用于控制、管理等非数值处理领域。与此相应,计算机处理的数据也由纯粹的数值发展到字符、表格、图形、图像、声音等具有一定结构的数据,处理的数据量也越来越大,这就给程序设计带来一个问题:应如何组织待处理的数据以及数据之间的关系(结构)。1968年高德纳(DonaU E. Knuth)教授开创了数据结构的最初体系,他所著的《计算机程序设计艺术》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。20世纪70年代初,数据结构作为一门独立的课程开始进入大学课堂。瑞士著名的计算机科学家沃思提出了以下公式:算法+数据结构=程序。这也说明了数据结构的重要性。

目前“数据结构”在计算机科学中是一门综合性的专业基础课,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。学习该课程的最终目的是使学生在学会利用计算机解决实际问题的同时学会有效地存储、组织和处理数据的方法,设计出相应的结构清晰、可移植性好、高效的算法,使学生获得求解问题的能力,即从实际问题中抽象出数学模型,选择合适的“数据结构”表示,再把解决问题的算法程序化,为以后课程的学习和计算机应用软件的研发奠定坚实的理论和实践基础。

2. 数据结构的研究内容

数据结构主要研究和讨论以下4方面问题。

- (1) 数据的各种逻辑结构和物理结构,以及它们之间的相应关系。
- (2) 对每种结构定义相适应的各种运算。

(3) 设计出相应的算法。

(4) 分析算法的效率。

3. 数据结构的基本概念

(1) 数据:所有能被计算机处理的符号的集合(信息)。

(2) 数据元素:是数据这个集合中的一个个体。

(3) 数据对象:具有相同特性的数据元素的集合。

(4) 数据项:数据元素常常还可以分为若干个数据项,数据项是数据具有意义的最小单位。

(5) 数据结构:是带有结构的数据元素的集合。结构就是数据元素之间的关系,描述数据元素之间的运算及运算规则,数据元素及其相互关系称为数据结构。

(6) 逻辑结构:指数据元素之间的结构关系。

(7) 物理结构:指数据结构在机内的存放及表示方式。

(8) 数据处理:指对数据进行查找、插入、删除、合并、排序、统计以及简单计算等的操作过程。

任务2 建立数据结构



任务描述

了解数据结构的逻辑结构与存储结构概念,认识建立数据结构的常用方法。



任务实现

数据结构是指同一数据元素类中各数据元素之间存在的关系,包括逻辑结构、存储结构(物理结构)和数据的运算。

数据的逻辑结构是从具体问题抽象出来的数学模型,是描述数据元素及其关系的数学特性,有时把逻辑结构简称为数据结构。逻辑结构是在计算机存储中的映像,形式地定义为 (K, R) (或 (D, S)),其中, K 是数据元素的有限集, R 是 K 上的关系的有限集。

1. 逻辑结构

数据结构有4种基本的逻辑结构。

(1) 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,如图1-1所示。



图1-1 线性结构

(2) 树型结构

结构中的数据元素之间存在着一对多的层次关系,如图1-2所示。

(3) 图状结构

结构中的数据元素之间存在着多对多的任意关系,如图1-3所示。

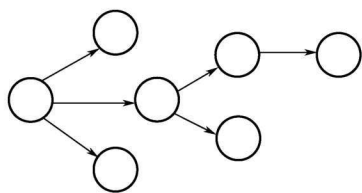


图 1-2 树型结构

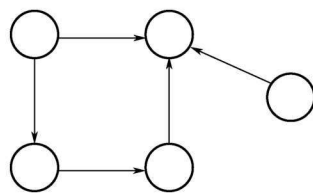


图 1-3 图状结构

(4) 集合结构

结构中的数据元素之间除了“同属于一个集合”的关系外,别无其他关系,则称这种结构为集合。在集合中,各元素是“平等”的,它们的共同关系是都属于同一个集合。如图 1-4 所示。

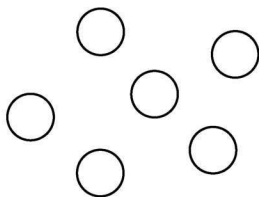


图 1-4 集合结构

(5) 生活中数据结构应用

① 图书馆的书目检索问题。每本书有一条记录,记录包括:书号、书名、作者、出版社、出版年月。书目的检索按记录中的任意数据项作为关键字进行查询。记录在数据库中按顺序排列,对象之间是简单的线性关系。这类数学模型称为线性的数据结构;算法:查找;模型:线性结构。

② 人机对弈问题。将每一种可能的棋盘格局存储在计算机里,对弈过程就是根据当前棋局搜索最优的棋局对策。棋盘格局之间的关系是一棵倒立的树结构;算法:搜索最优棋盘格局;模型:树结构。如图 1-5 所示。

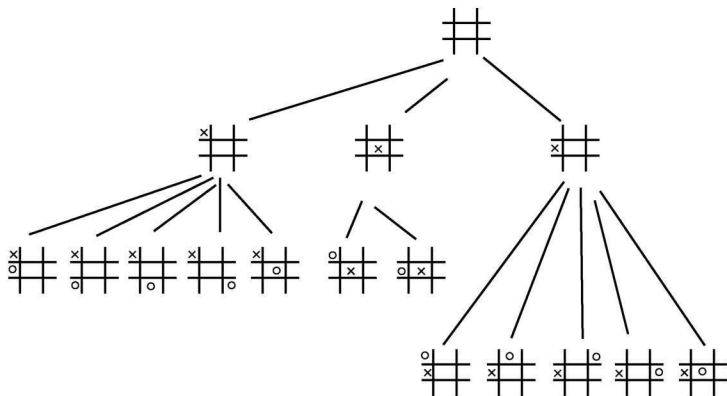


图 1-5 人机对弈问题的树结构

③ 煤气管道铺设问题。实际生活中如果进行煤气管道铺设项目,则需要涉及图的知识。例如图 1-6 中的(a)和(b)所示涉及的是图结构;其中算法是计算如何规划总投资最少;模型就是一个典型的图结构。

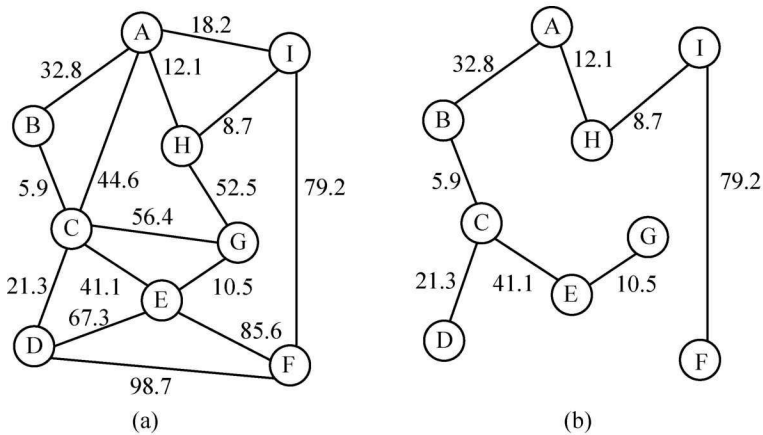


图 1-6 煤气管道铺设的图结构

因此,我们可以认为数据结构是一门讨论“描述现实世界实体的数学模型(非数值计算)及其上的操作在计算机中如何表示和实现”的学科。

2. 存储结构

数据结构在计算机中的表示(映像)称为数据的物理(存储)结构。它包括数据元素的表示和关系的表示。数据元素之间的关系有两种不同的表示方法:顺序映像和非顺序映像,并由此得到两种不同的存储结构:顺序存储结构和链式存储结构。

(1) 顺序存储方法:它是把逻辑上相邻的结点存储在物理位置相邻的存储单元里,结点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。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。

(2) 链接存储方法:它不要求逻辑上相邻的结点在物理位置上亦相邻,结点间的逻辑关系是由附加的指针字段表示的。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。

(3) 索引存储方法:除建立存储结点信息外,还建立附加的索引表来标识结点的地址。

(4) 散列存储方法:就是根据结点的关键字直接计算出该结点的存储地址。

数据结构中,逻辑上可以把数据结构分成线性结构和非线性结构。线性结构的顺序存储结构是一种顺序存取的存储结构,线性表的链式存储结构是一种随机存取的存储结构。线性表若采用链式存储表示时所有结点之间的存储单元地址可连续可不连续。逻辑结构与数据元素本身的形式、内容、相对位置、所含结点个数都无关。

任务3 解密算法与性能分析



任务描述

了解算法的概念及特性,算法设计的要求,初步理解算法的性能分析。



任务实现

1. 算法的概念与特性

(1) 算法概念

算法是一个有限的指令集,遵循指令流可以完成特定的功能。

(2) 算法的基本特性

有穷性:在执行有穷步骤之后一定能结束。

确定性:对于每种情况下所应执行的操作,在算法中都有确切的规定。

可行性:算法中的所有操作都可以通过已经实现的基本操作运算有限次进行实现。

有输入输出:每个算法都要有算法需要加工的数据,经加工后获得一定的结果。

(3) 算法与程序的区别

算法是转化的过程,是把输入转换成输出,而程序是用某种程序设计语言对算法的具体实现;程序可以是无穷的,而算法是有穷的;程序可以是错误的,算法必须是正确的;程序是用程序设计语言描述,可以在机器上执行;算法还可以用框图,自然语言等方式描述。

2. 算法的设计要求

一个优秀的算法应该具备的设计要求主要有以下几点。

(1) 正确性

算法的正确性是指算法至少具有输入、输出和加工处理无歧义,并且可以正确反映问题的需求,以及正确得到问题的答案。

关于“正确”的理解一共有4个层次:

- ①算法程序没有语法错误;
- ②算法程序能够根据正确的输入的值得到满足要求的输出结果;
- ③算法程序能够根据错误的输入的值得到满足规格说明的输出结果;
- ④算法程序对于精心设计的,极其刁难的测试数据都能满足要求的输出结果。

第④点比较难,因为我们几乎不可能对所有情况都逐一进行验证。一般来说,能够满足前3条就是一个正确性比较好的算法了。

(2) 可读性

算法设计的另一个目的是为了便于阅读、理解和沟通,如果写的代码只有你和上帝能看懂,那这个算法只能说明很失败,因为算法越难理解,就越难找到错误,对于调试和修改就更难了。

(3) 健壮性

当输入的数据不合法的时候,算法也能给出相关的处理,而不是产生异常或者错误。

(4)时间效率高和空间存储量低

在满足以上几点以后,我们还可以考虑对算法程序进一步优化,尽量满足时间效率高和空间存储量低的需求。

3.算法的性能分析

算法的效率是指算法的时间效率,一个算法所需要的时间取决于多个方面。

①算法采用的策略。

②问题的规模。

③采用的语言。

④机器的运算速度。

⑤编译程序产生的机器代码的质量。

随着计算机硬件和软件的提升,一个算法的执行时间是算不太精确的。只能依据统计方法对算法进行估算。我们抛开硬件和软件的因素,算法的好坏直接影响程序的运行时间。算法复杂度分为时间复杂度和空间复杂度,一个好的算法应该具有执行时间短,所需空间少的特点。

(1)算法的时间复杂度

时间复杂度常用大写 O 符号表示。一般情况下,算法的基本操作重复执行的次数是模块 n 的某一函数 $f(n)$,算法的时间复杂度为 $T(n) = O(f(n))$ 。 $f(n)$ 越小,算法的时间复杂度越低,算法的效率越高。

举例如下:

```
int value=0;           // 执行了 1 次
for(int i=0; i<n; i++) { // 执行了 n 次
    value+=i;
}
```

这个算法执行了 $1+n$ 次,如果 n 无限大,可以把 1 忽略,也就是说这个算法执行了 n 次,因此该算法的时间复杂度就是 $O(n)$ 。

再举例如下:

```
for(int i=0; i<n; i++) {
    for(int j=i; j<n; j++) {
        //执行语句
    }
}
```

当 $i=0$ 时内层的 for 循环执行了 n 次,当 i 等于 1 时内层的 for 循环执行了 $n-1$ 次,当 i 等于 2 时内层的 for 循环执行了 $n-2$ 次……所以算法的执行次数是

$$n+(n-1)+(n-2)+(n-3)+\cdots+1$$

根据时间复杂度算法的注意事项如下。

①去掉运行时间中的所有加法常数。

②只保留最高阶项。

③去掉与这个最高阶相乘的常数。

因此,该算法的时间复杂度为 $O(n^2)$ 。

(2)算法的空间复杂度

算法的空间复杂度是对一个算法在运行过程中临时占用存储空间大小的量度,记做 $S(n) = O(f(n))$ 。一个算法在计算机存储器上所占用的存储空间,包括存储算法本身所占用的存储空间,算法的输入输出数据所占用的存储空间和算法在运行过程中临时占用的存储空间这三个方面。

算法的执行时间和存储空间的花费是相互矛盾的,通常算法执行效率高,则存储空间会增加,反之也是一样的。不过通常是以算法的时间复杂度来判断算法的优劣。

实战训练版块

任务1 求简单算法的时间复杂度



任务描述

根据算法时间复杂度的计算步骤,给出下面三个算法的时间复杂度。



任务实现

(1) $++x;$

$s+=x;$

时间复杂度为 $O(1)$

(2) $\text{for}(i=0; i<n; ++i) \{$

$++x;$

$s+=x;$

$\}$

时间复杂度为 $O(n)$

(3) $\text{for}(j=0; j<n; ++j)$

$\text{for}(k=0; k<n; k++) \{$

$++x;$

$s+=x;$

$\}$

$\}$

时间复杂度为 $O(n^2)$



算法分析

三个算法中的基本操作语句为 $\{ ++x; s+=x; \}$, 其中(1)算法中语句的执行次数为 1 次, 因此时间复杂度为 $O(1)$; (2)算法中语句的执行次数为 n 次, 因此时间复杂度为 $O(n)$; (3)算法中语句的执行次数为 n^2 , 因此时间复杂度为 $O(n^2)$ 。

任务2 求矩阵相乘算法的时间复杂度



任务描述

下面给出了两个 $n \times n$ 阶矩阵相乘算法中的各条语句,计算出该算法的时间复杂度。



任务实现

```
for(i=0;i<n;++i) {                               //执行 n+1 次
    for(j=0;j<n;++j) {                             //执行 n^2+n 次
        C[i][j]=0;                                 //执行 n^2 次
        for(k=0;k<n;k++)                           //执行 n^3+n^2 次
            C[i][j]=C[i][j]+a[i][k]*b[k][j];      //执行 n^3 次
    }
}
```

时间复杂度为 $O(n^3)$



算法分析

算法中的基本操作语句为 $\{ C[i][j]=C[i][j]+a[i][k]*b[k][j]; \}$, 可用 n 的最高次幂来表示算法的时间复杂度, 因此两个矩阵相乘的算法时间复杂度为 $O(n^3)$ 。

任务3 求简单选择排序算法的时间复杂度



任务描述

下面给出了选择升序排序算法中的各条语句,计算出该算法的时间复杂度。



任务实现

```
public void selectSort(int[] a) {
    int n=a.length;
    for(int k=0;k<n-1;k++) {
        int min=k;
        for(int i=k+1;i<n;i++)
            if(a[i]<a[min])
                min=i;
        if(k!=min) {
            int temp=a[k];
            a[k]=a[min];
            a[min]=temp;
        }
    }
}
```

```
}  
}
```

时间复杂度为 $O(n^2)$



算法分析

这个算法执行的比较次数为 $(n-1)+(n-2)+\cdots+1=n(n-1)/2, T(n)=O(n^2)$ 。

项目二 线 性 表

线性表是最基本、最简单、也是最常用的一种数据结构。在实际应用中,线性表都是以栈、队列、字符串、数组等形式来实现的。掌握线性表的特性,对于数据运算的可靠性和提高操作效率都是至关重要的。

本次项目旨在了解线性表的概念及其结构特征,学习线性表的两种存储结构以及相关的运算,并比较两种存储结构性能优劣,最后通过两个实例掌握线性表的应用过程。

基础理论版块

任务 1 认识线性表



任务描述

理解线性表的概念及其结构特征,初步了解线性表的基本运算。



任务实现

1. 线性表的概念

如果一个数据元素序列满足:

①除第一个和最后一个数据元素外,每个数据元素只有一个前驱数据元素和一个后继数据元素;

②第一个数据元素没有前驱数据元素;

③最后一个数据元素没有后继数据元素。

则称这样的数据结构为线性结构。

简言之,线性表是一个线性结构,它是一个由 $n(n \geq 0)$ 个数据元素 $a_1, a_2, \dots, a_n$ 组成的有限序列,其中数据元素的个数 n 定义为表的长度,当 $n=0$ 时称为空表。应用中常将非空的线性表($n>0$)记作

$$(a_1, a_2, \dots, a_n)$$

其中数据元素 $a_i(1 \leq i \leq n)$ 只是一个抽象的符号,其具体含义在不同的情况下可以不同。

生活中线性表的例子有很多,例如:26 个英文字母组成的字母表、各种经济数据表格、一副扑克牌的点数也是线性表结构。从以上的例子可以看出,非空的线性表有且只有一个开始的结点 a_1 ,有且只有一个终端结点 a_n ,其余的内部结点 $a_i(2 \leq i \leq n-1)$ 都有且只有一个直接前驱 a_{i-1} 和一个直接后继 a_{i+1} 。

(1) 线性表的特征

①必存在唯一的一个“第一元素”;