普通高校系列教材·信息技术

数据结构学习参考

普通高校系列教材（信息技术）编委会组编
李从利 卢 伟 编

南京大学出版社

内容简介

本书是全国高校教材《数据结构》的配套辅导材料。全书共分成五大部分：第一部分学习方法指导，第二部分习题解答，第三部分典型题解，第四部分模拟测试，第五部分实验指导。其中第一部分给出了作者经多年教学经验总结出来的学习方法，可供广大考生参考；第二部分的习题解答不但给出了教材每章所附习题的解答，而且还结合相应知识点进行了分析与思考；第三部分的典型题解严格按照考试大纲的要求精心选择了100余题，既具有一定的代表性，又避免陷入题海过多增加考生的负担；结合大纲的给定难度要求，本书在第四部分提供了两套模拟试卷，考生可自行考核以检验学习效果；最后为增强考生的实际编程能力，编写了几份实验指导书。

本书内容丰富、结构清晰、针对性强、覆盖面广、叙述严谨，不仅对考试内容进行了高度概括，还通过分析、解答和进一步思考的形式提供了大量的典型题解，从而能够大大提高考生的应试能力。

本书是专门为相应配套教材而编写的，适合于计算机学生及专业人员使用，同时也可作为同层次的其他专业学生的学习参考书。

图书在版编目(CIP)数据

《数据结构》学习参考/李从利编. —南京：南京大学出版社，2003.5

ISBN 7-305-04075-4

Ⅰ.数… Ⅱ.李… Ⅲ.数据结构-高等学校-教学参考资料 Ⅳ.TP311.12

中国版本图书馆CIP数据核字(2003)第037595号

书　　名	《数据结构》学习参考
作　　者	李从利　卢　伟
出版发行	南京大学出版社
社　　址	南京市汉口路22号　邮编 210093
电　　话	025-3596923　025-3592317　传真 025-3303347
网　　址	www.njupress.com
电子函件	nupress1@public1.ptt.js.cn
经　　销	全国新华书店
印　　刷	合肥学苑印刷厂印刷
开　　本	787×1092 1/16　印张：11.875　字数：285千
版　　次	2003年6月第1版　2003年6月第1次印刷
书　　号	ISBN 7-305-04075-4/TP·263
定　　价	17.00元

普通高校系列教材(信息技术)编委会

出 版 前 言

近年来，我国的高等教育事业有了长足的发展，高校招生人数年年递增，越来越多的年轻人有机会接受正规的高等教育。这一举措无疑对我国的社会进步和经济发展有着重要的意义。但是人们也深刻地认识到，高等教育质量的好坏是一个不容忽视的关键性问题，而保证教育质量的一个重要环节就是抓好教材建设。但是，教材内容陈旧、教学手段落后的现象一直存在着。尤其像计算机技术这样的新兴领域发展迅猛，知识更新日新月异，教学内容落后于新技术新知识的矛盾显得尤为突出。基于上述两方面考虑，在南京大学出版社的鼎立相助下，一个以组编高校信息、电子类专业教材为主要任务的教材编委会成立了。

针对我国高等教育的现状和信息、电子技术的发展趋势，编委会组织部分高校的专家教授进行了深入的专题研讨。大家一致认为，在当前情况下组编一套紧跟新技术发展、符合高校教学需要、满足大学生求知欲望的系列教材势在必行，这将有助于规范教学体系、更新教学内容、把握教学质量，培养合格人才。专家们还对教材的结构、内容、体例及配套服务等提出了具体要求。为了能使这套教材逐步完善，并促进全国各地高校教学质量的提高，编委会决定在教材之外认真做好三件事：第一，为每本教材配备一本供学生使用的学习参考书，其主体内容为学习方法指导、习题分析与解答、典型题解或课程设计、模拟测试卷及解答、实验指导书；第二，定期对教材内容进行修订，及时补充新技术新知识，并根据具体情况组编新的教材；第三，有计划地组织各地高校教师进行教学交流与研讨，通过这种途径来提高偏远地区的师资水平。我们相信，通过各方面的大力支持和大家的不懈努力，这套教材会逐步被广大师生所接受，并在使用过程中得到完善、充实。

大家都知道，组编这样一套系列教材是个牵涉面很广的大工程。这个工程不仅在起步阶段需要得到各级教育主管部门、各高等院校、出版社的大力支持和协助，而且在使用过程中也离不开各位专家、教授、学生的热心呵护和指导。因此，殷切期待所有的能人志士关心我们，帮助我们，向我们提出好的建议或意见，为我们指出教材中的不足之处。

最后，感谢所有为本套系列教材出版付出辛勤劳动的同志们。

普通高校教材(信息技术)编委会

2001 年 8 月

编者的话

《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，在计算机学科中起着承前启后的作用，在计算机技术的各个领域中也有着广泛的应用，因而要从事计算机科学与技术工作，就要求相应人员必须具有良好的数据结构基础。

数据结构涉及数据的组织、存储以及运算的一般方法，其原理及算法较为抽象，对刚刚接触计算机学科的学生来讲，了解与掌握其中的原理显得尤为困难；有些章节学习起来难度很大，在解答相应习题时往往会感到无从下手，似乎教学内容与习题毫不相干。事实上解答习题所需的各种方法和技术都在教科书中，只不过其呈现形式多样化，需仔细体会才能掌握。作者在编写该书时，对多年来的教学经验进行了系统的总结，将自己对课程的理解和体会与初学者的学习角度相结合，对难以理解的原理及算法进行了通俗化的处理，以求高度概括、简单易懂。我们希望通过一些典型习题的解析和具体的实现环节给学生一些启发，帮助学生更好地学习和掌握课程内容，理解和掌握算法设计所需的方法和技术，为整个专业的学习打下良好的基础。

本书以考核知识点与考核要求为主线，再结合作者多年来的教学经验，从整体上对教材的结构和内容进行了规划、编排。全书共分成五大部分，它们各有特色、相互配合、相互补充。本书为相应教材的配套书，故在实际使用时，应注意以下几点：第一，在同步的学习过程中以教材为主，课后习题应首先自己独立完成，然后再与本书相应解答进行比较分析；第二，由于算法设计的不惟一性，对算法设计题本书给出的一种或几种解答，要在理解和领会的基础上自己上机动手编写程序，这样才会取得良好的效果，切忌照抄照搬；第三，要有 C 语言的基础，本书的算法是用 C 语言描述的，所以考生应较为熟练地掌握 C 语言的基本内容。

由于时间仓促和作者水平有限，本书难免存在问题和缺陷，恳请广大读者批评指正。

编　者

2002 年 8 月

目　录

第一部分　学习方法指导……(1)
一、课程的性质与要求 ……(1)
二、学习方法指导 ……(1)
三、重点、难点介绍……(4)
第二部分　习题解答 ……(29)
第 1 章 ……(29)
第 2 章 ……(31)
第 3 章 ……(39)
第 4 章 ……(47)
第 5 章 ……(54)
第 6 章 ……(63)
第 7 章 ……(76)
第 8 章 ……(83)
第 9 章 ……(93)
第 10 章 ……(101)
第三部分　典型题解……(107)
第四部分　模拟测试……(161)
模拟测试卷(一)……(161)
模拟测试卷(一)参考答案……(164)
模拟测试卷(二)……(168)
模拟测试卷(二)参考答案……(171)
第五部分　实验指导……(176)
实验一　栈和队列……(176)
实验二　树……(177)
实验三　图……(179)
实验四　查找……(180)
实验五　排序……(181)

第一部分　学习方法指导

一、课程的性质与要求

1. 课程性质

“数据结构”课程是计算机及其应用专业(本、专科)一门重要的专业基础课程。用数字计算机解决任何实际问题都离不开数据表示和数据处理,而数据表示和处理的核心问题之一是数据结构及其实现——这正是数据结构课程的基本内容。从这个意义上来说,数据结构课程在基础知识学习和动手技能培养两个方面都处于关键性地位。本课程不仅为数据库及其应用、操作系统概论等后继软件课程提供了必要的知识基础,也为计算机及其应用的专业人员提供了必要的技能训练。

2. 课程要求

根据数据结构教学大纲的要求,本课程的任务是介绍一些最常用的数据结构,阐述各种数据结构中数据元素之间内在的逻辑关系及其在计算机中的存储表示,并讨论在这些数据结构上所施加的各种运算(操作)的实现及应用。另外,对算法的时间和空间性能进行了必要的分析和比较。

通过该课程的学习,应达到知识和技能两方面的目标:

① 知识方面:从数据结构及其实现这两个层次及其相互关系的角度,系统地学习和掌握常用的基本数据结构(包括线性表、栈和队列、数组及广义表、二叉树、图、查找表和文件等等)及其不同的实现(包括不同的存储结构和算法),了解并掌握分析、比较和选择不同数据结构及其不同存储结构、不同运算实现(即算法)的原则和方法,为后继课程的学习打下坚实的基础。

② 技能方面:系统地学习和掌握在不同存储结构上实现的不同算法及其设计思想,从中体会并掌握存储结构的选择和算法设计的思维模式及技巧,使分析问题和解决问题(包括实际动手编程等)的能力得到提高。

在学习本课程前,要求学生必须基本掌握 C 语言程序设计的基本概念和基本技术,有条件的可以再掌握一些离散数学的知识效果会更好。

二、学习方法指导

1. 本课程学习主线

根据教材的内容安排,对每一种数据结构的讨论都基于以下几个方面来展开:

① 逻辑结构上的特点。

② 定义在逻辑结构上的一些基本运算。

③ 数据在不同存储结构上的表示。

④ 基本算法在不同存储结构上的具体实现。

⑤ 算法时间性能、空间性能的分析和比较。

各种数据结构在上述五个方面紧密相关，彼此之间存在着或多或少的联系，因此，在学习过程中，大家一定要注意掌握这条学习的主线，比较各种数据结构在不同存储结构的异同。下面我们结合教材的章节安排，整理出各章的知识点以及相应的重点和难点(见表 1-1)。

表 1-1　知识点、重点和难点汇总

章节	章节名	知　识　点	重　　点	难　　点
第 1 章	绪论	1. 数据、数据元素和数据项的概念 2. 数据的逻辑结构 3. 运算的概念 4. 数据结构的概念 5. 存储结构和运算实现 6. 算法描述及分析	逻辑结构和数据结构的概念	算法的时间复杂度分析
第 2 章	线性表	1. 线性结构的定义及基本特征 2. 线性表的概念 3. 顺序表和单链表 4. 插入、删除和定位运算在顺序表和单链表上的实现 5. 循环链表和双链表 6. 顺序表和链表的比较	线性结构的定义及特点；线性表的运算；顺序表和单链表的组织方法和算法设计	单链表、双链表上的算法设计
第 3 章	栈和队列	1. 栈和队列的定义 2. 栈的顺序实现 3. 栈的链接实现 4. 队列的顺序实现 5. 队列的链接实现	栈和队列的特点；顺序栈和链栈、链队列上基本运算的实现和简单算法	出入栈的顺序；循环队列的组织；队满、队空的条件及算法设计
第 4 章	串	1. 串的基本概念 2. 串的存储结构 3. 串的基本运算	串的基本概念；串的顺序存储结构	串的基本运算(模式匹配算法)
第 5 章	数组与广义表	1. 数组及广义表的概念 2. 数组的顺序存储结构 3. 矩阵的压缩存储 4. 广义表的链式存储结构 5. 广义表的基本运算	数组的逻辑结构；矩阵的顺序存储；广义表的定义	矩阵的压缩存储；广义表的基本运算
第 6 章	树	1. 树的基本概念及术语 2. 二叉树的定义及性质 3. 二叉树的顺序存储结构 4. 二叉树的链式存储结构 5. 二叉树的遍历 6. 树和森林的存储及与二叉树的相互转换 7. 树的应用	树形结构的概念；二叉树的定义、存储结构和遍历算法；树、森林与二叉树的相互转换	二叉树的遍历算法；树的应用

（续表）

章节	章节名	知 识 点	重 点	难 点
第7章	图	1. 图的概念 2. 图的存储结构 3. 图的遍历 4. 生成树和最小生成树 5. 拓扑排序和最短路径	图的存储结构和连通图的遍历	最小生成树；拓扑排序；最短路径
第8章	查找表	1. 集合的基本概念 2. 查找表的基本概念 3. 静态查找表的实现 4. 二叉排序树 5. 散列表的基本概念 6. 散列函数的构造方法 7. 动态查找表在开散列表上的实现 8. 动态查找表在闭散列表上的实现	二分查找；二叉排序树的查找；散列表的查找；查找性能的分析	二叉排序树的插入算法及散列表中的解决冲突的方法
第9章	排序	1. 排序的基本概念 2. 插入排序 3. 交换排序 4. 选择排序 5. 归并排序	各种排序过程的理解和掌握；快速排序；堆排序	堆的建立及调整；各种排序方法适用场合的选择
第10章	文件	1. 文件的基本概念 2. 顺序文件 3. 索引文件 4. 散列文件	文件的基本概念；顺序文件、索引文件和散列文件的组织方式和操作方式	顺序文件、索引文件和散列文件的组织方式和操作方式

2. 学习方法指导

“数据结构”课程涉及到数据的组织、存储以及运算的具体方法，其原理及算法较为抽象，对刚刚接触计算机学科的学生来说，了解与掌握其中的原理难度较大；同时教材中有些章节的内容，晦涩难懂，不易掌握。为此，作者结合多年的教学实践和对该课程的认识和体会，提出几点学习方法，希望能对广大读者有所帮助。

① 把握知识体系。数据结构课程中的知识体系具有良好的结构性。从总体上说，课程的主要内容是围绕着线性表、串、栈、队列、数组、二叉树、图、查找表和文件这九种常用的数据结构和排序运算来开展的。对于每种数据结构，都是从前面所介绍的五个方面加以介绍的。对于排序运算，在讨论了它的各种典型的、常见的算法的同时，还对它们的时间复杂性和空间复杂性进行了详细的比较。因此，只要按上述体系对课本内容进行分类，就能全面把握整个知识体系。

② 要善于比较、小结。由于课程中的知识体系具有结构性，故在学习过程中应注意从“纵向”和“横向”两个方面对比有关内容以便加深理解和记忆。纵向对比包括将一种数据结构与它的各种不同的实现加以比较；横向对比则指将具有相同逻辑结构的不同数据结构（如线性表、栈、队列和串）进行的比较，同一数据结构的不同存储结构和实现同一运算的不同算法（如各种查找算法）的比较等。比较的结果最好能汇总成图、表的形式。另外，在学完每一章、节后，要及时进行小结，最好用自己的语言进行描述，这样便于理解和记忆。

③ 要循序渐进，切忌囫囵吞枣。由于课程内容前后之间具有密切的联系，因此在学习新的内容之前，首先要掌握其前续知识点，并且要领会基本概念、基本思想。在阅读算法之前，一定要先弄清其基本思想、基本步骤。因为即使读懂了算法但不知其基本思想，根本不能算掌握算法，更谈不上应用算法。在学习过程中，千万要掌握好学习的节奏，要一步一个脚印，切忌囫囵吞枣，只讲速度，不求质量。

④ 多练习、多上机。教材的每一章后面都有精心选择的习题以供练习，要求学生在认真掌握课本内容的同时独立完成课后习题，一方面可以检查学习的效果，另一方面也可以提高技能水平。此外，在设计算法时，尽可能根据实际问题的需要，选择合适的数据结构，设计出完整的算法，有条件的同学最好能在机器上运行通过。

⑤ 要善于掌握算法的思想、实现方法。算法的基本思想、实现方法的描述在教材中占有相当大的比例。在学习每一种算法之前，我们建议：首先要理解算法的要求，在此基础上模拟问题的求解，想一想自己是否能提供算法描述，如果自己给出了算法描述，则与课本上的算法进行对比。如果差别很大则仔细阅读书上算法，看一看差别在哪里。在理解算法的基础上，要提出一些变换问题。例如，算法的主要实现功能能否由其他的方法代替？算法的各部分之间的关系改动后会发生什么变化？如果问题的要求发生改变，则应如何修改算法？将所提的问题记录下来，以便与老师或他人进行讨论，同时在复习时也会有很大帮助。

⑥ 要结合教学大纲进行复习。在进行每一章的学习之前和结束之后，应仔细阅读大纲的有关规定和要求，对一些难度较大的知识点，一定要结合大纲规定的掌握层次进行学习。

以上提出的几点学习方法，仅供大家学习参考，同学们应该根据自己的学习计划和要求制定合适的学习方法，以便节省时间、提高效率。

三、重点、难点介绍

下面给出的重点、难点介绍中，对各知识点的介绍不像教材那样详细，而是对它们进行提炼和分析，具体细节可参考教材。

1. 链表

链表是“数据结构”课程中第一次接触到的动态结构，因此许多初学者感到难以掌握，编写算法时困难重重，容易出错。一般来说，引起出错的原因有以下几种：

① 指针与动态链表的有关概念不清。

② 链表的含义不清。

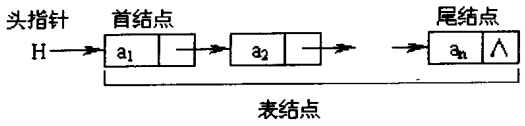
③ 动态变量的含义、标识及操作方法不清。

④ 混淆了链表的几种常见形式。

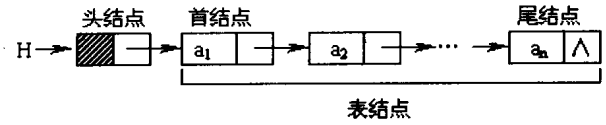
⑤ 算法的构思及设计方法出错。

⑥ 算法的编码不正确。

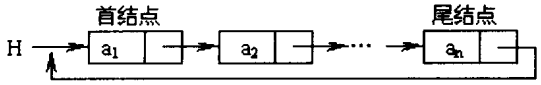
针对以上容易出错的地方，此处给出链表结构常见的几种组织形式，并结合链表是否为空的判断方法加以介绍。图 1－1 所示为各种线性链表结构示意图。



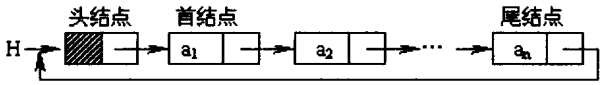
(a)(基本)单链表



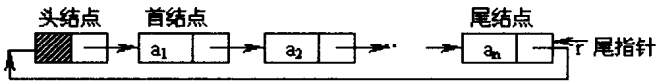
(b)带头结点的单链表



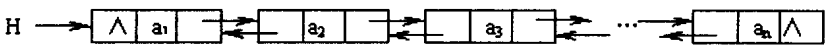
(c)循环单链表



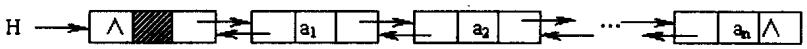
(d)带头结点的循环单链表



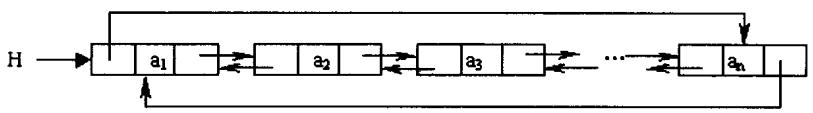
(e)带尾指针的循环单链表



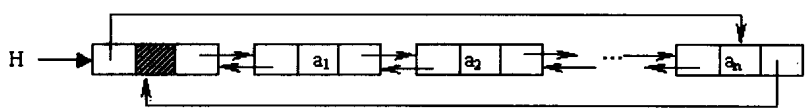
(f)(基本)双链表



(g)带头结点的双链表



(h)双循环链表



(i)带头结点的双循环链表

图 1-1　各种线性链表结构示意图

下面分别对上述链表进行简单介绍。

(1)(基本)单链表

这是最简单、最直接的链表形式,如图 1 - 1(a)所示。图中链表的每个元素与其直接后继的地址合在一起构成链表的一个结点,链表中的每个结点对应线性表中的一个元素。通常,称链表中值为 a_1 的结点为首结点,称链表中值为 a_n 的结点为尾结点,称指向链表中第一个结点的指针 H 为头指针(以下几种链表形式中结点说明相同)。因为由头指针的值可获得所有结点的指针(地址),因此常以头指针作为链表的名称,故图 1 - 1(a)所示的链表可称为链表 H。

很显然,判断单链表为空的条件为 H = = NULL。

(2)带头结点的单链表

在一些基本单链表的运算中(如插入、删除),常存在着这样的问题:对首结点的和其余表结点的某些操作因结构方面的原因而存在着差异,这样则给算法的设计和修改带来不便。解决这一问题的常用方法就是在链表的首结点之前设置一个附加结点(称为头结点),并让头指针指向这一结点,如图 1 - 1(b)所示。这样,首结点变成了链表中的第二个结点。这样的链表被称为带头结点的单链表。从图中可以看出,头结点中的数据域没有存放元素。

由于头指针始终指向头结点,所以判断这种链表为空的条件为 H - > next = = NULL。

(3)单循环链表

如果将单链表中尾结点的后继指针改为指向链表中第一个结点(表头),可以使某些运算的实现变得更为简单,这样就构成了单循环链表,如图1 - 1(c)、(d)所示。其中,图1 - 1(c)所示的为无头结点的单循环链表,图 1 - 1(d)所示的为带头结点的单循环链表。循环链表与非循环链表在表尾结点的判断及与此相关的运算实现上存在差异。循环链表尾结点的后继指针指向表头,而非循环链表的尾结点的后继指针为空(NULL)。

判断无头结点的循环单链表为空的条件为 H = = NULL。判断带头结点的循环单链表为空的条件为 H - > next = = H。

在有些情况下,为了能快速获得表头结点和表尾结点指针的值,而将指示表头结点的指针改为指向表尾,此时指针称为尾指针,这样即可得到带尾指针的单循环链表,如图1 - 1(e)所示。判断此种链表为空的条件为 r - > next = = r。

(4)双链表

在上述各种单链表中,要实现求解某结点的前驱结点的运算是不方便的。为此,可在链表每一结点中再设置一个指向其前驱结点的指针,这样,每个结点中有两个指针,这种形式的链表称为双链表。双链表也可设置头结点和循环形式,因此共有四种形式的双链表,分别如图 1 - 1(f)、(g)、(h)、(i)所示。

这四种形式链表为空的判断条件与前述几种形式的链表相同,这里不再赘述。

(5)其他形式的链表

教材中还介绍了二叉链表和三叉链表、十字链表等其他形式的几种链表,这些内容将在第三部分的典型习题分析讲解中进行介绍。

在与链表有关的算法设计中,遍历链表是最基本的也是最为典型的算法。所谓的遍历链表是指按某种次序"访问"链表所有结点一次,且仅一次。在掌握了遍历链表的算法的基本思想之后,即可对其作适当变化以用于许多问题的求解。遍历算法较为简单,请读者参考教材。

下面结合该知识点,给出几道习题的分析及解答。

【例1】顺序表中逻辑上相邻的元素,其物理位置_____相邻,故为_____存取结构;单链表中逻辑上相邻的元素,其物理位置_____相邻,故又为_____存取结构。

【分析】顺序表的特点是以数据元素在存储空间中的物理相邻来表示其逻辑相邻的关系。因此,只要确定了线性表的起始地址和一个数据元素占用的存储单元的大小,就可以随机存取线性表中任一数据元素。因此,顺序表是一种随机存取的结构。而单链表不同,它是通过指针来表示元素之间的逻辑关系,则逻辑上相邻的两个数据元素其物理位置不一定相邻,这就造成了元素之间的逻辑关系和物理存储次序的不一致。要存取链表中的结点,都必须从头指针开始顺序进行,故链式存储结构是一种顺序存取结构,也称作非随机存取结构。

【解答】一定,随机,不一定,顺序

【例2】在一个具有n个结点的有序单链表中插入一个新结点使得链表仍然有序,其算法的时间复杂度为_____。

A) $O(\log_2 n)$　　B) $O(1)$　　C) $O(n)$　　D) $O(n^2)$

【分析】按照最坏情况下时间复杂度的估算,在一个有序单链表中插入一个新结点,尽管插入时不需要移动元素,但寻找其合适的插入位置需消耗时间,故算法的时间复杂度为$O(n)$。

【解答】C)

【讨论】关于链表建立、插入、删除等算法的时间复杂度的估算,尽管它们都不需要像顺序表一样移动大量的元素,但由于链式存储结构是一种顺序存储机构,故其定位操作需消耗时间,这一点一定要注意。

【例3】单链表中,增加头结点的目的是为了_____。

A) 使单链表至少有一个结点

B) 标示表结点中首结点的位置

C) 方便运算的实现

D) 用于标示单链表

【分析】对单链表来讲,增加头结点是为了使空表和非空表统一,算法处理一致,方便单链表上各种运算的实现。

【解答】C)

【例4】试设计一个算法,该算法用于删除带头结点的单链表中值相同的多余结点。

【分析】由于题目中没有给出该单链表中结点是否有序,故必须按照结点无序的情况进

行分析。首先,用一个指针 p 指向单链表的第一个表结点,然后用另一个指针 q 查找链表中的结点元素,同时让指针 s 指向 q 所指结点的前驱结点。在查找过程中,若找到满足条件 q->data = p->data 的结点时,删除 q 指针所指结点,同时向后移动 q 指针,直到 q 为空时为止(由于是单链表,故结束条件为 p==NULL),表明与 p 指针所指结点值相同的元素均删除;然后再修改 p,使得 p 指针后移,重复上述操作,直到 p 为空时算法结束。

【解答】设链表头指针为 head,算法描述为:

```
void Dele_Same(LinkList *head)
/*删除带头结点的单链表中值相同的多余结点*/
{ LinkList *p,*q,*s;
p=head->next;
q=p;
while(p!=NULL)
{ s=q;
  q=q->next;
  do
  { while((q!=NULL)&&(q->data!=p->data))
    /*查找值相同的结点*/
    { s=q;
      q-q->next;
    }
    if(q!=NULL)/*删除结点,后移 q 指针*/
    { s->next=q->next;
      free(q);
      q=s->next;
    }
  }while(q!=NULL)/*一趟删除结束*/
  p=p->next;/*进行下一趟删除*/
  q=p;
}
}/*Dele_Same*/
```

【讨论】将该算法与在顺序表上删除值相同元素的算法进行比较,试述两者有何区别。若该单链表为不带头结点的单链表,则如何修改源程序?

【例 5】假设有一个单循环链表,其结点结构中含有 3 个域:data、next 和 prior。其中 data 为数据域;next 为指针域,指向后继结点;prior 为指针域,值为空(NULL)。试设计算法将该单链表修改为双向循环链表。

【分析】由于原单链表为循环链表且结点结构满足双链表的要求,故只要修改单链表中的 prior 指针,使其指向前驱结点即可。

【解答】算法描述为:

```
void Change(LinkList *head)
```

```
/* 将单循环链表修改为双循环链表 */
{ LinkList *p, *q;
  p = head;
  do
  { q = p->next;
    q->prior = p; /* 修改指针 prior,使其指向前驱结点 */
    p = q;
  }while(p! = head);
}/* Change */
```

【讨论】在该题中,有无头结点是否有影响?若题目中不是单循环链表,而是普通的单链表,则如何实现?

通过以上几个习题可以发现,对于链表的考核,主要集中在链表的概念、与顺序表的区别以及单链表的算法上,故在熟练掌握链表知识的基础上,还必须明白链表中有无头结点时的区别。

2. 栈和队列

栈和队列是计算机科学中两个非常重要的、经常使用的数据结构。它们是两种特殊的线性表,因为它们的逻辑结构与线性表相同,只是在运算规则上比线性表多增加了一些限制,故可把它们看成是运算受限的线性表。在考察时,也经常把它们放在一起进行考核。

(1)栈和队列的定义

栈是限制在表的一端进行插入和删除的线性表。根据这一定义可知,每次删除(出栈)的总是当前栈顶的元素,即最后插入(入栈)的元素。因此,栈又被称为后进先出的线性表。

队列也是一种运算受限的线性表,但与栈不同的是,它只能在表的一端(队尾)进行插入(入队列)运算,而在表的另一端(队头)进行删除(出队列)运算。根据这一定义可知,每次删除的元素总是当前的队头元素,每次插入的元素总是当前的队尾元素。也就是说,先入队列的元素总是先离开队列,因此队列又被称为先进先出的线性表。

(2)出、入栈和队列的元素序列

这是一个经常考察到的知识点。由于栈是先进后出的线性表,且在入栈的过程中可以出栈,这样一来入栈和出栈的元素序列就有多种,但可以证明这样一个定理:若借助栈由输入序列 1,2,…,n,得到输出序列 $P_1, P_2, \cdots, P_n$(它是输入序列的一个排列),则输出序列中不可能出现这样的情况:存在 $i<j<k$ 使得 $P_j<P_k<P_i$。这一定理的证明,我们稍后进行。

队列是一种先进先出的线性表,即最先入队列的元素一定先出队列,这样一来对任意多个元素来说,它们入队的顺序和出队的顺序一致。

(3) 栈满、栈空、队满、队空的判断

栈和队列都有两种存储方式(顺序、链式),即有顺序栈、链栈、顺序队列、链队列。在出栈或出队列时,一定要考虑到栈或队列中是否有元素,在入栈或入队列时,一定要考虑到栈

或队列是否已满。相应条件判断，请参见教材。

(4)循环队列

为了解决顺序队列出现的“假溢出”现象，将队列看作是一个首尾相连的循环队列，即用循环数组实现。此时队首仍在队尾之前，进行插入和删除运算时仍遵循“先进先出”的原则。

若 front 为头指针，rear 为尾指针，则判断队空的条件为 front = = rear；判断队满的条件为 front = =(rear + 1)mod m。

(5)栈和队列的应用

栈是实现递归函数调用必不可少的数据结构，可以用栈来保存地址、参数及局部变量的值。队列在程序设计中也经常采用，根据它的先进先出的特点，经常被用在树和图的遍历算法中。

下面结合该知识点，给出几道例题并进行分析、解答。

【例 1】栈的逻辑特点是_____，队列的逻辑特点是_____；二者的共同点是允许在它们的_____处插入和删除数据元素。

【分析】由于只能在栈顶处执行插入、删除操作，每次操作总是栈顶上最新进栈的元素，使得先进栈的数据元素后出栈，所以栈的逻辑特点是先进后出(或后进先出)；而对队列来说，插入和删除操作必须在队列的两端进行，数据元素的入队顺序始终与出队顺序一致，所以队列的逻辑特点是先进先出(或后进后出)。栈和队列都是操作受限的线性表，两者的共同点是所有的操作只能在端点处进行。

【解答】先进后出(或后进先出)，先进先出(或后进后出)，端点

【例 2】试证明：若借助栈由输入序列 1,2,…,n，得到输出序列 $P_1,P_2,\cdots,P_n$，则输出序列中不可能出现这样的情况：存在 $i<j<k$ 使得 $P_j<P_k<P_i$。

【证明】可用反证法进行：

若存在 $i<j<k$ 使得 $P_j<P_k<P_i$，则一方面由 $i<j<k$ 知三者的出栈顺序为：P_i,P_j,P_k(下标为其出栈序号)；另一方面，由 $P_j<P_k<P_i$，可知其入栈顺序为 P_j、P_k、P_i。由于 P_i 最后入栈，最先出栈，因此 P_i，出栈前，三个数都在栈中，如果在栈中的次序如图 1-2(a)所示，则和出栈顺序矛盾，反之如果在栈中的次序如图 1-2(b)所示，则和入栈顺序矛盾，由此得到证明。

图 1-2　元素可能在栈中的次序

【例 3】假定有四个元素 A,B,C,D 依次进栈，进栈过程中允许出栈，试写出所有可能的出栈序列。

【分析】由四个元素 ABCD 所有的可能组合为 24 种，只要挑选出所有不可能的组合(出

栈序列),剩下的即为所求。

【解答】所有不可能的组合有 10 种,分别为:

D,C,A,3　　C,D,A,B　　C,A,D,B　　C,A,B,D、
D,A,C,B　　D,A,B,C、　　A,D,B,C、　　D,B,A,C、
D,B,C,A　　B,D,A,C

按照第二题所述定理,每种序列中划线部分为不可能得出栈序列。故所有可能的组合为 24 - 10 = 14 种,分别为:ABCD、ABDC、ACBD、ACDB、BACD、ADCB、BADC、BCDA、BCAD、BDCA、CBAD、CBDA、CDBA、DCBA。

【例 4】假设以一个一维向量 data[0..maxsize - 1]存放一个循环队列中的元素,同时设变量 rear 指示循环队列中队尾元素的位置。试编写出此循环队列的队满条件,并设计出相应的出队列和入队列的算法。

【解答】假定队尾指针 rear 指示队尾元素实际存放的存储单元的位置,且设 q 为指向 SeqQueue 类型的队列指针,则相应的入队列和出队列的算法描述为:

① 入队列算法。

```
int EnQueue(SeqQueue *q,ElemType x)
/*若循环队列不满,将元素 x 入队列*/
{
  if((q->rear+1)%maxsize==q->front)
  return ERROR; /*若队列满则返回错误信息*/
  else
  { q->data[q->rear]=x; /*元素入队列*/
    q->rear=(q->rear+1)%maxsize;/*队尾指针加 1*/
    return OK;
  }
}/*EnQueue*/
```

② 出队列算法。

```
ElemType DeQueue(SeqQueue *q)
/*若循环队列非空,队头元素出队列且返回其值,否则返回空元素*/
{ ElemType temp;
  if(q->front==q->rear)
    return ERROR; /*若队列空则返回错误信息*/
  else
  { temp=q->data[q->front]; /*暂存队头元素的值*/
    q->front=(q->front+1)%maxsize;/*队头指针加 1*/
  }
  return temp;
}/*DeQueue*/
```

3. 二叉树

二叉树是数据结构中非常重要的一种非线性数据结构,在各种形式的考试中二叉树都