

2004年版新大纲

全国计算机等级考试

真题详解 (笔试+上机)

二级Visual FoxPro程序设计

史德芬 编著



- 开放式全真模拟考试环境
- 多套笔试、上机模拟练习与真题试卷
- 自动计时，自动阅卷，自动评分，现场评析



 科学出版社
北京科海电子出版社

全国计算机等级考试

真题详解（笔试+上机）

二级 Visual FoxPro 程序设计

史德芬 编著

科学出版社

北京科海电子出版社

内 容 提 要

本书是依据教育部考试中心制订的《全国计算机等级考试大纲(2004年版)》中对二级 Visual FoxPro 程序设计的要求,以高等教育出版社出版的《全国计算机等级考试二级教程——Visual FoxPro 程序设计》指定教材为蓝本编写而成。内容上做到了既不超纲,又不降低水平;讲解简明扼要,理论联系实际。为了适合自学考试,本书通过对笔试试题的分析与解答,对一些重点和难点问题加以分析、归纳与总结,使考生能够在最短的时间内掌握考试要点。同时,本书还对上机考试作了详细的介绍,包括上机考试操作说明、上机试题分析。最后给出了 2003 年 9 月、2004 年 4 月和 2004 年 9 月(笔试+上机)全真试卷及其分析与解答。

本书内容丰富、结构清晰、针对性强、叙述严谨,不仅适用于报考全国计算机等级考试二级 Visual FoxPro 程序设计的考生使用,同时也可以作为其他人员学习 Visual FoxPro 的参考用书。

与本书配套使用的光盘提供了开放式全真模拟考试环境,内容覆盖所有考点的笔试、上机模拟练习题,具有现场评析、自动阅卷和评分等功能,使考生提前熟悉考试环境,学练结合,全面提升应试能力。

图书在版编目(CIP)数据

全国计算机等级考试真题详解(笔试+上机) 二级 Visual FoxPro

程序设计/史德芬编著. —北京:科学出版社, 2004

ISBN 7-03-014493-7

I. 全… II. 史… III. 关系数据库—数据库管理系统,

Visual FoxPro—水平考试—自学参考资料

IV. TP311.138

中国版本图书馆 CIP 数据核字(2004)第 107930 号

责任编辑:陈轶 / 责任校对:科海

责任印刷:科海 / 封面设计:付剑飞

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>

北京科普瑞印刷有限责任公司印刷

科学出版社发行 各地新华书店经销

2005 年 1 月第一版

开本:16 开

2005 年 1 月第一次印刷

印张:20.875

印数:1-5 000

字数:508 千字

定价:32.00 元(1CD)

(如有印装质量问题,我社负责调换)

前 言

全国计算机等级考试是面向社会的计算机应用能力水平考试。自1994年开考以来持续发展,报考人数逐年递增。该项考试在促进计算机基础知识的普及和计算机应用技术的推广、考核工作人员计算机的实际应用能力等方面起了促进作用,在社会上享有良好的信誉。

2004年全国计算机等级考试作了重大调整,二级考试除了考试科目有所增减外,还统一添加了公共基础知识部分。本书作者针对新大纲中有关二级Visual FoxPro的考试要求,从笔试和上机考试两个方面进行分析与总结,希望能为广大考生应试提供有益的帮助。

全书包括4部分,各部分内容如下:

第1部分是“公共基础知识”,这是2004年版新大纲增加的内容,含有基本数据结构与算法、程序设计方法,软件工程和数据库设计基本知识。该部分在笔试考试中占30分(10道选择题,5道填空题),不涉及上机操作,故本部分重点放在知识点的归纳与总结上,并结合大量的选择题和填空题进行分析。

第2部分是“考试要点与笔试真题解析”,与新版计算机等级考试大纲紧密结合,将笔试中的难点和常考的知识点融入到典型考题的细致分析与解答中,便于考生准确把握笔试的重点和难点。

在章节顺序上,该部分与高等教育出版社出版的《全国计算机等级考试二级教程——Visual FoxPro程序设计》一致。在试题选择上,该部分试题全部来自2002年和2003年笔试真题试卷。

第3部分是“上机考试操作说明与真题解析”,在详细介绍上机考试环境和操作方法之后,对上机试题中的基本操作题、简单应用题和综合应用题进行分类解析,并结合上机考试真题进行详细的分析与解答,总结考试规律,点拨解题技巧,让考生对上机考试有一个全面的认识。在试题选择上,该部分全部为2003年和2004年上机考试真题。

第4部分是“全真试题(笔试+上机)分析与解答”,针对2003年9月、2004年4月和2004年9月3次最新的笔试试卷和上机试卷进行分析解答,使考生熟悉最新的考试内容和题型,掌握解题技巧,以便在最短时间内有所突破,取得好成绩。

本书作者长期从事计算机等级考试的教学和培训工作,具有丰富的考前辅导经验,对计算机等级考试颇有研究,对考试题型特点、知识点分布把握得比较准确。

本书由史德芬编著,史德俊、陈环、何炜、曾向红、江春芳、李平业、于学锋、李业玲、朱庆忠、李明琴等参与了资料收集、整理和编写等工作。

由于时间仓促,书中难免存在错误和缺陷,恳请广大读者批评指正。

联系方式: sdr@163.com

作者

2004年12月

光 盘 说 明

光盘内容

本光盘中的全国计算机等级考试超级模拟软件是由北京无忧电脑技术开发有限责任公司研制的,这套模拟软件提供了开放式的考试环境,具有自动计时、自动阅卷评分、完全模拟真实考场等功能,能让考生尽快熟悉等级考试的考试环境,掌握全国计算机等级考试的出题规律,顺利地通过考试。

本套模拟软件(二级Visual FoxPro考试)包括笔试和上机两部分。笔试部分包括选择题和填空题;上机部分包括基本操作题、简单应用题和综合应用题。

安装指南

1. 将光盘插入光驱中。
2. 从Windows桌面上双击“我的电脑”图标。
3. 双击表示光驱的图标项。
4. 双击名为autorun.exe的文件来运行这个安装程序。
5. 按照屏幕上的提示信息完成整个安装过程。

注意:如果启动了自动播放的特性,当将光盘放到光驱中时,autorun.exe会自动运行。

使用说明

安装成功后,系统将自动在桌面上创建“二级Visual FoxPro练习”和“二级Visual FoxPro考试”的快捷方式,并在“程序”菜单项中增加“全国计算机等级考试二级Visual FoxPro超级模拟软件”。

考生可先将模拟软件光盘放入光驱中,若进行练习,可双击桌面上的“二级Visual FoxPro练习”快捷方式(或单击“程序”菜单项中的“全国计算机等级考试二级Visual FoxPro超级模拟软件”子菜单中的选项),打开模拟软件。

软件操作界面包括“上机练习”、“笔试练习”、“帮助主题”、“客户服务”和“退出系统”5个选项,考生可根据自己的需要选择练习选项。

注意:在“上机练习”中,需要先安装Visual FoxPro 6.0,才能完成基本操作题、简单应用题和综合应用题,读者可从网上下载。

考生若进行模拟考试,可双击桌面上的“二级Visual FoxPro考试”快捷方式(或单击“程序”菜单项中的“全国计算机等级考试二级Visual FoxPro超级模拟软件”子菜单中的选项),弹出“考试登录”对话框。

考生在“准考证号”文本框中输入准考证号码,然后单击“考号验证”按钮,按照系统提示进行操作即可。进入系统后,可按照题目的提示进行操作。考试过程和方法与“上机练习”一样。

注意:考试系统中二级Visual FoxPro的准考证号为:270199990001~270199990005。

本套模拟软件的具体使用说明可参看光盘中的“二级VFP用户手册”文档和help.hlp文件。

目 录

第1部分 公共基础知识

第1章 数据结构与算法	2
1.1 考试要点	2
1.1.1 算法基本知识	2
1.1.2 数据结构的基本概念	3
1.1.3 线性表	4
1.1.4 栈和队列	5
1.1.5 线性链表及其基本运算	6
1.1.6 树的基本知识	8
1.1.7 查找和排序算法	11
1.2 试题解析	13
1.2.1 选择题	13
1.2.2 填空题	15
第2章 结构化程序设计基础	16
2.1 考试要点	16
2.1.1 程序设计方法与风格	16
2.1.2 结构化程序设计概念	16
2.1.3 面向对象的程序设计	17
2.2 试题解析	17
2.2.1 选择题	17
2.2.2 填空题	18
第3章 软件工程基础	19
3.1 考试要点	19
3.1.1 软件工程基本知识	19
3.1.2 结构化分析方法	21
3.1.3 结构化设计方法	22
3.1.4 软件测试	22
3.1.5 程序调试	25
3.2 试题解析	25
3.2.1 选择题	25
3.2.2 填空题	28

第4章 数据库设计基础	29
4.1 考试要点	29
4.1.1 数据库概念	29
4.1.2 数据模型	30
4.1.3 关系代数及数据库规范化	31
4.1.4 数据库设计方法及步骤	32
4.2 试题解析	32
4.2.1 选择题	32
4.2.1 填空题	35

第2部分 考试要点与笔试真题解析

第5章 Visual FoxPro基础知识	37
5.1 考试要点	37
5.1.1 数据库基础知识	37
5.1.2 关系数据库	39
5.1.3 Visual FoxPro的特点与工作方式	40
5.1.4 数据与数据运算	41
5.2 笔试真题解析	43
5.2.1 选择题	43
5.2.2 填空题	73
第6章 Visual FoxPro数据库的基本操作	81
6.1 考试要点	81
6.1.1 数据库及其建立	81
6.1.2 建立数据库表	82
6.1.3 自由表	83
6.1.4 表的基本操作	83
6.1.5 索引	84
6.1.6 多表操作	86
6.1.7 视图与查询	86
6.2 笔试真题解析	87
6.2.1 选择题	87
6.2.2 填空题	121
第7章 关系数据库标准语言SQL	125
7.1 考试要点	125
7.1.1 查询功能	125
7.1.2 操作功能	126

7.1.3 定义功能.....	127
7.2 笔试真题解析	128
7.2.1 选择题.....	128
7.2.2 填空题.....	161
第8章 项目管理器、设计器和向导的使用	168
8.1 考试要点	168
8.1.1 项目管理器.....	168
8.1.2 表单设计器.....	168
8.1.3 报表设计器.....	169
8.1.4 应用程序向导	170
8.2 笔试真题解析	171
8.2.1 选择题.....	171
8.2.2 填空题.....	196
第9章 Visual FoxPro程序设计	200
9.1 考试要点	200
9.1.1 命令文件的建立和运行	200
9.1.2 结构化程序设计	201
9.1.3 过程与过程调用	203
9.2 笔试真题解析	205
9.2.1 选择题.....	205
9.2.2 填空题.....	228

第3部分 上机考试操作说明与真题解析

第10章 上机考试操作方法说明	235
10.1 上机考试环境	235
10.2 上机考试的题型、分值及考试时间	235
10.3 上机考试操作方法.....	236
10.4 上机考试技巧	240
第11章 上机试题汇集.....	242
11.1 上机试题1	242
11.2 上机试题2	246
11.3 上机试题3	249
11.4 上机试题4	254
11.5 上机试题5	256
11.6 上机试题6	259

11.7 上机试题7	261
11.8 上机试题8	263
11.9 上机试题9	266
11.10 上机试题10	268

第4部分 全真试题（笔试+上机）分析与解答

2003年9月全国计算机等级考试二级笔试试卷	273
试卷解析	280
2004年4月全国计算机等级考试二级笔试试卷	284
试卷解析	290
2004年4月全国计算机等级考试二级上机试题	294
试题解析	297
2004年9月全国计算机等级考试二级笔试试题	305
试卷解析	313
2004年9月全国计算机等级考试二级上机试题	316
试题解析	319

第 1 部分



公共基础知识

教育部考试中心最新颁布了《全国计算机等级考试考试大纲》（2004年版），对部分开考课程作了调整，并从2005年4月份开始按新大纲开考。其中，在考核内容方面，二级仍然定位为程序员，考核内容主要包括公共基础知识和程序设计，具体如下：

- 所有科目对公共基础知识作统一要求，使用统一的公共基础知识大纲和教程。
- 公共基础知识主要涉及数据结构与算法、程序设计方法、软件工程、数据库基础知识等4个部分。
- 二级公共基础知识在各科笔试中所占的比重为30%（30分），题型为10道选择题和5道填空题。
- 二级上机考试中将取消对DOS部分的考核（占30分）。

为了便于考生正确把握新大纲对公共基础知识的考核要求，作者仔细研读新大纲，并结合多年的教学经验，精心编写了本部分辅导材料。

本部分特色如下所述：

- 按新大纲对知识点进行详细的论述，侧重归纳、总结，以便读者在短时间内掌握核心内容。
- 按新大纲的考核要求筛选等级考试试题，尽可能体现考试的重点和要求，并作解答与分析。

第1章 数据结构与算法

1.1 考试要点

1.1.1 算法基本知识

1. 算法

算法是对具体问题求解过程及步骤的一种描述，它是指令的有限序列，其中每一条指令表示一个或多个操作。

算法具有以下4个重要特征：

- 可行性——算法中描述的操作都是可以通过已经实现的基本运算执行有限次来实现的。
- 确定性——算法中每一条指令必须有确切的含义，不会产生二义性。并且在任何条件下，算法只有惟一的一条执行路径，即对于相同的输入只能得出相同的输出。
- 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，且每一步都可在有限时间内完成。
- 拥有足够的情报——一个算法执行的结果总是与输入的初始数据有关，不同的输入将会有不同的输出。当输入不够或输入错误时，算法本身无法执行或导致执行错误。所以，当算法拥有足够的情报来提供输入时，此算法才是有效的。

2. 算法的复杂度

算法复杂度是指一个算法在执行时所需的计算工作量和内存空间，主要包括时间复杂度和空间复杂度。

- 时间复杂度——通常用算法在执行过程中所需基本运算的执行次数来度量算法的工作量。基本运算反映了算法运算的主要特征，因此用基本运算的次数来度量算法工作量是客观的也是实际可行的，有利于比较同一问题的几种算法的优劣。
- 空间复杂度——指算法所占用的存储空间，包括算法程序所占用的空间、输入的初始数据所占的存储空间以及算法执行过程中所需要的额外空间。其中额外空间包括算法程序执行过程中的工作单元以及某种数据结构所需要的附加存储空间。

讨论算法复杂度的意义在于，若希望设计一个“好”的算法，往往要考虑使算法程序执行的时间尽可能短，算法执行过程中所需的存储空间尽可能小，这样的算法才是高效率的算法，才算是“好”的算法。

1.1.2 数据结构的基本概念

1. 数据结构的定义

数据是对客观事物的符号化表示,从计算机的角度来说,数据就是指所有能输入到计算机中并能被计算机处理的内容的总称。它是计算机处理加工的“原料”。数值、字符、汉字、图形、图像、声音、动画等均归为数据的范畴。数据元素是数据的基本单位,在计算机中通常作为一个整体进行考虑和处理。

简单地讲,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。数据结构的形式表示(定义)为 $\text{Data_Structure}=(D, S)$ 。其中 D 是数据元素的有限集合, S 是 D 上关系的有限集合。

2. 数据的逻辑结构与存储结构

通俗地说,数据结构是指带有结构的数据元素的集合。在此,结构实际上就是指数据元素之间前后的关系。其中,数据元素之间的前后数据关系是指它们的逻辑关系,而与它们在计算机中的存储位置无关。因此,以上所述的数据结构实际上是数据的逻辑结构,也就是反映数据元素之间逻辑关系的数据结构。数据的逻辑结构有两个要素:一是数据元素的集合,通常记为 D ;二是 D 上的关系,反映 D 中各数据元素间的关系,通常记为 R 。因此数据的逻辑结构可以表示为 $B=(D, R)$,这又称为数据结构的二元关系表示方式。

数据的逻辑结构在计算机存储空间中的实际存放形式称为数据的存储结构。由于数据元素在计算机存储空间中的位置关系可能与逻辑关系不同,因此,为了表示存放在计算机存储空间中的各数据元素之间的前后关系,在数据的存储结构中,不仅要存放各数据元素的信息,还需要存放各数据元素之间前后关系的信息。

3. 数据结构的图形表示

一个数据结构还可以直观地用图形来表示。在数据结构的图形表示中,数据集合中的每一个数据元素用中间标有元素内容的方框表示,这称为数据结点。为进一步表示各数据元素间的前后关系,通常用一条有向线段从前一数据元素指向后一数据元素。例如,一天3个时段(上午、下午、晚上)的数据结构可以用图1.1来表示,而反映家庭成员间辈分关系的数据结构可以用图1.2来表示。由此可见,用图形方式表示一个数据结构非常方便,也非常直观。

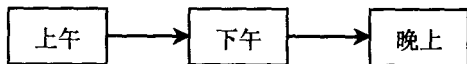


图 1.1

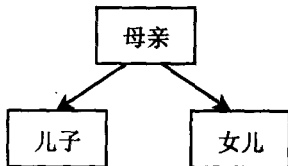


图 1.2

4. 线性结构与非线性结构

根据数据结构中各数据元素之间前后相关联的复杂程度,通常将数据结构分为两大类型:线性结构和非线性结构。

(1) 线性结构

如果一个非空的数据结构满足两个条件：一是有且只有一个根结点，二是每一个结点最多有一个前件一个后件，则称该数据结构为线性结构。

线性结构又称为线性表。如果一个数据结构不是线性结构，则称为非线性结构。例如图1.1中反映的一天3个时间段关系的数据结构是线性结构。

(2) 非线性结构

图1.2中反映的家庭成员间辈分关系的数据结构不是线性结构，而是非线性结构。显然，在非线性结构中，各数据元素之间的前后关系要比线性结构复杂。另外，线性结构与非线性结构都可以是空的数据结构。

1.1.3 线性表

1. 线性表的概念

线性表是数据结构中最简单、最重要的结构形式之一，是最经常遇到的一种操作对象，在程序设计语言和程序设计中广泛使用。

线性表是由 n ($n \geq 0$) 个数据元素 $a_1, a_2, a_3, \dots, a_n$ 构成的一个有限序列。除了第一个数据元素外，每一个数据元素有且只有一个前件；除了最后一个数据元素外，每一个数据元素有且只有一个后件。线性表是一种线性结构，数据元素在线性表中的位置只取决于它们自己的序号，即数据元素之间的相对位置是线性的。

2. 线性表的顺序存储结构

线性表的顺序存储结构是指线性表中所有元素所占的存储空间是连续的，且线性表中各数据元素在存储空间中是按逻辑顺序依次存放的。由此可见，在线性表的顺序存储结构中，其前后两个元素在存储空间中是紧邻的，且前一元素一定存储在后一元素的前面。

3. 线性表的插入运算

一般情况下，要在第 i ($1 \leq i \leq n$) 个元素之前插入一个新元素时，首先要从最后一个（即第 n 个）元素开始，到第 i 个元素之间共 $n-i+1$ 个元素依次向后移动一个位置，移动结束后，第 i 个位置就被空出，然后将新元素插入到第 i 项，插入后线性表的长度就增加了1。

显然，在线性表采用顺序存储结构时，如果插入运算在线性表末尾进行，即在第 n 个元素之后（可以认为是在第 $n+1$ 个元素之前）插入新元素，则只要在表的末尾增加一个元素即可，不需要移动表中的元素；如果要在线性表的第1个元素之前插入一个新元素，则需要移动表中所有的元素。

4. 线性表的删除运算

线性表的删除运算是指，要删除第 i ($1 \leq i \leq n$) 个元素时，要从第 $i+1$ 个元素开始，直到第 n 个元素共 $n-i$ 个元素依次向前移动一个位置。删除结束后，线性表的长度减小1。

显然，在线性表采用顺序存储结构时，如果删除运算在线性表的末尾进行，即删除第 n 个元素，则不需要移动表中的元素；如果要删除线性表中的第1个元素，则需要移动表中所有的元素。平均情况下，要在线性表中删除一个元素，需要移动表中一半的元素。

1.1.4 栈和队列

1. 栈的定义

栈是限定仅在表的一端进行插入或删除操作的线性表，所以它是一种特殊的线性表。即在这种线性表的结构中，一端是封闭的，不允许进行插入与删除元素；另一端是开口的，允许插入与删除元素。

对栈来说，允许插入或删除操作的一端称为栈顶，通常用指针 top 来指示栈顶的位置；相应地，不允许插入和删除操作的一端称为栈底。

栈顶元素总是最后被插入的元素，从而也是最先能被删除的元素；栈底元素总是最先被插入的元素，从而也是最后才能被删除的元素。即栈是按照“先进后出”或“后进先出”的原则组织数据的。不含数据元素的空表称为空栈。

2. 栈的顺序存储结构及其运算

在程序设计语言中，用一维数组 $s(1:m)$ 作为栈的顺序存储空间，其中 m 为栈的最大容量。在栈的顺序存储空间 $s(1:m)$ 中， $s(bottom)$ 通常为栈底元素（在栈非空的情况下）， $s(top)$ 为栈顶元素。 $top=0$ 表示栈空， $top=m$ 表示栈满。

栈的基本运算有入栈、退栈、读栈顶元素3种。

(1) 入栈运算

入栈运算是指在栈顶位置插入一个新元素。这个运算分为两个基本操作：首先将栈顶指针 top 加1，然后将新元素插入到栈顶指针指向的位置。当栈顶指针已经指向存储空间的最后一个位置时，说明栈空间已满，不可能再进行入栈操作，这种情况称为栈“上溢”错误。

(2) 退栈运算

退栈运算是指取出栈顶元素并赋给一个指定的变量。这个运算分为两个基本操作：首先将栈顶指针指向的栈顶元素赋给一个指定的变量，然后将栈顶指针退一（即 top 减1）。当栈顶指针为0时，说明栈空，不可能进行退栈操作，这种情况称为栈“下溢”错误。

(3) 读栈顶元素运算

读栈顶元素是指将栈顶元素赋给一个指定的变量。必须注意，这个运算不删除栈顶元素，只是将它的值赋给一个变量，因此在这个运算中，栈顶指针不会改变。当栈顶指针为0时，说明栈空，读不到栈顶元素。

3. 队列的定义

队列是指允许在一端进行插入而在另一端进行删除的线性表。允许插入的一端称为队尾，通常用一个称为尾指针（ $rear$ ）的指针指向队尾元素，即尾指针总是指向最后被插入的元素；允许删除的一端称为队头，通常用一个队头指针（ $front$ ）指向队头元素的前一个位置。显然，在队列这种数据结构中，最先插入的元素将最先能够被删除，反之，最后插入的元素将最后才能被删除。因此，队列又称为“先进先出”或“后进后出”的线性表。

4. 队列的顺序存储结构及其运算

与栈类似，在程序设计语言中，用一维数组 $s(1:m)$ 作为队列的顺序存储空间。在实际应

用中,队列的顺序存储结构一般采用循环队列的形式。所谓循环队列,就是将队列存储空间最后一个位置绕到第一个位置,形成逻辑上的环状空间,供队列循环使用。在循环队列中,用队尾指针rear指向队列中的队尾元素,用队头指针front指向队头元素的前一个位置。因此,从队头指针front指向的后一个位置直到队尾指针rear指向的位置之间所有的元素均为队列中的元素。注意,循环队列的初始状态为空,即 $rear=front=m$ 。

循环队列主要有两种基本运算,即入队运算与退队运算。

(1) 入队运算

入队运算是指在循环队列的队尾加入一个新元素。这个运算分两个基本操作:首先将队尾指针进一,即 $rear=rear+1$,并当 $rear=m+1$ 时置 $rear=1$;然后将新元素插入到队尾指针指向的位置。当循环队列非空($s=1$)且队尾指针等于队头指针时,说明循环队列已满,不能进行入队运算,这种情况称为“上溢”。

(2) 退队运算

退队运算是指在循环队列的队头位置退出一个元素并赋给指定的变量。这个运算分两个基本操作:首先将队头指针进一,即 $front=front+1$,并当 $front=m+1$ 时置 $front=1$;然后将队头指针指向的元素赋给指定的变量。当循环队列为空($s=0$)时,不能进行退队运算,这种情况称为“下溢”。

1.1.5 线性链表及其基本运算

线性表的顺序存储结构对于小线性表或长度固定的线性表来说较为方便。但对于大线性表特别是元素变动频繁的大线性表不宜采用。因此,对于大线性表,特别是元素变动频繁的大线性表,通常考虑使用链式存储结构的线性链表来处理。

在链式存储结构中,可以将每个存储数据元素的单元(又称存储结点或结点)分成两部分:一部分用于存放数据元素值(数据域),另一部分存放指向该结点的前一元素或后一元素的指针(指针域)。在链式存储结构中,存储数据的存储空间可以是不连续,各数据结点的存储顺序与数据元素之间的逻辑关系可以不一致,而数据元素之间的逻辑关系是由指针来确定的。

链式存储方式既可以用于表示线性结构,也可以表示非线性结构。线性表的链式存储结构称为线性链表。

1. 线性单链表及其运算

在定义的链表中,若只含有一个指针域来存放下一个元素地址,称这样的链表为单链表或线性链表。

单链表的常见运算有插入运算和删除运算。

(1) 插入运算

插入运算的思想是,将值为x的新结点插入到表的第i个结点的位置上,即插入到 a_{i-1} 与 a_i 之间。具体步骤如下:

- ① 找到 a_{i-1} 的存储位置p。

- ② 生成一个数据域为x的新结点*s。
- ③ 令结点*p的指针域指向新结点。
- ④ 新结点的指针域指向结点 a_i 。

(2) 删除运算

删除运算的思想是将表的第i个结点删去。具体步骤如下:

- ① 找到 a_{i-1} 的存储位置p (因为在单链表中, 结点 a_i 的存储地址是在其直接前趋结点 a_{i-1} 的指针域next中)。
- ② 令 $p \rightarrow \text{next}$ 指向 a_i 的直接后继结点 (即把 a_i 从链上摘下)。
- ③ 释放结点 a_i 的空间, 将其归还给存储池。



注意: 设单链表的长度为n, 则删去第i个结点仅当 $1 \leq i \leq n$ 时是合法的。当 $i=n+1$ 时, 虽然被删结点不存在, 但其前趋结点却存在, 它是终端结点。因此被删结点的直接前趋*p存在并不意味着被删结点就一定存在, 仅当*p存在 (即 $p! = \text{NULL}$) 且*p不是终端结点 (即 $p \rightarrow \text{next}! = \text{NULL}$) 时, 才能确定被删结点存在。

2. 双向链表及其运算

双向链表中有两条方向不同的链, 即每个结点中除next域存放后继结点地址外, 还增加一个指向其直接前趋的指针域prior。



注意: 双向链表是由头指针head惟一确定的; 带头结点的双向链表的某些运算将变得方便。将头结点和尾结点链接起来, 即为双向循环链表。

双向链表的常见操作有插入和删除本结点操作。由于双向链表的对称性, 在双向链表中能方便地完成各种插入和删除操作。注意: 与单链表上的插入和删除操作不同的是, 在双链表中插入和删除必须同时修改两个方向上的指针。

3. 循环链表及其运算

循环链表是一种首尾相接的链表。循环链表可分为单循环链表和多重链的循环链表。

- 单循环链表——在单链表中, 将终端结点的指针域NULL改为指向表头结点或开始结点即可。
- 多重链的循环链表——将表中结点链在多个环上。

循环链表的特点是无须增加存储量, 仅对表的链接方式稍作改变, 即可使得表处理起来更加方便灵活。

例如, 在链表上实现将两个线性表 $(a_1, a_2, \dots, a_n)$ 和 $(b_1, b_2, \dots, b_m)$ 连接成一个线性表 $(a_1, \dots, a_n, b_1, \dots, b_m)$ 的运算。

若采用单链表或头指针表示的单循环表, 都需要遍历第一个链表以找到结点 a_n , 然后将结点 b_1 链到 a_n 的后面。但若采用尾指针表示的单循环链表, 则只需修改指针即可, 无须遍历。



注意：① 循环链表中没有NULL指针。涉及遍历操作时，其终止条件就不再是像非循环链表那样判别 p 或 $p \rightarrow next$ 是否为空，而是判别它们是否等于某一指定指针，例如，头指针或尾指针等。

② 在单链表中，从一已知结点出发，只能访问到该结点及其后续结点，无法找到该结点之前的其他结点。而在单循环链表中，从任一结点出发都可访问到表中所有结点，这一优点使某些运算在单循环链表上易于实现。

1.1.6 树的基本知识

1. 树的基本概念

树是一种简单的非线性结构。在树这种数据结构中，所有数据元素之间的关系具有明显的层次特性。树的定义是：树(Tree)是 n ($n \geq 0$)个结点的有限集 T ， T 为空时称为空树，否则它满足如下两个条件：

- ① 有且仅有一个特定的称为根(Root)的结点；
- ② 其余的结点可分为 m ($m \geq 0$)个互不相交的子集 $T_1, T_2, \dots, T_m$ ，其中每个子集本身又是一棵树，并称其为根的子树(SubTree)。



注意：树的这种递归定义刻画了树的固有特性，即一棵非空树是由若干棵子树构成的，而子树又可由若干棵更小的子树构成。

下面介绍树结构中有的基本术语。

(1) 结点的度(Degree)

树中的一个结点拥有的子树数称为该结点的度(Degree)。

- 一棵树的度是指该树中结点的最大度数。
- 度为零的结点称为叶子(Leaf)或终端结点。
- 度不为零的结点称为分支结点或非终端结点。
- 除根结点之外的分支结点统称为内部结点。
- 根结点又称为开始结点。

(2) 孩子(Child)和双亲(Parents)

树中某个结点的子树之根称为该结点的孩子(Child)或儿子，相应地，该结点称为孩子的双亲(Parents)或父亲。同一个双亲的孩子称为兄弟(Sibling)。

(3) 祖先(Ancessor)和子孙(Descendant)

若树中结点 k 到 k_s 存在一条路径，则称 k 是 k_s 的祖先(Ancessor)， k_s 是 k 的子孙(Descendant)。一个结点的祖先是从小根结点到该结点路径上所经过的所有结点，而一个结点的子孙则是以该结点为根的子树中的所有结点。

约定结点 k 的祖先和子孙不包含结点 k 本身。