

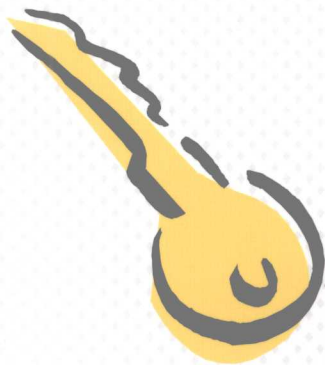


全国 计算机等级考试 过关 宝典

——二级公共基础知识

翟自强 马志强 主编

- ◆按照最新大纲 紧扣考试重点
- ◆精心编排内容 知识易懂易记
- ◆大量模拟习题 细致全面解析
- ◆真题实战热身 助你顺利过关



内容简介

全国计算机等级考试过关宝典—— 二级公共基础知识

翟自强 马志强 主 编

王宝和 巩国忠

张丹阳 张振河 副主编

李玉石 李红英

ISBN 7-309-04743-3

出 版 社
人 社 出 版 社
社 址
话 号
址 网
编 印
册 登
本 开
张 印
数 字
次 册
次 印
售 印
价 宝

 **天津大学出版社**
TIANJIN UNIVERSITY PRESS

天津大学出版集团 天津大学出版中心 天津大学出版集团 天津大学出版中心

天津大学出版集团 天津大学出版中心

内容简介

本书按教育部考试中心指定教程的章节分类编排,并按考试大纲的要求对近5年来的试题进行详细的分析,对相关知识点进行详尽的介绍,使考生能够熟悉全国计算机等级考试二级公共基础知识的内容,并抓住考试的重点与难点,掌握考试中经常出现的题型和每种题型的解法,同时也使考生熟悉专家们的出题思路、命题规律,从而提高应试复习的效率,做到有的放矢。

本书还根据最新大纲(2008年版)的要求,对二级公共基础知识按章节进行了样题汇编与详解。另外,本书还提供了公共基础知识模拟练习题10套,方便每位考生自测使用,并附有最新的历年试题及答案。本书针对性强,特别适合参加全国计算机等级考试二级考试的考生,也可以作为各类大专院校学生学习的参考书。

图书在版编目(CIP)数据

全国计算机等级考试过关宝典. 二级公共基础知识/
翟自强,马志强主编. —天津:天津大学出版社,2009.5
ISBN 978-7-5618-3036-9

I. 全… II. ①翟…②马… III. 电子计算机—水平考试—自学参考资料 IV. TP3

中国版本图书馆CIP数据核字(2009)第073574号

出版发行 天津大学出版社
出版人 杨欢
地 址 天津市卫津路92号天津大学内(邮编:300072)
电 话 发行部:022-27403647 邮购部:022-27402742
网 址 www.tjup.com
印 刷 天津泰宇印务有限公司
经 销 全国各地新华书店
开 本 185mm×260mm
印 张 10.25
字 数 256千
版 次 2009年5月第1版
印 次 2009年5月第1次
印 数 1-6 000
定 价 20.00元

凡购本书,如有缺页、倒页、脱页等质量问题,烦请向我社发行部门联系调换

版权所有 侵权必究

前 言

多年来,职业认证已经成为计算机行业中一个重要的组成部分,并将变得更加重要。如果你已经是一位 IT 业者,那么将不得不时刻面对新的挑战以及日趋激烈的竞争,将不可避免地遭遇一系列的考试和认证,唯有顺利通过才能铺平成功之路。

全国计算机等级考试(以下简称等级考试)是经原国家教委批准并由考试中心于 1994 年面向社会推出的、用于测试应试者计算机应用知识与能力的非学历教育证书考试。其目的在于推进计算机知识的普及,促进计算机技术的推广应用,为人员择业、人才流动提供权威性证明。

考试主要针对非计算机专业人员,其中又以在校大、中学生居多,兼有机关、企事业单位人员等,在很大程度上满足了计算机初学者自我测试的要求,同时也满足了用人单位对大批掌握和具有计算机知识与能力的人才的实际需要。

本书旨在向读者介绍全国计算机等级考试二级公共基础知识方面的笔试内容,并帮助读者通过全国计算机等级考试二级公共基础知识部分的考试。

作者集多年从事计算机等级考试培训的工作经验和体会,精心编写本教材。全书内容编排合理,紧扣新考试大纲,循序渐进,突出重点。对全国计算机等级考试二级公共基础知识的题型进行分章分类,并为每道题做出详细的解析,还为考生提供了最新的历年试题,供考生快速掌握各章知识考点及题型,更容易地掌握二级公共基础知识的考试内容。

本书参编人员有翟自强、马志强、王宝和、巩国忠、吴迪、张振河、李玉石、李红英、于筱荔、张丹阳、李强、冯光、柴君、朱云霞、王宇、刘薇,由翟自强、马志强任主编,王宝和、巩国忠、张丹阳、张振河、李玉石、李红英任副主编。

本书面向准备参加全国计算机等级考试二级各科的考生,适用于普通高校、高职高专以及各类培训学校作为考前辅导的培训教材以及学生自学。

编者

目 录

第 1 部分 公共基础知识考试大纲及分析

公共基础知识考试大纲	(1)
全国计算机等级考试二级公共基础知识考纲分析	(3)

第 2 部分 公共基础知识教程及习题

第 1 章 数据结构与算法	(4)
1.1 学习目标与要求	(4)
1.2 内容要点	(4)
1.2.1 算法	(4)
1.2.2 数据结构的基本概念	(7)
1.2.3 线性表及其顺序存储结构	(8)
1.2.4 栈和队列	(9)
1.2.5 线性链表	(11)
1.2.6 树与二叉树	(14)
1.2.7 查找技术	(17)
1.2.8 排序技术	(19)
1.3 例题分析	(22)
1.4 本章自测习题	(24)
1.5 本章自测习题答案	(31)
第 2 章 程序设计基础	(32)
2.1 学习目标与要求	(32)
2.2 内容要点	(32)
2.2.1 程序设计方法与风格	(32)
2.2.2 结构化程序设计	(33)
2.2.3 面向对象的程序设计	(34)
2.3 例题分析	(36)
2.4 本章自测习题	(37)
2.5 本章自测习题答案	(39)
第 3 章 软件工程基础	(40)
3.1 学习目标与要求	(40)
3.2 内容要点	(40)
3.2.1 软件工程基本概念	(40)
3.2.2 结构化分析方法	(43)

3.2.3	结构化设计方法	(45)
3.2.4	软件测试	(48)
3.2.5	程序的调试	(51)
3.3	例题分析	(52)
3.4	本章自测习题	(53)
3.5	本章自测习题答案	(57)
第4章	数据库设计基础	(58)
4.1	学习目标与要求	(58)
4.2	内容要点	(58)
4.2.1	数据库系统的基本概念	(58)
4.2.2	数据模型	(62)
4.2.3	关系代数	(66)
4.2.4	数据库设计与管理	(69)
4.3	例题分析	(74)
4.4	本章自测习题	(76)
4.5	本章自测习题答案	(80)
第3部分 公共基础知识习题解析		
第1章	数据结构	(82)
第2章	程序设计基础	(102)
第3章	软件工程基础	(109)
第4章	数据库设计基础	(124)
第4部分 公共基础知识模拟练习		
模拟练习1		(140)
模拟练习2		(141)
模拟练习3		(142)
模拟练习4		(143)
模拟练习5		(145)
模拟练习6		(146)
模拟练习7		(147)
模拟练习8		(149)
模拟练习9		(150)
模拟练习10		(151)
附录1	2008年4月全国计算机等级考试公共基础部分笔试试题	(153)
	2008年4月全国计算机等级考试公共基础部分笔试试题答案	(154)
附录2	2008年9月全国计算机等级考试公共基础部分笔试试题	(155)
	2008年9月全国计算机等级考试公共基础部分笔试试题答案	(156)

第1部分 公共基础知识考试 大纲及分析

公共基础知识考试大纲

► 基本要求

- ◆ 掌握算法的基本概念。
- ◆ 掌握基本数据结构及其操作。
- ◆ 掌握基本排序和查找算法。
- ◆ 掌握逐步求精的结构化程序设计方法。
- ◆ 掌握软件工程的基本方法,具有初步应用相关技术进行软件开发的能力。
- ◆ 掌握数据库的基本知识,了解关系数据库的设计。

► 考试内容

一、基本数据结构与算法

- ◆ 算法的基本概念;算法复杂度(时间复杂度与空间复杂度)的概念和意义。
- ◆ 数据结构的定义;数据的逻辑结构与存储结构;数据结构的图形表示;线性结构与非线性结构的概念。
 - ◆ 线性表的定义;线性表的顺序存储结构及其插入与删除运算。
 - ◆ 栈和队列的定义;栈和队列的顺序存储结构及其基本运算。
 - ◆ 线性单链表、双向链表与循环链表的结构及其基本运算。
 - ◆ 树的基本概念;二叉树的定义及其存储结构;二叉树的前序、中序和后序遍历。
 - ◆ 顺序查找与二分法查找算法;基本排序算法(交换类排序,选择类排序,插入类排序)。

二、程序设计基础

- ◆ 程序设计方法与风格。
- ◆ 结构化程序设计。
- ◆ 面向对象的程序设计方法以及对象、方法、属性及继承与多态性。

三、软件工程基础

- ◆ 软件工程基本概念,软件生命周期概念,软件工具与软件开发环境。
- ◆ 结构化分析方法:数据流图,数据字典,软件需求规格说明书。
- ◆ 结构化设计方法:总体设计与详细设计。
- ◆ 软件测试的方法:白盒测试与黑盒测试,测试用例设计,软件测试的实施,单元测试、集成测试和系统测试。
 - ◆ 程序的调试:静态调试与动态调试。

四、数据库设计基础

- ◆数据库的基本概念:数据库,数据库管理系统,数据库系统。
- ◆数据模型,实体联系模型及 E-R 图,从 E-R 图导出关系数据模型。
- ◆关系代数运算,包括集合运算及选择、投影、连接运算,数据库规范化理论。
- ◆数据库设计方法和步骤:需求分析、概念设计、逻辑设计和物理设计的相关策略。

►考试方式

公共基础知识有 10 道选择题和 5 道填空题,共 30 分。

二级公共基础知识

公共基础知识

数据库设计基础

数据库的基本概念

数据模型,实体联系模型及 E-R 图

关系代数运算

数据库设计方法和步骤

数据库设计方法和步骤:需求分析、概念设计、逻辑设计和物理设计的相关策略。

数据库设计方法和步骤:需求分析、概念设计、逻辑设计和物理设计的相关策略。

考试内容

公共基础知识

数据库设计基础

数据库设计基础

公共基础知识

数据库设计基础

数据库设计基础

数据库设计基础

数据库设计基础

数据库设计基础

公共基础知识

数据库设计基础

数据库设计基础

数据库设计基础

公共基础知识

数据库设计基础

数据库设计基础

数据库设计基础

数据库设计基础

公共基础知识

数据库设计基础

全国计算机等级考试二级公共基础 知识考纲分析

计算机二级考试是以程序设计为主的计算机等级考试,目的是促进考生学习程序设计的热情,提高考生的程序设计水平。而程序设计离不开算法、软件工程等知识。本课程作为计算机二级考试的公共基础课程,从理论的角度对数据结构、软件工程、结构化程序设计与面向对象的程序设计、数据库基础知识进行了简单的介绍,以此来扩展考生的知识面,并使之对程序设计知识有系统的了解。

公共基础知识在二级等级考试的各项科目中占有30分的笔试题目,其中选择题占10题,每题2分,共20分;填空题5个空,每空2分,共10分。本课程一共有4个章节。第1章主要介绍算法的基本概念,数据结构的基本概念和定义,线性表及其基本运算,二叉树的基本概念、存储结构及其应用,并介绍了一些常用的算法;第2章主要介绍程序设计的方法与风格,结构化程序设计,面向对象的程序设计方法以及对象、方法、属性及继承与多态性;第3章主要介绍软件工程的基本概念、结构化分析方法、结构化设计方法、软件测试的基本方法和程序的调试方法,从工程的角度对软件开发进行了介绍;第4章主要介绍数据库、数据库管理系统、数据库系统的基本概念、数据模型、实体联系模型及E-R图等基本概念、关系代数理论中的基本运算,数据库设计的基本方法和步骤。

1. 数据结构与算法

本章知识用于提高编制程序的效率以及对较复杂的问题进行求解。本章内容在计算机专业基础课中也是比较难的一门,学习本章的内容必须进行理解,死记硬背是无效的。对于等级考试,本章考核重点主要在二叉树,同时这也是本章的难点。考核形式主要为二叉树的遍历问题(如给图求遍历序列,给前序、中序遍历求后序遍历等)、二叉树的节点问题(如给出一些条件然后求叶子节点个数),排序和查找也经常涉及。排序主要考核计算时间复杂度,查找主要考核计算最佳/最坏比较次数。其余知识点主要考核概念。

2. 程序设计基础与软件工程基础

这两章以概述的形式简介了规范化开发软件的方法。与数据结构不同,这两章内容主要靠记忆。程序设计基础的内容与大纲改革前相比添加了面向对象程序设计的内容,考生对本章进行几次细读后了解即可;软件工程基础主要考核内容为结构化分析及结构化设计方法(即SA及SD,约占50%),信息量较大,其次是软件测试(约占20%),考生需要将相关的概念及规则背下来,在以后有机会开发程序时就可深刻理解。

3. 数据库设计基础

数据库是当前处理数据的核心,目前大部分软件都是基于数据库的,因此学习数据库知识对程序开发也是很有帮助的。本章主要的考核点是关系模型、关系代数及数据库系统的基本概念,其余知识了解即可。其中数据库的设计和管理可以结合软件工程来看,考生会发现这两者有很多相似之处。除了关系代数会考一些简单的计算问题外,其余都是以概念题的形式考核,考生需要仔细阅读。

第2部分 公共基础知识教程及习题

第1章 数据结构与算法

1.1 学习目标与要求

- ▶ 了解算法的基本概念和一些常用的算法,学会计算算法的时间复杂度。
- ▶ 掌握数据结构的基本概念,并了解数据的逻辑结构和存储结构,学会利用图形表示数据结构。
- ▶ 了解线性表的基本概念,并掌握线性表的顺序存储结构以及顺序存储的线性表的基本运算。
- ▶ 了解栈和队列的基本概念,并掌握它们的基本运算。
- ▶ 了解线性链表的基本概念,并掌握线性链表的基本运算。同时,了解循环链表的基本概念和基本操作。
- ▶ 理解树的概念,尤其是二叉树的基本概念和相关性质,掌握二叉树的存储结构和遍历技术。
- ▶ 掌握查找技术,学会利用顺序查找和二分查找在数列中查找指定的数据。
- ▶ 学会利用相关的排序技术实现无序数列的排序操作。

1.2 内容要点

1.2.1 算法

1. 算法的基本概念

算法是指对解题方案进行的准确而完整的描述。换句话说,算法是对特定问题求解步骤的一种描述。

1) 算法的基本特征

(1) 可行性

由于算法是为在某一个特定计算工具上解决某一实际问题而设计的,因此,它总是受到计算工具的限制,使执行产生偏差。

如:计算机的数值有效位是有限的,当大数与小数进行运算时,往往会因为有效位数的影

响而使小数丢失,因此,在设计算法时应该考虑到这一点。

(2) 确定性

每一条指令的含义明确,无二义性,并且在任何条件下,算法只有唯一的一条执行路径,即相同的输入只能得出相同的输出。

(3) 有穷性

算法的有穷性是指计算在一定时间内是能够完成的,即算法应该在计算有限个步骤后正常结束。

例如,数学中的无穷级数只能在计算机中求有限项,即计算过程是有穷的。

(4) 拥有足够的情报

算法中各种运算总是要施加到各个运算对象上,而这些运算对象又可能具有某种初始状态,这就是算法执行的起点或依据。因此,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结果输出。当输入不够或输入错误时,算法将无法执行或执行有错。一般说来,当算法拥有足够的情报时,此算法才是有效的;而当提供的情报不够时,算法可能无效。

综上所述,所谓算法,是一组严谨定义运算顺序的规则,并且每一个规则都是有效而且是明确的。

2) 算法的基本要素

算法的基本要素一是数据对象的运算和操作;二是算法的控制结构。

(1) 算法中对数据的运算和操作

算法实际上是按解题要求从环境能进行的所有操作中选择合适的操作组成的一组指令序列,亦即算法是计算机所能够处理的操作组成的指令序列。

(2) 算法的控制结构

算法的功能不仅取决于所选用的操作,而且还与各操作之间的顺序有关。

在算法中,操作的执行顺序又称算法的控制结构。算法控制结构有三种,即顺序结构、选择结构和循环结构。

描述算法的常用工具有流程图、N-S 结构图和算法描述语言等。

3) 算法设计的基本方法

为用计算机解决实际问题而设计的算法,即是计算机算法。

通常的算法设计有如下几种。

(1) 列举法

列举法的基本思想是,根据提出的问题,列举出所有可能的情况,并用问题中给定的条件检验哪些是满足条件的,哪些是不满足条件的。列举法通常用于解决“是否存在”或“有哪些可能”等问题。

例如,我国古代的趣味数学题“百钱买百鸡”、“鸡兔同笼”等,均可采用列举法解决。

使用列举法时,要对问题进行详细分析,将与问题有关的知识条理化、完备化、系统化,从中找出规律。

(2) 归纳法

归纳法的基本思想是,通过列举少量的特殊情况,经过分析,最后找出一般的关系。归纳是一种抽象,即从特殊现象中找出一般规律。但由于在归纳法中不可能对所有情况进行列

举,因此,该方法得到的结论只是一种猜测,还需要进行证明。

(3) 递推

递推,即从已知的初始条件出发,逐次推出所要求的各个中间环节和最后结果。其中,初始条件或问题本身已经给定,或是通过对问题的分析与化简而确定。

递推的本质也是一种归纳,递推关系式通常是归纳的结果。

例如,斐波那契数列就是采用递推方法解决问题的。

(4) 递归

在解决一些复杂问题时,为了降低问题的复杂程度,通常是将问题逐层分解,最后归结为一些最简单的问题。这种将问题逐层分解的过程,并没有对问题进行求解,而只是当解决了最后的问题,即那些最简单的问题后,再沿着原来分解的逆过程逐步进行综合,这就是递归的方法。

递归分为直接递归和间接递归两种方法。如果一个算法直接调用自己,称为直接递归调用;如果一个算法 A 调用另一个算法 B,而算法 B 又调用算法 A,则此种递归称为间接递归调用。

(5) 减半递推技术

减半递推即将问题的规模减半,然后重复相同的递推操作。

例如,求解一元二次方程。

(6) 回溯法

有些实际问题很难归纳出一组简单的递推公式或直观的求解步骤,也不能使用无限的列举。对于这类问题,只能采用试探的方法,通过对问题的分析,找出解决问题的线索,然后沿着这个线索进行试探。如果试探成功,就得到问题的解;如果不成功,再逐步回退,换别的路线进行试探。这种方法,即称为回溯法。

例如,人工智能中的机器人下棋。

2. 算法复杂度

算法的复杂度包括时间复杂度和空间复杂度。

1) 时间复杂度

时间复杂度指实现该算法需要的计算工作量。算法的工作量用算法所执行的基本运算次数来计算,即

$$\text{算法工作量} = f(n)$$

对同一个问题,如果算法执行所需要的基本次数取决于某一特定输入时,可以用以下两种方法分析算法的工作量。

(1) 平均性态

平均性态用各种特定输入下的基本运算次数的加权平均值度量算法的工作量。

设 x 是某个可能输入中的某个特定输入, $p(x)$ 是 x 出现的概率, $t(x)$ 是算法在输入为 x 时执行的基本运算次数,则算法的平均性态定义为:

$$A(n) = \sum_{x \in D_n} p(x)t(x)$$

D_n 表示当规模为 n 时,算法执行时所有可能输入的集合。

(2) 最坏情况复杂度

最坏情况复杂度指在问题规模为 n 时,算法所执行的基本运算的最大次数。它定义为:

$$W(n) = \max_{x \in D_n} \{t(x)\}$$

例如,在具有 n 个元素的数列中搜索一个数 x ,平均性态

$$A(n) = \sum_{i=1}^{n+1} p_i t_i = \sum_{i=1}^n \frac{q}{n} i + (1-q)n = \frac{(n+1)q}{2} + (1-q)n$$

即该数在数列中任何位置出现的概率是相同的,也有可能不存在,存在的概率为 q 。

如果有一半的机会存在,则概率 q 为 $1/2$,平均性态为:

$$A(n) = \frac{(n+1) \times \frac{1}{2}}{2} + \left(1 - \frac{1}{2}\right)n \approx \frac{3}{4}n$$

如果查找的元素一定在数列中,则每个数存在的概率即为 1 ,则平均性态为:

$$A(n) = \frac{n+1}{2} = \frac{2}{2}$$

最坏情况分析:即要查找的元素 x 在数列的最后或不在数列中。显然,它的最坏情况复杂度为:

$$W(n) = \max \{t_i | 1 \leq i \leq n+1\} = n$$

2) 空间复杂度

算法的空间复杂度指执行该算法需要的内存空间。算法所占用的内存空间包括算法程序所占的空间、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(如执行过程中工作单元以及某种数据结构所需要的附加存储空间)等。

1.2.2 数据结构的基本概念

1. 概念

数据结构主要研究和讨论以下三个方面的问题:

- ①数据集中各数据元素之间固有的逻辑关系,即数据的逻辑结构;
- ②在对数据进行处理时,各数据元素在计算机中的存储关系,即数据的存储结构;
- ③对各种数据结构进行的运算。

数据结构是指相互有关联的数据元素的集合。它包括以下两个方面:

- ①表示数据元素的信息;
- ②表示各数据之间的前后件关系。

1) 数据的逻辑结构

数据的逻辑结构是指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:

- ①数据元素的集合,记作 D ;
 - ②数据之间的前后件关系,记作 R ;
- 则数据结构 $B = (D, R)$ 。

2) 数据的存储结构

数据的逻辑结构在计算机存储空间中的存放形式称为数据的存储结构或数据的物理结构。即数据存储时,不仅要存放数据元素的信息,而且要存储数据元素之间的前后件关系的信息。

通常的数据存储结构有顺序、链接、索引等。

(1) 顺序存储

顺序存储是把逻辑上相邻的节点存储在物理位置相邻的存储单元里,节点间的逻辑关系由存储单元的邻接关系体现。由此得到的存储表示称为顺序存储结构。

(2) 链接存储

它不要求逻辑上相邻的节点在物理位置上亦相邻,节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。

(3) 索引存储

索引存储是指除建立存储节点信息外,还建立附加的索引表标识节点的地址。

数据的逻辑结构反映数据元素之间的逻辑关系,数据的存储结构(也称数据的物理结构)是数据的逻辑结构在计算机存储空间中的存放形式。同一逻辑结构的数据可以采用不同的存储结构,但不同的存储结构会影响数据处理效率。

2. 数据结构的图形表示

数据结构的图形表示有两个元素:中间标有元素值的方框表示数据元素,称为数据节点;用有向线段表示数据元素之间的前后件关系,即有向线段从前件节点指向后件节点。

注意:在结构图中,没有前件的节点称为根节点,没有后件的节点称为终端节点,也称叶子节点。

3. 线性结构与非线性结构

如果一个数据元素都没有,该数据结构称为空数据结构。在空数据结构中插入一个新的元素后,数据结构变为非空数据结构。将数据结构中的所有元素删除,则该数据结构变成空数据结构。

非空的线性数据结构满足如下条件:

①有且只有一个根节点;

②每一个节点最多只有一个前件,也最多只有一个后件。

线性结构又称线性表。在线性结构表中插入或删除元素,该线性表仍然应满足线性结构。

如果一个数据结构不满足线性结构,则称为非线性结构。

1.2.3 线性表及其顺序存储结构

1. 基本概念

线性表是最常用的数据结构,它由一组数据元素组成。

注意:这里的数据元素是一个广义的概念,并不仅仅是指一个数据,如,矩阵、学生记录表等。

非空线性表的结构特征是有且只有一个根节点,它无前件;有且只有一个终端节点,它无后件;除根节点和终端节点之外,所有的节点有且只有一个前件和一个后件。线性表中节点的个数称为线性表的长度 n 。当 $n=0$ 时,称为空表。

2. 顺序存储结构

顺序存储结构的特点如下:

①线性表中所有元素的存储空间是连续的;

②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。

通常,顺序存储结构中,线性表中每一个数据元素在计算机存储空间中的存储地址由该元素在线性表中的位置序号唯一确定。

由此可以看出,在线性表的顺序存储结构中,前后件两个元素在存储空间中是紧邻的,且前件元素一定存储在后件元素的前面,可以通过计算机直接确定第 i 个节点的存储地址。

线性表顺序存储结构的基本运算如下:

- ①在指定位置插入一个元素;
- ②删除线性表中的指定元素;
- ③查找某个或某些特定的元素;
- ④排序;
- ⑤按要求将一个线性表拆分为多个线性表;
- ⑥将多个线性表合并为一个线性表;
- ⑦复制线性表;
- ⑧逆置一个线性表。

3. 线性表的基本操作

1) 顺序表的插入运算

这是指在顺序存储结构的线性表中插入一个元素。

在一般情况下,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,直到第 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。插入结束后,线性表的长度就增加1。

顺序表的插入运算需要移动元素,在等概率情况下,平均需要移动 $n/2$ 个元素。

注意:找到插入位置后,将插入位置开始的所有元素从最后一个元素开始顺序后移。另外,在定义线性表时,一定要定义足够的空间,否则,将不允许插入元素。

2) 顺序表的删除运算

这是指在顺序存储结构的线性表中删除一个元素。

在一般情况下,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 个元素之间共 $n-i$ 个元素依次向前移动一个位置。删除结束后,线性表的长度就减小1。

进行线性表的删除运算时也需要移动元素,在等概率情况下,平均需要移动 $(n-1)/2$ 个元素。插入、删除运算不方便。

1.2.4 栈和队列

1. 栈及其基本运算

1) 栈

栈是一种特殊的线性表。它的插入和删除只能在表的一端进行,而另一端是封闭的,不允许进行插入和删除操作。

在栈中,允许插入和删除操作的一端称为栈顶,不允许插入和删除操作的一端则称为栈底。栈顶的元素总是最后被插入的元素,也是最先被删除的元素。它遵循的原则是:先进后出和后进先出。栈具有记忆作用。堆栈指针总是指向栈顶元素的。

2) 栈的顺序存储及其运算

在栈的顺序存储空间 $S(1:m)$ 中, $S(\text{bottom})$ 通常为栈底元素, $S(\text{top})$ 为栈顶元素。 $\text{top} = 0$ 表示栈空; $\text{top} = m$ 表示栈满。

3) 入栈运算

入栈运算即在栈的顶部插入一个新元素。操作方法是:将栈顶指针加1,再将元素插入至

指针所指的位置。

4) 退栈运算

退栈运算即将栈顶元素取出并赋给一个指定的变量。操作方法是:先将栈顶元素赋给指定的变量,再将栈顶指针减1。

5) 读栈顶元素

将栈顶元素赋给某一指定变量,但栈顶指针不变。

2. 队列及其基本运算

1) 队列

队列即是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,通常用一个尾指针指向队尾;允许删除的一端称为队首,通常用一个队首指针指向排队元素的前一个位置。

队列遵循的规则是:先进先出和后进后出。

2) 循环队列及其运算

队列的顺序存储结构一般采用循环形式。

循环队列,即是次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 指向队尾元素,用排头指针 front 指向排头元素的前一个位置。因此,从排头指针 front 指向的后一个位置到队尾指针 rear 指向的位置之间所有的元素均为队列中的元素。

循环队列的初始状态为空,即 $rear = front = m$ 。这里 m 即为队列的存储空间。

循环队列的基本运算包括入队运算和退队运算。

入队运算指每进行一次入队运算,队尾指针加1。当队尾指针 $rear = m + 1$ 时,即表示队列空间的尾部已经放置了元素,则下一个元素应该旋转到队列空间的首部,即 $rear = 1$ 。

退队运算指每退队一个元素,排头指针加1。当排头指针 $front = m + 1$ 时,即排头指针指向队列空间的尾部,退队后,排头指针指向队列空间的开始,即 $front = 1$ 。

在队列操作时,循环队列满时 $front = rear$,队列空时也有 $rear = front$,即在队列空或满时,排头指针和队尾指针均指向同一个位置。要判断队列空或满时,还应增加一个标志 s , s 值的定义:

$$s = \begin{cases} 0 & \text{表示队列空} \\ 1 & \text{表示队列满} \end{cases}$$

判断队列空与队列满的条件如下:

队列空的条件 $s = 0$

队列满的条件 $s = 1, front = rear$

(1) 入队运算

入队运算是在队尾加入一个新元素。这个运算有两个基本操作:首先,将队尾指针加1,即 $rear = rear + 1$,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 $front = rear$ 时,队列满,不能进行入队操作。此情况称“上溢”。

(2) 退队运算

退队运算将队首的元素赋给一个指定的变量。该运算也有两个基本操作:首先,将排头指针加1,即 $front = front + 1$,当 $front = m + 1$ 时,置 $front = 1$ 。然后,将排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。此种情况称为“下溢”。

1.2.5 线性链表

1. 基本概念

前面的线性表均采用顺序存储结构并在顺序存储结构下进行运算。

1) 顺序存储的优点

顺序存储的优点是结构简单、运算方便。

2) 顺序存储的缺点

顺序存储的缺点如下:

①插入或删除的运算效率很低。在顺序存储的线性表中,插入或删除数据元素时需要移动大量的数据元素。

②线性表的顺序存储结构下,线性表的存储空间不便于扩充。如果一个线性表分配顺序存储空间后,出现线性表的存储空间已满,再继续插入元素时,会发生“上溢”错误。

③线性表的顺序存储结构不便于对存储空间的动态分配。在实际应用时,可能有多个线性表同时使用存储空间,这样给存储空间的分配带来问题,有可能使有的队列空间不够或过多造成浪费。

基于上述情况,大线性表或元素变动频繁的大线性表不宜采用顺序存储结构,而应采用链式存储结构。

3) 链式存储结构

假设每一个数据节点对应一个存储单元,该存储单元称为存储节点,简称节点。

在链式存储方式中,要求每一个节点由两部分组成:一部分用于存放数据元素,称为数据域;另一部分用于存放指针,称为指针域。该指针用于指向该节点的前一个或后一个节点。

在链式存储结构中,存储数据结构的存储空间可以不连续,各数据节点的存储顺序与数据元素之间的逻辑关系不一致,而数据元素之间的逻辑关系是由指针域确定的。

链式存储结构既可以用于线性结构,也可用于非线性结构。

4) 线性链表

线性表的链式存储结构称为线性链表。

将存储空间划分成若干的小块,每块占用若干字节,这些小块称为存储节点。

将存储节点分为两个部分,一部分用于存储数据元素的值,称为数据域;另一部分用于存储元素之间的前后件关系,即存放下一个元素的存储序号(即存储地址),即指向后件节点,称为指针域,如图 2-1-1 和图 2-1-2 所示。

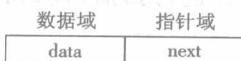


图 2-1-1 节点结构



图 2-1-2 一个非空的线性链表示意图