清华天骄

教育部考试中心指定教材配套辅导

NCR Examination

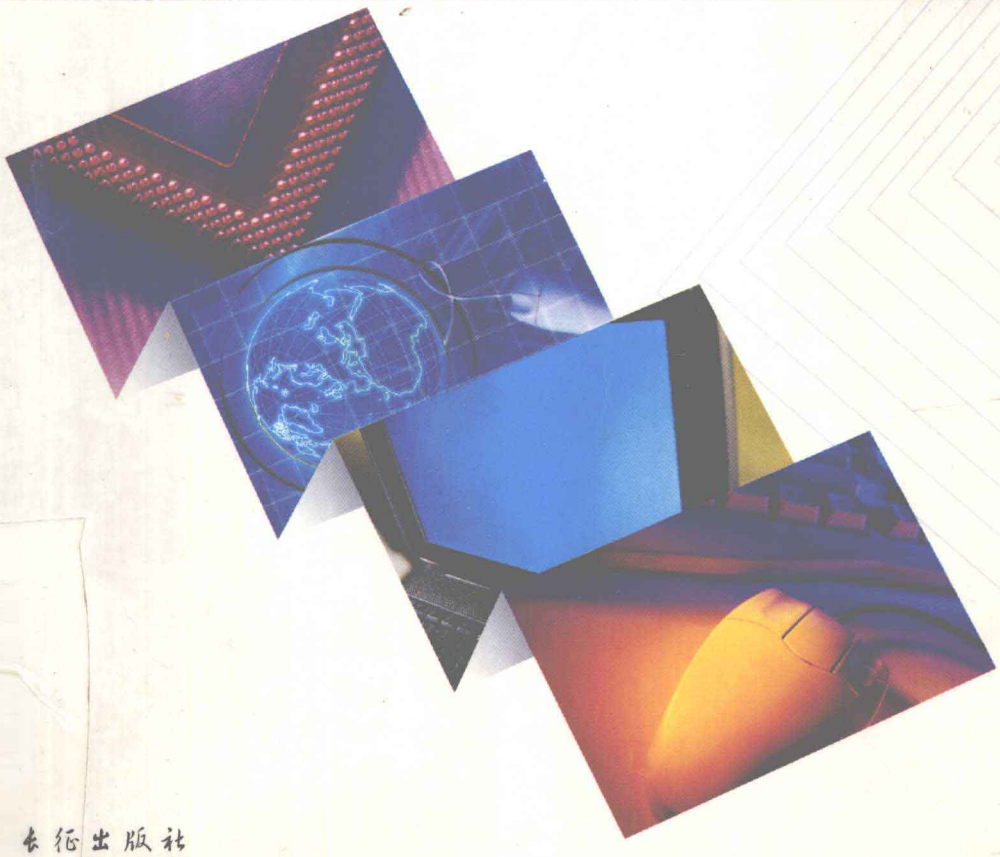
教育考试研究中心 组编

全国计算机等级考试

考点与题解

二级教程

公共基础知识



2006 等考王牌 最新版

长征出版社

【清华天骄】

等考系列之一

教育部考试中心指定教材配套辅导

全国计算机等级考试

考点与题解

二级教程

公共基础知识

教育考试研究中心组编

长征出版社

图书在版编目(CIP)数据

全国计算机等级考试考点与题解/教育考试研究中心组编.
—北京:长征出版社,2003
ISBN 7-80015-940-X

Ⅰ.全… Ⅱ.①李…②张… 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(2003)第 103117 号

全国计算机等级考试考点与题解

二级教程
——公共基础知识

教育考试研究中心组编

出版发行:长征出版社
地　　址:北京阜外大街 34 号
邮　　编:100832
电　　话:010-68586781
经　　销:全国新华书店
印　　刷:郑州文华印务有限公司
开　　本:850×1168 毫米　1/16
字　　数:7280 千字
印　　张:315 印张
版　　次:2005 年 10 月第 2 版
印　　次:2005 年 10 月第 1 次印刷
书　　号:ISBN 7-80015-940-X/G·281
总 定 价:473.00 元

随着计算机在各个领域愈来愈广泛地应用，信息科学正急剧地改变着人们的生产方式和生活方式。信息化社会必然对人们的素质及其知识结构提出新的要求，各行各业的人员不论年龄、专业和知识背景如何，都应掌握和应用计算机。国家教育部考试中心顺应社会发展的需要，于1994年推出“全国计算机等级考试”，其目的是以考促学，向社会推广普及计算机知识，为选拔人才提供统一、公正、客观和科学的标准。开考以来，截止2005年下半年，已顺利考过22次，千余个考点遍布全国30个省市。考生累计人数1500多万。累计获得证书人数500多万。根据我国计算机应用水平的实际情况。教育部考试中心于2004年对计算机等级考试大纲重新进行了修订，并正式颁布了新的考试大纲。

全国计算机等级考试的考核内容是根据应用计算机的不同要求，以应用能力为主，划分一、二、三、四个等级进行考核。基于此，我们严格依据教育部考试中心2004年颁布的全国计算机等级考试大纲和指定教材（《全国计算机等级考试二级教程公共基础知识》，高等教育出版社出版）编写了这本《二级教程公共基础知识考点与题解》。

本书内容共分四部分：第一部分是等级考试导引；第二部分是教材同步训练，内容包括考点速记、典型例题、强化练习、参考答案；第三部分是全真模拟试题；第四部分是附录。书中为广大考生提供了大量的题解分析和练习题目，选题内容、题型与考试一致，所选练习题带有典型性和启发性，对考点、重点、难点问题作了详尽的分析。

在编写过程中，充分考虑了等级考试的性质和考生学习及应试的特点，帮助考生在学习中把握重点，突破难点，掌握典型题例，以利在考试中发挥出水平并顺利通过考试。本书由刘文红主编，卢小林、刘丕顺副主编。

考生在学习过程中可访问全国计算机等级考试网站，网址是www.ncre.com.cn，可及时获得最新考试信息及本书补充内容。

衷心祝愿本书的出版对您的学习和应试有所帮助，也期望您通过全国等考网站对编写出版工作提出宝贵意见。

教育考试研究中心

第三部分 全真模拟试题

第四部分 附录

第一部分 等级考试导引

1 等级考试概述

全国计算机等级考试是由教育部考试中心主办，用于测试应试人员计算机应用知识与能力的等级水平考试。

全国计算机等级考试实行考试中心、各省承办机构两级管理的体制。

教育部考试中心聘请全国著名计算机专家组成“全国计算机等级考试委员会”，负责设计考试，审定考试大纲、试题及评分标准。教育部考试中心组织实施该项考试，组织编写考试大纲及相应的辅导材料、命制试卷，研制上机考试和考务管理软件，开展考试研究等。教育部考试中心在各省（自治区、直辖市）设立省级承办机构，各省（自治区、直辖市）承办机构根据教育部考试中心的规定设立考点，组织考试。

考试分笔试和上机两部分。考生的年龄、职业、学历不限，报考级别任选。成绩合格者由国家教育部考试中心颁发合格证书，笔试和上机成绩均在90分以上者为优秀，成绩优秀者在合格证书上加盖“优秀”字样。证书采用国际流行样式并有防伪标记。证书上印有考生本人的身份证号码，该证书全国通用。

全国计算机等级考试每年举行两次：第一次是每年4月的第一个星期日，考一、二、三级；第二次是每年9月的倒数第二个星期日，考一、二、三、四级。

各考试级别和基本要求如下：

一级考试

考试科目：在一级原来基础上，新增对金山 WPS Office 的考核，加上原有的一级和一级 B，

共三个科目。三个科目名称统一规范为:一级 MS Office、一级 B、一级 WPS Office。

考试形式:取消一级科目的纸笔考试,完全采取上机考试形式,各科上机考试时间均为 90 分钟。

考核内容:三个科目的考核内容包括微机基础知识和操作技能两部分。基础知识部分占全卷的 20%(20 分),操作技能部分占 80%(80 分)。各科目对基础知识的要求相同,以考查应知应会为主,题型为选择题。操作技能部分包括汉字录入、Windows 使用、文字排版、电子表格、演示文稿、因特网的简单应用。一级 B 因特殊行业和岗位需要,减少对演示文稿、因特网两部分的考核要求。

系统环境:一级科目中操作系统版本升级为 Windows 2000,MS Office 版本升级为 Office 2000,WPS Office 版本为 2003。

调整时间:一级 WPS Office 于 2004 年上半年试点,2004 年下半年在全国正式推广。调整后的一级 MS Office、一级 B 于 2004 年下半年在部分省试点,2005 年上半年在全国推广。

二级考试

新增科目:新增二级 Java、二级 Access、二级 C++ 三个科目。新增科目计划于 2004 年下半年试点,2005 年上半年在全国正式推广。

停考科目:逐步停考二级 Fortran、二级 Qbasic、二级 FoxBASE。二级 Fortran 于 2004 年上半年(第 19 考次)不再接收新考生报考,只接收补考。二级 Qbasic、二级 FoxBASE 于 2004 年下半年考试后停考,2005 年上半年不再接收新考生报考。

科目名称:对二级科目名称进行规范,根据应用性质和科目特点,将现有科目分成二级语言程序设计(C、C++、Java、Visual Basic、QBasic、Fortran)和二级数据库程序设计(FoxBASE、Visual FoxPro、Access)两类。

考核内容:二级仍然定位为程序员,考核内容主要包括基础知识和程序设计。所有科目对基础知识作统一要求,使用统一的基础知识大纲和教程。二级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、数据库基础知识共四个部分。二级基础知识在各科笔试中的比重为 30%(30 分),题型为 10 个选择题和 5 个填空题。二级上机考试中取消对 DOS 部分的考核(占 30 分)。

考试形式:二级所有科目的考试形式不变,仍包括笔试和上机考试两部分。

系统环境:二级各科目上机考试运行平台为:Access 2000、Jave JDK 1.4.0、Visual C++ 6.0、Visual Basic 6.0、Visual FoxPro 6.0、Turbo C 2.0。对逐步停考的三个科目,考试内容、考试形式、考试平台不作任何改动。

三级考试

三级划分为三级 PC 技术、三级信息管理技术、三级网络技术和三级数据库技术等 4 个科目,笔试时间均为 120 分钟,上机考试均为 60 分钟。

四级考试

四级考试考核计算机应用项目或应用系统的分析和设计的必备能力。笔试分选择题和论述题两种类型,其中选择题有中文和英文命题,英文占 1/3,论述题用中文命题。

四级考试的主要内容有计算机应用的基础知识,操作系统、软件工程和数据库系统的原理及应用知识,计算机系统结构、系统组成和性能评价的基础知识,计算机网络和通信的基础知

识，计算机应用系统安全和保密知识。要求应试者能综合应用上述知识，并能从事应用项目（系统）开发，即项目分析设计和组织实施的基本能力。四级考试笔试时间为180分钟，上机考试为60分钟。

当今世界，信息化是世界各国发展经济的共同选择。在实现国民经济信息化的过程中，必须解决全民普及计算机知识及应用技能的问题。随着计算机技术在我国各个领域的推广普及，计算机作为一种广泛应用的工具，其重要性日益受到社会的重视，越来越多的人开始学习计算机，操作和应用计算机成为人们必须掌握的一种基本技能。既掌握专业技术又具有计算机实际应用能力的人越来越受到重视和欢迎。许多单位部门已把掌握一定的计算机知识和应用技能作为干部录用、职称评定、上岗资格的重要依据之一。由于全国计算机等级考试具有较高的权威性、普遍性和正规性，这种考试得到了全社会的承认，这两年各高等学校在校学生中参加全国计算机等级考试的人越来越多，其证书对高校毕业生选择职业的成功率具有更重要的作用，成为我国规模最大、影响最大的计算机知识与能力的考试。

2 二级基础知识考试大纲

公共基础知识部分

基本要求：

1. 掌握算法的基本概念。
2. 掌握基本数据结构及其操作。
3. 掌握基本排序和查找算法。
4. 掌握逐步求精的结构化工程序设计方法。
5. 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。
6. 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

1. 算法的基本概念；算法复杂度的概念和意义（时间复杂度与空间复杂度）。

2. 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。

3. 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。

4. 栈和队列的定义；栈和队列的顺序存储结构及其基本运算。

5. 线性单链表、双向链表与循环链表的结构及其基本运算。

6. 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。

7. 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

1. 程序设计方法与风格。
2. 结构化程序设计。
3. 面向对象的程序设计方法，对象，方法，属性及继承与多态性。

（三）软件工程基础

1. 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。
2. 结构化分析方法，数据流图，数据字典，软件需求规格说明书。
3. 结构化设计方法，总体设计与详细设计。
4. 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。
5. 程序的调试，静态调试与动态调试。

（四）数据库设计基础

1. 数据库的基本概念：数据库，数据库管理系统，数据库系统。
2. 数据模型，实体联系模型 E－R 图，从 E－R 图导出关系数据模型。
3. 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。
4. 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。

第二部分 教材同步训练

第1章 数据结构与算法

考点速记

【考点一】 算法

一、算法的基本概念

所谓算法是指解题方案的准确而完整的描述。

1. 算法的基本特征

作为一个算法，一般应具有以下几个基本特征。

(1) 可行性(effectiveness)

针对实际问题设计的算法，人们总是希望能够得到满意的结果。但一个算法又总是在某个特定的计算工具上执行的，因此，算法在执行过程中往往要受到计算工具的限制，使执行结果产生偏差。

(2) 确定性(definiteness)

算法的确定性，是指算法中的每一个步骤都必须是有明确定义的，不允许有模棱两可的解释，也不允许有多义性。这一性质也反映了算法与数学公式的明显差别。在解决实际问题时，可能会出现这样的情况：针对某种特殊问题，数学公式是正确的，但按此数学公式设计的计算过

程可能会使计算机系统无所适从。这是因为根据数学公式设计的计算过程只考虑了正常使用的情况，而当出现异常情况时，此计算过程就不能适应了。

(3)有穷性(finiteness)

算法的有穷性，是指算法必须能在有限的时间内做完，即算法必须能在执行有限个步骤之后终止。数学中的无穷级数，在实际计算时只能取有限项，即计算无穷级数值的过程只能是有穷的。因此，一个数的无穷级数表示只是一个计算公式，而根据精度要求确定的计算过程才是有穷的算法。

算法的有穷性还应包括合理的执行时间的含义。因为，如果一个算法需要执行千万年，显然失去了实用价值。

(4)拥有足够的情报

一个算法是否有效，还取决于为算法所提供的情报是否足够。通常，算法中的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，这是算法执行的起点或是依据。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，算法本身也就无法执行或导致执行有错。一般来说，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，是一组严谨地定义运算顺序的规则，并且每一个规则都是有效的，且是明确的，此顺序将在有限的次数下终止。

2. 算法的基本要素

一个算法通常由两种基本要素组成：一是对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

每个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所组成的一组指令序列。因此，计算机算法就是计算机能处理的操作所组成的指令序列。

通常，计算机可以执行的基本操作是以指令的形式描述的。一个计算机系统能执行的所有指令的集合，称为该计算机系统的指令系统。计算机程序就是按解题要求从计算机指令系统中选择合适的指令所组成的指令序列。在一般的计算机系统中，基本的运算和操作有以下四类：

①算术运算：主要包括加、减、乘、除等运算。

②逻辑运算：主要包括“与”、“或”、“非”等运算。

③关系运算：主要包括“大于”、“小于”、“等于”、“不等于”等运算。

④数据传输：主要包括赋值、输入、输出等操作。

(2)算法的控制结构

一个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执行顺序有关。算法中各操作之间的执行顺序称为算法的控制结构。

算法的控制结构给出了算法的基本框架，它不仅决定了算法中各操作的执行顺序，而且也直接反映了算法的设计是否符合结构化原则。描述算法的工具通常有传统流程图、N－S 结构化流程图、算法描述语言等。一个算法一般都可以用顺序、选择、循环三种基本控制结构组合而成。

3. 算法设计基本方法

计算机解题的过程实际上是在实施某种算法，这种算法称为计算机算法。计算机算法不同于人工处理的方法。

(1)列举法

列举法的基本思想是,根据提出的问题,列举所有可能的情况,并用问题中给定运算速度快哪些是需要的,哪些是不需要的。因此,列举法常用于解决"是否存在"或"有多少种简单的问题型的问题,例如求解不定方程的问题。

列举法的特点是算法比较简单。但当列举的可能情况较多时,执行列举算法的工作量将会很大。因此,在用列举法设计算法时,使方案优化,尽量减少运算工作量,是应该重点注意的。通常,在设计列举算法时,只要对实际问题进行详细的分析,将与问题有关的知识条理化、完备化、系统化,从中找出规律;或对所有可能的情况进行分类,引出一些有用的信息,是可以大大减少列举量的。

列举原理是计算机应用领域中十分重要的原理。许多实际问题,若采用人工列举是不可想象的,但由于计算机的运算速度快,擅长重复操作,可以很方便地进行大量列举。列举算法虽然是一种比较笨拙而原始的方法,其运算量比较大,但在有些实际问题中(如寻找路径、查代、搜索等问题),局部使用列举法却是很有效的,因此,列举算法是计算机算法中的一个基础算法。

(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。显然,归纳法要比列举法更能反映问题的本质,并且可以解决列举量为无限的问题。但是,从一个实际问题中总结归纳出一般的关系,并不是一件容易的事情,尤其是要归纳出一个数学模型更为困难。从本质上讲,归纳就是通过观察一些简单而特殊的情况,最后总结出一般性的结论。

归纳是一种抽象,即从特殊现象中找出一般关系。但由于在归纳的过程中不可能对所有的情况进行列举,因此,最后由归纳得到的结论还只是一种猜测,还需要对这种猜测加以必要的证明。实际上,通过精心观察而得到的猜测得不到证实或最后证明猜测是错的,也是常有的事。

(3)递推

所谓递推,是指从已知的初始条件出发,逐次推出所要求的各中间结果和最后结果。其中初始条件或是问题本身已经给定,或是通过对问题的分析与化简而确定。递推本质上也属于归纳法,工程上许多递推关系式实际上是通过对实际问题的分析与归纳而得到的,因此,递推关系式往往是归纳的结果。

递推算法在数值计算中是极为常见的。但是,对于数值型的递推算法必须要注意数值计算的稳定性问题。

(4)递归

人们在解决一些复杂问题时,为了降低问题的复杂程度(如问题的规模等),一般总是将问题逐层分解,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,实际上并没有对问题进行求解,而只是当解决了最后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,这就是递归的基本思想。由此可以看出,递归的基础也是归纳。在工程实际中,有许多问题就是用递归来定义的,数学中的许多函数也是用递归来定义的。递归在可计算性理论和算法设计中占有很重要的地位。

递归分为直接递归与间接递归两种。如果一个算法 P 显式地调用自己则称为直接递归。如果算法 P 调用另一个算法 Q,而算法 Q 又调用算法 P,则称为间接递归调用。

递归是很重要的算法设计方法之一。实际上,递归过程能将一个复杂的问题归结为若干个较简单的问题,然后将这些较简单的问题再归结为更简单的问题,这个过程可以一直做下去,直

到最简单的问题为止。

有些实际问题,既可以归纳为递推算法,又可以归纳为递归算法。但递推与递归的实现方法是大不一样的。递推是从初始条件出发,逐次推出所需求的结果;而递归则是从算法本身到达递归边界的。通常,递归算法要比递推算法清晰易读,其结构比较简练。特别是在许多比较复杂的问题中,很难找到从初始条件推出所需结果的全过程,此时,设计递归算法要比递推算法容易得多。但递归算法的执行效率比较低。

(5)减半递推技术

实际问题的复杂程序往往与问题的规模有着密切的联系。因此,利用分治法解决这类实际问题是有效的。所谓分治法,就是对问题分而治之。工程上常用的分治法是减半递推技术。

所谓"减半",是指将问题的规模减半,而问题的性质不变;所谓"递推",是指重复"减半"的过程。

(6)回溯法

递推和递归算法本质上是对实际问题进行归纳的结果,而减半递推技术也是归纳法的一个分支。在工程上,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,并且也不能进行无限的列举。对于这类问题,一种有效的方法是"试"。通过对问题的分析,找出一个解决问题的线索,然后沿着这个线索逐步试探,对于每一步的试探,若试探成功,就得到问题的解,若试探失败,就逐步回退。换别的路线再进行试探。这种方法称为回溯法。回溯法在处理复杂数据结构方面有着广泛的应用。

二、算法复杂度

算法的复杂度主要包括时间复杂度和空间复杂度。

1. 算法的时间复杂度

所谓算法的时间复杂度,是指执行算法所需要的计算工作量。

为了能够比较客观地反映出一个算法的效率,在度量一个算法的工作量时,不仅应该与所使用的计算机、程序设计语言以及程序编制者无关,而且还应该与算法实现过程中的许多细节无关。为此,可以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。

算法所执行的基本运算次数还与问题的规模有关。例如,两个 20 阶矩阵相乘与两个 10 阶矩阵相乘,所需要的基本运算(即两个实数的乘法)次数显然是不同的,前者需要更多的运算次数。因此,在分析算法的工作量时,还必须对问题的规模进行度量。

综上所述,算法的工作量用算法所执行的基本运算次数来度量,而算法所执行的基本运算次数是问题规模的函数,即

$$算法的工作量 = f(n)$$

其中 n 是问题的规模。例如,两个 n 阶矩阵相乘所需要的基本运算(即两个实数的乘法)次数为 n^3,即计算工作量为 n^3,也就是时间复杂度为 n^3。

在具体分析一个算法的工作量时,还会存在这样的问题:对于一个固定的规模,算法所执行的基本运算次数还可能与特定的输入有关,而实际上又不可能将所有可能情况下算法所执行的基本运算次数都列举出来。例如,"在长度为 n 的一维数组中查找值为 x 的元素",若采用顺序搜索法,即从数组的第一个元素开始,逐个与被查值 x 进行比较。显然,如果第一个元素恰为 x,

则只需要比较1次。但如果x为数组的最后一个元素,或者x不在数组中,则需要比较n次才能得到结果。因此,在这个问题的算法中,其基本运算(即比较)的次数与具体的被查值x有关。

在同一个问题规模下,如果算法执行所需的基本运算次数取决于某一特定输入时,可以用以下两种方法来分析算法的工作量。

(1)平均性态(Average Behavior)

所谓平均性态分析,是指用各种特定输入下的基本运算次数的加权平均值来度量算法的工作量。

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

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

其中D_n表示当规模为n时,算法执行时所有可能输入的集合。这个式子中的t(x)可以通过分析算法来加以确定;而p(x)必须由经验或用算法中有关的一些特定信息来确定,通常是不能解析地加以计算的。如果确定p(x)比较困难,则会给平均性态的分析带来困难。

(2)最坏情况复杂性(Worst - Case Complexity)

所谓最坏情况分析,是指在规模为n时,算法所执行的基本运算的最大次数。它定义为

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

显然,W(n)的计算要比A(n)的计算方便得多。由于W(n)实际上是给出了算法工作量的一个上界,因此,它比A(n)更具有实用价值。

2. 算法的空间复杂度

一个算法的空间复杂度,一般是指执行这个算法所需要的内存空间。

一个算法所占用的存储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。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(例如,在链式结构中,除了要存储数据本身外,还需要存储链接信息)。如果额外空间量相对于问题规模来说是常数,则称该算法是原地(in place)工作的。在许多实际问题中,为了减少算法所占的存储空间,通常采用压缩存储技术,以便尽量减少不必要的额外空间。

【考点二】 数据结构的基本概念

利用计算机进行数据处理是计算机应用的一个重要领域。在进行数据处理时,实际需要处理的数据元素一般有很多,而这些大量的数据元素都需要存放在计算机中,因此,大量的数据元素在计算机中如何组织,以便提高数据处理的效率,并且节省计算机的存储空间,这是进行数据处理的关键问题。

数据结构作为计算机的一门学科,主要研究和讨论以下三个方面的问题:

①数据集合中各数据元素之间所固有的逻辑关系,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。所谓提高数据处理的效率,主要包括两个方面:一是提高数据处理的速度,二是尽量节省在数据处理过程中所占用的计算机存储空间。

一、什么是数据结构

计算机已被广泛用于数据处理。实际问题中的各数据元素之间总是相互关联的。所谓数据处理,是指对数据集合中的各元素以各种方式进行运算,包括插入、删除、查找、更改等运算,也包括对数据元素进行分析。在数据处理领域中,建立数学模型有时并不十分重要,事实上,许多实际问题是无法表示成数学模型的。人们最感兴趣的是知道数据集合中各数据元素之间存在什么关系,应如何组织它们,即如何表示所需要处理的数据元素。

简单地说,数据结构是指相互有关联的数据元素的集合。例如,向量和矩阵就是数据结构,在这两个数据结构中,数据元素之间有着位置上的关系。又如,图书馆中的图书卡片目录,则是一个较为复杂的数据结构,对于列在各卡片上的各种书之间,可能在主题、作者等问题上相互关联,甚至一本书本身也有不同的相关成分。

数据元素具有广泛的含义。一般来说,现实世界中客观存在的一切个体都可以是数据元素。

总之,在数据处理领域中,每一个需要处理的对象都可以抽象成数据元素。数据元素一般简称为元素。

在实际应用中,被处理的数据元素一般有很多,而且,作为某种处理,其中的数据元素一般具有某种共同特征。例如,{春,夏,秋,冬}这四个数据元素有一个共同特征,即它们都是季节名,分别表示了一年中的四个季节,从而这四个数据元素构成了季节名的集合。又如,{父亲,儿子,女儿}这三个数据元素也有一个共同特征,即它们都是家庭的成员名,从而构成了家庭成员名的集合。一般来说,人们不会同时处理特征完全不同且互相之间没有任何关系的各类数据元素,对于具有不同特征的数据元素总是分别进行处理。

一般情况下,在具有相同特征的数据元素集合中,各个数据元素之间存在有某种关系(即联系),这种关系反映了该集合中的数据元素所固有的一种结构。在数据处理领域中,通常把数据元素之间这种固有的关系简单地用前后件关系(或直接前驱与直接后继关系)来描述。

前后件关系是数据元素之间的一个基本关系,但前后件关系所表示的实际意义随具体对象的不同而不同。一般来说,数据元素之间的任何关系都可以用前后件关系来描述。

1. 数据的逻辑结构

前面提到,数据结构是指反映数据元素之间关系的数据元素集合的表示。更通俗地说,数据结构是指带有结构的数据元素的集合。因此,所谓结构实际上就是指数据元素之间的前后件关系。

由上所述,一个数据结构应包含以下两方面的信息:

①表示数据元素的信息;

②表示各数据元素之间的前后件关系。

在以上所述的数据结构中,其中数据元素之间的前后件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,上面所述的数据结构实际上是数据的逻辑结构。

所谓数据的逻辑结构,是指反映数据元素之间逻辑关系的数据结构。

由前面的叙述可以知道,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,它反映了 D 中各数据元素之间的前后件关系,通常记为 R。即一个数据结构可以表示成

$$B = (D, R)$$

其中 B 表示数据结构。为了反映 D 中各数据元素之间的前后件关系,一般用二元组来表示。

例如,假设 a 与 b 是 D 中的两个数据,则二元组(a,b)表示 a 是 b 的前件,b 是 a 的后件。这样,在 D 中的每两个元素之间的关系都可以用这种二元组来表示。

对于一些复杂的数据结构来说,它的数据元素可以是另一种数据结构。

例如,m × n 的矩阵

$$A=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{12} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

是一个数据结构。在这个数据结构中,矩阵的每一行

$$A_i=(a_{i1},a_{i2},\cdots,a_{in}),i=1,2,\cdots,m$$

可以看成是它的一个数据元素。即这个数据结构的数据元素的集合为

$$D=\{A_1,A_2,\cdots,A_m\}$$

D 上的一个关系为

$$R=\{(A_1,A_2),(A_2,A_3),\cdots,(A_i,A_{i+1}),\cdots,(A_{m-1},A_m)\}$$

显然,数据结构 A 中的每一个数据元素 $A_i(i=1,2,\cdots,m)$ 又是另一个数据结构,即数据元素的集合为

$$D_i=(a_{i1},a_{i2},\cdots,a_{in})$$

D_i 上的一个关系为

$$R_i=\{(a_{i1},a_{i2}),(a_{i2},a_{i3}),\cdots,(a_{ij},a_{i,j+1}),\cdots,(a_{i,n-1},a_{in})\}$$

2. *数据的存储结构*

数据处理是计算机应用的一个重要领域,在实际进行数据处理时,被处理的各数据元素总是被存放在计算机的存储空间中,并且,各数据元素在计算机存储空间中的位置关系与它们的逻辑关系不一定是相同的,而且一般也不可能相同。例如,一年四个季节的数据结构中,“春”是“夏”的前件,“夏”是“春”的后件,但在对它们进行处理时,在计算机存储空间中,“春”这个数据元素的信息不一定被存储在“夏”这个数据元素信息的前面,而可能在后面,也可能不是紧邻在前面,而是中间被其他的信息所隔开。又如,在家庭成员的数据结构中,“儿子”和“女儿”都是“父亲”的后件,但在计算机存储空间中,根本不可能将“儿子”和“女儿”这两个数据元素的信息都紧邻存放在“父亲”这个数据元素信息的后面,即在存储空间中与“父亲”紧邻的只可能是其中的一个。由此可以看出,一个数据结构中的各数据元素在计算机存储空间中的位置关系与逻辑关系是有可能不同的。

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(也称数据的物理结构)。

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逻辑关系(即前后件关系),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的前后件关系的信息。

一般来说,一种数据的逻辑结构根据需要可以表示成多种存储结构,常用的存储结构有顺序、链接、索引等存储结构。而采用不同的存储结构,其数据处理的效率是不同的。因此,在进行数据处理时,选择合适的存储结果是很重要的。

二、数据结构的图形表示