HZ BOOKS
华章科技

全国计算机等级考试命题研究小组最新成果
2014年最新无纸化考试专用指导
赠送价值100元的希赛等考重点培训视频

2014年
全国计算机等级考试
3年真题精解与过关全真训练题

二级 公共基础知识

希赛教育等考学院 卢艳芝 主编

紧扣考试大纲，提炼必考点，解析重点难点
连续3年考题训练解析
全真模拟试题与解析
赠送权威专家与辅导名师考试培训教程
在线测试，考前训练，加深记忆

机械工业出版社
China Machine Press

2014年
全国计算机等级考试
3年真题精解与过关全真训练题

二级公共基础知识

希赛教育等考学院 卢艳芝 主编

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

2014年全国计算机等级考试3年真题精解与过关全真训练题：二级公共基础知识 / 卢艳芝主编. —北京：机械工业出版社，2013.12

ISBN 978-7-111-44828-0

Ⅰ.2… Ⅱ.卢… Ⅲ.电子计算机－水平考试－题解 Ⅳ.TP3-44

中国版本图书馆CIP数据核字(2013)第275083号

本书由希赛教育等考学院组织编写，是全国计算机等级考试二级辅导和培训的指定教程。书中内容紧扣全国计算机等级考试2014年考试大纲，通过对历年试题进行科学分析、研究、总结、提炼而成。书中内容全面实用，涵盖了考试大纲规定的所有知识点，对考试大纲规定的内容有重点地进行了细化和深化。阅读本书，就相当于阅读了一本详细的、带有知识注释的考试大纲。准备考试的人员可通过阅读本书掌握考试大纲规定的知识，掌握考试重点和难点，熟悉内容的分布。

本书适合参加全国计算机等级考试的人员及广大计算机爱好者阅读。

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：朱秀英

北京瑞德印刷有限公司印刷

2014年1月第1版第1次印刷

185mm×260mm·11.75印张

标准书号：ISBN 978-7-111-44828-0

ISBN 978-7-89405-176-9（光盘）

定 价：39.00元（附光盘）

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88378991 88361066 投稿热线：（010）88379604

购书热线：（010）68326294 88379649 68995259 读者信箱：hzjsj@hzbook.com

前　言

全国计算机等级考试（NCRE）由教育部考试中心主办，面向社会，用于考查非计算机专业人员的计算机应用知识与能力。考试客观、公正，得到了社会的广泛认可。

本书根据全国计算机等级考试 2014 年考试大纲编写而成，在组织和写作上倾注了作者的许多精力和心血，相信能够提高考生通过率，为有效地完成“考试过关”提供帮助。考生可通过阅读本书，迅速掌握考试所涉及的知识点，全面梳理和系统学习考试大纲中的内容。

作者权威，阵容强大

希赛教育（www.educity.cn）专业从事人才培养、教育产品开发、教育图书出版，在职业教育方面具有极高的权威性。特别是在在线教育方面名列前茅，希赛教育的远程教育模式得到了国家教育部门的认可和推广。

希赛教育等考学院是国内知名的进行计算机等级考试在线教育的大型教育机构，在该领域取得了很好的效果。我们组织大纲制订者和阅卷组成员编写了考试辅导教材近 50 本，内容涵盖了计算机等级考试的主要级别。组织权威专家和辅导名师录制了考试培训视频教程，对历年考试进行了跟踪研究和比较研究，编写了权威的全真模拟试题。希赛教育的计算机等级考试培训采取统一教材、统一视频、统一认证教师的形式，采取线下培训与线上辅导相结合的方式，确保学员在通过考试的前提下能真正学到有用的知识。

本书由希赛教育等考学院卢艳芝主编，参加编写工作的还有胡钊源、张友生、桂阳、王勇、何玉云、张丹、谢顺、彭雪阳、余华山、刘洋波。参加编写的人员来自大学教学一线和企业研发团队，具有丰富的教学和辅导经验，对等级考试有深入的研究，具有极强的应试技巧、理论知识、实践经验和责任心。

在线测试，心中有数

希赛网在线测试平台（platform.educity.cn/oletest/）为考生准备了在线测试，其中有数十套全真模拟试题和考前密卷，考生可选择任何一套进行测试。测试完毕，系统自动判卷，立即给出分数。

对于考生做错的地方，系统会自动记忆，待考生第二次参加测试时，可选择“试题复习”。这样，系统就会自动把考生原来做错的试题显示出来，供考生重新测试，以加强记忆。

因此，读者可利用希赛网在线测试平台的在线测试系统检查自己的实际水平，加强考前训练，做到心中有数，考试不慌。

随书光盘，物超所值

本书附带的光盘内容为希赛网等考学院（www.educity.cn/ncre/）培训视频教程的节选，本视频教程对考试所有的难点和重点知识进行了精深讲解，可以保证既不漏掉考试必需的知识点，又不加重考生备考负担，使考生轻松、愉快地掌握知识点并领悟考试的真谛。更完整的培训视频教程，请访问希赛教育视频教学平台（platform.educity.cn/v/）。

诸多帮助，诚挚致谢

在本书出版之际，要特别感谢教育部考试中心计算机等级考试办公室的命题专家们，编者在本书中引用了部分考试原题，使本书能够尽量方便读者阅读。在本书的编写过程中，参考了许多相关的文献和书籍，编者在此对这些参考文献的作者表示感谢。

感谢机械工业出版社华章公司的李华君老师，他在本书的策划、选题的申报、写作大纲的确定，以及编辑、出版等方面，付出了辛勤的劳动和智慧，给予了我们很多的支持和帮助。

感谢参加希赛教育计算机等级考试辅导和培训的学员，正是他们的想法汇成了本书，他们的意见使本书更加贴近读者。

由于编者水平有限，且本书涉及的内容很广，书中难免存在错漏和不妥之处，诚恳地期望各位专家和读者不吝指正和帮助，对此，我们将十分感激。

希赛教育等考学院

目 录

第1章 数据结构与算法

根据考试大纲，本章要求考生掌握以下知识点：

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、二叉树的遍历（前序、中序和后序）。
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）。

1.1 考点精讲

计算机已经被广泛用于数据处理。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操作，包括插入、删除、查找、更改等，也包括对数据元素进行分析。在进行数据处理时，实际需要处理的数据元素一般很多，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。

算法是一个十分古老的研究课题，然而计算机的出现为这个课题注入了青春和活力，使算法的设计和分析成为计算机学科中最为活跃的研究热点之一。针对实际问题，例如要编制一个高效率的处理程序，那就需要解决如何合理地组织数据，建立合适的数据结构，设计好的算法来提高程序执行的效率这样的问题。

1.1.1 算法

算法（algorithm）是对解题方案的准确而完整的描述。但它不等于程序，也不等于计算方法。通常，程序的编制不可能优于算法的设计。算法是指令的有限序列，也就是说，能够对一定规范的输入，在有限时间内获得所要求的输出。当然，程序也可以作为算法的一种描述，但程序通常还需考虑很多与方法和分析无关的细节问题，这是因为在编写程序时要受到计算机系统运行环境的限制。

1. 算法的基本特征

一个算法，一般应具有以下几个基本特征：

1）可行性：算法的可行性，是指算法中指定的操作都可以通过基本运算执行有限的次数后实现。针对实际问题设计算法时必须考虑它的可行性，因为人们总是希望能够达到满意的结果。

2）确定性：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。

3）有穷性：算法的有穷性，是指一个算法必须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。另外，算法的有穷性还应包括合理的执行时间，如果一个算法需要执行很长时间甚至上千年才能终止，就失去了实用价值。

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，一般来说，当算法拥有足够的情报时，此算法才是有效的，否则，可能无效。

2．算法的基本要素

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。

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4类：

- 算术运算：主要包括加、减、乘、除等运算。
- 逻辑运算：主要包括“与”、“或”、“非”等运算。
- 关系运算：主要包括“大于”、“小于”、“等于”、“不等于”等运算。
- 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3种基本控制结构组合而成。

3．算法设计的基本方法

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，下面是工程上常用的几种算法设计方法，在实际应用时，各种方法之间往往存在着一定的联系。

（1）列举法

列举法的基本思想是，针对待解决的问题，列举所有可能的情况，并用问题中给定的条件来检验哪些是必需的，哪些是不需要的。因此，列举法常用于解决“是否存在”或“有多少种可能”等类型的问题。例如，我国古代的趣味数学题：“百钱买百鸡”、“鸡兔同笼”等求解不定方程的问题，均可采用列举法解决。

其特点是原理比较简单，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，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，就可以大大减少列举量。

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、查找、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，最后归纳出一般的关系。显然，归纳法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。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规律。但由于在归纳法中不可能对所有的情况进行列举，因此，该方法得到的结论只是一种猜测，还需要进行证明。

（3）递推法

递推，即是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，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得到的。因此，递推关系式通常是归纳的结果。

递推算法在数值计算中是极为常见的。例如，裴波那契数列是采用递推的方法解决问题的。但是，对于数值型的递推算法必须注意数值计算的稳定性问题。

（4）递归

在解决一些复杂问题或问题的规模比较大时，为了降低问题的复杂程序，通常将问题逐层分解，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，而只是在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。

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。递归过程能将一个复杂的问题归纳为若干较简单的问题，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，直到归结出最简单的问题为止。由此可以看出，递归的基础也是归纳。在实际工程中，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很重要的地位。

（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”，即将问题的规模减半，而问题的性质不变；所谓“递推”，是指重复“减半”的过程。例如，一元二次方程的求解，求解过程见下例。

例如，设方程 $f(x)=0$ 在区间$[a, b]$上有实根，且 $f(a)$ 与 $f(b)$ 异号。利用二分法求该方程在区间$[a, b]$上的一个实根。

用二分法求方程实根的减半递推过程如下：

首先取给定区间的中点 $c=(a+b)/2$。

然后判断 $f(c)$ 是否为 0。若 $f(c)=0$，则说明 c 即为所求的根，求解过程结束；如果 $f(c)\neq 0$，则根据以下原则将原区间减半：

若 $f(a)f(c)<0$，则取原区间的前半部分；

若 $f(b)f(c)<0$，则取原区间的后半部分。

最后判断减半后的区间长度是否已经很小：

若 $|a-b|<\varepsilon$，则过程结束，取 $(a+b)/2$ 为根的近似值；

若 $|a-b|\geq\varepsilon$，同重复上述的减半过程。

（6）回溯法

前面讨论的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，有些实际的问题很难归纳出一组简单的递推公式或直观的求解步骤，也不能无限地列举。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，换其他路线进行试探。这种方法即称为回溯法。

回溯法在处理复杂数据结构方面有着广泛的应用，如人工智能中的机器人下棋。

4．算法复杂度

算法的复杂度主要分为时间复杂度和空间复杂度。

（1）算法的时间复杂度

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，一个算法的工作量不仅与所使用的计算机、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，而算法所执行的基本运算次数是问题规模的函数，即

$$\text{算法的工作量}=f(n)$$

其中 n 是问题规模。例如，两个 n 阶矩阵相乘所需要的基本运算次数为 n^3，即计算量为 n^3，也就是时间复杂度为 n^3。

另外，在同一问题规模下，若算法执行所需的基本运算次数取决于某一特定的输入数据，则可以用平均性态分析和最坏情况分析两种方法来分析算法的工作量。顾名思义，平均性态分析即输入所有可能的平均值，相应的时间复杂度为算法的平均时间复杂度；最坏情况分析则是以最坏的情况估算算法执行时间的一个上界。一般情况下，后者更为常用。

（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，除了需要存储数据本身之外，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，则称该算法是原地工作的。

在许多实际问题中，为了减少算法的内存空间，一般采用压缩存储技术，以便尽量减少不必要的额外空间。

1.1.2 数据结构

数据结构（data structure）是指反映数据元素之间关系的数据元素集合的表示，其作为计算机的一门学科，主要研究和讨论以下3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）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。

讨论以上问题的主要目的是提高数据处理的效率。提高数据处理的效率主要包括两个方面：一是提高数据速度，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。

1．什么是数据结构

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。

数据的组织方式不同，通常对它进行处理时的效率也不同。例如，在对两个存放了相同元素的表进行查找时，如果一个表中的所有数据元素是没有规律的，而另外一个表中的元素是经过排序的，则对前者进行查找时，只能从第一个元素开始，逐个将表中的元素与被查数进行比较，直到表中的某个元素与被查数相等（即查找成功）或者表中所有元素与被查数都进行了比较且都不相等（即查找失败）为止。而对于后者，由于有序表中的元素是从小到大进行排列的，在查找时可以利用这个特点，使比较次数大大减少。可见数据元素在表中的排列顺序对查找效率是有很大影响的。

要理解数据结构，还需要理解以下基本概念：

- 数据（data）是对客观事物的符号表示，是信息的载体，它能够被计算机识别、存储和加工处理。在计算机学科中，数据就是计算机加工处理的对象，它可以是数值数据，也可以是非数值数据。
- 数据元素（data element）是数据的基本单位，在计算机程序中通常作为一个整体进行考虑和处理。在不同的条件下，数据元素又可称为元素、结点、顶点、记录等。例如，描述一年四季的季节名：春、夏、秋、冬可以作为季节的数据元素。
- 数据对象（data object）是具有相同性质的数据元素的集合。在某个具体问题中，数据元素都具有相同的性质（元素值不一定相等），属于同一数据对象，数据元素是数据元素类的一个实例。
- 数据结构（data structure）是指相互之间存在一种或多种特定关系的数据元素的集合。

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某种关系（即连续），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，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。例如，在考虑一年四个季节的顺序关系时，“春”是“夏”的前件（即直接前驱，下同），而“夏”是“春”的后件（即直接后继，下同）。同样，“夏”是“秋”的前件，“秋”是“夏”的后件；“秋”是“冬”的前件，“冬”是“秋”的后件。在考虑家庭成员间的辈分关系时，“父亲”是“儿子”和“女儿”的前件，而“儿子”与“女儿”都是“父亲”的后件。

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。

2．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。通俗地说，数据结构是指带有结构的数据元素的集合。结构实际上是指数据元素之间的前后件关系。

一个数据结构应包含以下两方面信息：

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。

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：一是用 D 表示数据元素的集合，二是用 R 表示数据元素之间的前后件关系。即一个数据结构可以表示为 $B=(D, R)$，其中 B 表示数据结构。这就是一个二元关系的表示方式。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。

例如，一年四季的数据结构可以表示成

$B=\{D，R\}$

$D=\{$春，夏，秋，冬$\}$

$R=\{$(春，夏)，(夏，秋)，(秋，冬)$\}$

例如，家庭成员数据结构可以表示成

$B=(D，R)$

$D=\{$父亲，儿子，女儿$\}$

$R=\{$(父亲，儿子)，(父亲，女儿)$\}$

例如，n 维向量 $X=(x_1，x_2，\cdots，x_n)$ 也是一种数据结构，即 $X=\{D，R\}$，

其中数据元素集合为

$D=\{x_1，x_2，\cdots，x_n\}$

关系为

$R=\{(x_1，x_2)，(x_2，x_3)，\cdots，(x_{n-1}，x_n)\}$

3．数据的存储结构

在进行数据处理时，计算机所处理的数据总是存放在计算机的存储空间中，一个数据结构中的各元素在计算机存储空间中的位置关系与逻辑关系有可能是不同的。

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（也称为数据的物理结构）。

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，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，选择合适的存储结构是很重要的。

4．数据结构的图形表示

数据结构除了用二元关系表示外，还可以直观地用图形表示。

在数据结构的图形表示中，对于数据集合 D 中的每一个数据元素用中间标有元素值的方框表示，一般称为数据结点，并简称为结点；为了进一步表示各数据元素之间的前后件关系，对于关系 R 中的每一个二元组，用一条有向线段从前件结点指向后件结点。

例如，一年四季的数据结构可以用如图 1-1 所示的图形来表示。

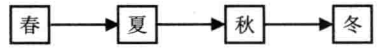


图 1-1　一年四季数据结构的图形表示

又例如，反映家庭成员间辈分关系的数据结构可以用如图 1-2 所示的图形表示。

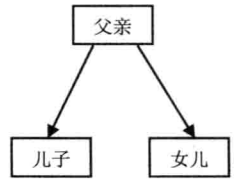


图 1-2　家庭成员间辈分关系数据结构的图形表示

显然，用图形方式表示一个数据结构是很方便的，并且也比较直观。

在数据结构中，没有前件的结点称为根结点；没有后件的结点称为终端结点（也称为叶子结点）。例如，在图 1-1 所示的数据结构中，元素“春”所在的结点（简称为结点“春”，下同）为根结点，结点“冬”为终端结点；在图 1-2 所示的数据结构中，结点“父亲”为根结点，结点“儿子”与结点“女儿”均为终端结点。数据结构中除了根结点与终端结点外的其他结点一般称为内部结点。

通常，一个数据结构中的结点可能是动态变化的。根据需要或在处理过程中，可以在一个数据结构中增加一个新结点（称为插入运算），也可以删除数据结构中的某个结点（称为删除运算）。插入与删除是对数据结构的两种基本运算。除此之外，对数据结构的运算还有查找、分类、合并、分解、复制和修改等。 在对数据结构的处理过程中，不仅数据结构中结点（即数据元素）的个数在动态变化，而且，各数据元素之间的关系也有可能在动态变化。例如，一个无序表可以通过排序处理而变成有序表；一个数据结构中的根结点被删除后，它的某一个后件可能变成了根结点；在一个数据结构中的终端结点后插入一个新的结点后，则原来的那个终端结点就不再是终端结点而成为内部结点了。有关数据的基本运算将在后面讲到具体数据结构时再介绍。

5．线性结构与非线性结构

如果一个数据结构中一个数据元素都没有，则称该数据结构为空的数据结构。一个空的数据结构中插入一个新的元素后，该数据结构就变为非空数据结构；在只有一个数据元素的数据结构中，将该元素删除后，该数据结构就变为空的数据结构。

根据数据结构中各数据元素之间前后件关系的复杂程度，一般将数据结构分为两大类：线性结构与非线性结构。

如果一个非空的数据结构满足如下两个条件：

1）有且只有一个根结点。

2）每一个结点最多有一个前件，也最多有一个后件。

则称该数据结构为线性结构。线性结构又称为线性表。

由此可以看出，在线性结构中，各数据元素之间的前后件关系是很简单的。例如，图 1-1 中一年四季这个数据结构就属于线性结构。

特别需要说明的是，在一个线性结构中插入或删除任何一个结点后还应是线性结构。根

据这一点，如果一个数据结构满足上述两个条件，但在此数据结构中插入或删除任何一个就不满足这两个条件时，则该数据结构不能称为线性结构。例如，图 1-3 的数据结构显然是满足上述两个条件的，但它不属于线性结构，因为在这个数据结构中删除结点 A 后，就不满足上述的条件 1）了。

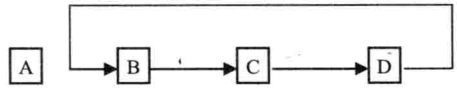


图 1-3　不是线性结构的数据结构特例

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 1-2 中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件关系要比线性结构复杂，因此，对非线性结构的存储与处理比线性结构要复杂得多。

线性结构与非线性结构都可以是空的数据结构。一个空的数据结构究竟是线性结构，还是非线性结构，要根据具体情况来确定。如果对该数据结构的运算是按线性结构的规则来处理的，则属于线性结构，否则属于非线性结构。

1.1.3　线性表

线性表的特点是数据元素之间的关系是线性的，数据元素可以看成是排列在一条线或一个环上。线性表（linear list）是最简单、最常用的一种数据结构。

1．线性表的基本概念

线性表由一组数据元素构成。数据元素的含义很广泛，在不同的情况下，它可以有不同的含义。例如，一个 n 维向量（x_1，x_2，…，x_n）是一个长度为 n 的线性表，其中的每一个分量就是一个数据元素。又例如，英文小写字母表（a，b，c，…，z）是一个长度为 26 的线性表，其中的每一个小写字母就是一个数据元素。再如，一年中的 4 个季节（春、夏、秋、冬）是一个长度为 4 的线性表，其中的每一个季节名就是一个数据元素。

数据元素可以是简单项（如上述例子中的数字、字母、季节名等）。在稍微复杂的线性表中，一个数据元素还可以由若干数据项组成。例如，某班的学生情况登记表是一个复杂的线性表，表中每一个学生的情况就组成了线性表中的每一个元素，每一个数据元素包括学号、姓名、性别、年龄、班级 5 个数据项，如表 1-1 所示。在这种复杂的线性表中，由若干数据项组成的数据元素称为记录，而由多个记录构成的线性表又称为文件。因此，上述学生情况登记表就是一个文件，其中每一个学生的情况就是一个记录。

表 1-1　学生情况登记表

学　号	姓　名	性　别	年　龄	班　级
2010011810205	于春梅	女	18	2010 级电信 016 班
2010011810260	何仕鹏	男	18	2010 级电信 017 班
2010011810284	王　爽	女	18	2010 级通信 011 班
2010011810360	王亚武	男	18	2010 级通信 012 班
…	…	…	…	…

综上所述，线性表是一个线性结构，它是一个含有 n（$n \geqslant 0$）个结点的有限序列。对于

其中的结点，有且仅有一个根结点，没有前件但有一个后件；有且仅有一个终端结点，没有后件但有一个前件。其他的结点都有且仅有一个前件和一个后件。一般地，一个线性表可以表示成一个线性序列：a_1，a_2，…，a_n，其中 a_1 是根结点，a_n 是终端结点。

线性结构的基本特征如下：

1）有且仅有一个根结点，无前件。

2）有且仅有一个终端结点，无后件。

3）除终端结点外，其他所有结点均有且仅有一个后件。

4）除根结点外，其他所有结点均有且仅有一个前件。

由 n（$n\geqslant 0$）个数据元素（结点）a_1，a_2，…，a_n 组成的有限序列称为线性表，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，常常将非空的线性表（$n>0$）记作：$(a_1,a_2,\cdots,a_n)$。

2．线性表的顺序存储

在计算机中存放线性表，一种最简单的方法是顺序存储，也称为顺序分配。

线性表的顺序存储指的是用一组地址连续的存储单元依次存储线性表的数据元素。

线性表的顺序存储结构具备如下两个基本特征：

1）线性表中的所有元素所占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。

由此可以看出，在线性表的顺序存储结构中，其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。如果线性表中各数据元素所占的存储空间（字节数）相等，则要在线性表中查找某一个元素是很方便的。

假设线性表中的第一个数据元素的存储地址（指第一字节的地址，即首地址）为 ADR（a_1），每个元素需要占用 k 个存储单元，则线性表中第 i 个元素 a_i 在计算机存储空间中的存储地址为

$$\mathrm{ADR}(a_i)=\mathrm{ADR}(a_1)+(i-1)\times k$$

即在顺序存储结构中，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。一般来说，长度为 n 的线性表 $(a_1,a_2,\cdots,a_n)$，设每个数据元素在内存中占 4 字节，起始元素的存储地址为 1010，则该线性表在计算机中的顺序存储结构如图 1-4 所示。

存储地址	数据元素
1010	a_1
1013	a_2
…	…
1010+(i−1) ×4	a_i
…	…
1010+(n−1) ×4	a_n

图 1-4　线性表的顺序存储结构示意图

在程序设计语言中，通常定义一个一维数组来表示线性表的顺序存储空间。在用一维数组存放线性表时，该一维数组的长度通常要定义得比线性表的实际长度大一些，以便对线性表进行各种运算，特别是插入运算。在一般情况下，如果线性表的长度在处理过程中是动态

变化的，则在开辟线性表的存储空间时，要考虑到线性表在动态变化过程中可能达到的最大长度。

在线性表的顺序存储结构下，可以对线性表做以下运算：

1）在线性表的指定位置处加入一个新的元素（即线性表的插入）。

2）在线性表中删除指定的元素（即线性表的删除）。

3）在线性表中查找某个（或某些）特定的元素（即线性表的查找）。

4）对线性表中的元素进行排序（即线性表的排序）。

5）将一个线性表分解成多个线性表（即线性表的分解）。

6）将多个线性表合并成一个线性表（即线性表的合并）。

7）复制一个线性表（即线性表的复制）。

8）逆转一个线性表（即线性表的逆转）等。

线性表的基本运算是插入运算和删除运算，下面两小节主要讨论这两种运算。

3．线性表的插入运算

线性表的插入运算是指在表的第 i（$1 \leqslant i \leqslant n+1$）个位置上，插入一个新结点 x，使长度为 n 的线性表：

$$(a_1, \cdots, a_{i-1}, a_i, \cdots, a_n)$$

变成长度为 $n+1$ 的线性表：$(a_1, \cdots, a_{i-1}, x, a_i, \cdots, a_n)$

首先举一个例子来说明如何在顺序存储结构的线性表中插入一个新元素。

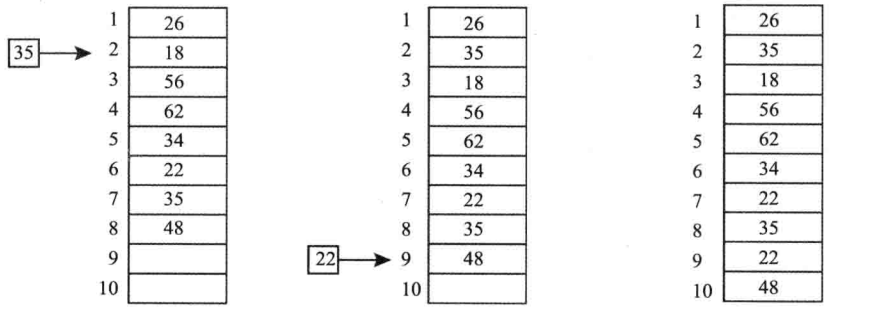
例如，图 1-5a 是一个长度为 8 的线性表，顺序存储在长度为 10 的存储空间中。现在要求在第 2 个元素（即 18）之前插入一个新元素 35。其插入过程如下：

首先从最后一个元素开始直到第 2 个元素，将其中的每一个元素均依次往后移动一个位置，然后将新元素 35 插入第 2 个位置。

插入一个新元素后，线性表的长度变成了 9，如图 1-5b 所示。

如果要再在线性表的第 9 个元素之前插入一个新元素 22，则采用类似的方法：将第 9 个元素往后移动一个位置，然后将新元素插入第 9 个位置。插入后，线性表的长度变成了 10，如图 1-5c 所示。

现在，为线性表开辟的存储空间已经满了，不能再插入新的元素了。如果再要插入，则会造成称为“上溢”的错误。



a）长度为 8 的线性表　　b）插入元素 87 后的线性表　　c）插入元素 22 后的线性表

图 1-5　线性表在顺序存储结构下的插入