National Computer Rank Examination

全国计算机等级考试专用辅导丛书

# 全国计算机等级考试专用辅导教程

# 二级公共基础知识

希赛教育等考学院 刘欣苗 主编

2013版

◆紧扣最新考试大纲，透彻精讲大纲规定考点
◆突出重点与难点，深入分析例题，讲练结合
◆提供最新真题解析，摸清考试规律，掌握实考难度

全面 实用 权威

访问希赛教育等考学院（www.educity.cn/ncre/）可获惊喜大礼！

◆海量模拟试题在线测试
◆模拟测试软件免费下载
◆配套学习资料倾情奉送
◆众考生与教师在线交流

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

National Computer Rank Examination

全国计算机等级考试专用辅导丛书

# 全国计算机等级考试专用辅导教程

# 二级公共基础知识

希赛教育等考学院 刘欣苗 主编

2013版

電子工業出版社
Publishing House of Electronics Industry
北京·BEIJING

## 内 容 简 介

本书由希赛教育等考学院组织编写，作为全国计算机等级考试二级的辅导和培训指定教程。书中内容紧扣教育部考试中心新推出的考试大纲，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。



**图书在版编目（CIP）数据**

全国计算机等级考试专用辅导教程：2013 版. 二级公共基础知识 / 刘欣苗主编. —北京：电子工业出版社，2013.1

（全国计算机等级考试专用辅导丛书）

ISBN 978-7-121-19204-3

Ⅰ.①全… Ⅱ.①刘… Ⅲ.①电子计算机－水平考试－自学参考资料 Ⅳ.①TP3

中国版本图书馆 CIP 数据核字（2012）第 295367 号

策划编辑：牛 勇
责任编辑：徐津平
印 刷：三河市鑫金马印装有限公司
装 订：三河市鑫金马印装有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编：100036
开 本：787×1092 1/16 印张：9 字数：230 千字
印 次：2013 年 1 月第 1 次印刷
定 价：19.80 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

# 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二级公共基础知识的最新考试大纲编写而成，在组织和写作上倾注了作者们的许多精力和心血，相信能够提高考试通过率，有效地为“考试过关”提供帮助。考生可通过阅读本书，快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。

## 重要通知，考生必读

根据教育部2012年12月颁发的最新文件规定，从2013年上半年开始，计算机等级考试中的二级考试科目采取无纸化考试。在无纸化考试中，传统考试的笔试部分被移植到计算机上完成，考核内容和要求不变。无纸化考试时间为120分钟，满分100分，其中选择题40分，上机操作题60分。总分达到60分，可以获得合格证书。

## 作者权威，阵容强大

希赛教育（http://www.educity.cn）专业从事人才培养、教育产品开发和教育图书出版，在职业教育方面具有很高的权威性，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（http://www.educity.cn/ncre/）是国内进行计算机等级考试在线教育的著名大型教育机构，在该领域取得了很好的效果。希赛教育等考学院组织大纲制订者和阅卷组成员已编写了数十本考试辅导教材，内容涵盖了计算机等级考试的主要科目，并组织权威专家和辅导名师录制了众多考试培训视频教程，持续对历年考试进行跟踪研究和比较研究，定期编写权威的全真模拟试题。希赛教育的计算机等级考试培训采用统一教材、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刘欣苗主编，参加编写工作的还有胡钊源、张友生、桂阳、王勇、何玉云、左水林、谢顺、邓旭光、胡光超、刘洋波等。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

上学吧在线考试中心（http://exam.shangxueba.com/）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，读者可利用上学吧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。

### 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的阅读。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。

感谢电子工业出版社牛勇老师，他在本书的策划、写作大纲的确定，以及编辑、出版等方面，付出了辛勤的劳动，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的原动力，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，编者诚恳地期望各位专家和读者不吝指正。对此，我们将十分感激！

欢迎与我们交流，电子邮箱：master@csai.cn。

希赛教育等考学院

# 目　录


**第 1 章　算法和数据结构** …… 1

1.1　算法 …… 1

1.1.1　算法的概念 …… 1

1.1.2　算法复杂度 …… 4

1.2　数据结构的基本概念 …… 5

1.2.1　什么是数据结构 …… 5

1.2.2　数据结构的图形表示 …… 7

1.2.3　线性结构与非线性结构 …… 8

1.3　线性表及其顺序存储结构 …… 9

1.3.1　线性表的基本概念 …… 9

1.3.2　线性表的顺序存储 …… 10

1.3.3　线性表的插入运算 …… 11

1.3.4　线性表的删除运算 …… 12

1.4　栈和队列 …… 14

1.4.1　栈及其基本运算 …… 14

1.4.2　队列及其基本运算 …… 15

1.5　线性链表 …… 19

1.5.1　线性链表的基本概念 …… 19

1.5.2　线性链表的基本运算 …… 23

1.5.3　循环链表的结构及其基本运算 …… 25

1.6　树与二叉树 …… 26

1.6.1　树的基本概念 …… 26

1.6.2　二叉树及其基本性质 …… 28

1.6.3　二叉树的存储结构 …… 31

1.6.4　二叉树的遍历 …… 32

1.7　查找技术 …… 34

1.7.1 顺序查找 ……34
1.7.2 二分法查找 ……34
1.8 排序技术 ……35
1.8.1 插入类排序 ……35
1.8.2 交换类排序 ……37
1.8.3 选择类排序 ……39
1.8.4 各种排序方法的比较 ……41
1.9 本章习题 ……41

**第2章 程序设计基础** ……44

2.1 程序设计方法与风格 ……44
2.2 结构化程序设计 ……46
2.2.1 结构化程序设计的原则 ……46
2.2.2 结构化程序的基本结构与特点 ……47
2.2.3 结构化程序设计原则和方法的应用 ……48
2.3 面向对象的程序设计 ……48
2.3.1 关于面向对象方法 ……48
2.3.2 面向对象方法的基本概念 ……51
2.4 本章习题 ……55

**第3章 软件工程基础** ……58

3.1 软件工程基本概念 ……58
3.1.1 软件定义与软件特点 ……58
3.1.2 软件危机与软件工程 ……59
3.1.3 软件工程过程与软件生命周期 ……60
3.1.4 软件工程的目标与原则 ……62
3.1.5 软件开发工具与开发环境 ……63
3.2 结构化方法 ……64
3.2.1 需求分析与需求分析方法 ……64
3.2.2 结构化分析方法 ……65
3.2.3 软件需求规格说明书 ……70
3.3 结构化设计方法 ……72
3.3.1 软件设计的基本概念 ……72
3.3.2 概要设计 ……75
3.3.3 详细设计 ……82
3.4 软件测试 ……86

3.4.1 软件测试的目的 87
3.4.2 软件测试的准则 88
3.4.3 软件测试技术与方法综述 88
3.4.4 软件测试的实施 95
3.5 程序的调试 97
3.5.1 基本概念 97
3.5.2 软件调试方法 99
3.6 本章习题 100

**第 4 章 数据库设计基础** 103

4.1 数据库系统的基本概念 103
4.1.1 数据、数据库、数据库管理系统 103
4.1.2 数据库系统的发展 106
4.1.3 数据库系统的基本特点 108
4.1.4 数据库系统的内部结构体系 109
4.2 数据模型 111
4.2.1 数据模型概述 111
4.2.2 E-R 模型 113
4.2.3 层次模型 117
4.2.4 网状模型 118
4.2.5 关系模型 118
4.2.6 数据操作 120
4.2.7 关系中的数据约束 121
4.3 关系代数 121
4.3.1 传统的集合运算 121
4.3.2 专门的关系运算 125
4.4 数据库设计 127
4.4.1 数据库设计概述 127
4.4.2 数据库设计的需求分析 128
4.4.3 数据库概念设计 129
4.4.4 数据库的逻辑设计 132
4.4.5 数据库的物理设计 133
4.4.6 数据库管理 133
4.5 本章习题 134

# 第 1 章　算法和数据结构

## 1.1　算　法

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科学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。本节首先讨论算法的有关知识。

### 1.1.1　算法的概念

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，能够对符合一定规范的输入，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要考虑它的可行性，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有明确的定义，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、循环三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，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，均可采用列举法进行解决。

列举的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，是可以大大减少列举量的。

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而得到的，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，斐波那契数列就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要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，而只是在解决了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。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。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。

（5）减半递推技术

实际问题的复杂程度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 $|a-b|\geqslant\varepsilon$，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。

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。

## 1.1.2 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，在衡量一个算法的工作量时，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：

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输入数据时，我们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。

# 1.2　数据结构的基本概念

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（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。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。

## 1.2.1　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，则对于前者进行查找时，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，以便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。在计算机科学中，所谓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、节点、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。例如，在考虑一年四个季节的顺序关系时，则“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，则“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。

**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。通俗地说，数据结构是指带有结构的数据元素的集合。所谓结构，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 $R$ 来表示数据元素之间的前后件的关系。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 $\boldsymbol{X}=(x_1,x_2,\cdots,x_n)$ 也是一种数据结构，即 $\boldsymbol{X}=\{\boldsymbol{D},\boldsymbol{R}\}$。

其中数据元素集合为

$D=\{x_1, x_2, \cdots, x_n\}$

关系为

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

**2. 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的，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。

## 1.2.2　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，一般称之为数据节点，并简称为节点。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节点指向后件节点。

例如，一年四季的数据结构可以用图 1-1 所示图形来表示。

又如，反映家庭成员间辈分关系的数据结构可以用图 1-2 所示图形表示。

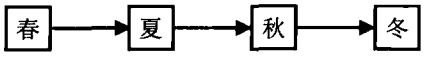


图 1-1　一年四季数据结构的图形表示

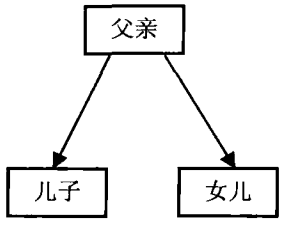


图 1-2　家庭成员间辈分关系数据结构的图形表示

显然，用图形方式表示一个数据结构是很方便的，并且比较直观。

在数据结构中，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（也称为叶子节点）。例如，在图 1-1 所示数据结构中，元素“春”所在的节点（简称为节点“春”，下同）为根节点，节点“冬”为终端节点；在图 1-2 所示数据结构中，节点

“父亲”为根节点，节点“儿子”与节点“女儿”均为终端节点。数据结构中除了根节点与终端节点外的其他节点一般称为内部节点。

通常，一个数据结构中的节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节点（称为插入运算），也可以删除数据结构中的某个节点（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，不仅数据结构中的节点（即数据元素）个数在动态地变化，而且，各数据元素之间的关系也有可能在动态地变化。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节点被删除后，它的某一个后件可能变成根节点；在一个数据结构中的终端节点后插入一个新的节点后，原来的那个终端节点就不再是终端节点而成为内部节点了。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。

### 1.2.3 线性结构与非线性结构

如果在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。一个空的数据结构在插入一个新的元素后就变为非空的数据结构；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）有且只有一个根节点。

（2）每一个节点最多有一个前件，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。如图1-1所示一年四季这个数据结构，它属于线性结构。

特别需要说明的是，一个线性结构在插入或删除任何一个节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，但当在此数据结构中插入或删除任何一个节点后就不满足这两个条件了，则该数据结构不能称为线性结构。例如，图1-3所示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这个类型，因为如果在这个数据结构中删除节点A，就不再满足上述的条件（1）。

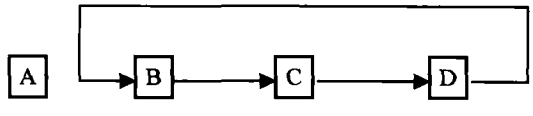


图1-3 不是线性结构的数据结构特例

如果一个数据结构不是线性结构，则称之为非线性结构。如图1-2中的家庭成员间辈分关系的数据结构，它不是线性结构，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属于线性结构还是属于非线性结构，这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。

## 1.3　线性表及其顺序存储结构

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。

### 1.3.1　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，在不同的具体情况下，它可以有不同的含义。例如，一个 $n$ 维向量$(x_1,x_2,\cdots,x_n)$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，一年中的四个季节（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个数据项组成。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 $n$（$n\geq 0$）个节点的有限序列。对于其中的节点，有且仅有一个根节点，没有前件但有一个后件；有且仅有一个终端节点，没有后件但有一个前件。其他的节点都有且仅有一个前件和一个后件。一般，一个线性表可以表示成一个线性序列：$a_1,a_2,\cdots,a_n$。其中 $a_1$ 是根节点，$a_n$ 是终端节点。