飞思考试中心
Fecit Examination Center

标准 严谨 高效 实用

配：多媒体光盘
考点速记卡

CD-ROM
光盘内容

三大系统

练习系统、考试系统、辅导系统，自动生成试卷、自动计时，试题评析

超量题库

15套权威试题，多种练习模式自由选择

教学支持

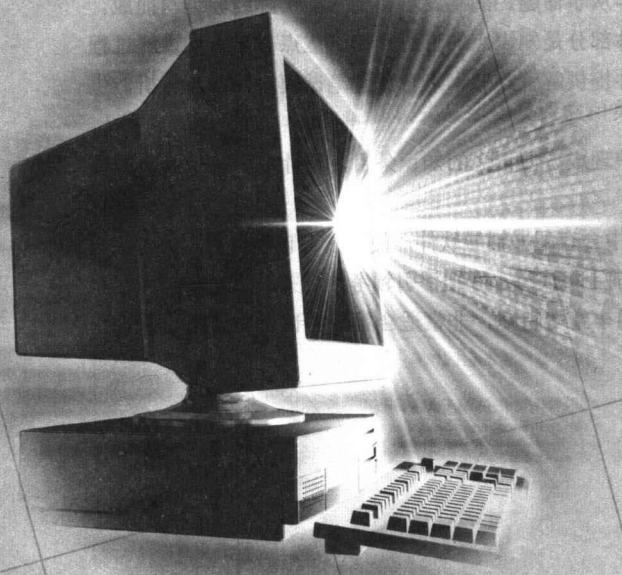
提供完整电子教案

全国计算机等级考试
实用应试教程
——二级公共基础知识

张伍荣　王军武　编著
飞思教育产品研发中心　监制

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

飞思考试中心
Fecit Examination Center



全国计算机等级考试
实用应试教程
——二级公共基础知识

张伍荣　王军武　　编著
飞思教育产品研发中心　　监制

电子工业出版社
Publishing House of Electronics Industry
北京·BEIJING

内容简介

本书紧扣最新版考试大纲，以高教版教程为基础，结合编者多年从事命题、阅卷及培训辅导的实际经验编写而成。本书编写目的是将同步辅导直接加到教程中，做到一本通。章节主体部分是知识点的讲解，精讲重点与难点；讲解过程中穿插真题和典型例题，并给出详细的解析；章节末安排适量习题并提供解答。书末附有数套笔试模拟试卷及解析，供考生考前实战演练。

本书配考点速记卡。即把一些重要考点及难记忆的知识点浓缩整理成方便记忆的知识条目，设计成卡片形式，方便考点携带和记忆。

本书配有上机盘。盘中含有电子教案，方便培训班老师教学；盘中含有配书辅导软件，便于读者自学自测。另外，盘中提供数套全真达标试题，并特别增加了试题评析功能，便于读者考前上机演练，手把手引领考生过关。

本书具有标准、严谨、实用、高效、考点全面、考题典型、练习丰富等特点，非常适合等级考试考生使用，并可作为高等院校或培训班的教材。

图书在版编目（CIP）数据

全国计算机等级考试实用应试教程．二级公共基础知识／张伍荣，王军武编著．—北京：电子工业出版社，2007.2
（飞思考试中心）
ISBN 978-7-121-03681-1

Ⅰ.全... Ⅱ.①张...②王... 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（2006）第 158935 号

责任编辑：赵红梅
印　　刷：北京牛山世兴印刷厂
装　　订：
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：850×1168　1/16　印张：9.25　　字数：251.6 千字
印　　次：2007 年 2 月第 1 次印刷
印　　数：7 000 册　　定　价：19.80 元（含光盘 1 张）

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，联系电话：（010）68279077；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

知己知彼 百战百胜

全国计算机等级考试是目前我国规模最大、参加人数最多的全国性计算机类水平考试，因其具有权威性、公平性和广泛性而在社会上享有良好的声誉，越来越多的单位把获得计算机等级考试证书作为人事录用、职称评定及职务晋升的标准之一。

为了给广大考生提供一套高效实用的标准应试教材，我们在广泛调研和充分论证的基础上，并听取资深专家及众多考生的建议，组织编写了这套《全国计算机等级考试实用应试教程》，其目的是引导考生在短时间内快速突破过关，并为广大培训学校提供一套规范实用的应试教材。

✦ 丛书书目

丛书第一批推出 5 本：

- ✧ 全国计算机等级考试实用应试教程——一级 MS Office/B
- ✧ 全国计算机等级考试实用应试教程——二级 C 语言程序设计
- ✧ 全国计算机等级考试实用应试教程——二级 Visual FoxPro 程序设计
- ✧ 全国计算机等级考试实用应试教程——二级公共基础知识
- ✧ 全国计算机等级考试实用应试教程——三级网络技术

✦ 丛书特色

（1）紧扣最新考试大纲，以高教版教程为基础，涵盖所有大纲规定考点。

（2）在全面覆盖考点的基础上，精讲重点与难点，深入分析例题，并提供实战训练。

（3）章节主体部分是知识点的讲解，讲解过程要突出重点和难点，并运用特殊标记对重要考点进行标识；讲解过程中穿插真题和典型例题，并给出详细的解析；章节末安排适量习题并提供解答。

（4）在正文中提供专门节进行上机辅导。

（5）在正文中提供数套模拟题，供考生考前实战演练。

（6）配送考点速记卡。即把一些重要考点及难记忆的知识点浓缩整理成方便记忆的知识条目，设计成卡片形式，方便考生携带和记忆。

（7）配多媒体上机盘。特点如下：

- ✧ 登录、抽题、答题、交卷等与真实上机考试完全一致，营造逼真的考试氛围。
- ✧ 自动生成试卷、自动计时，特别增加了试题评析功能，便于考生自学与提高。
- ✧ 在光盘中建立题库，提供“按章节”和“按题型”两种学习方式，读者既可以在学习过程中进行同步练习，还可以在考前进行题型的强化训练。
- ✧ 提供电子教案，方便培训班老师教学。
- ✧ 提供视频演示，手把手引领学生过关。

✦ 读者对象

本套丛书以全国计算机等级考试考生为主要读者对象，特别适合在较短时间内取得较大收获的广大应试考生，也可作为相关考试培训班的培训教材。

Preface

✦ 关于作者

丛书由一线教学及考试研究专家分工编写。作者们长期从事这方面的教学和研究工作，积累了丰富的经验，对等级考试颇有研究（其中大多数编写者多年参加真题阅卷及相关培训与辅导工作）。参与本丛书组织、指导、编写、审校和资料收集及光盘开发的人员有（以姓氏笔划为序）：于新豹、尹静、王乃和、王军武、石竹、孙虹、朱贵喜、许勇、何光明、吴婷、张伍荣、李海、杨明、陈玉旺、陈智、陈静、周松、范远宏、姚昌顺、赵传申、赵旭晖、骆健、陶安、葛武滇等，在此对诸位作者付出的辛勤劳动表示衷心的感谢。

✦ 特别致谢

在此，首先对丛书所选用的参考文献的著作者，及丛书所引用试题的出题老师和相关单位表示真诚的感谢。

感谢电子工业出版社对这套书的大力支持。

由于时间仓促，学识有限，书中不妥之处，敬请广大读者指正。

✦ 互动交流

读者的进步，是我们的心愿。您如果发现书中有任何疑惑之处，请与我们交流。

飞思教育产品研发中心

联系方式

咨询电话：（010）68134545　88254160

电子邮件：support@fecit.com.cn

服务网址：http://www.fecit.com.cn　http://www.fecit.net

通用网址：计算机图书、飞思、飞思教育、飞思科技、FECIT

目 录

CONTENTS

CONTENTS

第 1 章
数据结构与算法

1.1 算法

1.1.1 算法的基本概念

确切地说，算法是指解题方案准确而完整的描述。对于一个问题，如果可以通过一个计算机程序，在有限的存储空间内运行有限长的时间而得到正确的结果，则称这个问题是算法可解的。算法不等于程序，也不等于计算方法。

1．算法的基本特征

（1）可行性（effectiveness）：针对实际问题而设计的算法，执行后能够得到满意的结果。

（2）确定性（definiteness）：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（finiteness）：算法必须在有限时间内完成。有两重含义，一是算法中的操作步骤为有限个，二是每个步骤都能在有限时间内完成。

（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，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，同时是明确的，此顺序将在有限的次数后终止。

> ※**重点提示**：算法是解题方案的准确而完整的描述，它不等于程序，也不等于计算方法。其基本特征包括：可行性、确定性、有穷性、拥有足够的情报。

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。

基本的运算和操作有以下 4 类：①算术运算（包括加、减、乘、除等）；②逻辑运算（包括“与”、“或”、“非”等运算）；③关系运算（包括“大于”、“小于”、“等于”、“不等于”等）；④数据传输（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。一个算法一般都以用顺序、选择、循环 3 种基本控制结构组合而成。

※**重点提示**：算法由两个基本要素组成：一是算法中对数据的运算和操作，二是算法的控制结构。

3．算法设计的基本方法

计算机算法不同于人工处理的方法，工程上常用的算法有列举法、归纳法、递推法、减半递推法和回溯法等技术。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：其基本思想是，根据提出的问题，列举所有可能的情况，并用问题中给定的条件检测哪些是需要的，哪些是不需要的。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。

（2）归纳法：其基本思想是，通过列举少量的特殊情况，经过分析，最后找出一般的关系。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。

（3）递推法：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。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）递归法：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，再沿着原来的分解的逆过程逐步进行综合，这就是递归的基本思想。递归分为直接递归与间接递归两种。

（5）减半递推法：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。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：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，若试探成功，就得到问题的解，若试探失败，就逐步回退，换别的路线再逐步试探。

1.1.2 算法复杂度

※**重点提示**：算法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

算法的时间复杂度是指执行算法所需要的计算工作量。

由于算法执行时间需通过依据该算法编制的程序在计算机上运行时所消耗的时间来度量，而这种机器消耗时间与下列因素有关：

（1）书写算法的程序设计语言；

（2）编译产生的机器语言，代码质量；

（3）机器执行指令的速度；

（4）问题的规模。

这表明使用绝对的时间单位衡量算法的效率是不合适的。撇开这些与计算机硬件、软件有关的因素，可以认为一个特定算法“运行工作量”的大小，只依赖于问题的规模（通常用整数 n 表示），它是问题的规模函数。即：

算法的工作量=$f(n)$

为了便于比较同一个问题的不同算法，通常以算法中基本操作重复执行的频率作为度量标准。被看做算法基本操作的一般是最深层循环内的语句。因为算法中基本操作重复执行的频率 $T(n)$，是问题规模 n 的某个函数 $f(n)$，记为：

$$T(n)=O(f(n))$$

记号“O”读做“大 O”。它表示随问题规模 n 的增加，算法执行时间的增长率和 $f(n)$相应增加。一个没有循环算法的基本操作的执行频率与问题规模 n 无关，记为 $O(1)$，也称做常数阶。一个只有一重循环算法的基本操作的执行频率随问题规模 n 增大呈线性增大关系，记为 $O(n)$，也称做线性阶。其余常用的还有平方阶 $O(n^2)$，立方阶 $O(n^3)$，对数阶 $O(\log n)$，指数阶 $O(2^n)$，等等。

例如，下面算法是用来描述两个 $n\times n$ 矩阵相乘。在这个算法中，乘法操作、加法操作和赋值操作是矩阵相乘问题的基本操作，这些基本操作重复执行的频率为 n^3，即整个算法执行时间与问题规模 n 的三次方同阶，算法的时间复杂度为 $O(n^3)$。

```
void Mult_matrix( int c[][], int a[][], int b[][], int n)
{
    // a、b 和 c 均为 n 阶方阵，且 c 是 a 和 b 的乘积
    for (i=1; i<=n; ++i)
        for (j=1; j<=n; ++j) {
            c[i,j] = 0;
            for (k=1; k<=n; ++k)
                c[i,j] += a[i,k]*b[k,j];
        }
}
```

分析算法的工作量通过采用以下两种方法来分析：

（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。

（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。

※重点提示：算法的时间复杂度用来衡量算法执行过程中所需要的基本运算次数。

2. 算法的空间复杂度

算法的存储量是指算法执行过程中所需的最大存储空间。算法执行期间所需要的存储量应该包括以下三部分：（1）算法程序所占用的空间；（2）输入的初始数据所占用的存储空间；（3）算法执行过程中所需要的额外空间。

类似于算法的时间复杂度，通常以算法的空间复杂度作为算法所需存储空间的量度。算法空间复杂度可定义为：

$$S(n)=O(f(n))$$

表示随着问题规模 n 的增大，算法运行所需辅助存储量的增长率与 $f(n)$的增长率相同。

如果一个算法所需辅助空间都只是若干简单变量，和问题规模无关，这类所需额外空间相对于输入数据量来说是常量的算法，被称做是原地工作（in place）的算法，记为 $O(1)$。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。

※重点提示：算法空间复杂度是用来衡量算法执行过程中所需要的存储空间。

1.2 数据结构的基本概念

1.2.1 什么是数据结构

1. 数据结构研究的主要内容

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。

2. 研究数据结构的目的

研究数据结构的主要目的是为了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。

3. 数据结构的定义

数据结构是指相互有关联的数据元素的集合。数据元素之间的关系可以用前后件关系（或直接前驱与直接后继关系）来描述。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。

4. 数据的逻辑结构

数据的逻辑结构是对数据元素之间的逻辑关系的描述，它可以用一个数据元素的集合和定义在此集合中的若干关系来表示。数据的逻辑结构只抽象地反映数据元素之间的逻辑关系，即数据元素之间的前后件关系，而不管它在计算机中的存储表示形式。

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，它反映了数据元素之间的前后件关系，通常记为 R。一个数据结构 B 可以表示为：

$$B=(D, R)$$

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）$\}$。再如，家庭成员数据结构可以表示为：$B=(D, R)$，$D=\{$父亲，儿子，女儿$\}$，$R=\{$（父亲，儿子），（父亲，女儿）$\}$。

5. 数据的存储结构

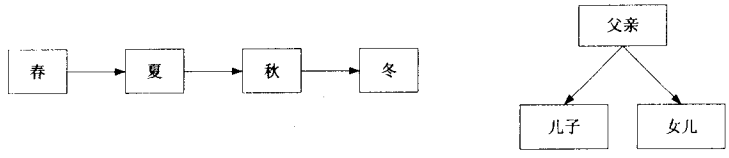
数据的存储结构是数据的逻辑结构在计算机存储空间中的存放形式，也称数据的物理结构。一个数据结构中的各数据元素在计算机存储空间的位置与逻辑关系有可能不同。

一种数据结构可根据需要采用不同的存储结构。常用的存储结构有顺序、链接、索引等存储方式。采用不同的存储结构，其数据处理的效率是不同的。

> ※**重点提示：数据的逻辑结构反映数据元素之间的逻辑关系，数据的存储结构是数据的逻辑结构在计算机存储空间中的存放形式。同一种逻辑结构的数据可以采用不同存储结构，但影响数据处理效率。**

1.2.2　数据结构的图形表示

一个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。例如图 1-1 所示的是上述一年四季的数据结构和家庭成员数据结构的图形表示法。



（a）一年四季数据结构　　（b）家庭关系数据结构

图 1-1　数据结构的图形表示

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，图 1-1（a）中“春”结点是根结点，“冬”结点是终端结点；图 1-1（b）中“父亲”结点是根结点，“儿子”结点和“女儿”结点是终端结点，

一个数据结构中的结点可能是在动态变化的。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）。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。

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。

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线性结构与非线性结构两大类型。

如果一个非空数据结构满足下列两个条件：

（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。例如，一年四季的数据结构就是一个线性结构。常见的线性结构有：线性表、栈与队列、线性链表。

如果一个数据结构不是线性结构，称之为非线性结构。例如，家庭成员的数据结构就是一个非线性结构。常见的非线性结构有：树、二叉树、图。

线性结构与非线性结构都可以是空的数据结构。对于空的数据结构，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。

> ※**重点提示：数据结构分为两大类型：线性结构与非线性结构。常见的线性结构有线性表、栈、队列和线性链表等；常用的非线性结构有树、二叉树和图等。**

1.3　线性表及其顺序存储结构

1.3.1　线性表的基本概念

线性表（Linear List）是简单最常用也是最基本的一种数据结构，它是由 $n(n\geqslant 0)$个相同类型的数据元素构成的有限序列：

$(a_1,a_2,\cdots,a_i,\cdots a_n)$

序列中数据元素的个数 n 定义为线性表的表长；n=0 时的线性表被称为空表。称为 i 在线性表中的位序。

例如，英文大写字母表（A，B，C，D，E，F，…X，Y，Z）是一个长度为 26 的线性表，其中的每一个大写字母就是一个数据元素。又如，同一花色的 13 张扑克牌（2,3,4,5,6,7,8,9,10,J,Q,K,A）是一个长度为 13 的线性表，其中每一张牌就是一个元素。

线性表有如下一些结构特征：

（1）数据元素在表中的位置由序号决定，数据元素之间的相对位置是线性的；

（2）对于一个非空线性表，有且只有一个根结点 a_1，它无前件，有且只有一个终端结点 a_n，它无后件，除根结点与终端结点外，其他所有结点有且只有一个前件，也有且只有一个后件。

在计算机内，线性表有两种基本的存储结构：一种是顺序存储，另一种是链式存储。

1.3.2 线性表的顺序存储结构

在计算机内存放线性表，一种最简单的方法是顺序存储，也称为顺序分配。线性的顺序存储结构具有以下两个基本特点：

（1）线性表中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，其前后元素在存储空间中是紧邻的，且前驱元素一定存储在后继元素的前面。

在顺序表的存储结构中，假设线性表中的第一个数据元素 a_1 的存储地址（指第一个字节的地址，即首地址）为 $Loc(a_1)$，每一个数据元素占 k 个字节，则序号为 i 的数据元素 a_i 的存储地址为：

$$Loc(a_i)=Loc(a_1)+(i-1)*k$$

图 1-2 所示的是一个长度为 n 的线性表 $(a_1,a_2,\cdots,a_i,\cdots,a_n)$ 在计算机中的顺序存储结构。

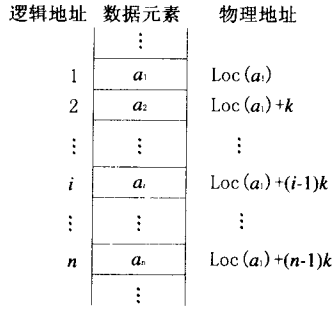


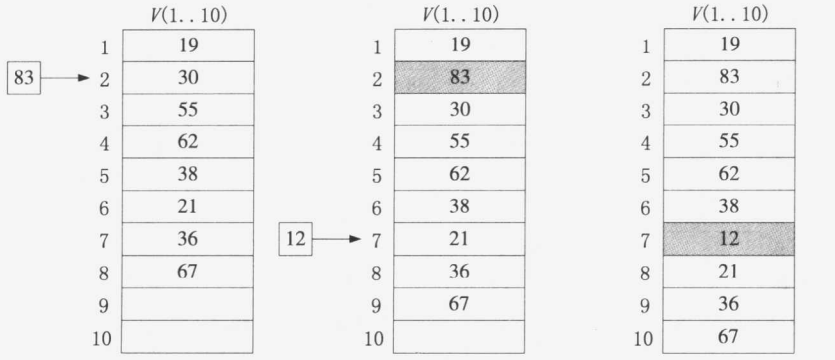
图 1-2 线性表的顺序存储

在线性表的顺序存储结构下，可以对线性表做以下运算：（1）插入；（2）删除；（3）查找；（4）排序；（5）分解；（6）合并；（7）复制；（8）逆转等。

1.3.3 顺序表的插入运算

先通过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。图 1-3（a）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 30）之前插入一个新元

素 83。插入过程如下：



（a）长度为8的线性表　（b）插入83后，长度变为9　（c）插入12后，长度变为10

图 1-3　线性表在顺序存储下的插入

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往后移动一个位置，然后将新元素 83 插入到第 2 个元素的位置。插入一个新元素后，线性表元素的个数变成 9 个。

如果在线性表的第 7 个元素之前插入一个新元素 12,则采用类似的方法：将第 9、8、7 个元素往后移动一个位置,将新元素 12 插入到第 7 个元素位置。插入后，线性表的长度变成了 10。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，则造成称为“上溢”的错误。

一般来说，设长度为 n 的线性表：

$$(a_1, a_2, \cdots, a_{i-1}, a_i, \cdots, a_n)$$

在表的第 i（$1 \leqslant i \leqslant n$）个元素之前插入一个新结点 x，插入后得到长度为 $n+1$ 的线性表：

$$(a_1, a_2, \cdots, a_{i-1}, x, a_i, \cdots, a_n)。$$

在一般情况下，要在第 i（$1 \leqslant i \leqslant n$）个元素之前插入一个元素，首先需从最后一个（即第 n 个）元素开始，直到第 i 个元素共（$n-i+1$）个元素依次向后移动一个位置，空出第 i 个位置，然后将新元素插入到第 i 项，插入结束后，线性表的长度增长 1。所以，若顺序表中结点个数为 n，在往每个位置插入的概率相等的情况下，插入一个结点的平均移动结点个数为 $n/2$。

※重点提示：顺序表的插入运算时需要移动元素，在等概率情况下，平均需要移动 $n/2$ 个元素。

1.3.4　顺序表的删除运算

先通过一个例子来说明如何在顺序存储结构的线性表中删除一个元素。图 1-4（a）所示的是一个长度为 8 的线性表，顺序存储在长度为 10 的存储空间中，现在要求删除线性表中的第 1 个元素（即删除元素 19）。删除过程如下：

从第 2 个元素开始直到最后一个元素，将其中的每一个元素均依次往前移动一个位置。此时，线性表的长度变成 7。如果再删除线性表中的第 6 个元素（即删除元素 36），则采用类似的方法，将第 7 个元素往前移动一个位置。此时，线性表的长度变成了 6。

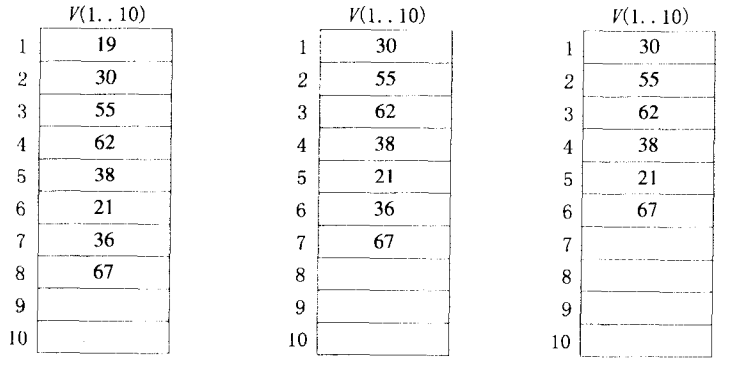


图 1-4　线性表在顺序存储下的删除

一般来说，设长度为 n 的线性表：

$$(a_1,a_2,\cdots,a_{i-1},a_i,a_{i+1}\cdots,a_n)$$

将表中第 i（$1\leqslant i\leqslant n$）个结点删去，删除后得到长度为 n-1 的线性表：

$$(a_1,a_2,\cdots,a_{i-1},a_{i+1},\cdots,a_n)$$

在一般情况下，要删除第 i（$1\leqslant i\leqslant n$）个元素时，则要从第 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，线性表的长度减少 1。所以，若顺序表中结点个数为 n，在往每个位置删除的概率相等的情况下，删除一个结点的平均移动结点个数为$(n-1)/2$。

※**重点提示**：进行顺序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。

由线性表在顺序存储结构下的插入与删除运算可以看出，在顺序存储表示的线性表中插入或删除一个数据元素，平均约需移动表中一半元素。这个数目在线性表的长度较大时是很可观的。这个缺陷完全是由于顺序存储要求线性表的元素依次紧挨存放造成的。因此，这种顺序存储表示仅适用于不常进行插入和删除操作、表中元素相对稳定的线性表。

1.4　栈和队列

1.4.1　栈及其基本运算

在日常生活中，很多事务是按照“后到达的比先到达的优先”的顺序处理的。例如，穿衣服的顺序是先衬衫，后制服，最后是大衣；脱衣服的顺序必须反过来；若想在衬衫和制服之间加一件毛衣时，必须先脱去大衣和制服，才能穿上毛衣。又如，枪支的子弹压入子弹夹时，总是要将子弹一粒粒按顺序压入，而射出时却是最后压入的子弹被最先打出来，而最先压入的子弹最后才被打出来。栈的结构特点和运算正是上述实践的抽象。

1．栈的定义

栈（stack）是一种只允许在表的一端进行插入或删除操作的特殊的线性表。表中允许进行插入与删除操作的一端称为栈顶（top），而不允许插入与删除操作的另一端称为栈底（bottom）。当栈中没有