全国1001所高校学子的明智选择

（第2版）

全国计算机等级考试全能教程

二级公共基础知识

全国计算机等级考试命题研究组◎编写

- **权威性：**
  命题研究组资深专家最新研究成果，作者长期工作在
- **全真性：**
  所有试题均紧扣最新大纲的要求，布局均与考试真题完全相当。
- **实战性：**
  配书光盘提供现场做题环境，便于考生在考前60天实战冲刺。
- **前瞻性：**
  在深入研究历年真题的基础上，提供严密的知识体例供学习及应试。

全国大学生最关注的等考品牌

北邮·等考

北京邮电大学出版社
www.buptpress.com

全国计算机等级考试全能教程

——二级公共基础知识(第2版)

全国计算机等级考试命题研究组　编写

北京邮电大学出版社
·北京·

内容简介

本书根据最新全国计算机等级考试最新考试大纲和官方教程，在研究历年真题的基础上编写而成。本书章节安排与官方教程一致。每章末尾设置有“应试加油站”，该板块汇集重点提示、解题技巧等部分，引导考生掌握重点内容，增强考生的解题能力和综合应用能力。在正文中提供一章进行无纸化考试专题辅导。本书附有5套无纸化考试样卷，供考生考前实战演练。

本书配有光盘，光盘中的配套软件完全模拟真题考试环境，便于考生实战演练。

本书可供全国计算机等级考试二级考生复习使用，特别适合考前冲刺使用，同时也非常适合相关等级考试培训班用作培训教材，以及大、中专院校师生的教学用书。

图书在版编目(CIP)数据

全国计算机等级考试全能教程. 二级公共基础知识 /全国计算机等级考试命题研究组编写.--2版.--北京：北京邮电大学出版社，2014.1

ISBN 978-7-5635-3735-8

Ⅰ.①全… Ⅱ.①全… Ⅲ.①电子计算机－水平考试－教材 Ⅳ.①TP3

中国版本图书馆CIP数据核字(2013)第245720号

书　　名：全国计算机等级考试全能教程——二级公共基础知识(第2版)

作　　者：全国计算机等级考试命题研究组

责任编辑：满志文　姚顺

出版发行：北京邮电大学出版社

社　　址：北京市海淀区西土城路10号(邮编：100876)

发 行 部：电话：010-62282185　传真：010-62283578

E-mail：publish@bupt.edu.cn

经　　销：各地新华书店

印　　刷：北京联兴华印刷厂

开　　本：889 mm×1 194 mm　1/16

印　　张：7.25

字　　数：242千字

版　　次：2014年1月第2版　2014年1月第1次印刷

ISBN 978-7-5635-3735-8　　定　价：19.80元

前　言

全国计算机等级考试为国内影响最大、参加人数最多的计算机类水平考试，在推广、普及计算机应用知识和技术中发挥了重要作用，并为用人单位的人员考核提供了客观、公正的评价标准。

为了引导考生顺利通过全国计算机等级考试，我们根据新大纲的要求，结合典型试题，按教育部考试中心指定教材的篇章结构，由从事全国计算机等级考试试题研究人员及在等级考试第一线从事命题研究、教学、辅导和培训的老师精心编写了《全国计算机等级考试全能教程——二级公共基础知识(第2版)》。

1. 本书特色

☑ **突出实用性和高效性**：书的章名、节名与教育部考试中心指定教程同步，每章还设计了以下板块。

- 考题链接：精选出常考题型与历年真题穿插在知识点中的讲解，一方面有利于考生对知识点的理解，另一方面也让考生明白试题是如何考的。
- 应试加油站：该板块汇集考试重点整理、解题技巧等部分，引导考生掌握重点内容，增强考生的解题能力和综合应用能力。
- 习题：针对本节知识点设计一部分题目，方便读者一点一练，巩固提高。

☑ **突出标准性与严谨性**：本书由从事全国计算机等级考试试题研究人员及在等级考试第一线从事命题研究、教学、辅导和培训的老师分工编写，层次清晰，结构严谨，导向准确。

☑ **结构科学，实用性强**：紧扣新大纲要求，精讲考点、重点与难点，深入分析典型范例，抓住等级考试题眼，并提供实战训练。

☑ **注重无纸化考试的辅导**：针对无纸化考试的特点，本书在深入研究无纸化考试题库的基础上，将真题进行分类，提炼出题型，按类型进行解析，便于考生专项攻克，提高复习效率。

☑ **全真模拟，实战提高**：根据新大纲、新考点、新题型进行最新命题，书末提供5套无纸化考试样题，供考生考前实战演练。

☑ **书盘结合，一本速通**：光盘中包括考试模拟系统，提供数套真题供考生联系，考试环境与真实考试一致，帮助考生顺利过关。

2. 读者对象

本书以全国计算机等级考试考生为主要读者对象，适合于考生在等考前复习使用，也可作为相关考试培训班的辅助教材，以及大、中专院校师生的参考书。

3. 本书作者

本书由史春联、韩雪、汪胡青主编，与编写与考试研究、光盘制作的人员有：陈玲、卢振侠、周建霞、周汉、毛辉杰、陈丽荣、洪秋妹、何光明、陈海燕、冯伯虎、顾锦江、朱杰、杨婷、郭丽红、吴海涛、徐劲松、余永红、赵卫滨、蒋晶、赵强等。

由于作者水平有限，书中难免存在疏漏和错误之处，恳请专家和广大读者批评指正。在学习过程中，遇到疑难问题，可以通过以下方式与我们联系：bjbaba@263.net。

编　者

全国计算机等级考试全能教程丛书

顾问委员会

成员名单 （排名不分先后）

陈　畅　陈海燕　迟冬祥　邓达平　丁为民　江家宝

焦风杰　李　海　刘家琪　卢振侠　骆　健　盛　可

史春联　史国川　孙　虹　唐瑞华　王　钢　王继水

王景胜　吴　婷　吴成林　吴晓维　谢书玉　杨　晋

杨张静　尹　静　应艳杰　张　博　张　剑　张居晓

赵　明　钟志水　谭　红　林　莉　徐文娟　王　强

邓祖明　张　强　王敏珍　吴文斗

本书主编　史春联　韩　雪　汪胡青

目　录

第 1 章　数据结构与算法 …… 1

1.1　算　法 …… 1
1.1.1　算法的基本概念 …… 1
1.1.2　算法复杂度 …… 3
1.2　数据结构的基本概念 …… 4
1.2.1　什么是数据结构 …… 4
1.2.2　数据结构的图形表示 …… 5
1.2.3　线性结构与非线性结构 …… 5
1.3　线性表及其顺序存储结构 …… 6
1.3.1　线性表的基本概念 …… 6
1.3.2　线性表的顺序存储结构 …… 6
1.3.3　顺序表的插入运算 …… 8
1.3.4　顺序表的删除运算 …… 8
1.4　栈和队列 …… 9
1.4.1　栈及其基本运算 …… 9
1.4.2　队列及其基本运算 …… 10
1.5　线性链表 …… 13
1.5.1　线性链表的基本概念 …… 13
1.5.2　线性链表的基本运算 …… 14
1.5.3　循环链表及其基本运算 …… 16
1.6　树与二叉树 …… 17
1.6.1　树的基本概念 …… 17
1.6.2　二叉树及其基本性质 …… 18
1.6.3　二叉树的存储结构 …… 20
1.6.4　二叉树的遍历 …… 21
1.7　查找技术 …… 22
1.7.1　顺序查找 …… 22
1.7.2　二分法查找 …… 23
1.8　排序技术 …… 23
1.8.1　交换类排序法 …… 23
1.8.2　插入类排序法 …… 25
1.8.3　选择类排序法 …… 27
1.9　应试加油站 …… 28
1.9.1　考试重点整理 …… 28
1.9.2　解题技巧 …… 28
1.10　过关练习与答案 …… 30
1.10.1　过关练习 …… 30
1.10.2　参考答案 …… 30

第 2 章　程序设计基础 …… 31

2.1　程序设计方法与风格 …… 31
2.2　结构化程序设计 …… 32
2.2.1　结构化程序设计的原则 …… 32
2.2.2　结构化程序的基本结构与特点 …… 32
2.2.3　结构化程序设计原则和方法的应用 …… 33
2.3　面向对象的程序设计 …… 33
2.3.1　关于面向对象方法 …… 33
2.3.2　面向对象方法的基本概念 …… 35
2.4　应试加油站 …… 37
2.4.1　考试重点整理 …… 37
2.4.2　解题技巧 …… 37
2.5　过关练习与答案 …… 38
2.5.1　过关练习 …… 38
2.5.2　参考答案 …… 38

第 3 章　软件工程基础 …… 39

3.1　软件工程基本概念 …… 39
3.1.1　软件定义与软件特点 …… 39
3.1.2　软件危机与软件工程 …… 40
3.1.3　软件工程过程与软件生命周期 …… 41
3.1.4　软件工程的目标和原则 …… 43
3.1.5　软件开发工具与软件开发环境 …… 44
3.2　结构化分析方法 …… 44
3.2.1　需求分析和需求分析方法 …… 45
3.2.2　结构化分析方法 …… 45
3.2.3　软件需求规格说明书 …… 48
3.3　结构化设计方法 …… 49
3.3.1　软件设计的基本概念 …… 49
3.3.2　概要设计 …… 51
3.3.3　详细设计 …… 54
3.4　软件测试 …… 56
3.4.1　软件测试的目的 …… 56
3.4.2　软件测试的准则 …… 56
3.4.3　软件测试技术和方法综述 …… 57
3.4.4　软件测试的实施 …… 59
3.5　程序的调试 …… 60

3.5.1 基本概念 …… 60
3.5.2 软件调试方法 …… 62
3.6 应试加油站 …… 62
3.6.1 考试重点整理 …… 62
3.6.2 解题技巧 …… 63
3.7 过关练习与答案 …… 64
3.7.1 过关练习 …… 64
3.7.2 参考答案 …… 65

第4章 数据库设计基础 …… 66

4.1 数据库系统的基本概念 …… 66
4.1.1 数据、数据库、数据库管理系统 …… 66
4.1.2 数据库系统的发展 …… 67
4.1.3 数据库系统的基本特点 …… 68
4.1.4 数据库系统的内部结构体系 …… 69
4.2 数据模型 …… 70
4.2.1 数据模型的基本概念 …… 70
4.2.2 E-R模型 …… 71
4.2.3 层次模型 …… 73
4.2.4 网状模型 …… 73
4.2.5 关系模型 …… 74
4.3 关系代数 …… 75
4.4 数据库设计与管理 …… 86
4.4.1 数据库设计概述 …… 86
4.4.2 数据库设计的需求分析 …… 86
4.4.3 数据库概念设计 …… 87
4.4.4 数据库的逻辑设计 …… 88
4.4.5 数据库的物理设计 …… 89
4.4.6 数据库管理 …… 89
4.5 应试加油站 …… 90
4.5.1 重点提示 …… 90
4.5.2 解题技巧 …… 91
4.6 过关练习与答案 …… 92
4.6.1 过关练习 …… 92
4.6.2 参考答案 …… 93

第5章 真题及答案解析 …… 94

5.1 真 题 …… 94
5.2 真题答案与解析 …… 96

第6章 模拟试题及答案解析 …… 99

模拟试题一 …… 99
模拟试题二 …… 100
模拟试题三 …… 100
模拟试题四 …… 101
模拟试题五 …… 102
模拟试题一答案解析 …… 103
模拟试题二答案解析 …… 103
模拟试题三答案解析 …… 104
模拟试题四答案解析 …… 105
模拟试题五答案解析 …… 106

第1章 数据结构与算法

1.1 算 法

1.1.1 算法的基本概念

所谓算法是指解题方法的准确而完整的描述。

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的时间而得到正确的结果，则称这个问题是算法可解的，但是算法并不等于程序，也不等计算机方法，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，使得给定类型的问题通过有限的指令序列、在有限的时间内被求解。其中每一条指令表示一个或多个操作。

1. 算法的基本特性

算法的主要特征有以下几个。

(1)可行性(effectiveness)。针对实际问题设计的算法，人们总是希望得到满意的结果。但一个算法总是在某个特定的计算机工具上执行的，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。在设计一个算法时必须要考虑它的可行性，否则是不会得到满意结果的。

(2)确定性(definiteness)。算法的确定性，是指算法中的每一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时间内做完，即算法必须能在执行有限的步骤之后终止。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。一个算法的执行结果总是与输入的初始数据有关，它可以有多个输入，也可以不要输入，但必须有一个或多个输出，不同的输入将会有不同的输出结果。通常情况下，当情报足够时，算法才是有效的，当情报不够时，算法可能无效。

综上所述，所谓算法，就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、明确的，且此顺序将在有限的次数内终止。

2. 算法的基本要素

一个算法通常由两种基本要素构成：一是对数据对象的运算和操作，二是算法的控制结构。

(1)算法中对数据对象的运算和操作。每个算法实际上是按照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，基本的运算和操作包括以下四类：

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”和“不等于”等运算。

④数据传输：主要包括赋值、输入和输出等操作。

(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，更直接反映了算法的设计是否符合结构化设计原则。描述算法的工具通常有传统流程图、N-S 图、伪码描述语言等。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

计算机中算法设计的方法有以下几种。

(1)列举法

列举法的基本思想是，根据提出的问题，列举所有可能的情况，并用问题中给定的条件检验哪些是需要的、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。

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，需重注意在用列举法设计算法时，应尽量使方案优化，减少运算工作量。

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。列举法虽然是一种比较笨拙且原始的方法，且运算量比较大，但在有些实际问题中，局部使用列举法却是很有效的，因此，列举法是计算机算法中的一种基础算法。

(2)归纳法

归纳法的基本思想是，通过列举少量的简单而又特殊的情况，经过分析，总结归纳出一般性的结论。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。

(3)递推

所谓递推，是指从给定的或已知的初始条件出发，逐步推理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简化而确定。

(4)递归

人们在解决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，一般总是将问题逐层分解，最后归结为一些简单的问题。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。

(5)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。一种有效的解决此类问题的方法是尝试。通过分析问题，找出一个解决的线索，沿此线索逐步试探，若试探成功，就得到问题的解，否则，就逐步回退，换线索再逐步试探。回溯法在处理复杂数据结构方面有着广泛的应用。

1.1.2 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。基本运算反映了算法运算的主要特征，因此，用基本运算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。例如，在考虑两个矩阵相乘时，可以将两个实数之间的乘法运算作为基本运算，而对于所用的加法(或减法)运算忽略不计。又如，当需要在一个表中进行查找时，可以将两个元素之间的比较作为基本运算。

算法所执行的基本运算次数还与问题的规模有关，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

$$\text{算法的工作量} = f(n)$$

其中 n 是问题的规模。它表示随着问题规模 n 的增大，算法执行时间的增长率和 $f(n)$ 的增长率相同，称为算法的时间复杂度。

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。

(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。

考题链接

【例 1-1】算法的时间复杂度是指_____。

A) 算法的执行时间　　B) 算法所处理的数据量

C) 算法程序中的语句或指令条数　　D) 算法在执行过程中所需要的基本运算次数

解　析：本题考查的知识点是时间复杂度的概念。算法的时间复杂度是指算法在执行过程中所需要的基本运算次数。

答　案：D

2. 算法的空间复杂度

一个算法的空间复杂度就是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（in place）工作的。在许多实际的问题中，为了减少算法的内存空间，一般采取压缩存储技术，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

1.2.1 什么是数据结构

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。

1. 数据的逻辑结构

数据结构是指反映数据元素之间的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。所谓结构实际上就是指数据元素之间的前后件关系。

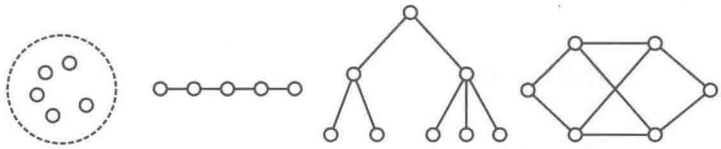
由上所述，一个数据结构应包含以下两方面的信息：

（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之间的逻辑关系，而上述的数据结构实际上是数据的逻辑结构，它是对数据元素之间的逻辑关系的描述。

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。根据数据元素之间关系的不同特性，通常有四类基本的逻辑结构，如图 1-1 所示。



(a) 集合结构　(b) 线性结构　(c) 树形结构　(d) 图形结构

图 1-1　数据的逻辑结构

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，数据结构包括数据的逻辑结构和数据的物理结构。数据的逻辑结构可以看成是从具体问题抽象出来的数学模型，它与数据的存储无关。研究数据结构的目的是为了在计算机中实现对它的操作，为此还需要研究如何在计算机中表示一个数据结构。

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物理结构，或称数据的存储结构。

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。

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。

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，反映 D 中数据元素之间的前后件关系，通常记为 R。一个数据结构可以表示如下：

$B=(D,R)$

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，一般用二元组表示。例如，家庭成员数据结构可以表示如下：

$B=(D,R)$

$D=\{$父亲，儿子，女儿$\}$

$R=\{$(父亲，儿子)，(父亲，女儿)$\}$

在数据结构的图形表示中，对于集合 D 中的每个数据元素用标有元素值的方框表示，通常称为数据结点，并简称为结点；对于关系 R 中的每个二元组，用一条有向线段从前件结点（或称为前驱结点）指向后件结点（或称为后继结点）。例如，上例中的家庭成员数据结构也可以用图 1-2 表示。

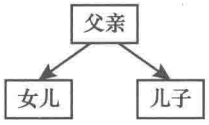


图 1-2 数据结构的图形表示

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（或叶子结点）。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，可以在数据结构中增加新结点（称为插入运算），或删除某个结点（称为删除运算），这两种运算是数据结构的基本运算，除此之外，还有查找、分类、合并、复制和修改等运算。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就变成了根结点；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。

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。在一个空的数据结构中插入一个新的元素后就变非空；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，又可以将数据结构分为两大类：线性结构和非线性结构。

如果一个非空的数据结构满足下列两个条件：

①有且只有一个根结点。

②每个结点最多有一个直接前件，也最多有一个直接后件。

则称该数据结构为线性结构，又称线性表。

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，但当在此数据结构中插入或删除任何一个结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。

如果一个数据结构不是线性结构，则称之为非线性结构。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。

线性结构和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。

考题链接

【例 1-2】下列叙述中正确的是_____。

A）线性表的链式存储结构与顺序存储结构所需要的存储空间是相同的

B）线性表的链式存储结构所需要的存储空间一般要多于顺序存储结构

C）线性表的链式存储结构所需要的存储空间一般要少于顺序存储结构

D）上述三种说法都不对

解　析：线性表的顺序存储结构使用一组地址连续的存储单元，而链式存储结构除了存放数据之外，还需要存放指向下一个元素的指针，因此选 B。

答　案：B

【例 1-3】在长度为 n 的线性表中，寻找最大项至少需要_____次。

A）1　　B）n　　C）$n-1$　　D）$\log_2 n$

解　析：线性表中顺序查找时，第一个就是最大项，次数为 1。

答　案：A

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)是最简单、最常用的一种数据结构。

线性表由一组数据元素构成。线性表是具有相同数据类型的 $n(n\geqslant 0)$ 个数据元素组成的有限序列，表中的每一个数据元素，除了第一个外，有且只有一个前件，除了最后一个外有且只有一个后件。即线性表或是一个空表，或可以表示为：

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

其中 $a_i(i=1,2,\cdots,n)$ 是属于数据对象的元素，通常也称为其线性表中的一个结点。n 为表长，$n=0$ 时称为空表。表中相邻元素之间存在着顺序关系。将 a_{i+1} 称为 a_i 的直接前件，a_{i+1} 称为 a_i 的直接后件。显然，线性表是一种线性结构。数据元素在线性表中的位置只取决于它们之间的序号，即数据元素之间的相对位置是线性的。

对于非空的线性表，有如下一些结构特征：

(1)有且只有一个根结点 a_1，无前件。

(2)有且只有一个终端结点 a_n，无后件。

(3)除根结点和终端结点外，其他所有结点有且仅有一个直接前件，也有且仅有一个直接后件。线性表中结点的个数 n 称为线性表的长度。当 $n=0$ 时，称为空表。

1.3.2　线性表的顺序存储结构

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。

在程序设计语言中，一维数组在内存中占用的存储空间就是一组连续的存储区域，因此，用一维数组来表示顺序表的数据存储区域是再合适不过的。考虑到线性表的运算有插入、删除等运算，即表的长度是可变的，因此，数组的容量需设计得足够大，假设用 data[MAXSIZE]来表示，其中 MAXSIZE 是一个根据实际问题定义的足够大的整数，线性表中的数据从 data[0]开始依次顺序存放，但当前线性表中的实际元素个数可能未达到 MAXSIZE 多个，因此需用一个变量 last 记录当前线性表中最后一个元素在数组中的位置，即 last 起一个指针的作用，始终指向线性表中最后一个元素，因此，当表为空时，last=－1。这种存储思想的具体描述可以是多样的。

从结构性上考虑，通常将 data 和 last 封装成一个结构作为顺序表的类型：

```
typedefstruct
{datatypedata[MAXSIZE];
intlast;
} SeqList;
```

定义一个顺序表："SeqList"。

这样表示的线性表如图 1-3(a)所示。表长＝L. last＋1，线性表中的数据元素 a_1 至 a_n 分别存放在 L. data[0]至 L. data[L. last]中。

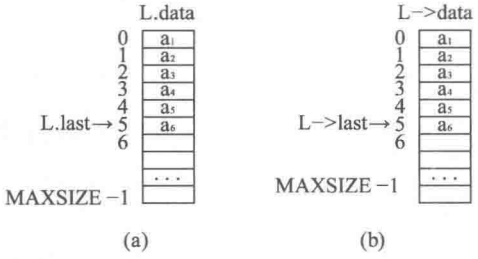


图 1-3 线性表的顺序存储示意图

由于算法通常用 C 语言描述，根据 C 语言中的一些规则，有时定义一个指向 SeqList 类型的指针更为方便：

```
SeqList * L ;
```

L 是一个指针变量，线性表的存储空间通过 L＝malloc(sizeof(SeqList)) 操作来获得。

L 中存放的是顺序表的地址，这样表示的线性表如图 1-3(b)所示。表长表示为(* L). last 或 L->last＋1，线性表的存储区域为 L->data，线性表中数据元素的存储空间为：

L->data[0]～ L->data[L->last]

在线性表的顺序存储结构下，可以对线性表进行各种处理。主要的运算有以下几种：

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按要求将一个线性表分解成多个线性表(即线性表的分解)。

⑥按要求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。

1.3.3 顺序表的插入运算

线性表的插入是指在表的第 $i(1\leqslant i\leqslant n+1)$ 个位置上插入一个值为 x 的新元素，插入后使原来表长为 n 的表：

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

成为表长为 $n+1$ 的表：

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

顺序表上完成这一运算通过以下步骤进行：

(1)将 $a_i\sim a_n$ 顺序向后移动，为新元素让出位置。

(2)将 x 置入空出的第 i 个位置。

(3)修改 last 指针(相当于修改表长)，使之仍指向最后一个元素。

下面是插入算法的时间性能分析。

顺序表的插入运算，其时间主要消耗在了数据的移动上，在第 i 个位置上插入 x，从 a_i 到 a_n 都要向后移动一个位置，共需要移动 $n-i+1$ 个元素，而 i 的取值范围为 $1\leqslant i\leqslant n+1$，即有 $n+1$ 个位置可以插入。设在第 i 个位置上作插入的概率为 P_i，则平均移动数据元素的次数：

$$E_{in}=\sum_{i=1}^{n+1}P_i(n-i+1)$$

设 $P_i=1/(n+1)$，即为等概率情况，则：

$$E_{in}=\sum_{i=1}^{n+1}P_i(n-i+1)=\frac{1}{n+1}\sum_{i=1}^{n+1}(n-i+1)=\frac{n}{2}$$

这说明：在顺序表上做插入操作需移动表中一半的数据元素，显然该算法的时间复杂度为 $O(n)$。

在一般情况下，如果插入运算在第 $i((1\leqslant i\leqslant n)$ 个元素之前进行，则原来第 i 个元素之后(包括第 i 个元素)的所有元素都必须移动。在平均情况下，要在线性表中插入一个新元素，需要移动表中一半的元素。因此，在线性表顺序存储的情况下，要插入一个新元素，其效率是很低的，特别是在线性表比较大的情况下更为突出，由于数据元素的移动而消耗较多的处理时间。

1.3.4 顺序表的删除运算

线性表的删除运算是指将表中第 $i(1\leqslant i\leqslant n)$ 个元素从线性表中去掉，删除后使原来表长为 n 的线性表：

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

成为表长为 $n-1$ 的线性表：

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

顺序表上完成这一运算的步骤如下：

(1)将 $a_{i+1}\sim a_n$ 顺序向前移动。

(2)修改 last 指针(相当于修改表长)使之仍指向最后一个元素。

下面是删除算法的时间性能分析。

与插入运算相同，其时间主要消耗在了移动表中元素上，当删除第 i 个元素时，其后面的元素 $a_{i+1}\sim a_n$ 都要向前移动一个位置，共移动了 $n-i$ 个元素，所以平均移动数据元素的次数：

$$E_{de}=\sum_{i=1}^{n}P_i(n-i)$$

在等概率情况下，$P_i=1/n$，则：

$$E_{de}=\sum_{i=1}^{n}P_i(n-i)=\frac{1}{n}\sum_{i=1}^{n+1}(n-i)=\frac{n-1}{2}$$

这说明：在顺序表上做删除运算时大约需要移动表中一半的元素，显然该算法的时间复杂度为$O(n)$。

在一般情况下，如果要删除第$i(1\leqslant i\leqslant n)$个元素，则原来第i个元素之后的所有元素都必须依次往前移动一个位置。在平均情况下，要在线性表中删除一个元素，需要移动表中一半的元素。因此，在线性表顺序存储的情况下，要删除一个元素，其效率也是很低的，特别是在线性表比较大的情况下更为突出，由于数据元素的移动而消耗较多的处理时间。

1.4 栈和队列

1.4.1 栈及其基本运算

1. 什么是栈

栈实际上也是线性表，只不过是一种特殊的线性表。在这种特殊的线性表中，其插入与删除运算都只在线性表的一端进行。即在这种线性表的结构中，一端是封闭的，不允许进行插入与删除元素；另一端是开口的，允许插入与删除元素。在顺序存储结构下，对这种类型线性表的插入与删除运算是不需要移动表中其他数据元素的。这种线性表称为栈。

栈(stack)是限定在一端进行插入与删除的线性表。

根据栈的定义可知，栈顶元素总是最后入栈的，因而是最先出栈；而栈底元素总是最先入栈的，因而是最后出栈。这种表是按照后进先出(Last In First Out，LIFO)的原则组织数据的，因此，栈也被称为“后进先出”的线性表。由此可以看出，栈具有记忆作用。

图1-4是一个栈的示意图，通常用指针top指示栈顶的位置，用指针bottom指向栈底。栈顶指针top动态反映栈的当前位置。

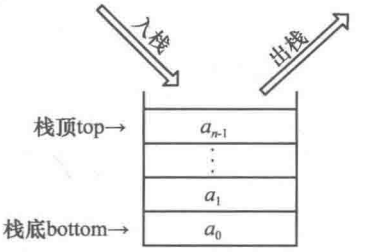


图1-4 栈的示意图

2. 栈的顺序存储

利用一组地址连续的存储单元依次存放自栈底到栈顶的数据元素，这种形式的栈称为顺序栈。可以使用一维数组来作为栈的顺序存储空间。设指针top指向栈顶元素的当前位置，以数组下标的一端作为栈底，通常以top=0时为空栈，在元素进栈时指针top不断地加1，当top等于数组的最大下标值时则栈满。栈的操作示意图如图1-5所示。

图1-5(a)是空栈，图1-5(b)是只有一个元素A入栈，图1-5(c)是A、B、C、D、E 5个元素依次入栈，图1-5(d)是在图1-5(c)之后E、D相继出栈，此时栈中还有3个元素，或许最近出栈的元素D、E仍然在原先的单元存储着，但top指针已经指向了新的栈顶，也就是说元素D、E已不在栈中了。

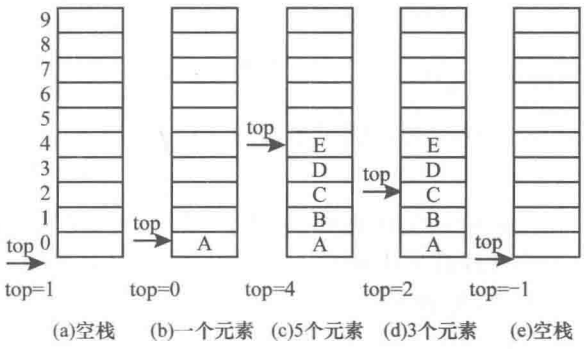


图 1-5　栈顶指针 top 与栈中数据元素的关系

3. 顺序栈的基本运算

(1)入栈运算

入栈运算是指在栈顶插入一个新元素，其基本步骤是首先将栈顶指针进一(即 top 加 1)，然后将新元素插入到栈顶指针指向的位置。

当栈顶指针已经指向存储空间的最后一个位置时，说明栈顶空间已满，不可能再进行入栈操作，这种情况称为栈“上溢”错误。

(2)退栈运算

退栈运算是指取出栈顶元素并赋给一个指定的变量，分两个基本步骤进行：首先将栈顶元素(栈顶指针指向的元素)赋给一个指定的变量，然后将栈顶指针退一(即 top 减 1)。

当栈顶指针为 0 时，说明栈空，不可能再进行退栈操作，这种情况称为栈“下溢”错误。

考题链接

【例 1-4】下列叙述中正确的是_____。

A) 在栈中，栈中元素随栈底指针与栈顶指针的变化而动态变化

B) 在栈中，栈顶指针不变，栈中元素随栈底指针的变化而动态变化

C) 在栈中，栈底指针不变，栈中元素随栈顶指针的变化而动态变化

D) 上述三种说法都不对

【例 1-5】一个栈的初始状态为空。首先将元素 5,4,3,2,1 依次入栈，然后退栈一次，再将元素 A,B,C,D 依次入栈，之后将所有元素全部退栈，则所有元素退栈(包括中间退栈的元素)的顺序为_____。

A) 54321ABCD　　B) 12345ABCD　　C) 1DCBA2345　　D) 12345DCBA

解　析：栈的特点是后进先出，所以最后入栈的最先出栈，首先将元素 5,4,3,2,1 依次入栈，然后退栈一次，第一次出栈为 1，A,B,C,D 依次入栈后栈内为 DCBA2345，因此输出顺序为 1DCBA2345。

答　案：C

(3)读栈顶元素

读栈顶元素是指将栈顶元素赋给一个指定的变量。需要注意的是，该运算不删除栈顶元素，只是将它的值赋给一个变量，因此，在该运算中，栈顶指针不会改变。

当栈顶指针为 0 时，说明栈空，读不到栈顶元素。

1.4.2　队列及其基本运算

1. 什么是队列

在计算机系统中，如果依次只能执行一个用户程序，则在多个用户程序需要执行时，这些用户程序必