高懿／主编　魏坤／副主编

考试无忧

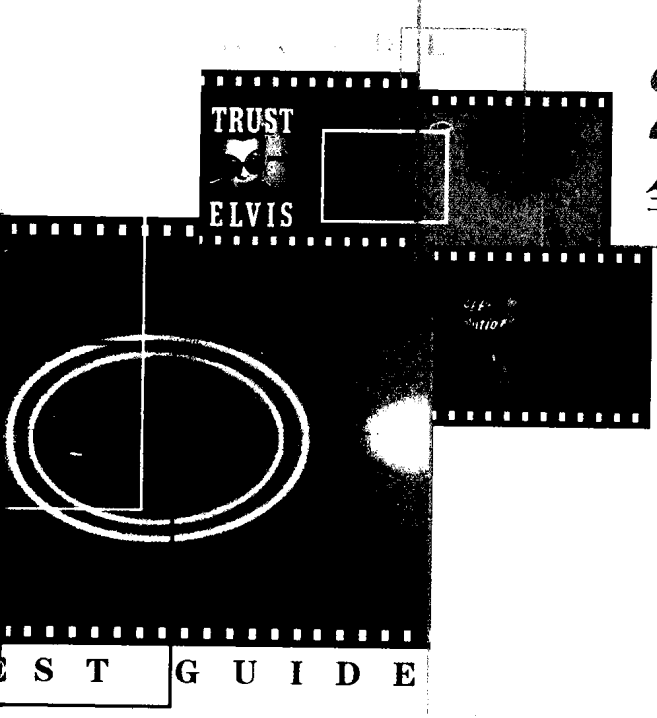
2009年
全国硕士研究生入学统一考试
计算机学科联考
全程辅导

拥有本书 = ***最新大纲*** + 考纲解析
+ 全真试题 + 权威预测 + 复试指南

一站式的复习宝典助您金榜题名

東南大學出版社

高懿 主编
魏坤 副主编



2009年
全国硕士研究生入学统一考试
计算机学科联考
全程辅导

EST GUIDE

东南大学出版社

作者简介

高懿，中国计算机学会会员，上海交通大学软件学院硕士研究生，研究方向：电子商务与管理信息系统。本科就读于南京大学国家理科基地班，曾担任数个国内最大型考研、计算机等级考试网络讨论版版主，毕业后曾任职于新东方、未来路等大型教育机构，辅导各类学生上万名，主编出版计算机辅导专著：《全国计算机等级考试四级应试指南》、《2008年全国硕士研究生入学统一考试计算机专业课辅导全书》。

魏坤，上海交通大学博士研究生，控制理论与控制工程专业。参加过重大国防预研项目，2010上海世博会安全项目和国家863计划多项，主要负责项目中的算法和软件的设计任务，发表学术论文多篇，在计算机理论和应用方面有着许多独特的理解和经验。长期从事计算机等级考试和考研辅导工作，有着丰富的计算机学科辅导经验。

图书在版编目(CIP)数据

2009年全国硕士研究生入学统一考试计算机学科联考全程辅导/高懿主编．—2版．—南京：东南大学出版社，2008.8(2008.10重印)

ISBN 978-7-5641-1343-8

Ⅰ.2… Ⅱ.高… Ⅲ.电子计算机—研究生—入学考试—自学参考资料 Ⅳ.TP3

中国版本图书馆CIP数据核字(2008)第126788号

2009年全国硕士研究生入学统一考试计算机学科联考全程辅导

主　　编：高　懿
策划编辑：张　煦
文字编辑：王小然
装帧设计：王　玥
出 版 人：江　汉
出版发行：东南大学出版社
社　　址：江苏省南京市四牌楼2号(210096)
经　　销：江苏省新华书店
印　　刷：南京玉河印刷厂
版　　次：2008年8月第2版　2008年10月第2次印刷
开　　本：787 mm×1092 mm　1/16
印　　张：29.25
字　　数：748千字
ISBN　978-7-5641-1343-8/TP·221
定　　价：48.00元

前言

去年的今天，本书的前身《2008全国硕士研究生入学统一考试计算机专业课辅导全书》正在最后的修改中，当时我的心情和现在一样忐忑，因为深感肩上的担子特别重，肩负着众多考生朋友们的期待。

该书出版后，我的邮箱几乎每天都可以受到来自四面八方的邮件，这些数以千计的邮件中，不论是同学们的鼓励，或者是指出不足之处，抑或是同行或者老师们的建议，都让我和我的团队非常的感动，真的很感谢大家对我们的帮助，这是我们继续前进的动力。

去年我们编写这本书的目的就是为了给全国所有立志于报考计算机学科研究生的同学们提供一个较为统一与规范的复习思路与方法，令人欣慰的是，我们的思路与做法与教育部考试中心的改革思路不谋而合。2008年4月，教育部正式发函各单位，根据文件精神，2009年全国硕士研究生入学统一考试计算机科学与技术专业的初试科目中，专业课调整为计算机学科专业基础综合，卷面满分值为150分，考试时间为180分钟，全国统一命题。计算机学科专业基础综合的考试内容包括：数据结构、计算机组成原理、操作系统和计算机网络，重点考查考生的基础知识、基本理论和分析问题解决问题的能力，考试内容及结构在大纲中确定。

在从官方渠道收到这份通知的第一天开始，我们就深感责任重大，立即开始不分昼夜地忙碌起来，因为需要花很大工作量来调整全书的章节，对部分章节还要增补许多习题与解答，难度可以想见，但是为了确保全书的质量，我们仍然坚持几易其稿，经过多次的讨论，最终定稿的这本书有如下几个特点：

1. 定位准确：全国第一本严格按照考研统一大纲编写的辅导读物，不但做到形式上与大纲严格对应，并且在其中章节写作时也非常小心与注意，尽量在提法与解答上按照高等教育出版社出版的权威教材的提法来处理，同时兼顾各大名校的出题习惯，力求使全书易读、易懂。

2. 实用性强：每一章分别按教育部颁发的新大纲规定，沿考点归纳、重点提示两条理论线索来梳理内容，同时包含权威预测、真题解析两部分，主要通过优质而有代表性的试题的训练，使大家能够迅速提高自己的应试技巧和能力，全书边讲边练，通过对知识逻辑结构的完善和题目的强化两方面的共同努力，使广大考生应试能力得到实质性的提高。

3. 内容新颖：由于计算机学科本身知识更新快、理论研究发展迅速的特点，本书为了适应时代需要，贴近大纲，选题力求跟上时代步伐，权威预测试题大多是教育部权威教材最新版本中的最具代表性习题，而真题解析则大多精选近两年的名校考研实考真题并配以解析，最大限度地贴近考试。

4. 自学性好：考虑到考研复习的特点，本书为了方便广大考生进行自习，在章节编排的顺序，习题的次序、代表性，解析的准确、易读性上狠下工夫，力求让考生自学方便，顺利考出理想的成绩。

5. 精炼实用：为了减轻考生的实际负担，本书力求习题经典，不以题海取胜，而是强调每一题的经典性，为广大考生提高效率，节约宝贵时间。但是并不是说本书就此压缩了信息量，去年广大读者反映我们的复试部分以及附录都非常实用，给了我们很大的鼓励。今年全国各高校初试统一之后，复试的比重一定会相应加大，所以这次我们同样重视复试，做了非常大的调整与修改，力求实用性与精炼性的统一。

虽然我们一直抱着诚惶诚恐的心态对待这本书，其间所耗费的时间甚至超过了第一版，但是由于种种原因，书中一定还是存在着诸多不足，需要我们继续努力，希望广大读者、老师、同行多提宝贵意见，使我们能够做得更好。如果对本书有任何的意见或建议，请发送邮件到：gaoyi83@yahoo.cn，我们一定会努力改正，争取为广大考生提供更好的帮助！

结稿之际，恰逢2008北京奥运盛典开幕之时，在此预祝广大考生和我国运动健儿一样顽强拼搏，实现梦想！

高懿　魏坤

2008年8月8日 于上海交通大学

目录

第一篇 初试篇

本部分是由2009年全国硕士研究生入学统一考试计算机科学与技术学科联考规定的初试内容，以及正式考纲制定而成，包括数据结构、计算机原理、操作系统、计算机网络。四部分内容有不同权重，前两门所占权重要稍高些，采用45、45、35、25的考分比例。

计算机学科专业基础综合考试考查目标要求考生比较系统地掌握上述专业基础课程的概念、基本原理和方法，能够运用所学的基本原理和基本方法分析、判断和解决有关理论问题和实际问题。

本篇严格按照新考纲编写，选择高等教育出版社出版的“十一五”国家级规划教材为蓝本，以教材习题难度为基准线，结合各高校历年真题与解析，力求准确到位。希望考生认真对待每道试题，只要强化训练，吃准重点，一定可以取得良好的成绩！

第一章 数据结构

一、考点归纳

◇ 基本能力要求

“数据结构”是计算机各专业的专业基础课，要求考生具有基本的专业理论基础及程序设计能力。在考查基本概念、基本知识、基本方法的基础上，注重考查学生运用基本知识来分析和解决实际问题的能力，注重考查利用各种数据结构来设计算法和程序的能力。具体有：

(1) 要求考生全面系统地掌握数据结构与算法的基本概念、数据的逻辑结构和存储结构及操作算法，并能灵活运用。

(2) 能够利用数据结构和算法的基本知识，为应用问题设计有效的数据结构和算法；能够分析算法的复杂性。

(3) 要求考生能够用一种程序设计语言描述数据结构和算法。

◇ 考查目标

1. 理解数据结构的基本概念；掌握数据的逻辑结构、存储结构及其差异，以及各种基本操作的实现。

2. 掌握基本的数据处理原理和方法的基础上，能够对算法进行设计与分析。

3. 能够选择合适的数据结构和方法进行问题求解。

◇ 主要考核内容

1. 线性表

(1) 线性表的定义和基本操作

(2) 线性表的实现

①顺序存储结构

②链式存储结构

③线性表的应用

2. 栈、队列和数组

(1) 栈和队列的基本概念

(2) 栈和队列的顺序存储结构

(3) 栈和队列的链式存储结构

(4) 栈和队列的应用
(5) 特殊矩阵的压缩存储
3. 树与二叉树
(1) 树的概念
(2) 二叉树
①二叉树的定义及其主要特征
②二叉树的顺序存储结构和链式存储结构
③二叉树的遍历
④线索二叉树的基本概念和构造
⑤二叉排序树
⑥平衡二叉树
(3) 树、森林
①书的存储结构
②森林与二叉树的转换
③树和森林的遍历
(4) 树的应用
①等价类问题
②哈夫曼(Huffman)树和哈夫曼编码
4. 图
(1) 图的概念
(2) 图的存储及基本操作
①邻接矩阵法
②邻接表法
(3)图的遍历
①深度优先搜索
②广度优先搜索
(4) 图的基本应用及其复杂度分析
①最小(代价)生成树
②最短路径
③拓扑排序
④关键路径
5. 查找
(1) 查找的基本概念
(2) 顺序查找法
(3) 折半查找法
(4) B-树
(5) 散列(Hash)表及其查找
(6) 查找算法的分析及应用
6. 内部排序
(1) 排序的基本概念

(2) 插入排序
①直接插入排序
②折半插入排序
(3) 气泡排序(bubble sort)
(4) 简单选择排序
(5) 希尔排序(shell sort)
(6) 快速排序
(7) 堆排序
(8) 二路归并排序(merge sort)
(9) 基数排序
(10) 各种内部排序算法的比较
(11) 内部排序算法的应用

二、重点提示

1. 什么是数据结构?

数据是描述客观事物的数字、字符以及所有能直接输入到计算机中并被计算机程序处理的符号的集合。数据元素是数据的基本单位,即是数据集合中的一个元素。可由若干个小数据项组成。数据对象是具有相同性质的数据元素的集合,是数据集合的一个子集。通常,一个数据对象中的数据元素不是孤立的,而是彼此之间存在着一定的联系,这种联系或者说数据的组织方式就是数据结构。

数据对象中数据元素之间的联系需要在对数据进行存储和加工中反映出来,因此,数据结构概念一般包括三方面的内容:数据之间的逻辑关系、数据在计算机中的存储方式以及在这些数据上定义的运算的集合。

2. 数据的逻辑结构和存储结构是什么?

数据的逻辑结构只抽象地反映数据元素之间的逻辑关系,即数据间抽象的相互关系。它与数据的存储无关,是独立于计算机的。数据的逻辑结构分为线性结构和非线性结构两大类。线性结构的逻辑特征是:有且仅有一个开始结点和一个终端结点,并且所有的结点都最多有一个直接前驱和一个直接后继。线性表就是一个典型的线性结构。非线性结构的逻辑特征是:一个结点可能有多个直接前驱和直接后继。树、图等都是非线性结构。

数据的存储结构是数据的逻辑结构在计算机存储器里的实现(亦称为映象)。它是依赖于计算机的,并有四种基本的存储映象方法:

(1) 顺序存储方法

该方法是把逻辑上相邻的结点存储在物理位置上相邻的存储单元内,结点间的逻辑关系由存储单元的邻接关系来体现。顺序存储方法主要用于线性的数据结构,非线性的数据结构也可以通过某种线性化方法来实现顺序存储。

(2) 链接存储方法

在链接存储方法中,逻辑上相邻的结点在物理位置上未必相邻,结点间的逻辑关系是由附加的指针字段表示的。

(3) 索引存储方法

该方法通常是在存储结点信息的同时,还建立一个附加的索引表,索引表中的每一项

称为索引项，索引项的一般形式是：关键字，地址。关键字是能唯一标识一个结点的那些数据项。

(4) 散列存储方法

在散列存储方法中，结点的存储地址是根据结点的关键字值直接计算出来的。上述四种基本的存储方法也可以组合起来对数据结构进行存储映象。

3. 算法的特征有哪些?

算法就是一种解题方法，是由若干条指令组成的有穷序列。算法的描述可以采用自然语言、数学语言、约定的符号语言以及图解等方式。

算法必须具有以下特征：

(1) 有穷性：一个算法必须在执行有穷步后结束；

(2) 确定性：算法的每一步必须是确切地定义的，无二义性；

(3) 可行性：算法中的所有待实现的运算必须在原则上能够由人使用笔和纸在做有穷次运算后完成；

(4) 输入：一个算法具有 0 个或多个输入的外界量，它们是算法开始前对算法最初给出的量；

(5) 输出：一个算法至少产生一个输出，它们是与输入有某种关系的量。

4. 算法分析主要注意算法的哪些方面?

求解同一个问题可以有多种不同的算法，评价一个算法的优劣除了正确性和简明性外，主要考虑两点：一是执行算法所耗费的时间；二是执行算法所耗费的存储空间，特别是辅助存储空间的耗费。就这两者而言，前者显得比后者更为重要，在数据结构中往往更注重对算法执行时间的分析。

一个算法所耗费的时间是该算法中每条语句的执行时间之和，而每条语句的执行时间是该语句执行次数(频度)与该语句一次执行所需时间的乘积。如果假定每条语句一次执行所需的时间均为单位时间，则一个算法的时间耗费就是该算法中所有语句的频度之和。

5. 简述线性表的存储方式

在实际应用中，线性表一般可用顺序存储结构和链式存储结构两种实现方法。有数组、栈、队列、串、链表等形式，其中栈、队列和串又称为受限的线性结构。这些形式在时间效率、空间效率、操作效率、可靠性等方面有着各自的特点。

(1) 线性表的顺序存储

线性表的顺序存储是最简单的存储方式。用顺序存储方法存储的线性表简称为顺序表(Sequential List)。

通常在设计前估算出所需的存储空间，定义一个足够大的数组，从数组的第一个元素开始，将线性表的结点依次存储在数组中，用数组元素的顺序存储来体现线性表中结点的先后次序关系。数组存储线性表的最大优点是能直接、随机地访问线性表中的任一结点，对表中任一结点都可在 O(1) 时间内直接存取，不会出现因为结点位置的不同而耗费的时间不同的情况。缺点主要有两个：一是数组大小通常固定，会与线性表可以任意增减结点的要求矛盾；二是对线性表的结点进行插、删等操作时必须移动数组中的其他元素，很不简便，效率也不高。

(2) 线性表的链接存储

线性表链接存储是用链表存储线性表。链接方式存储的线性表简称为链表(Linked

List)。最简单的方法是用单链表。

链表中结点的逻辑次序和物理次序不一定相同，即逻辑上相邻未必在物理上相邻，结点之间的相对位置由链表中的指针域指示，而结点在存储器中的存储位置是随意的。链表的每个表元除要存储线性结点的信息外，还要有一个成分(链域)存储其后继或前驱结点的指针。

用链表存储线性表的优点是利用线性表的每个表元的链域指针就能完成插或删的操作，不需移动任何表元。其缺点也主要有两条：一是每个表元增加了一个后继指针成分，要花费更多的存储空间；二是不便随机地直接访问线性表的任一结点。

6. 线性表有哪些基本操作？

常见的线性表运算有：

(1) 表的初始化

Init (L)，即构造一个空线性表 L。

(2) 求表长

Length(L)，即求线性表 L 中的结点数(元素个数)。

(3) 取任意结点

Node(L, i)，取线性表 L 中的第 i 个结点(1≤i≤Length(L))。

(4) 查找结点

Locate (L, x)，在 L 中查找值为 x 的结点，返回该结点在 L 中的位置。若 L 中有多个结点的值和 x 相同，则返回首次找到的结点位置；若 L 中没有结点的值为 x，则返回一个特殊值表示查找失败。

(5) 插入结点

Imsert(L, x, i)，在线性表 L 的第 i 个($1\leqslant i\leqslant n+1$，$n$ 为原表长)位置上插入一个值为 x 的新结点，表长增加 1。

(6) 删除结点

Delete(L, i)，删除线性表 L 的第 i 个($1\leqslant i\leqslant n$，n 为原表长)结点。删除后表 L 的长度减 1。

以上的运算是逻辑结构上定义的运算(其中查找、插入、删除是基本运算)，只给出了运算的功能("做什么")，至于实现问题("如何做")则需在存储结构确定后才考虑。在实际应用中，自然可以针对具体情况对上述运算进行扩充改进。

7. 给出单链表、循环链表和双向链表的概念和结点定义

(1) 单链表

一般的链表的结点结构如图 1.1 所示：

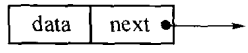


图 1.1 (单)链表结点图

其中 data 域是数据域，next 域是存放结点的直接后继的地址(位置)的指针域(链域)。链表是通过每个结点的链域将线性表的 n 个结点按其逻辑顺序链接在一起的。

每个结点只有一个链域的链表称为单链表(Single Linked List)。用箭头来表示链域中的指针，"^"表示空指针，则线性表(a_1, a_2, …, a_n)的单链表可表示为图 1.2 形式。

单链表的结点定义如下：

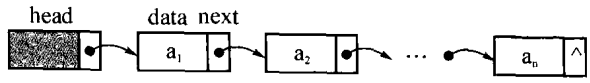


图 1.2 非空单链表图

```
typedef char DataType;          //假设结点的数据域类型为整型
typedef struct node
{ //结点类型定义
  DataType data;                //结点的数据域
  struct node * next;           //结点的指针域
} ListNode;
```

typedef ListNode * LinkList;//LinkList 和 ListNode * 是不同名字的同一个指针类型，

```
//命名的不同是为了概念上更明确
ListNode * p;                   //指向某一结点的指针
LinkList head;                  //单链表的头指针
```

(2) 循环链表

首尾相连的链表即是循环链表，有两种：

单循环链表：在单链表中，将终端结点的指针域 NULL 改为指向表头结点或开始结点即可；多重链的循环链表：将表中结点链在多个环上。

循环链表的特点是无须增加存储量，仅对表的链接方式稍作改变，即可使得表处理更加方便灵活。

不过，循环链表中没有 NULL 指针。涉及遍历操作时，其终止条件就不再是像非循环链表那样判别 p 或 p->next 是否为空，而是判别它们是否等于某一指定指针，如头指针或尾指针等。例如判断一个带头结点(head)的单循环链表(如图 1.3 所示)是否为空的条件是：head==head->next。

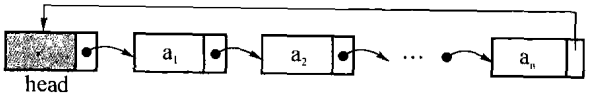


图 1.3a 非空单循环链表图

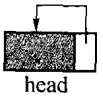


图 1.3b 空单循环链表

在单链表中，从一已知结点出发，只能访问到该结点及其后续结点，无法找到该结点之前的其他结点。而在单循环链表中，从任一结点出发都可访问到表中所有结点，这一优点使某些运算在单循环链表上易于实现。

(3) 双向链表

单链表的每个结点再增加一个指向其前趋的指针域 prior，这样形成的链表有两条不同方向的链，称之为双(向)链表，其结点结构可图示如下：

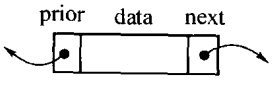


图 1.4 双(向)链表结点图

与单链表相同，双链表一般也由头指针 head 唯一确定。

双链表每一结点均有：数据域(data)，存储数据；左链域(prior)，指向前趋结点的指针；右链域(next)，指向后继的指针。这是一种对称结构(既有前趋势，又有后继)。

一般结点可以声明为(供参考)：

```
typedef int datatype;
typedef struct dnode
{
    datatype data;
    struct dnode * prior, * next;
} dlinklist;
dlinklist * head;
```

其中结构对称性表现为：设双链表中的一个结点为 * p，则有 p－＞prior－＞next＝p－＞next-＞prior＝p，结点 * p 的存储位置既存放在其前趋结点(* p－＞prior)的后继指针域中，也存放在其后继结点(* p－＞next)的前趋指针域中。

构造出的双链表可以表示为如图 1.5：

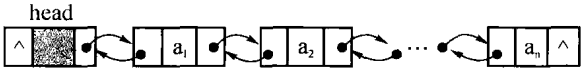


图 1.5　非空双向链表图

8. 如何选择线性表的存储实现方式？

总结如表 1.1 所示。

表 1.1

		顺序表	链表
基于存储空间考虑	内存分配方式	静态分配。程序执行之前必须明确规定存储规模。若线性表长度 n 变化较大，则存储规模难于预先确定估计过大将造成空间浪费，估计太小又将使空间溢出机会增多	动态分配只要内存空间尚有空闲，就不会产生溢出。因此，当线性表的长度变化较大，难以估计其存储规模时，以采用动态链表作为存储结构为好
	存储密度	为 1。当线性表的长度变化不大，易于事先确定其大小时，为了节约存储空间，宜采用顺序表作为存储结构	＜1
基于运算时间考虑	存取方法	随机存取结构，对表中任一结点都可在 O(1)时间内直接取得；操作主要是进行查找，很少做插入和删除操作时，采用顺序表做存储结构为宜	顺序存取结构，链表中的结点，需从头指针起顺着链扫描才能取得
	插入删除操作	在顺序表中进行插入和删除，平均要移动表中近一半的结点，尤其是当每个结点的信息量较大时，移动结点的时间开销就相当可观	在链表中的任何位置上进行插入和删除，都只需要修改指针。对于频繁进行插入和删除的线性表，宜采用链表做存储结构。若表的插入和删除主要发生在表的首尾两端，则采用尾指针表示的单循环链表为宜

9. 栈有哪些基本运算？

栈(Stack)是仅在表的一端进行插入和删除运算的线性表。

通常称插入、删除的这一端为栈顶(Top),另一端称为栈底(Bottom)。当表中没有元素时称为空栈。栈为后进先出(Last In First Out)的线性表,修改是按后进先出的原则进行,故又简称为 LIFO 表。每次删除(退栈)的总是当前栈中"最新"的元素,即最后插入(进栈)的元素,而最先插入的是被放在栈的底部,要到最后才能删除。

基本运算有:

(1) 栈初始化

Init(S),构造一个空栈 S。

(2) 判断栈是否为空

IsEmpty(S),若 S 为空栈,则返回 TRUE,否则返回 FALSE。

(3) 判断栈是否满

IsFull(S),若 S 为满栈,则返回 TRUE,否则返回 FALSE。该运算只适用于栈的顺序存储结构。

(4) 进栈

Push(S, x),若栈 S 不满,则将元素 x 插入 S 的栈顶。

(5) 退栈

Pop(S),若栈 S 非空,则将 S 的栈顶元素删去,并返回该元素。

(6) 取栈顶元素

Top(S),若栈 S 非空,则返回栈顶元素,但不改变栈的状态。

10. 顺序栈是如何实现的?

栈的顺序存储结构简称为顺序栈,它是运算受限的顺序表。

顺序栈可定义如下:

```
comst int nSTACKSIZE =256          //假定预分配的栈空间最多为 256 个元素
typedef int DataType;              //假定栈元素的数据类型为字符
typedef struct _SeqStack
{
    DataType  data[nSTACKSIZE];
    int   top;
} SeqStack;
```

顺序栈中的元素用向量(数组)存放。栈底位置是固定不变的,可设置在向量两端的任意一个端点。栈顶位置是随着进栈和退栈操作而变化的,用一个整型量 top(通常称 top 为栈顶指针)来指示当前栈顶位置。

顺序栈的基本操作实现如下:

1) 初始化顺序栈

```
void Init(SeqStack * S)
{  //将顺序栈置空
    S->top=-1;
}
```

2) 判断顺序栈是否空

```
int IsEmpty(SeqStack *S)
{
```