02142《数据结构导论》

考前资料

考试题型及分值分布

类别	题型	题量	分值/题	总分
选择题	单项选择题	15 题	2 分	30 分
非选择题	填空题	13 题	2 分	26 分
	应用题	5 题	6 分	30 分
	算法设计题	2 题	7 分	14 分

注：本资料题型根据考试大纲或历年真题进行预测，仅供复习参考

第一章　概论

★考点 1：数据结构引言（填空）

（1）数据结构是计算机组织数据和存储数据的方式，数据及数据的组织方式称为数据的逻辑结构。合理的数据结构可降低程序设计的复杂性，提高程序执行的效率。

（2）1976 年瑞士计算机科学家尼克劳斯·维尔特曾提出一个著名公式：算法+数据结构=程序。

★考点 2：数据、数据元素和数据项（单选、填空）

（1）数据是所有被计算机存储、处理的对象。

（2）数据元素是数据的基本单位，在程序中作为一个整体而加以考虑和处理。

（3）数据元素由数据项组成。在数据库中数据项又称为字段或域。它是数据的不可分割的最小标识单位。

数据结构是相互之间存在一种或多种特定关系的数据元素的集合。它包括数据的逻辑结构、数据的存储结构和数据的基本运算。

考点 3：数据的逻辑结构（单选）

（1）数据的逻辑结构是指数据元素之间的逻辑关系，其与数据元素本身的形式、内容、相对位置、个数无关。

（2）基本的逻辑结构有：集合、线性结构、树形结构和图结构。

★考点 4：数据的存储结构（填空）

（1）数据的逻辑结构在计算机中的实现称为数据的存储结构（或物理结构）。一般情况下，一个存储结构包括存储数据元素和数据元素之间的关联方式两个部分。

（2）表示数据元素之间的关联方式主要有顺序存储方式和链式存储方式。

★★★考点 5：时间复杂度（单选、算法设计）

（1）T(n)=O(f(n))称为算法的渐进时间复杂度，简称时间复杂度。

（2）时间复杂度常见的阶数有常数阶 O(1)（即算法的时间复杂度与输入规模 n 无关）；对数阶 $O(\log_2 n)$、线性阶 O(n)、多项式阶 $O(n^C)$和指数阶 $O(C^n)$，C 为大于 1 的正整数。常见的多项式阶有 $O(n^2)$和 $O(n^3)$，常见的指数阶有 $O(2^n)$。

时间复杂度的阶数大小：$O(1)<O(\log_2 n)<O(n)<O(n\log_2 n)<O(n^2)<O(n^3)<O(n^k)<O(2^n)$

（3）最坏时间复杂度是指，对相同输入数据量的不同输入数据，算法时间用量的最大值。平均时间复杂度是指，对所有相同输入数据量的各种不同输入数据，算法时间用量的平均值。

★★考点 6：空间复杂度（填空、算法设计）

（1）空间复杂度是对一个算法在运行过程中临时占用存储空间大小的量度。一个算法在执行期间所需要的存储空间量应包括：程序代码所占用的空间；输入数据所占用的空间；辅助变量所占用的空间。

（2）在估算算法空间复杂度时，一般只需要分析辅助变量所占用的空间。

第二章　线性表

考点 1：线性表的定义和特征（填空）

（1）线性表是一种线性结构，它是由 n（n≥0）个数据元素组成的有穷序列，数据元素又称结点。

（2）线性表中结点具有一对一的关系，如果结点数不为零，则除起始结点没有直接前驱外，其他每个结点有且仅有一个直接前驱；除终端结点没有直接后继外，其他每个结点有且仅有一个直接后继。

★考点 2：线性表顺序存储的类型定义（单选、填空）

（1）线性表顺序存储的方法：将表中的结点依次存放在计算机内存中一组连续的存储单元中，数据元素在线性表中的邻接关系决定它们在存储空间中的存储位置，即逻辑结构中相邻的结点其存储位置也相邻。

（2）用顺序存储实现的线性表称为顺序表。一般使用数组来表示顺序表。

★考点 3：顺序表实现算法的分析（单选、填空）

（1）在线性表的基本操作中，最频繁的操作是数据元素的比较和移动。在分析线性表的顺序表实现算法时，一个重要指标就是数据元素的比较和移动的次数。

（2）一般情况下元素比较和移动的次数为 n-i+1 次，插入算法的平均移动次数约为 n/2，其时间复杂度是 O(n)，在顺序表上做插入运算平均要移动表中一半的结点。。

★考点 4：单链表的类型定义（单选、填空）

（1）指针表示数据元素之间的逻辑关系。data 部分称为数据域，用于存储线性表的一个数据元素，next 部分称为指针域或链域，用于存放一个指针，该指针指向本结点所含数据元素的直接后继结点。

（2）head 称为头指针变量，该变量的值是指向单链表的第一个结点的指针。我们通常用头指针来表示一个单链表。

（3）链表中第一个数据元素结点称为链表的首结点，链表中最后一个数据元素结点称为尾结点或终端结点。尾结点指针域的值 NULL 称为空指针，它不指向任何结点，表示链表结束。

（4）在单链表的第一个结点之前增设一个类型相同的结点，称之为头结点，其他结点称为表结点。

★★考点 5：线性表的基本运算在单链表上的实现（单选、填空）

（1）在单链表存储结构中，线性表的表长等于单链表中数据元素的结点个数，即除了头结点以外的结点的个数。

（2）工作指针 p->next 为 NULL 时，说明已经走到了表的尾部。在单链表中，释放已移出结点 p 的空间使用语句 free(p)。

设单链表中指针 p 指向结点 A，要删除 A 之后的结点（若存在），则修改指针的操作为 p->next=p->next->next。（单选）

设 r 指向单链表的最后一个结点，要在最后一个结点之后插入 s 所指的结点，需执行的语句序列是 r->next=s;r=s;r->next=NULL。（单选）

★考点 6：其他链表（单选、填空）

（1）如果让最后一个结点的指针域指向第一个结点可以构成循环链表。

（2）双向循环链表结点结构为 prior、data、next，头结点的 prior 指向最后一个结点，最后一个结点的 next 指向头结点。双向循环链表的对称性可以表示为 p=p->prior->next=p->next->prior。

考点 7：序实现与链接实现的比较（单选）

从整体考虑，顺序表要预分配存储空间，如果预先分配得过大，将造成浪费，若分配得过小，又将发生上溢；单链表不需要预先分配空间，只要内存空间没有耗尽，单链表中的结点个数就没有限制。

第三章　栈、队列和数组

考点 1：栈的基本概念（填空）

（1）栈是运算受限的线性表，这种线性表上的插入和删除运算限定在表的某一端进行。允许进行插入和删除的一端称为栈顶，另一端称为栈底。不含任何数据元素的栈称为空栈。处于栈顶位置的数据元素称为栈顶元素。

（2）栈的修改原则是后进先出，因此，栈又称为后进先出线性表，简称后进先出表。栈的插入和删除运算分别称为进栈和出栈。

★★考点 2：栈的顺序实现（单选、应用）

（1）栈的顺序实现称为顺序栈。通常用一个一维数组和一个记录栈顶位置的变量来实现栈的顺序存储。

（2）顺序栈用类 C 语言定义中：maxsize 为顺序栈的容量。data[maxsize]为存储栈中数据元素的数组。top 为标志栈顶位置的变量，常用整型表示，范围为 0～(maxsize-1)。

在一个具有 n 个单元的顺序栈中，假定以地址低端（即 0 单元）作为栈底，以 top 为栈顶指针，当栈未满时进行进栈操作，此时 top++。（单选）

★★★考点 3：栈的链接实现（单选、填空、应用）

（1）栈的链接实现称为链栈，链栈可以用带头结点的单链表来实现。

（2）栈初始化时，生成一个结点，将该结点的 next 域设置为 NULL。

设栈的输入序列为 1，2，3，若输出的第一个元素为 3，则第二个输出的元素为 2。（填空）

设有一个链栈的输入序列为 A、B、C，当输出序列分别为 ABC 和 BCA 时，请写出对应的进栈和出栈过程。（应用）

考点 4：栈的简单应用和递归（单选）

如果在一个函数或数据结构的定义中又应用了它自身（作为定义项之一），那么这个函数或数据结构称为是递归定义的，简称递归的。

栈可以实现函数的嵌套调用和程序递归的处理。（单选）

★考点 5：队列的基本概念（单选、填空）

（1）队列是有限个同类型数据元素的线性序列，是一种先进先出的线性表。排队的规则是不允许“插队”。

（2）在操作系统中，为了保持多个进程 P1、P2、P3 和 P4 按某种次序依次执行，需要一个队列来实现这个过程。

★★★考点 6：队列的顺序实现（单选、填空、算法设计）

（1）顺序存储实现的队列称为顺序队列，顺序队列结构类型中有三个域：data、front 和 rear。其中 data 为一维数组，存储队列中数据元素。front 和 rear 定义为整型变量，实际取值范围是 0～(maxsize-1)。

（2）顺序队列需要预先定义队列的容量，一般将数组的首尾相接，形成循环队列，这样可以解决“假

溢出”问题。

（3）队列常考语句

①出队列操作使用的赋值语句是 SQ.front=SQ.front+1。

②循环队列满条件为(CQ.rear+1)%maxsize==CQ.front。

③循环队列空条件为 CQ.rear==CQ.front。

④判断队列空的条件为 front==rear。

⑤循环队列 SQ 为空的条件是 SQ.rear==SQ.front。

⑥设指针变量 front 表示链队列的队头指针，指针变量 rear 表示链队列的队尾指针，指针变量 s 指向将要入队列的结点 X，则入队列的操作序列为 rear->next=s;rear=s;。

★考点 7：队列的链接实现（填空）

（1）队列的链接实现实际上是使用一个带有头结点的单链表来表示队列，称为链队列。

（2）头指针指向链表的头结点，单链表的头结点的 next 域指向队列首结点，尾指针指向队列尾结点。

考点 8：数组的逻辑结构和基本运算（填空）

（1）若一维数组中的数据元素又是一维数组结构，则该数组称为二维数组。

（2）若一维数组中的元素又是一个二维数组结构，则称作三维数组。

★考点 9：数组的存储结构（单选）

由于下标从 0 开始，元素 a[i][j]之前已经有 i 行元素，每行有 n 个元素，在第 i 行，有 j+1 个元素，总共有 n*i+j+1 个元素，第一个元素与 a[i][j]相差 n*i+j+1-1 个位置，故 a[i][j]的位置为：loc[i，j] = loc[0，0] + (n*i+j)*k。

一个数组的第一个元素的存储地址是 100，每个元素占 2 存储单元，则第 5 个元素的存储地址是 108。（单选）

★★★考点 10：矩阵的压缩存储（单选、填空、应用）

（1）如果值相同的元素或者零元素在矩阵中的分布有一定规律，称此类矩阵为特殊矩阵。矩阵的非零元素个数很少的矩阵称为稀疏矩阵。稀疏矩阵可以采用三元组表示法进行压缩存储。

（2）对称矩阵有近一半的元素可以通过其对称元素获得，为每一对对称元素只分配一个存储单元，则可将 n^2 个元素压缩存储到含有 n(n+1)/2 个元素的一维数组中。

（3）以主对角线为界的上（下）半部分是一个固定的值 c 或零，这样的矩阵叫做下（上）三角矩阵。

把特殊矩阵 A[10][10]的下三角矩阵压缩存储到一个一维数组 M 中，则 A 中元素 a[4][3]在 M 中所对应的下标位置是 13。（单选）

第四章　树和二叉树

★考点 1：树的概念（单选、填空）

树是 n（n≥0）个结点的有限集合，一棵树满足以下两个条件：

（1）当 n=0 时，称为空树；

（2）当 n>0 时，有且仅有一个称为根的结点，除根结点外，其余结点分为 m（m≥0）个互不相交的非空集合 T_1，T_2，…，T_m，这些集合中的每一个都是一棵树，称为根的子树。

森林是 m（m≥0）棵互不相交的树的集合。树的每个结点的子树是森林。删除一个非空树的根结点，它的子树便构成森林。

★★考点 2：树的相关术语（单选、填空）

（1）树上任一结点所拥有的子树的数目称为该结点的度。

（2）度为 0 的结点称为叶子或终端结点。

（3）一棵树中所有结点的度的最大值称为该树的度。

★★考点 3：二叉树的基本概念（单选、填空、应用）

（1）二叉树是 n（n≥0）个元素的有限集合，该集合或者为空，或者由一个根及两棵互不相交的左子树和右子树组成，其中左子树和右子树也均为二叉树。

（2）二叉树有五种基本形态：①空二叉树；②左右子树均为空的二叉树；③右子树为空的二叉树；④左子树为空的二叉树；⑤左、右子树都非空的二叉树。

★★考点 4：二叉树的性质（单选、填空）

（1）二叉树第 i（i≥1）层上至多有 2^{i-1} 个结点。

（2）深度为 k（k≥1）的二叉树至多有 2^k-1 个结点。

（3）对任何一棵二叉树，若度数为 0 的结点（叶结点）个数为 n_0，度数为 2 的结点个数为 n_2，则 $n_0=n_2+1$。

（4）深度为 k（k≥1）且有 2^k-1 个结点的二叉树称为满二叉树。

设一棵完全二叉树中有 65 个结点，则该完全二叉树的深度为 7。（单选）

★考点 5：二叉树的顺序存储结构（填空）

（1）二叉树的顺序存储结构可以用一维数组来实现。

（2）对于任何完全二叉树来说，可以采用以编号作为数组的下标的方法将结点存入一维数组中，也就是将编号为 i 的结点存入一维数组的以 i 为下标的数组元素中。

★★★考点 6：二叉树遍历的递归实现（单选、填空、应用、算法设计）

（1）树的遍历有三种，为先序、中序和后序遍历。

（2）先序遍历：①访问根结点；②先序遍历左子树；③先序遍历右子树。

（3）中序遍历：①中序遍历左子树；②访问根结点；③中序遍历右子树。

（4）后序遍历：①后序遍历左子树；②后序遍历右子树；③访问根结点。

先序遍历与中序遍历结果相同的二叉树，所有结点只有右子树。（单选）

二叉树的中序序列中，结点 P 排在结点 Q 之前的条件是：在二叉树中 P 在 Q 的左边。（单选）

由先序序列的第一个结点可以确定这棵树的根结点。（填空）

考点 7：二叉树的层次遍历（单选）

二叉树的层次遍历，是指从二叉树的根结点的这一层开始，逐层向下遍历，在每一层上按从左到右的顺序对结点逐个访问。

★★考点 8：树、森林与二叉树的关系（填空、应用）

（1）任何一棵树可唯一地与一棵二叉树对应。相应地，一棵二叉树也唯一地对应一棵树，即树与二叉树可以互相转换。

（2）森林转换成二叉树的方法如下：①将每棵树转换成相应的二叉树；②将①中得到的各棵二叉树的根结点看作是兄弟连接起来。

设森林 F 中有三棵树，第一、第二、第三棵树的结点个数分别为 M1、M2 和 M3。与森林 F 对应的二叉树根结点的右子树上的结点个数是 M2+M3。（填空）

考点 9：树的分类与判定（填空）

用于描述分类过程的二叉树称为判定树。一棵判定树描述了一种分类方法，因而由给定的判定树很容易写出相应的分类算法。

★考点 10：哈夫曼树与哈夫曼算法（单选）

初始森林中共有 n 棵二叉树，则最终求得的哈夫曼树中共有 2n-1 个结点，其中 n 个叶结点是初始森林中的 n 个结点，并且哈夫曼树中没有度数为 1 的分支结点。

第五章　图

★★★考点 1：图的定义和术语（单选、填空、应用）

（1）任何两点之间都有边的无向图称为无向完全图。一个具有 n 个顶点的无向完全图的边数为 n(n-1)/2。任何两点之间都有弧的有向图称为有向完全图。一个具有 n 个顶点的有向完全图的弧数为 n(n-1)。

（2）无向图中一个顶点的度是指图中与该顶点相邻接的顶点数。如果 G 是一个有向图，则把以顶点 v 为终点的弧的数目称为 v 的入度。

（3）在图中，序列中顶点不重复出现的路径称为简单路径。第一个顶点和最后一个顶点相同的路径称为回路或环。含有 n 个顶点的连通图中任意一条简单路径，其长度最大为 n-1。

设有 10 个顶点的无向图，若它为连通图，则它具有的边数最少为 9。（单选）

根据图的定义，图中顶点的最少数目是 1。（填空）

★★考点 2：邻接矩阵（填空、应用）

（1）邻接矩阵就是用矩阵来描述图中顶点之间的关联关系，无向图的邻接矩阵是一个对称矩阵。

（2）无向带权图的邻接矩阵的建立方法是：首先将矩阵 A 的每个元素都初始化为最大值，然后读入边及权值(i，j，w_ij)，将 A 的相应元素置成 w_{ij}。

★考点 3：邻接表（应用）

（1）邻接表是顺序存储与链式存储相结合的存储方法。

（2）每一个单链表设一个表头结点，每一个表头结点有两个域 vertex 和 firstarc，vertex 用来存放顶点 v_i 的信息，firstarc 用来指向邻接表中的第一个结点。

（3）单链表中每一个结点称为表结点，包括两个域：邻接点域（adjvex）和链域（nextarc）。

★考点 4：连通图的广度优先搜索（应用）

连通图广度优先搜索的基本思想是：从图中某个顶点 v_i 出发，在访问了 v_i 之后依次访问 v_i 的所有邻接点，然后依次从这些邻接点出发按广度优先搜索方法遍历图的其他顶点，重复这一过程，直至所有顶点都被访问到。广度优先搜索遍历类似于树的按层次遍历的过程。

★★考点 5：最小生成树（单选、填空）

（1）对于有 n 个顶点的无向图，所有生成树中都有且仅有 n-1 条边。一个图的最小生成树是指该图的所有生成树中权总和最小的生成树。

（2）求最小生成树有 Prim 方法和 Kruskal 方法。

（3）Dijkstra 算法用于求单源最短路径问题。

任何一个带权的无向连通图的最小生成树有一棵或多棵。（单选）

★★考点 6：拓扑排序（单选、填空）

（1）设图 G 中有 n 个顶点，e 条弧，采用邻接表存储，则拓扑排序算法的时间复杂度为 O(n+e)。

（2）完成拓扑排序的前提条件是 AOV 网中不允许出现回路。

（3）任何一个无环有向图，其全部顶点可以排成一个拓扑序列。

第六章　查找

考点 1：查找表的基本概念

（1）查找表是由同一类型的数据元素构成的集合，它是一种以查找为“核心”，同时包括其他运算的非常灵活的数据结构。

（2）若对查找表只进行前两项操作，则称此类查找表为静态查找表。若在查找过程中，向表中插入不存在的数据元素，或者从表中删除某个数据元素，则称此类表为动态查找表。

考点 2：顺序表上的查找（填空）

（1）静态查找表最简单的实现方法是以顺序表作为存储结构。

（2）顺序查找算法的平均查找长度为(n+1)//2。

★★考点 3：有序表上的查找（单选、应用）

（1）如果顺序表中数据元素是按照键值大小的顺序排列的，则称为有序表。

（2）二分查找算法每进行一次键值与给定值的比较，查找区间的长度至少减小为原来二分之一。

设有序表中的元素为(13，18，24，35，47，50，62)，则在其中利用二分法查找值为 24 的元素需要经过比较的次数是 3。（单选）

考点 4：二叉排序树上的查找（单选）

在二叉排序树上进行查找，若查找成功，则是从根结点出发走了一条从根结点到待查结点的路径；若查找不成功，则是从根结点出发走了一条从根到某个叶子的路径。因此与二分查找类似，关键字比较的次数不超过二叉树的深度。

依次输入键值序列 50，72，43，85，75，20，35，45，65，30，建立对应的二叉排序树以后，查找元素 35 要进行元素间的比较次数为 4。（单选）

★考点 5：二叉排序树的插入（应用）

（1）在二叉排序树上进行插入的原则是：必须要保证插入一个新结点后，仍为一棵二叉排序树。这个结点是查找不成功时查找路径上访问的最后一个结点的左孩子或右孩子。

（2）入键值序列的键值排列顺序不同，建造的二叉排序树的结构也会不同。

考点 6：二叉排序树的查找分析（单选）

二叉排序树上的平均查找长度是介于 O(n)和 O($\log_2 n$)之间的，其查找效率与树的形态有关。二叉排序树的平均查找长度 ASL≤1+$\log_2 n$。

★考点 7：常用散列法（单选）

（1）数字分析法又称数字选择法，其方法是收集所有可能出现的键值，排列在一起，对键值的每一位进行分析，选择分布较均匀的若干位组成散列地址。

（2）除留余数法是一种简单有效且最常用的构造方法，其方法是选择一个不大于散列表长 n 的正整数

p，以键值除以 p 所得的余数作为散列地址。

（3）平方取中法以键值平方的中间几位作为散列地址。

（4）基数转换法将键值看成另一种进制的数再转换成原来进制的数，然后选其中几位作为散列地址。

对于线性表(7，34，55，25，64，46，20，10)进行散列存储时，若散列函数为 H(K)=K%9，则散列地址为 1 的元素个数是 4。（单选）

在散列函数 H(k)=kMODm 中，一般来讲，m 应取素数。（单选）

★★★考点 8：散列表的实现（单选、应用）

（1）用线性探测法解决冲突，可能要探测多个散列地址，这些位置上的键值不一定都是同义词。非同义词之间对同一个散列地址的争夺现象称为“堆积”。

（2）二次探测法的基本思想：生成的后继散列地址不是连续的，而是跳跃式的，以便为后续数据元素留下空间从而减少堆积。二次探测法的缺点是不易探测到整个散列表的所有空间，也就是说，上述后继散列地址可能难以包括散列表的所有存储位置。

（3）多重散列法的优点是不易产生“堆积”，缺点是计算量较大。

（4）公共溢出区法的插入首先在基本表上进行，假如发生冲突，则将同义词存入溢出表。这样，基本表不可能发生“堆积”。

第七章　排序

考点 1：排序的定义（填空）

（1）排序就是将一组对象按照规定的次序重新排列的过程，排序往往是为检索服务的。

（2）相同键值的两个记录在排序前后相对位置的变化情况是排序算法研究中经常关注的一个问题，这个问题称为排序算法的稳定性。

（3）稳定性是排序方法本身的特性，与数据无关。

★考点 2：直接插入排序算法（应用）

（1）直接插入排序是一种简单的排序方法，它的基本思想是依次将每个记录插入到一个已排好序的有序表中去，从而得到一个新的、记录数增加 1 的有序表。

（2）直接插入排序的算法时间复杂度为 $O(n^2)$，若待排序记录的数量很大时，一般不选用直接插入排序。

★★考点 3：冒泡排序（单选、应用）

冒泡排序法是一种交换排序方法，其过程是首先将第一个记录的键值和第二个记录的键值进行比较，若为逆序（即 R[1].key>R[2].key），则将这两个记录交换，然后继续比较第二个和第三个记录的键值。

★考点 4：快速排序（单选、填空）

快速排序方法是不稳定的，就平均时间性能而言，快速排序方法最佳，其时间复杂度为 $O(n\log_2 n)$。但在最坏情况下，即对几乎已是排好序的输入序列，该算法的效率较低，近似于 $O(n^2)$。另外，对于较小的 n 值该算法效果不明显；反之，对较大的 n 值效果较好。

★考点 5：堆排序（单选）

（1）若在输出堆顶的最小键值之后，使得剩余的 n-1 个键值重新建成一个堆，则可得到 n 个键值中的次最小值。如此反复执行，便能得到有序序列，这个过程称之为堆排序。

（2）堆排序在待排序记录较少时不适用，但对记录数很多时是很有效的，对于 n 个记录进行排序所需的平均时间是 $O(n\log_2 n)$。在最坏情况下，其时间复杂度也为 $O(n\log_2 n)$。

考点 6：归并排序

（1）归并排序是与插入排序、交换排序、选择排序不同的一类排序方法，其不同之处在于要求待排序列是由若干个有序子序列组成。归并排序中的核心操作是两个有序子序列的合并。

（2）二路归并排序即是将两个有序表合并成一个有序表的排序方法