2021 天勤计算机考研八套模拟卷 • 卷六

数据结构篇选择题答案解析

1．C。

I：有序表插入的时候是不能指定位置的，因为这样可能使得插入后的表不再是有序表。正确的插入思想是：先通过元素比较找到插入的位置，再在该位置上插入，故 I 错误。

II：从单链表插入和删除的语句描述中可以看出，无论是插入还是删除操作，都必须找到其前驱结点，故 II 正确。

III：删除双链表中间某个结点时，需要修改前后两个结点的各一个指针域，共计两个指针域，故 III 正确。

IV：当一个较短的有序表中所有元素均小于另一个较长的有序表中所有的元素，所需比较次数最少。假如一个有序表为 1、3、4，另一个有序表为 5、6、7、8、12，这样只需比较 3 次即可，故答案应该是 n 和 m 中较小者，即 min(n，m)，故Ⅳ错误。

2．A。

I：该选项旨在让考生知道一个公式。对于 n 个不同元素进栈，出栈序列的个数为

$$\frac{1}{n+1}C_{2n}^{n}$$

可以马上得出，当 n=3 时，出栈序列个数为

$$\frac{1}{4}C_{6}^{3}=\frac{6\times5\times4}{4\times3\times2\times1}=5$$

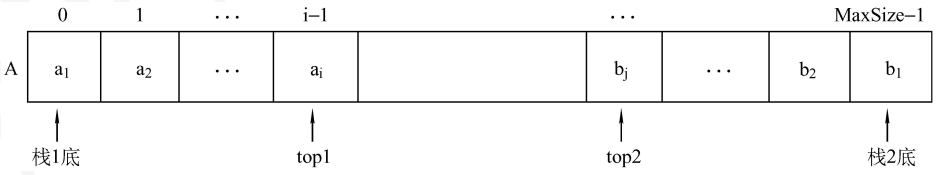
故Ⅰ正确。

II：链式栈一般采用单链表，栈顶指针即为链头指针。进栈和出栈均在链头进行，每次都要修改栈顶指针，链空即栈空（top==NULL），故Ⅱ错误。

III：由于栈中数据的操作只有入栈和出栈，且时间复杂度均为 O(1)，因此并没有减少存取时间，故 III 错误。

补充知识点：共享栈

解析：两个栈共享一个数组 A[0…MaxSize−1]的空间，从而构成共享栈。数组 A 的两端是固定的，而栈底也是固定的，为此将下标为 0 的一端作为栈 1 的栈底，其栈顶指针为 top1，将下标为 MaxSize−1 的一端作为栈 2 的栈底，其栈顶指针为 top2，如下图所示。



共享栈

栈 1 的四要素如下：

① 栈空条件：top1==−1。

② 栈满条件：top1==top2−1。

③ 元素 x 进栈：top1++；将元素 x 插入 A[top1]处。

④ 出栈元素：弹出 A[top1]元素；top1−−。

栈 2 的四要素如下：

① 栈空条件：top2==MaxSize。

② 栈满条件：top2==top1+1。

③ 元素 x 进栈：top2--；将元素 x 插入 A[top2]处。

④ 出栈元素：弹出 A[top2]元素；top2++。

注：以上都默认指针指向当前元素的下一个位置。

3．D。

对于元素 a（i，j）而言，前面有 j-1 列，第 1 列到第 j-1 列的元素个数分别为 1～j-1 个，由等差数列求和公式可算得一共有 j×(j-1)/2 个元素，故 k=j×(j-1)/2+i-1（注意 B 数组是从 0 开始存元素，因此要减去 1）。

4．A。

对于一棵二叉树（包括子树），它的遍历序列对应的结构应该是：先序遍历：|根|左子树|右子树|，中序遍历：|左子树|根|右子树|，后序遍历：|左子树|右子树|根|，由题目中给出的先序序列的第一个结点我们找到树的根 A，然后在中序序列中找到 A，并以 A 为分界将中序序列划分为|C_ED|A|_GFI_|，所以 C_ED 为左子树，_GFI_为右子树，再对应到后序遍历序列上，这里左子树结点的个数等于中序遍历序列中左子树结点的个数，因此 C_ _B 为左子树，HGJI_为右子树，这样把中序序列和后续序列中的左右子树一对比，则 CBED 为左子树，FGHIJ 为右子树。答案选 A。

总结：根据树的遍历结果来还原二叉树，或者根据其中的两个遍历序列来求第三个遍历序列这是历年考试的热点，考生需要记住的是无论怎样的序列，要想构建二叉树必须有中序序列。这是很显然的，这里说明一下原因：我们知道二叉树的定义是递归的，那么我们在构建二叉树的时候势必也会用到递归这种方法，而**这种形式的递归我们把它称之为分治（从中间分开来治理）法，而在三种遍历序列中只有中序遍历的结构才体现了这种思想。**

5．B。

首先，赫夫曼编码遵循的原则为：一个编码不能是任何其他编码的前缀。比如 1 和 10 就不行，因为 1 是 10 的前缀。既然 1 和 01 已经使用了，所以 1 和 01 开头的码字不能再使用。又由于赫夫曼树的高度为 5，故赫夫曼编码的长度不能超过 4，只剩下 0000、0001、0010、0011 等 4 种编码（这种编码方式可得到最多），故选 B 选项。

注意：本题选的是最多还可以对多少个字符编码，所以不能选取 001、000 等编码。例如选取 001，就意味着 0010 和 0011 不能使用，这样可编码的字符就少了 1 个。

总结：

（1）有 n 个叶子结点的赫夫曼树的结点总数为 2n-1。

（2）高度为 h 的赫夫曼树中，至少有 2h-1 个结点，至多有 2^h-1 个结点。

（3）赫夫曼树中一定没有度为 1 的结点。

（4）赫夫曼树中两个权值最小的结点一定是兄弟结点。

（5）树中任一非叶子结点的权值一定不小于下一层任一结点的权值。

补充例题：一棵赫夫曼树共有 215 个结点，对其进行赫夫曼编码，共能得到多少个码字？

解析：求多少个码字就是求有多少个叶结点，从（1）可以得到，叶结点的个数为 108 个，故可以得到 108 个码字。

6．D。

I：总结如下：

① 对于一个具有 n 个顶点的无向图，若采用邻接矩阵表示，则该矩阵大小是 n^2。

② 在含有 n 个顶点 e 条边的无向图的邻接矩阵中，非零元素的个数为 2e。

③ 在含有 n 个顶点 e 条边的无向图的邻接矩阵中，零元素的个数为 n^2-2e。

④ 在含有 n 个顶点 e 条边的有向图的邻接矩阵中，非零元素的个数为 e。

⑤ 在含有 n 个顶点 e 条边的有向图的邻接矩阵中，零元素的个数为 n^2-e。

根据③，故Ⅰ正确。

II：无向图采用邻接表表示时，每条边存储两次，所以其边表结点个数为偶数，故边表结点为奇数只能是有向

图，故 II 正确。

III：深度优先遍历算法是先访问一个顶点 v，然后是离开顶点越远越优先访问，即相当于二叉树的先序遍历，故 III 错误。

IV：采用广度优先遍历算法遍历一个图时，每个顶点仅遍历一次，所以最多只能进队 1 次，故Ⅳ错误。

7．D。

A：最小生成树是指权值之和为最小的生成树，但是不唯一，故 A 选项错误。

B：由广度优先遍历和深度优先遍历算法可知，深度优先算法构造的生成树的树高大于等于广度优先算法构造的生成树的树高，故 B 选项错误。

C：当最小生成树不唯一时，这两种算法构造的最小生成树可能相同，也可能不同，故 C 选项错误。

D：Prime 算法的时间复杂度为 $O(n^2)$，适合稠密图；Kruskual 算法的时间复杂度为 $O(elog_2e)$，适合稀疏图，故 D 选项正确。

8．B。

一棵 m 阶 B+树满足下列条件：

① 每个分支结点至多有 m 棵子树。

② 根结点或者没有子树，或者至少有两棵子树。

③ 除根结点外，其他每个分支结点至少有$\lceil m/2 \rceil$棵子树。

④ 具有 n 个关键字的结点含有 n 棵子树。

⑤ 所有叶子结点包含全部关键字及指向相应记录的指针，而且叶子结点按关键字的大小顺序链接。

⑥ 所有分支结点中仅包含它的各个子结点中**最大关键字**及指向子结点的指针。

⑦ B+树中，所有非终端结点可以看成是索引部分，故可用于文件的索引结构。

注意：B+和 B-树都支持“随机”检索。这里的随机检索之所以加双引号，是因为它与一般意义的随机检索有所不同。一般意义上，例如一个顺序表，给出元素地址即可直接取到元素称之为随机检索，而 B+树和 B-树中的从根结点到关键字所在结点的查找过程也叫随机检索。

综上所述，可知 II、III、IV 正确，I 错误，故选 B 选项。

补充知识点：很多考生被 B+树和 B-树的基本概念弄混，下面做一个小结。

解析：m 阶 B+树和 m 阶 B-树的主要差异如下：

① 在 B+树中，具有 n 个关键字的结点含有 n 棵子树；而在 B-树中，具有 n 个关键字的结点至少含有（n+1）棵子树。

② 在 B+树中，每个结点（除根结点外）中的关键字个数 n 的取值范围是$\lceil m/2 \rceil \leqslant n \leqslant m$，根结点 n 的取值范围是 $2 \leqslant n \leqslant m$；而在 B-树中，除根结点外，其他所有非叶子结点的关键字个数 n 的取值范围是$\lceil m/2 \rceil - 1 \leqslant n \leqslant m-1$，根结点 n 的取值范围是 $1 \leqslant n \leqslant m-1$。

记忆方式：“B-”中有个“-”号，自然关键字个数相对于 B+减掉了 1。

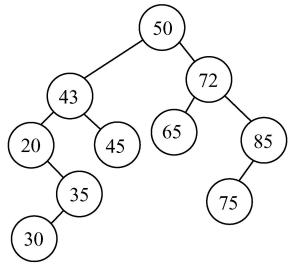
③ 在 B+树中，所有叶子结点包含了全部关键字，即其他非叶子结点中的关键字包含在叶子结点中；而在 B-树中，关键字是不重复的。

④ 在 B+树中，所有非叶子结点仅仅是起到了索引的作用，即结点中的每个索引项只含有对应子树的最大关键字和指向子树的指针，不含有该关键字对应记录的存储地址。而在 B-树中，每个关键字对应一个记录的存储地址。

⑤ 在 B+树上有两个头指针，一个指向根结点，另一个指向关键字最小的叶子结点，所有叶子结点链接成一个链表；而在 B-树中，叶子结点并不会有指针相连。

9．B。

由题可以建立出如下图所示的一棵二叉排序树。



查找元素 30 一次经过比较的元素为 50，43，20，35，30，共有 5 次元素间的比较，因此本题选 B 选项。

10．A。

首先需要知道快速排序的一个特性，即元素越无序，快速排序越快；元素越有序，快速排序越慢。但是一般情况下，有序的元素序列比较少，大部分情况都是杂乱无章的一堆数，所以说快速排序是所有排序中性能最好的排序方法。有些同学可能会有疑问，快速排序最差的时间复杂度是 $O(n^2)$，而有不少排序算法最坏的时间复杂度是 $O(n\log_2 n)$, 比如堆排序。为什么快速排序的性能是最好的呢？因为快速排序出现最坏性能的情况实在是太少发生了，所以要看综合的性能，不能只看最坏的（记住就好，在此不举例子了）。本题 A 选项是一个有序序列，所以速度肯定最慢。

总结：如果元素基本有序，使用直接插入排序效果最好；如果元素完全没序，使用快速排序效果最好。

11．C。

A：产生初始归并段的工作应该由置换—选择排序完成，故 A 选项错误。

设输入的关键字满足 $k_1>k_2>\cdots>k_n$，缓冲区大小为 m，用置换-选择排序方法可产生$\lceil n/m \rceil$个初始归并段。

B：因为最佳归并树是针对排序之后的初始归并段操作，所以归并排序不可能由最佳归并树完成，故 B 选项错误。

C：最佳归并树仿造赫夫曼树的构造过程，以初始归并段的长度为权值，构造具有最小带权路径长度的赫夫曼树，可以有效地减少归并过程中的读写记录数，以加快外部排序的速度，故 C 选项正确。

D：增大归并路数应该是由败者树来完成的，故 D 选项错误。