第一章　数据结构

1.1　选择题

1. ▲ 下列程序段的时间复杂度是 ____

```
int sum = 0;
    for(int i = 1; i < n; i *= 2)
        for(int j = 0; j < i; j++)
            sum++;
```

2. 关于线性表的顺序存储和链式存储结构的描述中, 正确的是 ( )

   (1) 线性表的顺序结构优于其链式存储结构

   (2) 链式存储结构比顺序存储结构能更方便地表示各种逻辑结构

   (3) 若频繁使用插入和删除操作, 则顺序存储结构更优于链式存储结构

   (4) 顺序存储结构和链式存储结构都可以进行顺序存取

   A. 1,2,3　　**B.** 2,4　　**C.** 2,3　　**D.** 3,4

3. 对于一个头指针为 $head$ 的带头结点的单链表, 判断该表为空表的条件是 ( ), 对于不带头结点的单链表, 判断空表的条件是 ( )

   A. $head == NULL$　　**B.** $head \to next == NULL$

   C. $head \to next == head$　　**D.** $head \neq NULL$

4. 一个链表最常用的操作为在末尾插入结点和删除节点, 则选用 ( ) 最节省时间.

   A. 带头结点的双循环链表　　**B.** 单循环列表

   C. 带尾结点的单循环链表　　**D.** 单链表

5. 设对 $n(n>1)$ 元素的线性表运算只有 4 种, 删除第一个元素, 删除最后一个元素, 在第一个元素之前插入一个元素, 在最后一个元素之后插入一个元素, 则最好使用 ( )

   A. 只有尾结点指针没有头结点指针的循环单链表

   B. 只有尾结点指针没有头结点指针的非循环双链表

   C. 只有头结点指针没有尾结点指针的循环双链表

   D. 既有头结点有又有尾结点的循环单链表

6. 假定利用数组 $a[n]$ 存储一个栈, 初始栈顶指针 $top == -1$, 则元素 x 进栈的操作为 ()

   A. $a[--top]=x$ **B.** $a[top--]=x$ **C.** $a[++top]=x$ **D.** $a[top++]=x$

7. 和顺序栈相比, 链栈有一个比较明显的优势, 即 ()

   A. 通常不会出现栈满的情况 **B.** 通常不会出现栈空的情况

   C. 插入操作更容易实现 **D.** 删除操作更容易实现

8. 链栈 (不带头结点) 执行 Pop 操作, 并将出栈元素存在 x 中, 应该执行 ()

   A. $x=top; top=top\to next$ **B.** $x=top\to data$

   C. $top=top\to next; x=top\to data$ **D.** $x=top\to data; top=top\to next$

9. 三个不同元素进栈, 能够得到 () 不同的出栈序列

10. 一个栈的输入序列为 $1,2,\ldots,n$ 输出序列的第一个元素为 i, 则第 j 个输出元素是 ()

    A. 不确定 **B.** n - i **C.** n - i - 1 **D.** n - i + 1

11. 设栈的初始状态为空, 当字符序列 n1_ 作为栈的输入时, 输出长度为 3, 且可用做 C 语言标识符的序列有 () 个

    A. 3 **B.** 4 **C.** 5 **D.** 6

12. 设有一个顺序共享栈 Share[0:n-1], 其中第一个栈顶指针 top1 的初始值为-1, 第二个栈顶指针 top2 的初始值为 n, 则判断共享栈满的条件是 ()

    A. $top2-top1==1$ **B.** $top1-top2==1$

    C. $top1==top2$ **D.** 以上对不对

13. ♦ 若元素a,b,c,d,e,f 依次进栈, 允许进栈, 出栈交替进行, 但不允许连续 3 次进栈, 退栈操作, 不可能得到的出栈序列是 ()

A. dcebfa **B.** cbdaef **C.** bcaefd **D.** afedcb

14. ♦ 一个栈的入栈序列为$1, 2, 3 \ldots, n$ 出栈序列为 $P_1, P_2, \ldots, P_n$. 若 $P_2 = 3$ 则 P_3 可能的取值的个数是 ()

A. n-3 **B.** n-2 **C.** n-1 **D.** 无法确定

15. ♦ 若栈 S1 中保存整数, 栈 S2 中保存运算符, 函数 F() 依次执行如下各步操作:

(1) 从 S1 中依次弹出两个操作数 a 和 b

(2) 从 S2 中弹出一个运算符 op

(3) 执行相应的运算 b op a

(4) 将运算结果压入 S1 中

假定 S1 中的操作数一次是5,8,3,2(2 在栈顶),S2 中的运算符依次是*,-,+(栈顶). 调用 3 次 F() 后,S1 栈顶保存的值是 ()

A. -15 **B.** 15 **C.** -20 **D.** 20

16. 循环队列存储在数组 $A[0 \ldots n]$ 中, 其中 $rear$ 为队尾指针,$front$ 为队首指针. 则入队时的操作为 ___; 出队时的操作为 ____; 判断队空的操作为 ___; 判断队满的操作为 ___, 当前队列中元素的个数为 ____.

17. 用链式存储方法的队列进行删除操作时需要 ()

A. 仅修改头指针 **B.** 仅修改尾指针

C. 头尾指针都要修改 **D.** 头尾指针可能都要修改

18. 假设循环单链表表示的队列长度为 n, 队头固定在链表尾, 若只设置头指针, 则进队操作的时间复杂度为 ()

A. $O(n)$ **B.** $O(1)$ **C.** $O(n^2)$ **D.** $O(n \log_2 n)$

19. ♦ 已知循环队列存储在一维数组 $A[0 \ldots n-1]$ 中, 且队列非空时 $front$ 和 $rear$ 分别指向队头元素和队尾元素. 若初始队列为空, 且要求第一个进入队列的元素存储在 $A[0]$ 处, 则初始时 $front$ 和 $rear$ 的值分别是 ()

A. 0,0 **B.** 0, n-1 **C.** n-1,0 **D.** n-1,n-1

20. 循环队列放在一维数组 $A[0\ldots M-1]$ 中, end1 指向对首元素,end2 指向队尾元素的后一个位置. 假设队列两端均可进行入队与出队操作, 队列中最多能容纳 M-1 个元素. 初始时为空, 下列判断队满和队空的条件中, 正确的是 ()

A. 队空:$end1 == end2$ 队满: $end1 == (end2+1)\ mod\ M$

B. 队空:$end1 == end2$ 队满: $end2 == (end1+1)\ mod\ (M-1)$

C. 对空:$end2 == (end1+1)\ mod\ M$ 队满: $end1 == (end2+1)\ modM$

D. 对空:$end1 == (end2+1)\ mod\ M$ 队满: $end2 == (end1+1)\ mod(M-1)$

21. ♦ 已知操作符包含 $+,-,*,/,(,)$. 将中缀表达式 $a+b-a*((c+d)/e-f)+g$ 转换为等价的后缀表达式 (逆波兰表示法), 用栈来实现. 初始时栈为空, 转换过程中栈中至多保存 () 个操作符.

22. ▲ 有一个 $n\times n$ 的对称矩阵 A, 将其下三角部分按行存放在一维数组 B 中, 而 $A[0][0]$ 存放在 $B[0]$ 中, 则第 $i+1$ 行对角元素 $A[i][i]$ 存放在 B 中的 () 处

A. (i+3)i/2 **B.** (i+1)i/2 **C.** (2n-i+1)i/2 **D.** (2n-i-1)i/2

23. ♦ 由一个 100 阶的三对角矩阵 M, 其元素 $m_{i,j}(1\le i,j\le 100)$ 按行优先依次压入下标从 0 开始的一维数组 N 中. 元素 $m_{30,30}$ 在 N 中的下标是 ()

24. ▲ 在 KMP 算法中, 串ababaaababaa 的 PM 数组,Next 数组,Nextval 数组分别为?

25. ♦ 设主串 $T=abaabaabcabaabc$ 模式串 $S=abaabc$ 采用 KMP 算法进行模式匹配, 到匹配成功为止, 在匹配过程中进行的单个字符间的比较次数是 ()

26. 树的路径长度是从树根到每个结点的路径长度的 ( )

A. 总和 **B.** 最小值 **C.** 最大值 **D.** 平均值

27. (判断正误)

(1) 度为 2 的有序树就是二叉树

(2) 结点按完全二叉树层序编号的二叉树中, 第 i 个结点的左孩子编号为 $2i$

28. 具有 10 个叶结点的二叉树中有 ( ) 个度为 2 的结点

29. 设二叉树有 $2n$ 个结点, 且 $m < n$, 则不可能存在 ( ) 的结点

A. n 个度为 0 **B.** 2m 个度为 0 **C.** 2m 个度为 1 **D.** 2m 个度为 2

30. 已知一颗完全二叉树的第 6 层 (设根为第一层) 有 8 个结点, 则完全二叉树的结点个数最少是 ( )

31. 一颗完全二叉树上有 1001 个结点, 其中叶结点的个数是 ( )

32. ♦ 对于任意一颗高度为 5 且有 10 个结点的二叉树, 若采用顺序存储结构保存, 每个节点占 1 个存储单元 (仅保存结点的数据信息), 则存放该二叉树需要的存储单元数量至少是 ( )

33. 在下列关于二叉树遍历的说法中, 正确的是 ( )

A. 若有一个结点是二叉树中某个子树的中序遍历结果序列的最后一个结点，则它一定是该子树的前序遍历结果序列的最后一个结点。

B. 若有一个结点是二叉树中某个子树的前序遍历结果序列的最后一个结点，则它一定是该子树的中序遍历结果序列的最后一个结点。

C. 若有一个叶结点是二叉树中某个子树的中序遍历结果序列的最后一个结点，则它一定是该子树的前序遍历结果序列的最后一个结点。

D. 若有一个叶结点是二叉树中某个子树的前序遍历结果序列的最后一个结点，则它一定是该子树的中序遍历结果序列的最后一个结点。

34. 设 n, m 为一颗二叉树上的两个结点, 在后序遍历时,n 在 m 前的充分条件是 ( )

A. n 在 m 的右方 **B.** n 是 m 的祖先 **C.** n 在 m 的左方 **D.** n 是 m 的子孙

35. 在二叉树中的两个结点 m 和 n, 若 m 是 n 的祖先, 则使用 ( ) 可以找到从 m 到 n 的路径

36. 若二叉树中结点的先序序列是 $\ldots a \ldots b \ldots$, 中序序列是 $\ldots b \ldots a \ldots$ 则 ( )

A. 结点 a 和结点 b 分别在某结点的左子树和右子树中

B. 结点 b 和结点 a 的右孩子中

C. 结点 b 在结点 a 的左孩子中

D. 结点 a 和结点 b 分别在某结点的两颗分非空子树中

37. 线索二叉树是 ( ) 结构

A. 逻辑 **B.** 逻辑和存储 **C.** 物理 **D.** 线性

38. 一颗左子树为空的二叉树的先序线索化后, 其中空的链域的个数是 ( )

A. 不确定 **B.** 0 个 **C.** 1 个 **D.** 2 个

39. 二叉树在线索化后, 仍然不能有效求解的问题是 ( )

A. 先序线索二叉树求先序后继 **B.** 中序线索二叉树求中序后继

C. 中序线索二叉树求中序前驱 **D.** 后序线索二叉树求后序后继

40. 若 X 是二叉中序线索树中一个有左孩子的结点, 且 X 不为根, 则 X 的前缀为 ( )

A. X 的双亲 **B.** X 的右子树中最左节点

C. X 的左子树中最右结点 **D.** X 的左子树中最右的叶结点

41. ( ) 遍历仍然需要栈的支持.

A. 先序线索树 **B.** 中序线索树 **C.** 后序线索树 **D.** 所有线索树

42. ♦ 先序序列为a,b,c,d 的不同二叉树的个数是 ( )

43. ♦ 若结点 p 和 q 在二叉树 T 的中序遍历序列中相邻, 且 p 在 q 之前, 则下列 p 和 q 的关系中, 不可能的是 ( )

(1) q 是 p 的双亲

(2) q 是 p 的右孩子

(3) q 是 p 的右兄弟

(4) q 是 p 的双亲的双亲

A. 1 **B.** 3 **C.** 2,3 **D.** 2,4

44. 利用二叉链表存储森林时, 根结点的右指针是 ( )

A. 指向最左兄弟 **B.** 指向最右兄弟 **C.** 一定为空 **D.** 不一定为空

45. 森林 $T=(T_1,T_2,\ldots,T_m)$ 转换为二叉树 BT 的过程为: 若 m=0, 则 BT 为空, 若 $m\neq 0$

A. 将中间子树 $T_{mid}(mid = (1+m)/2)$ 的根作为 BT 的根; 将 $(T_1, T_2, \ldots, T_{mid-1})$ 转换为 BT 的左子树; 将 $(T_{mid+1}, \ldots, T_m)$ 转换为 BT 的右子树

B. 将子树 T_1 的根作为 BT 的根, 将 T_1 的子树森林转换为 BT 的左子树; 将 $(T_2, T_3 \ldots, T_m)$ 转换 BT 的右子树

C. 将子树 T_1 根作为 BT 的根, 将 T_1 的左子森林转换为 BT 的左子树; 右子森林转换右子树, 其他类似

D. 将森林 T 的根作为 BT 的根, 将 $(T_1, \ldots, T_m)$ 转换为该根下的结点, 得到一棵树, 然后将这课树转换为二叉树

46. 设 F 是一个森林,B 是由 F 转换为来的二叉树. 若 F 中有 n 个非终端节点, 则 B 中右指针域为空的结点数目是 ( )

47. ♦ 将森林转换为对应的二叉树, 若二叉树中, 结点 u 是结点 v 的父结点的父结点, 则原来的森林中,u 和 v 可能具有关系是 ( )

    (1) 父子关系

    (2) 兄弟关系

    (3) u 的父结点和 v 的父结点是兄弟关系

    A. 2 **B.** 1,2 **C.** 1,3 **D.** 1,2,3

48. ♦ 已知一颗有 2011 个结点的树, 其叶结点个数为 116, 则该树对应的二叉树中无右孩子的结点个数为 ( )

    A. 115 **B.** 116 **C.** 1895 **D.** 1896

49. ♦ 若森林 F 有 15 条边,25 个结点, 则 F 包含树的个数是 ( )

50. ♦ 若将一颗树 T 转换为对应的二叉树 BT, 则下列对 BT 的遍历中, 其遍历序列与 T 的后根遍历序列相同的是 ( )

    A. 先序遍历 **B.** 中序遍历 **C.** 后序遍历 **D.** 层序遍历

51. 在有 n 个叶节点的哈夫曼树中, 非叶结点的总数是 ( )

52. 设哈夫曼编码的长度不超过 4, 若已对两个字符编码为 1 和 01, 则还最多可以对 ( ) 个个字符编码

53. 一下对于哈夫曼树的说法中, 错误的是 ( )

A. 对应一组权值构造出来的哈夫曼树一般不是唯一的

B. 哈夫曼树具有最小的带权路径长度

C. 哈夫曼树中没有度为 1 的结点

D. 哈夫曼树中除了度为 1 的节点外, 还有度为 2 的结点和叶结点

54. 若度为 m 的哈夫曼树中, 叶结点的数目为 n, 则非叶结点的数目为 ( )

55. ♦ 已知字符集a,b,c,d,e,f 若各字符出现的次数分别为6,3,8,2,10,4 则对应字符集中的各字符的哈夫曼编码可能是 ( )

A. 00,1011,01,1010,11,100

B. 00,100,110,000,0010,01

C. 10,1011,11,0011,00,010

D. 0011,10,11,0010,01,000

56. ♦ 对应任意给定的含有 n 个字符的有限集合 S, 用二叉树表示 S 的哈夫曼编码集和定长编码集, 分别得到二叉树 T_1 和 T_2. 下列叙述正确的是 ( )

A. T_1 和 T_2 的结点数相同

B. T_1 的高度大于 T_2 的高度

C. 出现频次不同的字符在 T_1 中处于不同的层

D. 出现频次不同的字符在 T_2 中处于相同的层

57. 以下关于图的叙述中, 正确的是 ( )

A. 图与树的区别在于图的边数大于等于顶点数

B. 假设有图 $G = \{V, \{E\}\}$, 顶点集 $V' \subseteq V, E' \subseteq E$ 则 V' 和 $\{E'\}$ 构成 G 的子图

C. 无向图的连通分量是指无向图的极大连通子图

D. 图的遍历就是从图中的某一顶点出发遍历图中的其余顶点

58. 以下关于图的说法, 正确的是 ( )

A. 强连通有向图的任何顶点到其他顶点都有弧

B. 图的任意顶点的入度都等于出度

C. 有向完全图一定是强连通有向图

D. 有向图的边集的子集和顶点集的子集可构成原有向图的子图

59. 对于一个有 n 个顶点的图; 若是连通无向图, 其边的个数至少是 ( ); 若是强连通有向图, 其边的个数至少为 ( )

60. 在有 n 个顶点的有向图中, 顶点的度最大可以达到 ( )

61. 设无向图 $G=(V,E), G'=(V',E')$ 若 G' 是 G 的生成树, 则下列不正确的是 ( )

    (1) G' 为 G 的连通分量

    (2) G' 为 G 的无环子图

    (3) G' 为 G 的极小连通子图且 $V'=V$

    A. 1,2 **B.** 3 **C.** 2,3 **D.** 1

62. ♦ 下列关于无向连通图特性的叙述中, 正确的是 ( )

    (1) 所有顶点的度之和为偶数

    (2) 边数大于顶点数减一

    (3) 至少有一个顶点的度为一

    A. 1 **B.** 2 **C.** 1,2 **D.** 1,3

63. 带权有向图 G 用临接矩阵存储, 则 v_i 的入度等于邻接矩阵中 ( )

    A. 第 i 行非 ∞ 的元素个数 **B.** 第 i 列非 ∞ 的元素个数

    C. 第 i 行非 ∞ 且非 0 的元素个数 **D.** 第 i 列非 ∞ 且非 0 的元素个数

64. 无向图 G 中包含 N(N>15) 个顶点, 以临接矩阵形式存储时共占用 N^2 个存储单元 (其他辅助空间忽略不计); 以临接表形式存储时, 每个表结点占用 3 个存储单元, 每个头结点占用 2 个存储单元 (其他辅助空间忽略不计). 若令图 G 的临接矩阵存储所占空间小于等于临接表存储所占空间, 该图 G 所包含的边的数量至少是 ( )

65. n 个顶点的无向图的邻接表中最多有 ( ) 个边表节点

66. 假设有 n 个顶点,e 条边的有向图用邻接表表示, 则删除与某个顶点 v 相关的所有边的时间复杂度是 ( )

67. 对于一个有 n 个顶点,e 条边的图采用临接表表示时, 进行 DFS 遍历的时间复杂度是 ( ), 空间复杂度是 ( ); 进行 BFS 遍历的时间复杂度是 ( ), 空间复杂度是 ( )

68. 对于一个有 n 个顶点,e 条边的图采用临接矩阵表示时, 进行 DFS 遍历的时间复杂度是 ( ), 空间复杂度是 ( ); 进行 BFS 遍历的时间复杂度是 ( ), 空间复杂度是 ( )

69. 图的广度优先生成树的树高比深度优先生成的树高 ( )

A. 小或相等 **B.** 小 **C.** 大或相等 **D.** 大

70. 从无向图的任意顶点出发进行一次深度优先遍历便可以访问所有顶点, 则该图一定是 ( )

A. 完全图 **B.** 连通图 **C.** 有回路 **D.** 一棵树

71. 一下叙述中, 正确的是 ( )

A. 最短路径一定是简单路径

B. Dijkstra 算法不适合求有环路的带权图的最短路径

C. Dijkstra 算法不适合求任意两个顶点的最短路径

D. Floyd 算法求两个顶点的最短路径,$path_k - 1$ 一定是 $path_k$ 的子集

72. 若一个有向图的顶点不能排成一个拓扑序列, 则可以判断该有向图 ( )

A. 含有多个出度为 0 的顶点 **B.** 是一个强连通图

C. 含有多个入度为 0 的顶点 **D.** 含有顶点数大于 1 的强连通分量

73. 下列关于图的说法中, 正确的是 ( )

(1) 有向图中顶点 V 的度等于其临接矩阵中第 V 行中 1 的个数

(2) 无向图的邻接矩阵一定是对称矩阵, 有向图的邻接矩阵一定是非对称矩阵

(3) 在带权图 G 的最小生成树 G_i 中, 某条边的权值可能会超过为选边的权值

(4) 若有向无环图的拓扑序列唯一, 则可以唯一确定该图

A. 1,2,3 **B.** 3,4 **C.** 3 **D.** 4

74. 已知带权图为 $G = (V, E)$, 其中 $V = \{v_1, v_2, \ldots, v_{10}\}$, 边集合为 $E = \{< v_1, v_2 > 5, < v_1, v_3 > 6, < v_2, v_5 > 3, < v_3, v_5 > 6, < v_3, v_4 > 3, < v_4, v_5 > 3, < v_4, v_7 > 1, < v_4, v_8 > 4, < v_5, v_6 > 4, < v_5, v_7 > 2, < v_6, v_{10} > 4, < v_7, v_9 > 5, < v_8, v_9 > 2, < v_9, v_{10} > 2\}$ 则 G 的关键路径长度为 ( )

75. 下列关于关键路径的说法中, 正确的是 ( )

(1) 改变网上某一关键路径上的某一关键路径, 必将产生不同的关键路径

(2) 在 AOE 图中, 关键路径上活动的时间延长多少, 整个工期的时间也就随之延长多少

(3) 缩短关键路径上任意一个关键活动的持续时间可缩短关键路径长度

(4) 缩短所有关键路径上共有的任意一个关键活动的持续时间可缩短关键路径的长度

(5) 缩短多条关键路径上共有的任意一个关键活动的持续时间可缩短关键路径长度

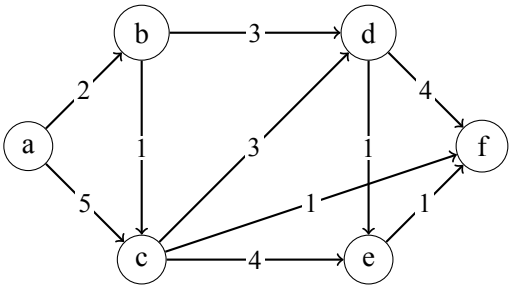
A. 2,5 **B.** 1,2,4 **C.** 2,4 **D.** 1,4

76. ♦ 若用临接矩阵存储有向图, 矩阵中主对角线以下的元素全为零, 则关于该图拓扑序列的结论是 ( )

A. 存在, 且唯一 **B.** 存在, 且不唯一

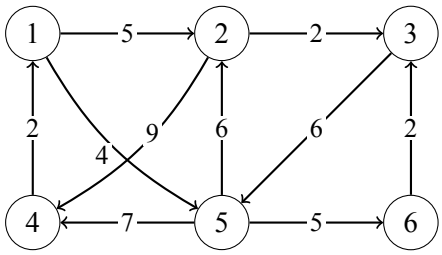
C. 存在, 可能唯一 **D.** 无法确定是否存在

77. ♦ 对下列图所示的有向带权图, 若采用 Dijkstra 算法求源点 a 到其他个顶点的最短路径, 则得到的的第一条最短路径的目标顶点是 b, 第二条最短路径的目标顶点是 c, 后续得到的其余各最短路径的目标顶点一次是 ( )



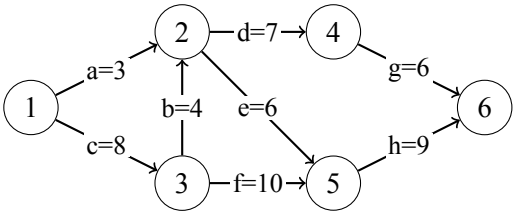
A. d,e,f **B.** e,d,f **C.** f,d,e **D.** f,e,d

78. ♦ 使用 Dijkstra 算法求下图中从顶点 1 到其他个顶点的最短路径, 依次得到的各最短路径的目标顶点是 ( )



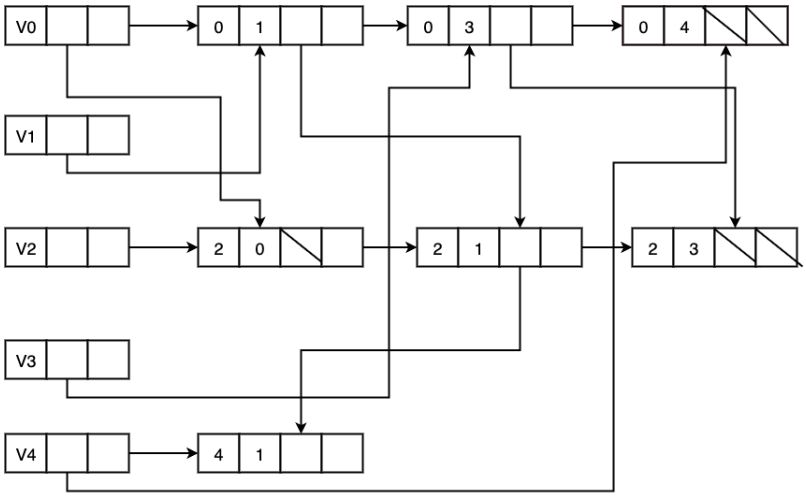
A. 5,2,3,4,6 **B.** 5,2,3,6,4 **C.** 5,2,4,3,6 **D.** 5,2,6,3,4

79. ♦ 下列所示的 AOE 网表示一项包含 8 个活动的工程, 活动 d 的最早开始时间和最迟开始时间分别是 ( )



A. 3,7 **B.** 12,12 **C.** 12,14 **D.** 15,15

80. 图 G 利用十字链表法表示如下, 请问图 G 可能的拓扑排序为 ( )



A. V_2, V_0, V_3, V_1, V_4 **B.** V_0, V_3, V_1, V_4, V_2 **C.** V_2, V_0, V_4, V_3, V_1 **D.** 不存在拓扑序列

81. 以下对于最小生成树的描述, 正确的是 ( )

(1) 所有无向连通图的最小生成树一定有多个

(2) $Prim$ 和 $Kruskal$ 算法构建的最小生成树一定不同

(3) 只要无向图中不存在相同权值的边, 则该无向图的最小生成树唯一

(4) 只要无向图中存在权值相同的边, 则该无向图的最小生成树一定不唯一

(5) 在具有 n 个顶点的无向图 G 中, 含有 n 个顶点,n-1 条边的 G 的子图就是 G 的生成树

(6) 生成树就是最小生成树

A. 3 **B.** 3,4 **C.** 全部正确 **D.** 全部错误

82. 以下说法中错误的是 ( )

(1) 求从源点到其余顶点的 Dijkstra 最短路径算法中弧上权不能为负的原因是在实际应用中无意义

(2) 若图用临接矩阵表示, 则利用 Dijkstra 算法求每一对不同顶点之间的最短路径的算法时间为 $O(n^3)$

(3) Floyd 算法求每对不同顶点对的算法中允许弧上的权为负, 但不能有权和为负的回路

A. 1,2,3 **B.** 1 **C.** 1,3 **D.** 2,3

83. () 可以求无向图的所有连通分量.

84. 存在一张无向连通图 $G=(V,E), |V|=n, |E|=e$ 分布使用 $Prim, Kruskal$ 算法来产生图 G 的最小生成树, 则时间复杂度分别是 ( ).

85. 已知 7 个城市 (分别编号 $0 \sim 6$) 之间修建道路的耗费分别为:
(0,1)22,(0,2)9,(0,3)10,(1,3)15,(1,4)15,(1,6)12,(2,3)4,(2,5)3,(3,5)5,(3,6)23,(4,6)20,(5,6)32
要修建路网让每两个城市间都可以互通 (直达或途径其他城市), 最小的耗费是 ( );

86. 由 n 个数据元素组成的两个表: 一个递增有序, 一个无序. 采用顺序查找算法, 对有序表从头开始查找, 发现当前元素已不小于待查元素时, 停止查找, 确定查找不成功, 已知查找任意元素的概率是相同的, 则在两种表中成功查找 ( )

A. 平均时间后者小 **B.** 平均时间两者相同

C. 平均时间前者小 **D.** 无法确定

87. 在一个顺序存储的有序线性表上查找一个数据时, 既可以采用折半查找, 也可以采用顺序查找, 但前者比后者的查找速度 ( )

A. 必然快 **B.** 取决于表是递增还是递减

C. 在大部分情况下要快 **D.** 必然不快

88. 折半查找过程所对应的判断树是一颗 ( )

A. 最小生成树 **B.** 平衡二叉树 **C.** 完全二叉树 **D.** 满二叉树

89. 折半查找和二叉排序树的时间性能 ( )

A. 相同 **B.** 有时不相同 **C.** 完全不同 **D.** 无法比较

90. 对表长为 n 的有序表进行折半查找, 其判定树的高度为 ( )

A. $\lceil \log_2 (n+1) \rceil$ **B.** $\log_2 (n+1) - 1$ **C.** $\lceil \log_2 n \rceil$ **D.** $\lfloor \log_2 n \rfloor$ - 1

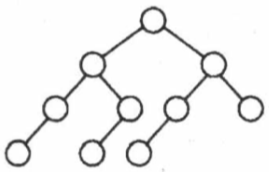
91. 具有 12 个关键字的有序表中, 对每个关键的查找概率相同, 折半查找算法查找成功的平均查找长度是 ( ), 折半查找失败的平均查找长度是 ( )

92. 为提高查找效率, 对有 65025 个元素的有序顺序表建立索引顺序结构, 在最好的情况下查找到表中已有元素最多需要执行 ( ) 次关键字比较

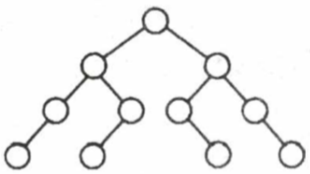
93. ♦ 已知一个长度为 16 的顺序表 L, 其元素按关键字有序排列, 若采用折半查找法查找一个 L 中不存在的元素, 则关键字的比较次数最多是 ( )

94. ♦ 下列二叉树中，可能成为折半查找判定树（不含外部结点）的是

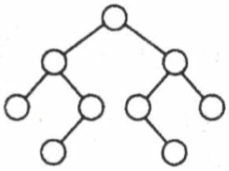
A.



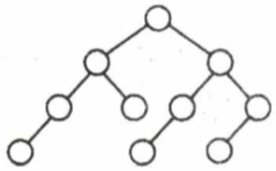
B.



C.



D.



95. 在含有 n 个结点的二叉排序中查找某个关键字的结点时, 最多进行 ( ) 比较

96. 构建一颗具有 n 个节点的二叉排序树时, 最理想情况下的深度为 ( )

97. 含有 20 个节点的平衡二叉树的最大深度为 ( ), 具有 5 层结点的 AVL 树至少有 ( ) 个结点.

98. 下列关于红黑树的说法中, 正确的是 ( )

A. 红黑树的红结点的数目最多和黑结点的数目相同

B. 若红黑树的所有结点都是黑色的, 那么它一定是一棵满二叉树

C. 红黑树的任何一个分支结点都有两个非空孩子结点

D. 红黑树的子树也一定是红黑树

99. ▲ 将关键字序列1,2,3,4,5,6,7 一次插入初始为空的红黑树 T, 则 T 中红结点的个数是 ( )

100. ♦ 现有一颗无重复关键字的平衡二叉树, 对其进行中序遍历得到一个降序序列, 下列关于该平衡二叉树的叙述中, 正确的是 ( )

A. 根结点的度一定是 2　　**B.** 树中最小元素一定是叶结点

C. 最后插入的元素一定是叶结点　　**D.** 树中最大元素一定是无左子树

101. ♦ 在任意一颗非空平衡二叉树 T_1 中, 删除某结点 v 之后形成平衡二叉树 T_2, 再将 v 插入 T_2 形成平衡二叉树 T_3 下列关于 T_1, T_3 的描述中, 正确的是 ( )

(1) 若 v 是 T_1 的叶结点, 则 T_1 和 T_3 可能不相同

(2) 若 v 不是 T_1 的叶结点, 则 T_1 和 T_3 一定不相同

(3) 若 v 不是 T_1 的叶节点, 则 T_1 和 T_3 一定相同

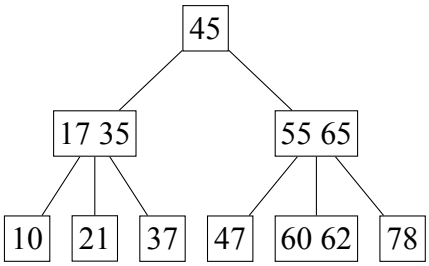
A. 1　　**B.** 2　　**C.** 1,2　　**D.** 1,3

102. 下列关于 B 与 B+ 树的描述中, 不正确的是 ( )

A. B 数和 B+ 树都能有效的支持顺序查找　　**B.** B 树和 B+ 树都能有效的支持随机查找

C. B 树和 B+ 树都是平衡的多叉树　　**D.** B 树和 B+ 树都可以用于文件索引结构

103. ♦ 已知一颗 3 阶 B 树, 如下图所示. 删除关键字 78 得到一颗新 B 树, 其最右叶结点中的关键字是 ( )



104. ♦ 在一颗高度为 2 的 5 阶 B 树中, 所含有的关键的个数至少是 ( )

A. 5　　**B.** 7　　**C.** 8　　**D.** 14

105. ♦ 下列应用中, 适合使用 B+ 树的是 ( )

A. 编译器中的词法分析　　**B.** 关系数据库系统的索引

C. 网络中的路由表的快速查找　　**D.** 操作系统的磁盘空闲块管理

106. 散列表查找成功时, 平均查找长度仅和 () 有关.

107. 在开放定址法中散列到同一地址而引起的堆积问题是由于 ( ) 而引起的

A. 同义词之间发生冲突 **B.** 非同义词之间发生冲突

C. 同义词之间或非同义词之间发生冲突 **D.** 散列表溢出

108. 下列关于散列冲突处理方法中, 正确的是 ( )

(1) 采用在平方探测法处理冲突时不容易产生聚集

(2) 采用线性探测法解决冲突时, 所有同义词在散列表中一定相邻

(3) 采用链地址法处理冲突时, 若限定在链首插入, 则插入任意一个元素的时间是相同的

(4) 采用链地址法处理冲突时容易引起聚集现象

A. 1,3 **B.** 1,2,3 **C.** 3,4 **D.** 1,4

109. 对包含 n 个元素的散列表进行查找, 平均查找长度为 ( )

A. 为 $O(\log_2 n)$ **B.** 为 $O(1)$ **C.** 不直接依赖于 n **D.** 直接依赖于表长 m

110. ♦ 现有长度为 11 且初始为空的散列表 HT, 散列函数 $H(k) = k\%7$, 用线性探测再散列法法解决冲突, 将关键字序列87,40,30,6,11,22,98,20 依次插入 HT 后,HT 查找失败的平均查找长度是 ( )

A. 4 **B.** 5.25 **C.** 6 **D.** 6.29

111. ♦ 下列因素中, 影响哈希方法的平均查找长度是 ( )

(1) 装填因子

(2) 散列函数

(3) 冲突解决策略

A. 1,2 **B.** 1,3 **C.** 2,3 **D.** 1,2,3

112. 下列关于排序的叙述中, 正确的是 ( )

A. 稳定的排序方法优于不稳定的排序方法

B. 对同一线性表使用不同的排序方法进行排序, 得到的排序结果可能不同

C. 排序方法都是在顺序表上实现的, 在链表上无法实现排序方法

D. 在顺序表上实现的排序方法在链表上也可以实现

113. 对于任意 7 个关键字进行基于比较的排序, 至少要进行 () 次关键字之间的比较

A. 13 **B.** 14 **C.** 15 **D.** 16

114. 用直接插入排序算法对下列 4 个表进行排序 (从小到大), 比较次数最少的是 ( )

A. 94,32,40,90,80,46,21,69 **B.** 21,32,46,40,80,69,90,94

C. 32,40,21,46,69,94,90,80 **D.** 90,69,80,46,21,32,94,40

115. 对序列98,36,-9,0,47,23,1,8,10,7 采用希尔排序, 下列序列 ( ) 是增量为 4 的一趟排序结果

A. 10,7,-9,0,47,23,1,8,98,36 **B.** -9,0,36,98,1,8,23,47,7,10

C. 36,98,-9,0,23,47,1,8,7,10 **D.** 以上都不对

116. 若用冒泡排序算法对序列10,14,26,29,41,52 从大到小进行排序, 则需要进行 ( ) 比较

A. 3 **B.** 10 **C.** 15 **D.** 25

117. 对下列关键字序列用到了快排进行排序, 速度最快的情形是 () 速度最慢的是 ()

A. 21,25,5,17,9,23,30 **B.** 25,23,30,17,21,5,9

C. 21,9,17,30,25,23,5 **D.** 5,9,17,21,23,25,30

118. 对于下列 4 个序列, 以第一个关键字为基准用快速排序算法进行排序, 在第一趟过程中移动记录次数最多的是 ()

A. 92,96,88,42,30,35,110,100 **B.** 92,96,100,110,42,35,30,88

C. 100,96,92,35,30,110,88,42 **D.** 42,30,35,92,100,96,88,110

119. 设线性表中每个元素有两个数据项 k_1, k_2 现对线性表按以下规则进行排序, 先看数据项 k_1, 若比其值小的元素在前, 大的元素在后, 与其值相同再看 k_2, 小的元素在前, 大的元素在后. 满足这种要求的算法是 ( )

A. 先按 k_1 进行直接插入排序, 在按 k_2 进行简单选择排序

B. 先按 k_2 进行直接插入排序, 在按 k_1 进行简单选择排序

C. 先按 k_1 进行简单选择排序, 在按 k_2 进行直接插入排序

D. 先按 k_2 进行简单选择排序, 在按 k_1 进行直接插入排序

120. 若只想得到 1000 个元素组成的序列中第 10 个最小元素之前的部分排序的序列, 则用 () 方法最快.

A. 冒泡排序 **B.** 快速排序 **C.** 希尔排序 **D.** 堆排序

121. 在含有 n 个关键字的小根堆中, 关键字最大的记录可能存储在 ( ) 位置

A. n/2 **B.** n/2 + 2 **C.** 1 **D.** n/2 - 1

122. 构建 n 个记录的初始堆, 其时间复杂度为 (), 对 n 个记录进行堆排序, 最坏情况下时间复杂度是 ()

A. $O(n)$ **B.** $o(n^2)$ **C.** $O(\log_2 n)$ **D.** $O(n \log_2 n)$

123. 已知小根堆为8,15,10,21,34,16,12 删除关键字 8 之后需要重新建堆, 关键字之间的比较次数是 ()

A. 1 **B.** 2 **C.** 3 **D.** 4

124. 将序列6,1,5,9,8,4,7 建成大根堆时, 正确的序列变化时 ()

A. 6,1,7,9,8,4,5→6,9,7,1,8,4,5→9,6,7,1,8,4,5→9,8,7,1,6,4,5

B. 6,9,5,1,8,4,7→6,9,7,1,8,4,5→9,6,7,1,8,4,5→9,8,7,1,6,4,5

C. 6,9,5,1,8,4,7→9,6,5,1,8,4,7→9,6,7,1,8,4,5→9,8,7,1,6,4,5

D. 6,1,7,9,8,4,5→7,1,6,9,8,4,5→7,9,6,1,8,4,5→9,7,6,1,8,4,5→9,8,6,1,7,4,5

125. 下列关于大根堆 (至少包含两个元素) 的叙述中, 正确的是 ()

(1) 可以将堆视为一颗完全二叉树

(2) 可以采用顺序存储方式保存堆

(3) 可以将堆视为一棵二叉排序树

(4) 堆中的次大值一定在根的下一层

A. 1,2 **B.** 2,3 **C.** 1,2,4 **D.** 1,3,4

126. 若对 27 个元素值只进行三趟多路归并排序, 则选取的归并路数最少是 ()

A. 2 **B.** 3 **C.** 4 **D.** 5

127. 将两个各有 N 个元素的有序表合并为一个有序表, 最少的比较次数 (), 最多比较次数是 ()

A. N **B.** 2N-1 **C.** 2N **D.** N-1

128. 若要求排序是稳定的, 且关键字为实数, 则在下列排序中应该选用 ()

A. 直接插入排序 **B.** 选择排序 **C.** 基数排序 **D.** 快速排序

129. 下列排序算法中属于稳定排序的是 (), 平均时间复杂度为 $O(n\log n)$ 的是 (), 在最好的情况下, 时间复杂度可以达到线性的时间有 ()

A. 冒泡排序 **B.** 堆排序 **C.** 选择排序 **D.** 直接插入排序

E. 希尔排序 **F.** 归并排序 **G.** 快速排序

130. 若序列的原始状态为1,2,3,4,5,10,6,7,8,9 要想使得排序过程中元素比较次数最少, 则应该采用的是 ()

A. 插入排序 **B.** 选择排序 **C.** 希尔排序 **D.** 冒泡排序

131. ♦ 下列排序方法中, 若将顺序存储转换为链式存储, 则算法时间效率会降低的是 ()

A. 插入排序 **B.** 选择排序 **C.** 冒泡排序

D. 希尔排序 **E.** 堆排序

132. 设有 5 个初始归并段, 每个归并段有 20 个记录, 采用 5 路平衡归并排序, 若不采用败者树, 使用传统的顺序选择出最小记录 (简单选择排序) 的方法, 总的比较次数为 (); 若采用败者树最小的方法, 总的比较次数约为 ()

A. 20 **B.** 300 **C.** 396 **D.** 500

133. 在做 m 路平衡归并排序过程中, 为实现输入/内部归并/输出的并行处理, 需要设置 () 个输入缓冲区和 () 输出缓冲区.

A. 2 **B.** m **C.** 2m-1 **D.** 2m

134. ♦ 已知三叉树 T 中的 6 个叶结点的权分别是2,3,4,5,6,7,T 的带权路径长度最小是 ()

A. 27 **B.** 46 **C.** 54 **D.** 56

135. ♦ 设外存上有 120 个初始归并段, 进行 12 路归并时, 为实现最佳归并, 则需要补充的虚段个数是 ()

A. 1 **B.** 2 **C.** 3 **D.** 4

136. 下面算法中, 语句"x*=2;" 执行的次数是 ( )

```
int x = 1;
for (int i = 0; i < n; ++i)
    for (int j = 1; j < n; ++j)
        x *= 2;
```

137. 下列说法中不正确的是 ()

A. 数据元素是数据的基本单元

B. 数据项是数据元素中不可分割的最小可标记单位

C. 数据可由若干数据元素组成

D. 数据项可由若干个数据元素组成

138. 数据的四种基本存储结构是指 ____,____,____,____

139. 线性表是具有 n 个 ( ) 的有限序列.

A. 表元素 **B.** 数据元素 **C.** 数据项 **D.** 信息项

140. 对于没有尾指针的单链表, 将 n 个元素采用头插法建立单链表的时间复杂度为 ( ), 采用尾插法建立单链表的时间复杂度为 ( ).

141. (多选) 某线性表用带头结点的循环单链表存储, 头指针为 $head$, 当 $head \to next \to next \to next == head$ 成立的时候, 线性表的可能长度是 ( ).

142. (多选) 以下选项中正确的是 ( )

A. 静态链表即有顺序存储的优点, 又有动态链表的优点, 所以, 它存取表中 i 个元素的时间与 i 无关

B. 静态链表中能容纳的元素个数在表定义时就确定了, 在后续操作中不能增加

C. 静态链表与动态链表在元素的插入, 删除上类似, 不需要做元素的移动

D. 静态链表相比于动态链表有可能浪费存储空间

143. 稀疏矩阵的压缩存储的缺点在于 ( )

A. 无法得到矩阵的维数信息

B. 无法根据行列号查找矩阵的元素

C. 无法随机存取

D. 使矩阵的逻辑关系变得更加复杂

144. 若以行优先顺序存储三维数组 $A[80][20][40]$, 其中元素 $A[0][0][0]$ 所在的地址为 0, 且每个元素占有 4 个存储单元, 则 $A[20][10][3]$ 的地址为 ____.

145. ♦ 在一颗度为 4 的树中, 若有 20 个度为 4 的结点,10 个度为 3 的结点,1 个度为 2 的结点,10 个度为 1 的结点, 则树 T 的叶结点个数是 ____

146. 若二叉树非空, 具有 n 个结点且深度也是 n 的二叉树有 ( ) 种

147. 以下说法中正确的是 ( )

A. 完全二叉树中, 一个叶节点的左侧叔结点有孩子结点, 则该左侧叔结点一定不是叶结点

B. 任何一颗非空二叉树内 $n_0 = n_2 - 1$

C. 除了完全二叉树外, 其它任何二叉树都不适合顺序存储结构

D. 结点按完全二叉树层序编号的二叉树中 (从 0 开始), 第 i 个结点的左孩子 (若存在) 的编号为 2i

148. (多选) 假设一颗二叉树 T 的节点为 31, 则下列说法正确的是 ( )

A. T 的最小高度为 5, 最大高度为 31

B. T 中最少有一个叶子结点, 最多有 15 个叶子结点

C. 若 T 中分支结点的度均为 1, 则 T 的所有可能的形态共有 $2^3 1$ 种

D. 若 T 中分支结点的度均为 2, 则 T 必为完全二叉树, 可能为满二叉树

149. n 个结点的 k 叉树 ($k \geq 2$) 的 k 叉链表中有 _____ 个空指针

150. 若二叉树有两个结点 p, q, 对该树进行中序遍历,p 在 q 的前面则,()

A. p 是 q 的祖先 **B.** q 是 p 的祖先 **C.** p 在 q 的左边 **D.** q 在 p 的左边

151. (判断正误) 在二叉树的先序序列, 中序序列和后序序列中, 所有叶子结点的先后顺序相同.

152. 由层次序列 $ABCDEF$ 和中序序列 $BADCFE$, 可以唯一确定一颗二叉树, 则 T 的先序序列为 _____

153. 中缀表达式 $A + B * C - D/E$ 对应的前缀表达式是 _____

154. (多选) 一颗二叉树采用二叉链表表示, 若要采用递归的方法将其所有结点的左右子树交换位置, 则采用 () 遍历方法比较合适.

A. 先序 **B.** 中序 **C.** 后序 **D.** 层序

155. (多选) 下列关于先序线索树中查找结点的先序后继的说法中, 错误的是 ( )

A. 当指定结点不是叶结点时, 若指定结点有左孩子, 则左孩子就是他的先序后继, 若指定结点没有左孩子, 则右孩子是它的先序后继

B. 当指定结点是叶结点, 若指定结点是某结点 X 的左子树中先序遍历序列的最后一个结点, 且节点 X 有右孩子, 则指定结点的先序后继就是结点 X 的右孩子

C. 当指定结点是叶结点, 若指定结点是某结点 X 的左子树先先序遍历序列的最后一个结点, 但节点 X 没有右孩子, 则指定结点没有先序后继

D. 当指定结点是叶结点, 若指定节点不是任意结点左子树先序遍历的最后一个结点, 则指定节点先序后继是根结点

156. n 个结点的线索二叉树上含有的线索数为 _____

157. 高度为 H 的后序线索二叉树中,p 是其中一个结点,q 是 p 的左孩子,p 的右子树高为 H_1, 请问 q 到 q 的后继结点的连线路径上最多经过 _____ 结点 (不含路径的两个端点)

158. 对于一个线索化的二叉树, 其中 p 所指结点无左子树的充要条件是 ( )

A. $p \to lChlid == NULL$

B. $p \to ltag == 1$

C. $p \to ltag == 1 \&\& p \to lChlid == NULL$

D. 以上都不对

159. 一个具有 n 个非叶结点完全二叉线索树, 含有 _____ 条线索

160. 如果森林 F 采用" 孩子-兄弟" 表示法对应的二叉树是 16 个结点的完全二叉树, 森林 F 中树的数目和最大数的结点个数分别是 ( )

A. 2,8 **B.** 2,9 **C.** 4,8 **D.** 4,9

161. 设有 4 叉哈夫曼树, 结点到 4 个孩子结点的路径分别编码为00,01,10,11. 现对关键字序列 1,1,2,3,5,8,13,21 构建 4 叉哈夫曼树并进行编码, 下列说法正确的是 ( )

A. 最小带权路径长度为 108

B. 关键字 2 对应的编码长度为 6

C. 编码长度为 6 的关键字有 4 个

D. 010000,010001,0101,0110,0111,00,10,11 是一个合法序列

162. 在顺序有序表中 {2,5,7,10,14,15,18,23,35,41,52}, 用折半查找法查找关键字 14 的关键字比较次数为 () 用折半查找法查找关键字 6 的比较次数为 ()

163. 即希望较快查找有便于线性表动态变化的查找算法是 ()

A. 顺序查找 **B.** 折半查找 **C.** 索引顺序查找 **D.** 哈希法查找

164. 下列说法中, 正确的是 ()

A. 如果数据元素保持有序, 则查找时就可以采用折半查找法

B. 折半查找与二叉查找树的时间性能在最坏情况下相等

C. 折半查找法的速度一定比顺序查找法块

D. 折半查找法查找一个元素平均需要 $\log_2 n$ 次关键字比较

165. 下列说法中正确的是 ( )

A. 任何一颗含有 n 个节点的二叉树, 可以通过 $O(n)$ 次旋转, 转换为另一棵含有 n 个结点的二叉查找树

B. 满足任何一分支结点的值都小于其右孩子的值, 大于其左孩子的值的二叉树就是二叉查找树

C. 假设一棵 BST 中查找一个关键字 k, 查找结束语一个叶节点, 设 A 集合为查找路径左侧关键字的集合,B 是查找路径上的集合,C 是查找路径右侧关键字的集合. 则 $\forall a \in A, b \in B, c \in C, a \leq b \leq c.$

D. 一个序列仅能构成一种 AVL 树

166. 一颗具有 N 个结点的二叉排序树, 查找某个关键字的节点, 最多进行 () 次比较, 最少进行 () 次比较, 理想情况下查找叶子结点最多需要比较 () 次

167. 若平衡二叉树的结点树为 21, 则该树的高度至多是 ( )

168. 将关键字 $1, 2, 3, \ldots, 2016$ 插入初始为空的平衡二叉树, 假设只有一个根节点的二叉树高度为 0, 那么最终二叉树的高度为 ( )

169. 在 AVL 树中插入一个结点后造成了不平衡, 设最低的不平衡结点为 A, 已知在 A 的左孩子平衡因子为 0, 右孩子的平衡因子为 1, 则应该做 ( ) 型调整使其平衡.

A. LL **B.** LR **C.** RL **D.** RR

170. 在一颗具有 20 个关键字的 3 阶 B 树中, 含有关键字的结点个数最多是 () 最少是 ()

171. 已知一颗 5 阶 B 树有 53 个关键字, 且每个结点的关键字都达到了最少状态, 则它的深度是 (不包含叶子结点)()

172. (多选) 下列关于红黑树的说法中, 不正确的是 ( )

A. 若红黑树黑高为 h, 则最多有 $2^{2h} - 1$ 个内部结点

B. 若红黑树黑高为 h, 则最少有 $2^h - 1$ 个内部节点

C. 含有 n 个内部节点的红黑树, 高度不超过 $2\log_2(h+1)$

D. 红黑树中, 红结点的数量不会超过内部节点总数的一半

E. 具有 n 个关键字的红黑树中红的内部结点数与黑的内部结点树之比最大为 $2:1$

F. 如果一个结点是黑色的, 则它的父结点和孩子结点都可能是黑色

G. 插入 n 个结点形成的红黑树, 它至少有 1 个红色结点

H. 在通常情况下, 和含有相同结点数目的 AVL 树相比, 红黑树的查询效率较好

173. ♦ 高度为 5 的 3 阶 B 树含有的关键字个数至少是 ( )

174. 依次将关键字 5,6,9,13,8,2,12,15 插入初始为空的 4 阶 B 树后, 根结点中包含的关键字是 ()

175. 采用链地址法解决冲突的哈希表中, 查找成功的平均查找长度 ()

A. 直接与关键字个数有关　　**B.** 直接与装填因子有关

C. 直接与表的容量有关　　**D.** 直接与哈希函数有关

176. 采用开放定地法解决冲突的哈希查找中, 发生集聚的原因主要是 ()

A. 数据元素过多　　**B.** 负载因子过大

C. 哈希函数选择不当　　**D.** 解决冲突的算法选择不好

177. 下列说法中正确的是 ( )

A. 散列函数越复杂越好, 因为这样随机性好, 冲突概率小

B. 在散列查找中, 比较操作一般也是不可避免的

C. 若散列表的负载因子小于 1, 则可避免碰撞的产生

D. 若填充因子为 1, 则向散列表中散列元素时一定会产生冲突

178. ♦ 用哈希方法处理冲突时可能会产生堆积线性, 下列选择中, 会受堆积现象直接影响的是 ()

A. 存储效率　　**B.** 散列函数　　**C.** 装填 (装载) 因子　　**D.** 平均查找长度

179. 对数据序列 {8,9,10,4,5,6,20,1,2} 采用冒泡排序 (从后向前次序进行, 要求升序), 需要进行的趟数至少是 ()

180. 在一次遍历比较序列中查找最大数, 最小数. 最大值放在最右端, 最小的放在最左端, 同样缩小范围再次比较, 放在次右端, 次左端, 对数组 4,7,8,3,5,6,10,9,1,2 进行双向冒泡排序, 求排序趟数 ( )

181. 下列序列可能是快排第一趟所得到的序列是 ()

A. 68,11,18,69,23,96,73　　**B.** 93,73,68,11,69,23,18

C. 68,73,93,11,69,23,18　　**D.** 68,11,69,23,18,93,73

182. 在排序过程中, 对尚未确定最终位置的所有元素进行一遍处理称为一趟, 下列序列中, 不可能是快速排序第二趟的结果是 ()

A. 5,2,16,12,28,60,32,72 **B.** 2,16,5,28,12,60,32,72

C. 2,12,16,5,28,32,72,60 **D.** 5,2,12,28,16,32,72,60

183. 一组关键字为 {46,79,56,38,40,84} 则利用堆排序的方法建立大顶堆的初始堆为 ()

A. 76,46,56,38,40,84 **B.** 84,79,56,38,40,46

C. 84,79,56,46,40,38 **D.** 84,79,56,46,38,40

184. 对序列 {22,16,71,59,24,7,67,70,51} 进行堆排序形成小根堆, 删除 2 个堆顶元素后的剩余小根堆是 ()

185. 对一个初始状态为递增的序列进行按递增顺序的排序, 用 () 最省时, 用 () 最费时.

A. 直接插入排序 **B.** 堆排序 **C.** 快速排序 **D.** 归并排序

186. (多选) 如果一台计算机具有多核 CPU, 可以同时执行相互独立的任务. 归并排序的各个归并段也可以并行执行, 因此称归并排序是可以并行执行的, 以下排序方法不可以并行执行的有 ()

A. 基数排序 **B.** 快速排序 **C.** 冒泡排序 **D.** 堆排序

187. ♦ 对大部分元素已有序的数据进行排序, 直接插入排序比简单选择排序效率更高, 其原因是 ()

(1) 直接插入排序过程中元素之间的比较次数更少

(2) 直接插入排序过程中所需要的辅助空间更少

(3) 直接插入排序过程中的元素的移动次数更少

A. 1 **B.** 3 **C.** 1,2 **D.** 1,2,3

188. (多选题) 下列关于败者树和小根堆的描述中正确的是 ()

A. 败者树是从下往上维护, 每上一层, 只需要和败者结点比较一次即可

B. 堆在维护时从上往下, 每下一层, 需和左右子结点都比较, 需要比较 2 次

C. 败者树内的结点存储的内容是对应元素数值

D. 败者树每一次维护, 必定要从叶结点一直走到根结点, 不可能从中间停止

E. 堆维护一个结点位置的过程中必定要从根结点一直走到叶结点不可能从中间停止

F. 对于相同规模的处理对象, 败者树的构建需要的空间是小顶堆的 2 倍

189. (多选) 下列关于选择-置换排序的说法中, 正确的是 ()

A. 创建初始文件过程中, 需要不断的从内存工作区中选择不小于旧的 MINIMAX 的最小值, 此过程需要利用败者树实现

B. 不断选择新的 MINIMAX 记录时, 为防止新加入的关键字值更小, 每个叶结点都附加一个序号位, 当进行关键字比较时, 先比较序号, 序号大的位胜者; 序号相同的关键字值小的为胜者

C. 置换选择排序算法得到的初始归并段的长度可以超过内存容量限制, 且获得的归并短的平均长度为内存工作区大小的两倍

190. 设内存工作却能容纳 m 个记录, 那么对磁盘上的 n 个记录进行 k 路平衡归并排序, 需要做多少遍归并排序 ()

1.2 综合题

1. 选择-置换排序的功能是? 图示过程是?

2. 图示败者树的维护过程

3. 最佳归并树的虚拟节点是? 如何确定需要多少虚拟结点? 为啥需要虚拟结点?

1.3 选择题答案

1. 答案: $O(N)$

假设第 t 次时 $2^t \geq n$, 此时 sum 执行的总次数为

$$1+2+4+\ldots+2^t = \frac{1-2^t}{1-2} = 2^t - 1$$

又因为 $t = \log_2 n$ 故而总的计算执行次数为 $n-1 \sim O(n)$

2. 答案: B

3. 答案: B,A

4. 答案: A;

分析时间复杂度为, 对于 A 选项, 分别为 $O(1), O(1)$; 对于 B,C,D 选项, 分别为 $O(n), O(n)$

5. 答案: C

6. 答案: C;

用数组模拟栈, 具体的操作完全没比较记忆. 毕竟 $top = 0$ 或者 $top = -1$ 是完全不同的, 掌握原理才是根本.

考虑本题, 由于 $top = -1$, 每次进栈前, 应该先让 $top+ = 1$ 在执行赋值操作, 故应该为 $top[++top] = x$; 出栈操作为 $top--$;

> 区分 $++x$ 和 $x++$
>
> $++x$ 不会产生临时的副本, 而是直接将 $x+ = 1$ 的值返回给调用者;
>
> $x++$ 则会产生临时的 x 值, 并将 x 的值返回给调用者后, 将 $x+ = 1$;

7. 答案: A

顺序栈用数组实现, 而数字的大小是指定的 (不考虑动态申请数组空间的情况), 而链栈用链表实现申请结点空间较为容易. 故后者不容易出现栈满情况.

8. 答案: D

向将栈中的数据保存进 x 中, 再将 top 指针后移 (惰性删除, 并不直接释放空间).

9. 答案: 5 个

对于 n 不大的情况, 可以直接穷举. 假设三个元素为 a, b, c 则穷举出栈序列即可

(1)a, b, c (2)a, c, b (3)b, a, c

(4)b, c, a (5)c, b, a

求 n 个不同元素的出栈序列的个数

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

比如上题就可以用公式计算

$$\frac{1}{4} \cdot \frac{6*5*4}{3*2*1} = 5$$

10. 答案: A

11. 答案: A

C 语言中变量标识符 (变量名) 只能以 _ 或者字母开头不能以数组开头. 故只有 3 中可能的输出即

_1n　　n1_　　n_1

12. 答案: A

第一个栈从 $0 \ldots, top1$; 第二个栈从 $top2, \ldots, n-1$ 只要两个栈指针相遇即 $top1 + 1 == top2$ 的时候共享栈就满了, 此时 $top2 - top1 == 1$

13. 答案: D

14. 答案: C, 除 $P_2 = 3$ 外其余全部数

15. 答案: B

16. 答案: 入队:$rear = (rear + 1)\%(n + 1); A[rear] = x$

出队:$front = (front + 1)\%(n + 1)$

判空:$rear = front$

判满:$(rear + 1)\%(n + 1) == front$

当前队列中的元素: $(read - front + n + 1)\%(n + 1)$

17. 答案: D

如果队列中元素不止一个, 仅修改头指针; 但如果队列中的元素仅一个的时候, 就需要修改头指针和尾指针即 $rear == front$

18. 答案: A

插入即在链头插入, 此时时间复杂度是 $O(1)$ 但为了保持循环单链表的性质, 需要找到链尾 (队头) 元素, 此时需要 $O(n)$ 的时间复杂度.

19. 答案: B

20. 答案: A

21. 答案: 5 个

可以用栈直接做, 也可以转化为二叉表达式树做 (我感觉后者更好); 当然前者也不能不会.

22. 答案: A

23. 答案: (1) 可以记公式 $k = 2i + j - 3$

(2) 把三对角线矩阵画出来, 观察. 第一个行元素为两个, 在 $m_{30,30}$ 所在行之前有 28 行这些行有 3 个元素; 本行在 $m_{30,30}$ 之前仅有 $m_{30,29}$ 故 $2 + 28 \times +3 + 2 - 1 = 87$

24. 答案: (1) PM 数组结果 001231123456

(2) Next(1) 数组 -100123112345

(3) Next(2) 数组 011234223456

(4) Nextval 数组 010104210104

25. 答案: 10 次

26. 答案: 树的路径长度: 从树根到每个结点的路径长度的总和

树的带权路径长度 (WPL):

树中所有叶子结点的权值 X 该叶子到根的路径长度的总和

二者的区别在于, 前者计算内部节点与叶子结点; 而后者仅计算叶结点.

27. 答案: 错; 这二者的区别在于, 对于二叉树, 若一个结点仅有一个孩子, 这个子结点是左孩子还是有孩子是确定的; 而对于度为 2 的有序树, 这个孩子是左孩子还是右孩子是无所谓的错误; 左孩子不一定存在.

28. 答案: 基本公式, 对于二叉树有如下公式 $n_0 = n_2 + 1$;

故 $10 - 1 = 9$ 个度为 2 个结点

29. 答案: C

由于 $n_0 = n_2 - 1$ 故一颗二叉树的总结点数目为 $2n = n_1 + 2n_2 + 1$ 则 $n_1 = 2(n - n_2) - 1$ 显然 n_1 必须是奇数, 必然不可能有 $2m$(偶数) 个度为 1 的结点

30. 答案: 39

由于完全二叉树的特性可知, 当高度为 6 的时候结点数目最小且前 5 层必然是满二叉树,

故此时完全二叉树的结点个数最小是 $2^5 - 1 + 8 = 39$ 个结点

31. 答案: 501

解法一: 对于完全二叉树, 若其结点数为 n, 则最后一个分支结点 (含子结点) 的序号为 $x = \lfloor n/2 \rfloor$ 当 $n_i > x$ 的时候说明该结点是叶子结点. 故本题, 最后一个分支结点的序号为 $\lfloor 1001/2 \rfloor = 500$ 故叶结点的序号范围为 $501 \sim 1001$, 总数为 501

解法二: 由于 $n = 2n_0 + n_1 - 1$ 由于完全二叉树的定义可知 $n_1 = 0$ 或者 $n_1 = 1$ 带入可以知 $n_1 = 0, n_0 = 501$

32. 答案: 31

这道题比较容易错, 数字存储二叉树必须按照满二叉树存储, 因为并不能事先知道那些叶结点会有; 故这道题的答案是 $2^5 - 1 = 31$

33. 答案: C

34. 答案: D

35. 答案: 后序遍历

在遍历过程中, 从根 (或子树根 m 出发) 后序遍历会先走完 m 的整棵子树才回溯, 因此可以在访问到 n 时沿着递归栈或显式栈回溯, 从而得到 m→⋯→n 的路径

36. 答案: C

37. 答案: C

逻辑结构＝“是什么关系”——只关心数据之间的逻辑关系（线性、树、图等），与机器怎么存、存在哪儿无关。

物理结构＝“怎么存”——关心在内存/磁盘里到底怎么摆放（顺序表、链表、索引、散列、线索化等实现细节）。

二叉树本身是一种逻辑结构, 而线索二叉树是加上前后指针后的链式结构.

38. 答案: D

39. 答案: D

40. 答案: C

41. 答案: C

后序线索树中，根的后序后继是父结点，但普通线索二叉树没有保存父指针；当右子树非空时，无法通过线索直接得到该后继，仍需借助栈来回溯。

42. 答案: 14

简单来说等价于求 4 个元素的卡特兰数, 即

$$n = \frac{1}{n+1}C_{2n}^{n} = 14$$

任何一棵 n 个结点的二叉树, 其先序遍历序列与中序遍历序列的对应关系等价于 n 个元素的入栈顺序与出栈顺序的对应关系;

先序遍历与入栈顺序一致

中序遍历与出栈顺序一致

故上述问题转换为入栈顺序为a,b,c,d, 则出栈序列的个数是啥?

43. 答案: B

44. 答案: D

森林与二叉树的转换依据 (左孩子右兄弟表示法) 若森林内只有一颗树, 则右指针为空; 但若不止一棵树则右指针非空.

45. 答案: B

将森林的每棵树视为兄弟结点, 再按照左孩子右兄弟的规则来转换.

46. 答案: n+1

二叉树 B 中右指针为空的结点表明该结点无无兄弟结点.

其中森林中所有根结点中仅最右边的根的右指针为空, 所有中间结点的孩子中, 必然有且仅有一个孩子右指针为空 (n 个)

综上有 $n+1$ 个右指针为空

47. 答案: B

这个 III, 老头想问的是" 在原来森林中 u 的父结点和 v 的父结点" 是不是兄弟, 而不是问转换后的二叉树"u 的父结点和 v 的父结点" 在原来的森林中是不是兄弟关系.

48. 答案: 1896

这道题和 46 题考察的内容一致. 每个分支结点的最右孩子必然无右指针, 根结点也必然无 (单棵树), 故 $n - n_0 + 1 = 1896$

49. 答案: 10

对于一棵树其结点与边数满足 (n=e-1), 对于每棵树其节点数比边数多 1, 而本题结点比边数多 $25 - 15 = 10$ 棵树.

50. 答案: B

对于一颗多叉树与二叉树的转换后的遍历关系有如下:

先根遍历 (多叉树) $\rightarrow$ 先序遍历 (二叉树)

后根遍历 (多叉树) $\rightarrow$ 中序遍历 (二叉树)

多叉树没有所谓的中根遍历.

51. 答案: $n-1$

(二叉) 哈夫曼树的重要特征: 仅包含度为 0 或者度为 2 的节点. 又因为非空二叉树满足 $n_2 = n_0 - 1$ 故非叶结点总数为 $n-1$

52. 答案: 4

哈夫曼编码是前缀编码,3 位编码可以是 (000) 此时四位编码可以是 (0010,0011); 同理 3 为编码是 (001) 时候, 四位编码为 (0010,0011) 是不是答案就是 3 呢? 并不是, 并没有说一定要有 3 位编码, 若只用 4 为编码此时可能的编码有 (0000,0001,0010,0011) 有 4 种编码方式.

53. 答案: D

考虑哈夫曼树的构造过程, 每次选取和最小的两个结点作为叶结点. 而这两个结点何者为左结点, 何者为右结点是不确定, 故哈夫曼树通常是不唯一的.

54. 答案: $\lceil (n-1)/(m-1) \rceil$

由于哈夫曼树的特性可知,m 叉哈夫曼树仅有度为 0 和度为 m 的结点. 设总结点数为 $N = n_0 + n_m$ 又因为 N 个结点的哈夫曼树有 $N-1$ 条分支, 则 $mn_m = N - 1 = n_m + n_0 - 1$ 即 $n_m = (n\_1)/(m-1)$

55. 答案: A

注意构建哈夫曼树, 至于左右孩子谁为 0 谁为 1 是不确定. 可以带入选项判断.

56. 答案: D

57. 答案: C

树无环且连通, 边数恰好是 n-1; 而图没有这种限制.

B 选项第一眼很容易理解错, 子图还要求 E' 中每一条边对应的结点都存在于 V' 中, 否则不是合法图.

对于非连通图, 通过一个结点并不能一次遍历其余所有结点

58. 答案: C

A: 强连通 (连通) 图仅保证结点和结点之间有路径而不保证有弧 (边)

B: 只有无向图的入度等于出度, 而有向图并不一定满足

D: 如上题 C 一致.

59. 答案: $n-1; n$

最小的连通无向图即一颗树, 此时边为 (n-1); 而对于有向图, 构成一个有向环, 此时边为 (n);

60. 答案: $2n-2$

由于 408 数据结构仅考虑简单图, 每个结点最多只能与其余 $n-1$ 个结点存在两条互相指向的弧, 即每个顶点出度 = 入度 =$n-1$, 综上顶点的度之和为 $2n-2$

61. 答案: D

62. 答案: A

对于无向图 I 总是成立, 考虑一棵树其边数等于顶点数减一但此时是连通的, 所以 II 错误; 考虑成环的无向图, 此时每个点的度都为 2 的倍数, 故 III 错误.

63. 答案: D

这道题讲道理有歧义, 除非默认 0 表示无边否则应该选 B.

64. 答案: $(N^2-2N)/6$

不要忘记无向图表要存两次

65. 答案: $n(n-1)$

临接表有两部分, 一部分为顶点表 (存顶点) 一部分为边表 (存边); 从而当其是一个完全图的时候边数最多此时有 $n(n-1)$ 条边, 从而边表结点有 $n(n-1)$ 个

66. 答案: $O(n+e)$

分类加, 分步乘, 这里是分类;

删除顶点 v 的出边, 直接顺着顶点表, 删除这个顶点即可.$O(n)$

删除顶点 v 的入边, 遍历整张临接表的所有顶点, 并把以含 v 的边表结点删除, 此时每边至多遍历一次 $O(e)$

综上, 最终的时间复杂度为 $O(n+e)$

67. 答案: $O(n+e), O(n); O(n+e), O(n)$

空间复杂度不包含临接表的开销, DFS 主要是递归栈的消耗, 由于要遍历每个点一次最多需要压入 n 个点;BFS 使用队列实现, 最多同时有 n 个顶点入队.

68. 答案: $O(n^2), O(n); O(n^2), O(n)$

69. 答案: A

70. 答案: B; 一棵树可以通过一次 DFS 遍历所有点, 但并非所有连通图都是一棵树.

71. 答案: A

若环的权值之和为负数, 此时通过反复过环路径长度为无穷小, 不存在所谓最短路; 若环权值为 0, 此时完全不需要经过这个环可以直接删除; 若环权值之和为正数 $w > 0$, 设环形路径为 $P = s \to \ldots \to u \to C \to u \ldots \to t$ 其中 C 是一个环路, 把 C 删除得到新的路径为 $P' = s \to \ldots \to u \to u \ldots \to t$ 显然 $W(P') = W(P) - W(C) < W(P)$ 因此最短路必然不包含正权回路.

$Dijkstra$ 算法能够处理不含负权环路的最短路径而非不含环路, 对于零 (正) 权回路的最短路问题一样可以做

$Dijkstra$ 每次可以确定两个顶点间的最短路, 只要每个顶点都使用一次就可以确定任意顶点间的最短路

Floyd 维持的最优子结构是距离值的递推, 即

$$d_{ij}^k = \min\{d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\}$$

而非维护路径集合. Floyd 算法更新规则为若 $d_{ik}^{k-1} + d_{kj}^{k-1} < d_{ij}^{k=1}$ 直接把从 i 到 j 的新路径设置为

$$path[i][j] = path[i][k] + path[k][j]$$

整段可能被完全抛弃所以不符合路径子集.

72. 答案: D; 拓扑排序存在 $\iff$ 有向图无环

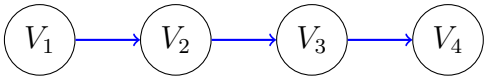
73. 答案: C

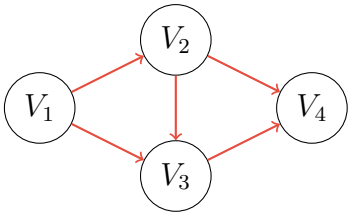
I: 有向图应该考虑行与列, 度 = 入度 + 出度 $= \sum_{j=1}^{n} A[j][i] + \sum_{j=1}^{n} A[i][j]$

II: 有向图只要入度等于出度, 其临接矩阵也是对称矩阵

III: 最小生成树只能保证其权值之和为所有生成树中权值最少, 但边权最少的边并不一定能组成一个生成树

IV: 考虑如下两图, 其拓扑排序一致但图不一致.





74. 答案: 21

75. 答案: C

76. 答案: C

题设条件只能保证有向图无环, 而不能保证拓扑排序唯一. 当前仅当每次确定拓扑序时只能找到一个入度为 0 的点, 此时拓扑排序唯一.

在离散数学中确实有拓扑序列唯一的充要条件 **G** 的 **Hanse** 图是一条链

77. 答案: C

需要熟练理解下面的过程, 很重要!

顶点	第一轮	第二轮	第三轮	第四轮	第五轮
b	$(a,b)2$	-	-	-	-
c	$(a,c)5$	$(a,b,c)3$	-	-	-
d	∞	$(a,b,d)5$	$(a,b,d)5$	$(a,b,d)5$	-
e	∞	∞	$(a,b,c,e)7$	$(a,b,c,e)7$	$(a,b,d,e)6$
f	∞	∞	$(a,b,c,f)4$	-	-
集合 S	(a,b)	(a,b,c)	(a,b,c,f)	(a,b,c,f,d)	(a,b,c,f,d,e)

78. 答案: 5,2,6,3,4

79. 答案: 12 和 14

AOE 网络的计算过程

具体参看笔记, 这里只给出上题的计算过程的表格.

时间的最早开始时间 (拓扑) 前驱结点的最早开始时间 + 对应活动之和的最大值

时间的最迟开始时间 (拓扑) 后继结点的最迟开始时间-对应活动之差的最小值

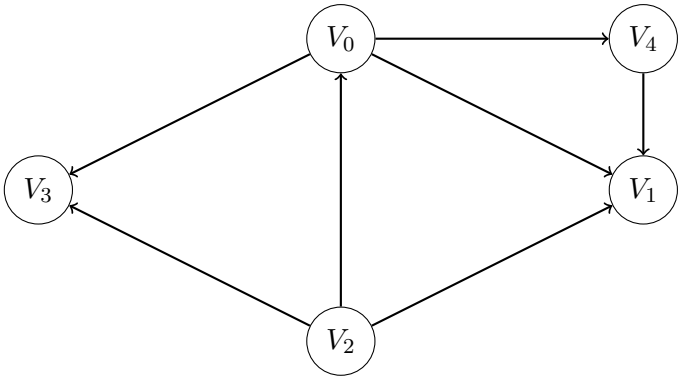
事件 (按拓扑序)	最早开始时间	最迟开始时间
1(源点)	0	0
3	0 + 8 = 8	$\min\{12-4, 18-10\} = 8$
2	$\max\{3, 8+4\}$ = 12	$\min\{19-7, 18-6\} = 12$
5	12+6=18	27-9=18
4	12+7=19	27-6=21
6(汇点)	$\max\{19+6, 18+9\} = 27$	27(关键路径长度)

事件的最早开始时间该事件 (弧) 对应的弧尾所表示时间的最早开始时间

事件的最迟开始时间该事件 (弧) 对应弧头所示的最迟开始时间与该活动持续时间之间

活动	最早开始时间	最迟开始时间
a	$EST[1] = 0$	$LST[2] - 3 = 9$
b	$EST[3] = 8$	$LST[2] - 4 = 8$
c	$EST[1] = 0$	$LST[3] - 8 = 0$
d	$EST[2] = 12$	$LST[4] - 7 = 14$
e	$EST[2] = 12$	$LST[5] - 6 = 12$
f	$EST[3] = 8$	$LST[5] - 10 = 8$
g	$EST[4] = 19$	$LST[6] - 6 = 21$
h	$EST[5] = 18$	$LST[6] - 9 = 18$

80. 答案: C



81. 答案: ?

82. 答案: ?

83. 答案: dfs, bfs

84. 答案: $O(n^2), O(e\log_2 e)$

85. 答案: 50

86. 答案: B

对于顺序查找, 不管线性表是有序的, 成功查找第一个元素的比较次数都是 1, 成功查找的第二个元素比较次数都是 2, 依次类推, 每个元素查找成功的比较次数只和位置有关而与线性表是否有序无关.

87. 答案: C

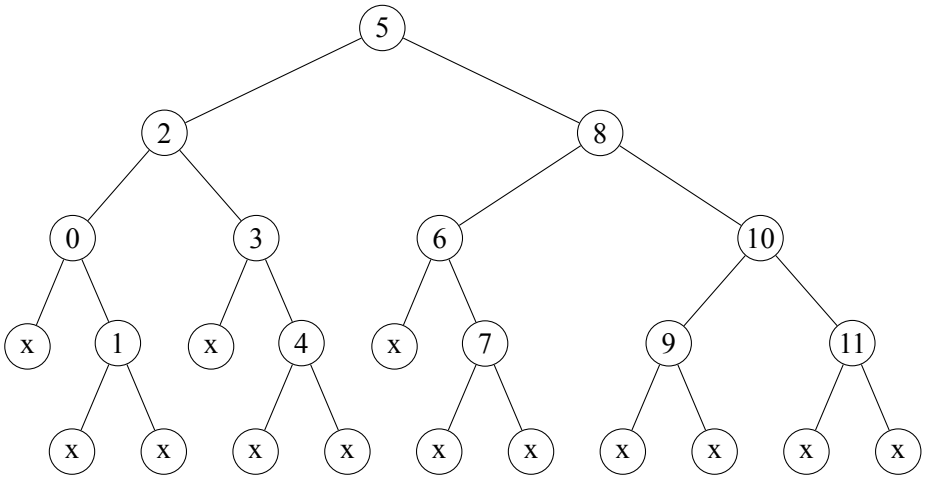
88. 答案: B

89. 答案: B

90. 答案: A

91. 答案: $\frac{37}{12}, \frac{49}{12}$

这种题比较理想的做法是画出该关键字序列的判定树, 用虚拟失败结点和叶结点计算失败与成功查找长度. 不妨假设含 12 个关键字的有序表为 $0, 1, \ldots, 11$ 此时其折半查找判定树为.



查找成功的

$$ASL = (1 + 2x2 + 3x4 + 4x5)/12 = \frac{37}{12}$$

查找失败的

$$ASL = (3x3 + 4x10)/12 = \frac{49}{12}$$

注意查找失败计算的时候不要多加虚拟的失败节点!

92. 答案: 16

为了使查找效率最高, 每个索引块的大小应该是 $\sqrt{n}=\sqrt{65025}=255$, 此时索引项的个数为 $\frac{65025}{255}=255$, 若此时采用折半查找, 效率最高,$2x\log_2(255+1)=16$

93. 答案: 查找失败最多只需要查找整个树高即 $\lceil\log_2(n+1)\rceil$

94. 答案: A

这道题第一眼可以以为是考察折半查找的判定树必然是一棵平衡树, 然后发现所有选项都是平衡树. 然后想呀想, 二分还有啥特性呢? 其决策树必然是一棵排序树, 但好像没啥用. 还有啥特性呢?

其实谜底在谜面上, 二分最重要的当然是确定分界点呀, 因此不妨根据中序有序把数字全部还原回去看看分界点的确定是否满足二分的要求.

95. 答案: n

二叉排序树最差会退化为单链表.

96. 答案: $\lceil\log_2 n+1\rceil$

97. 答案: 6,12

关键在于 AVL 结点树的递推公式,$n_0=0, n_1=1, n_h=n_{h-1}+n_{h-2}$

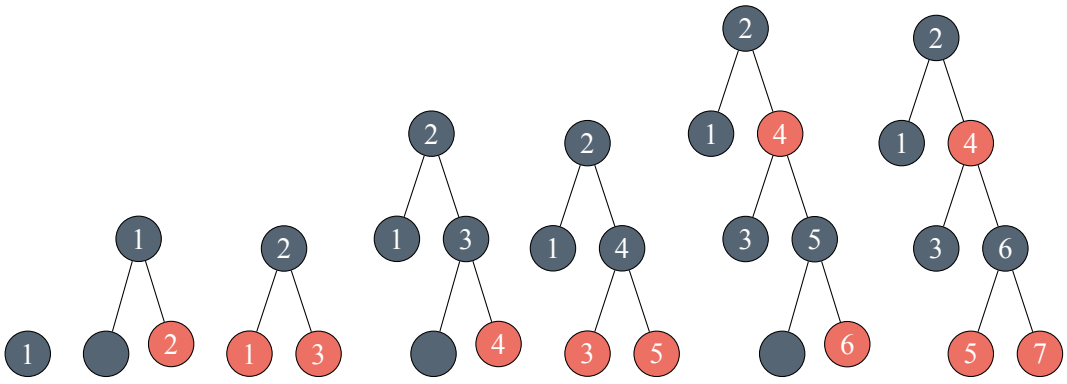
98. 答案: B

对于 B 选项, 红黑树的重要特征之一是黑平衡, 所以当一个红黑树全为黑结点必然是满二叉树, 否则就会破坏黑平衡条件

对于 D 选项, 由于红黑树要求根为黑结点, 所有根节点为红色的都不是红黑树.

99. 答案: 3 个

红黑树的插入过程比较重要 (考察可能性比删除高, 删除太难了.)



100. 答案: D

根据题设有左子树 > 根 > 右子树. 最小元素只能保证无右子树而不能保证是叶子结点. 最大元素可以保证无左子树.

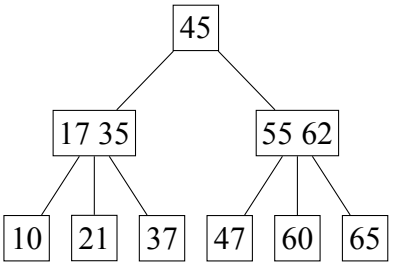
101. 答案: A

102. 答案: A

103. 答案: 65

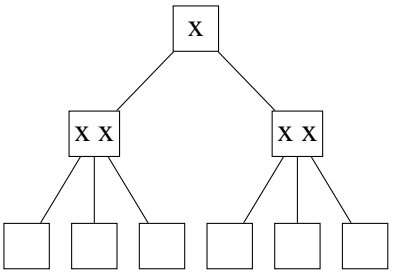
对于 3 阶 B 树, 其关键字范围为 $\lceil 3/2 \rceil \sim 3-1=1 \sim 2$

其左兄弟的关键字为 $2 \geq \lceil 3/2 \rceil$ 属于够接的情况, 删除后的 B 树如下



104. 答案: A

B 树的根结点至少有两个子结点 (包含一个关键字), 其余分支结点的关键字范围为 $\lceil 5/2 \rceil - 1 \sim 5-1=2 \sim 4$, 所以包含关键字最少的情况如下, 个数为 (1+2+2)



105. 答案: B

106. 答案: 填充因子

107. 答案: C

108. 答案: A

109. 答案: C; 仅和填充因子有关

110. 答案: C

哈希表的查找成功和查找失败的平均查找长度是重点, 需要掌握.

查找成功计算关键字的比较次数, 查找失败计算" 插槽".

最后哈希表如下所示

散列地址	0	1	2	3	4	5	6	7	8	9	10	11
关键字	98	22	30	87	11	40	6	20				

对于算出关键字算出的地址为 0, 需要比较 $0 \sim 8$ 地址的关键字才能确定失败; 对于关键字算出地址为 1, 需要比较 $1 \sim 8$, 一次类推; 需要注意原关键字序列算不出 7, 哈希表中的 20 是被线性探测改到的位置, 所以只有 7 个位置是可能的.

$$ASL_{fail} = \sum_{i=0}^{6} \frac{9-i}{7} = 6$$

111. 答案: D

112. 答案: B

注意并非所有排序方法都可以用于链表, 例如折半插入排序 (用于要使用二分) 链表就无法实现. 但大部分应该还是可以的.

113. 答案: A

对于任意 n 个关键字排序的比较次数, 其下界为 $\lceil \log_2(n!) \rceil$

114. 答案: B

插入排序中, 越解决正序插入次数越少.

115. 答案: A

116. 答案: C

117. 答案: A,D

118. 答案: B

119. 答案: D

120. 答案: D

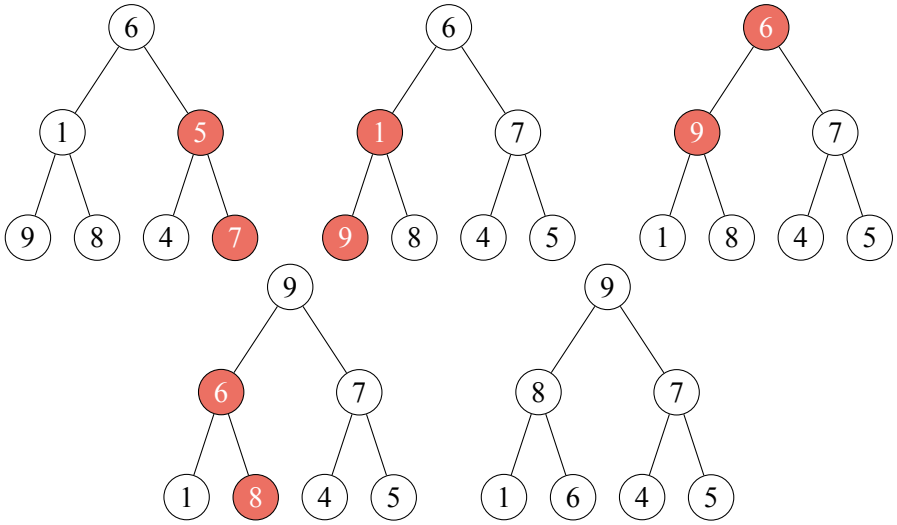
121. 答案: B

小根堆中最大元素必然位于叶结点, 而堆是一棵完全二叉树, 其最后一个非叶结点的编号为 $\lfloor n/2 \rfloor$, 所以关键字的存储范围为 $\lfloor n/2 \rfloor + 1 \sim n$

122. 答案: 这个题目一点都不好, 建堆要考虑是自上而下 ($O(n)$) 还是自下而上 ($O(n \log_2 n)$)

123. 答案: C

124. 答案: A



125. 答案: C

126. 答案: B

127. 答案: A,B

128. 答案: A

129. 答案: I,IV,VI

II,VI,VII

I,IV

130. 答案: A

131. 答案: DE

132. 答案: C, B

不采用败者树, 在 5 个记录中选出最小的需要 4 次比较, 从 100 个记录中选出最小的需要 99 次操作. 总共需要 $4 \times 99 = 396$

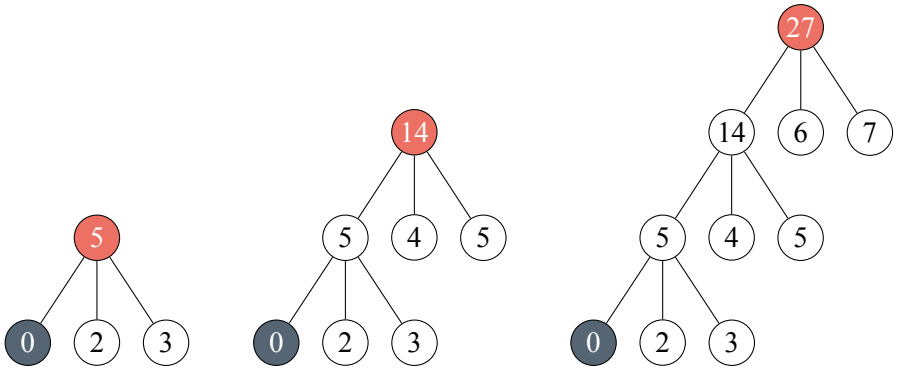
采用败者树, 败者树的高度为 $\lceil \log_2 5 \rceil = 3$, 每次确定一个关键字的最小记录不超过树高, 共 100 个记录, 需要比较的次数不多于 $3 \times 100 = 300$

133. 答案: D, A

由于要求并行, 所以需要 2m 个输入缓冲区,m 个用于读输入缓冲,m 个用于输入到内部排序;2 个外部缓冲区,1 个用于内部归并输出缓冲, 一个用于缓冲输出.

134. 答案: B

按照二叉 huffman 树的方法构建三叉 huffman 树, 构建过程如下



注意要加入虚拟结点!

135. 答案: B

注意多叉 huffman 树需要补充的都是叶子结点, 且其只有 n_{12} 和 n_0 的结点, 不妨设 $n_0 = 120 + n_{补}$ 又因为 $n_0 = (12-1)n_1 2 + 1$ 从而 $n_{12} = (120 - 1 + n_{补})(12-1)$ 由于 n_{12} 是整数, 从而 $n_{补} = 2$

136. 答案: