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++;
```

Solution

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. 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

Solution

选 B

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.** $head \neq NULL$

Solution

B,A

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

Solution

选 A, 可以注意分析时间复杂度. 容易得到

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. 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

Solution

和上题一样逐一分析时间复杂度, 选 C. 此时四种操作均可以达到 $O(1)$ 的时间复杂度.

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.** $a[top++] = x$

Solution

选 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. 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

Solution

选 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. 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.** $x = top \to data; top = top \to next$

Solution

选 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. 三个不同元素进栈, 能够得到 () 不同的出栈序列

Solution

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. 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

Solution

选 A, 不确定

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

Solution

答案选 A,3 个

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. 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

Solution

选 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. ♦ 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

Solution

没有技巧, 枚举的时候要考虑全面即可. 选 D

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

Solution

除 $P_2=3$ 外其余全部数, 即 $n-1$ 选 C

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

Solution

选 B, 模拟的时候注意题目要求 $b\ op\ a$,b 是第二次弹出的,a 是第一次弹出的.

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 ___

Solution

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

出队:$front = (front + 1)\%(n + 1)$

判空:$rear = front$

判满:$(rear + 1)\%(n + 1) == front$

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

Solution

选 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. 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.** $O(n\log_2 n)$

Solution

选 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. ♦ 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

Solution

选 B

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: $end2 == (end1+1)\ mod(M-1)$

Solution

选 A

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.

Solution

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. ▲ 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

Solution

选 A

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 ()

Solution

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

Solution

(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. ♦ 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 ()

Solution

10 次

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

Solution

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. (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 $2i$

Solution

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. 具有 10 个叶结点的二叉树中有 ( ) 个度为 2 的结点

Solution

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. 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

Solution

选 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. 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 ( )

Solution

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. 一颗完全二叉树上有 1001 个结点, 其中叶结点的个数是 ( )

Solution

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. ♦ 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 ( )

Solution

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. 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。

Solution

选 C, 这种题举反例即可, 注意题设.

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

Solution

选 D

C 选项要加上二者位于同一层, 才是充分条件.

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

Solution

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. 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

Solution

选 C

37. 线索二叉树是 ( ) 结构

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

Solution

选 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. 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

Solution

选 D

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

Solution

选 D

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

Solution

选 C

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

Solution

选 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. ♦ 先序序列为a,b,c,d 的不同二叉树的个数是 ( )

Solution

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. ♦ 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

Solution

选 B

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

Solution

选 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. 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 $m\neq 0$, 则 ( )

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

Solution

选 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. 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 ( )

Solution

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. ♦ 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

Solution

选 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. ♦ 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

Solution

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

Solution

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. ♦ 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

Solution

选 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. 在有 n 个叶节点的哈夫曼树中, 非叶结点的总数是 ( )

Solution

$n - 1$

(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. 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

Solution

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. 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

Solution

选 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. 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 ( )

Solution

$\lceil (n-1)/(m-1) \rceil$

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. ♦ 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

Solution

选 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. ♦ 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

Solution

选 D

57. 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

58. n 个顶点的无向图的邻接表中最多有 ( ) 个边表节点

59. 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 ( )

60. 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 ( )

61. 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 ( )

62. 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

63. 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

64. 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

65. 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

66. 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 ( )

67. 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

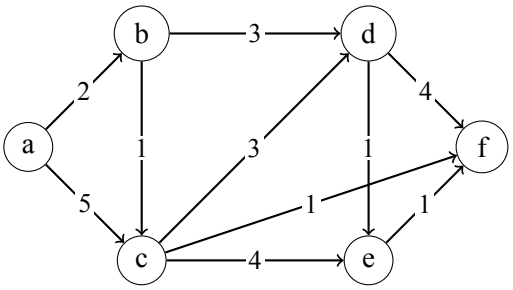
A. 2,5 **B.** 1,2,4 **C.** 2,4 **D.** 1,4

68. ♦ 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

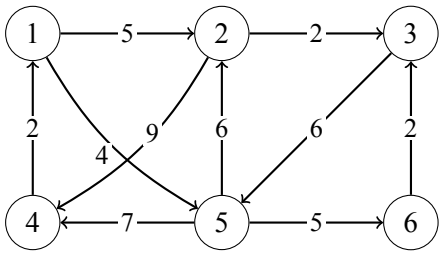
C. 存在, 可能唯一　　**D.** 无法确定是否存在

69. ♦ 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 ( )



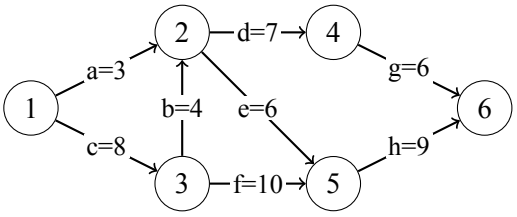
A. d,e,f　　**B.** e,d,f　　**C.** f,d,e　　**D.** f,e,d

70. ♦ 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

71. ♦ 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

72. 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

73. 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

74. 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

75. 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

76. 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

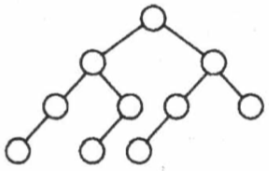
77. 具有 12 个关键字的有序表中, 对每个关键的查找概率相同, 折半查找算法查找成功的平均查找长度是 ( ), 折半查找失败的平均查找长度是 ( )

78. 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亦有元素最多需要执行 ( ) 次关键字比较

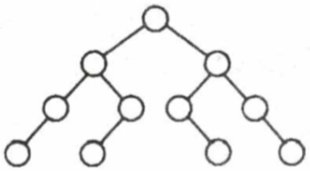
79. ♦ 已知一个长度为 16 的顺序表 L, 其元素按关键字有序排列, 若采用折半查找法查找一个 L 中不存在的元素, 则关键字的比较次数最多是 ( )

80. ♦ 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

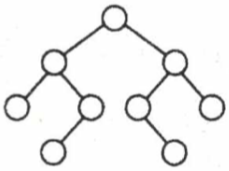
A.



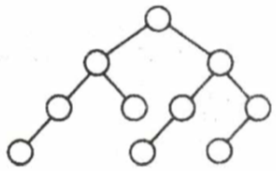
B.



C.



D.



81. 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

82. 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 ( )

83. 含有 20 个节点的平衡二叉树的最大深度为 ( )

84. 具有 5 层结点的 AVL 树至少有 ( ) 个结点

85. 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

86. ▲ 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 ( )

87. ♦ 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

88. ♦ 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

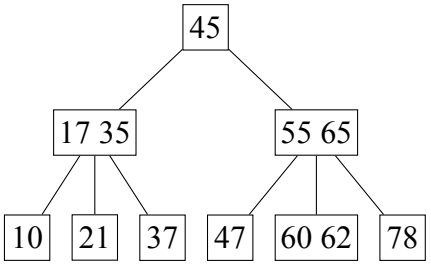
A. 1 **B.** 2 **C.** 1,2 **D.** 1,3

89. 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+ 树都是平衡的二叉树 **D.** B 树和 B+ 树都可以用于文件索引结构

90. ♦ 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 ( )



91. ♦ 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

92. ♦ 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

93. 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

94. 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

95. 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

96. ♦ 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

97. ♦ 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

98. 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

99. 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

100. 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

101. 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

102. 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

103. 一组记录的关键码为46,79,56,38,40,84, 采用快速排序, 以第一个记录为基准, 从小到大得到的一次划分结果为 ( )

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

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

104. 快速排序算法在 () 情况下不利于其发挥长处

A. 要排序的数据量太大 **B.** 要排序的数据中包含多个相同的值

C. 要排序的数据个数为奇数 **D.** 要排序的数据基本有序

105. 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

106. 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

107. 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

108. 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

109. 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

110. 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.** $O(n \log_2 n)$

111. 对关键字23,17,72,60,25,8,68,71,52 进行堆排序, 输出两个最小关键码后的剩余堆堆是 ()

A. 23,72,60,25,68,71,52 **B.** 23,25,52,60,71,72,68

C. 71,25,23,52,60,72,68 **D.** 23,25,68,52,60,72,71

112. 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

113. 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

114. 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

115. 若对 27 个元素值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

116. 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

117. 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

118. 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

119. 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

120. ♦ 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

121. 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

122. 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

123. ♦ 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

124. ♦ 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

1.2 综合题