

国开电大 2025《11252 数据结构（本）》期末考试题库小抄（按字母排版）
总题量（610）：单选题（400）判断题（177）复合题（33）

单选题（400）微信号：zydz_9527

1. [(] (1) 对关键字序列 (36, 69, 46, 28, 30, 74) 采用快速排序，以第一个关键字为分割元素，经过一次划分后的结果序列为__1__
... .C. 38, 36, 30, 46, 69, 74
D. 28, 36, , 30, 46, 69, 74; __1__ 答案：D
2. [(] (1) 对关键字序列 (36, 69, 46, 28, 30, 74) 采用快速排序，以第一个关键字为分割元素，经过一次划分后的结果序列为__1__
... .C. 38, 36, 30, 46, 69, 74
D. 28, 36, , 30, 46, 69, 74; __2__ 答案：A

3. [(] (1) 对关键字序列 (56, 51, 71, 54, 46, 106), 利用快速排序，以第一个关键字为分割元素，经过一次划分后结果为__1__;
... 60, 80, 30, 39, 47, 46) D. 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__ 答案：C
4. [(] (1) 对关键字序列 (56, 51, 71, 54, 46, 106), 利用快速排序，以第一个关键字为分割元素，经过一次划分后结果为__1__;
... 60, 80, 30, 39, 47, 46) D. 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__ 答案：D
5. [(] (1) 设查找表为 (1, 10, 11, 14, 23, 27, 29, 55, 68) , 画出对上述查找表进行折半查找所对应的判定树，为了成功查找到元素14... A. 24/9
B. 25 /9 C. 3
D. 2. 5
; __1__ 答案：C
6. [(] (1) 设查找表为 (1, 10, 11, 14, 23, 27, 29, 55, 68) , 画出对上述查找表进行折半查找所对应的判定树，为了成功查找到元素14... A. 24/9
B. 25 /9 C. 3
D. 2. 5
; __2__ 答案：B
7. [(] (1) 一组记录的关键字序列为 {45, 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__... 进行了__2__次元素间的比较。
A. 8 B. 11 C. 9
D. 10; __1__ 答案：C
8. [(] (1) 一组记录的关键字序列为 {45, 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__... 进行了__2__次元素间的比

- 较。
A. 8 B. 11 C. 9
D. 10; __2__ 答案：D
9. [(] (1) 一组记录的关键字序列为 {45, 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__... 2 次元素间的比较。
A. 8 B. 11 C. 9
D. 10; __1__ 答案：C
10. [(] (1) 一组记录的关键字序列为 {45, 40, 65, 43, 35, 95} 写出利用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为__1__... 2 次元素间的比较。
A. 8 B. 11 C. 9
D. 10; __2__ 答案：D
11. [(] (1) 一组记录的关键字序列为 (47, 80, 57, 39, 41, 46) , 利用堆排序的方法建立的初始堆为__1__ (堆顶元素是最小元素, 采用树的形...
C. 41, 57, 80, 47, 46
D. 41, 80, 46, 47, 57 ; __1__ 答案：B
12. [(] (1) 一组记录的关键字序列为 (47, 80, 57, 39, 41, 46) , 利用堆排序的方法建立的初始堆为__1__ (堆顶元素是最小元素, 采用树的形...
C. 41, 57, 80, 47, 46
D. 41, 80, 46, 47, 57 ; __2__ 答案：A
13. [(] (1) 以2, 3, 4, 7, 8, 9作为叶结点的权，构造一棵哈夫曼树，该树的带权路径长度为__1__
A, 66 B. ... 结点的哈夫曼编码为__2__。
A. 0001 B. 1110 C. 001
D. 110; __1__ 答案：B
14. [(] (1) 以2, 3, 4, 7, 8, 9作为叶结点的权，构造一棵哈夫曼树，该树的带权路径长度为__1__
A, 66 B. ... 结点的哈夫曼编

- 码为__2__。
A. 0001 B. 1110 C. 001
D. 110; __2__ 答案：C
15. [(] (1) 以3, 4, 5, 8, 9, 作为叶结点的权，构造一棵哈夫曼树。该树的带权路径长度为__1__。
A, 64... 2 。
A. 010 B. 0101
C. 000 D. 0111; __1__ 答案：B
16. [(] (1) 以3, 4, 5, 8, 9, 作为叶结点的权，构造一棵哈夫曼树。该树的带权路径长度为__1__。
A, 64... 2 。
A. 010 B. 0101
C. 000 D. 0111; __2__ 答案：C
17. [(] (1) 以3, 4, 5, 8, 9, 作为叶结点的权，构造一棵哈夫曼树。该树的带权路径长度为__1__。
A, 64 ...
A. 010 B. 0101
C. 000 D. 0111; Cloze Subject 答案：B
18. [(] (1) 以给定权重值5, 6, 17, 18, 25, 30, 为叶结点，建立一棵哈夫曼树，该树的中序遍历序列为__1__
5, 11, 28, 6, 17, 5... 夫曼为__2__。
A. 1001 B. 011 C. 001
D. 0001; __1__ 答案：B
19. [(] (1) 以给定权重值5, 6, 17, 18, 25, 30, 为叶结点，建立一棵哈夫曼树，该树的中序遍历序列为__1__
5, 11, 28, 6, 17, 5... 夫曼为__2__。
A. 1001 B. 011 C. 001
D. 0001; __2__ 答案：D
20. [(] (1) 以给定权重值5, 6, 17, 18, 25, 30, 为叶结点，建立一棵哈夫曼树，该树的中序遍历序列为__1__
A. 5, 11, 28, 6... 2 。
A. 1001 B. 011 C. 001
D. 0001; __1__ 答案：B

21. [()] (1) 以给定权重值5, 6, 17, 18, 25, 30, 为叶结点, 建立一棵哈夫曼树, 该树的中序遍历序列为 1

A. 5, 11, 28, 6... 2 .
A. 1001 B. 011 C. 001

D. 0001; _2_ 答案: D

22. [()] (1) 已知某二叉树的后序遍历序列是debca, 中序遍历序列是dbeac, 该二叉树的根结点是_1_

A. e ... a B.
c, a, b, , d, e C. a, b, d, e, c D.

a. c, b, d, e, ; _1_ 答案: D

23. [()] (1) 已知某二叉树的后序遍历序列是debca, 中序遍历序列是dbeac, 该二叉树的根结点是_1_

A. e ... a B.
c, a, b, , d, e C. a, b, d, e, c D.

a. c, b, d, e, ; _2_ 答案: C

24. [()] (1) 已知某二叉树的先序遍历序列是aecdb, 中序遍历序列是eadcb, 该二叉树的根结点是_1_;

A. e ..., b, c, a B.
c, a, b, , d, e C. a, b, d, e, c D.

a. c, b, d, e, ; _1_ 答案: D

25. [()] (1) 已知某二叉树的先序遍历序列是aecdb, 中序遍历序列是eadcb, 该二叉树的根结点是_1_;

A. e ..., b, c, a B.
c, a, b, , d, e C. a, b, d, e, c D.

a. c, b, d, e, ; _2_ 答案: A

26. [B] 把数据存储在计算机中, 并具体体现数据元素间的逻辑结构称为 ()。答案: 物理结构

27. [B] 表达式 $a*(b+c)-d$ 的后缀表达式是 ()。答案: $abc+*d-$

28. [()] () 不属于线性表的基本操作。答案: 求子表

29. [C] 采用分块查找时, 若线性表中共有324个元素, 查找每个元素的概率相同, 假设采用

顺序查找来确定结点所在的块, 每块应分 () 个结点最佳。答案: 18

30. [C] 采用分块查找时, 数据的组织方式为 ()。答案: 把数据分成若干块, 每块 (除最后一块外) 中的数据个数相等

31. [C] 采用邻接表存储的图的广度优先搜索遍历算法类似于二叉树的 ()。答案: 层次遍历

32. [C] 采用邻接表存储的图的深度优先搜索遍历算法类似于二叉树的 ()。答案: 先序遍历

33. [C] 采用顺序查找法对长度为 n (n 为偶数) 的线性表进行查找, 采用从前向后的方向查找。在等概率条件下成功查找到前 $n/2$ 个元素的平均查找长度为 ()。答案: $(n+1)/2$

34. [C] 采用顺序查找方法查找长度为 n 的线性表时, 每个元素的平均查找长度为 ()。答案: $(n+1)/2$

35. [C] 采用折半查找方法查找长度为 n 的线性表时, 其算法的时间复杂度为 ()。答案: $O(\log_2 n)$

36. [C] 常对数组进行的两种基本操作是 ()。答案: 查找和修改

37. [C] 串的长度是指 ()。答案: 串中所含字符的个数

38. [C] 串函数StrCmp("abA", "aba")的值为 ()。答案: -1

39. [C] 串函数Strcat(a, b)的功能是进行串 ()。答案: 连接

40. [C] 串是 ()。答案: 有限个字符的序列

41. [C] 串是一种特殊的线性表, 其特殊性体现在 ()。答案: 数据元素是一个字符

42. [C] 串与普通的线性表相比较, 它的特殊性体现在 ()。答案: 数据元素是一个字符

43. [C] 从未排序序列中依次取出元素与已经

排好序的序列中的元素作比较。将其放入已排序序列的正确的位置上, 此方法称为 ()。答案: 插入排序

44. [D] 带头结点的单向链表 L 为空的判定条件是 ()。答案: $L \rightarrow next == NULL$

45. [D] 带头结点的链表为空的判断条件是 () (设头指针为head)。答案: $head \rightarrow next == NULL$

46. [D] 带头结点的双向循环链表 L 为空表的条件是 ()。答案: $L \rightarrow next == L$

47. [D] 当利用大小为100的数组顺序存储一个队列时, 队列的最大长度为 ()。答案: 99

48. [D] 当利用大小为 N 的数组顺序存储一个栈时, 假定用 $top == -1$ 表示栈空, 则入栈应该执行 () 语句修改 top 指针。答案: $top++$

49. [D] 当利用大小为 N 的数组顺序存储一个栈时, 假定用 $top == N$ 表示栈空, 则入栈应该执行 () 语句修改 top 指针。答案: $top--$

50. [D] 当两个元素出现逆序的时候就交换位置, 这种排序方法称为 ()。答案: 交换排序

51. [()] () 的一个重要应用是解决主机和打印机之间速度不匹配的问题。答案: 队列

52. [()] () 的一个重要应用是在程序设计实现递归调用。答案: 栈

53. [D] 对二叉排序树进行 () 遍历, 可以使遍历所得到的序列是有序序列。答案: 中序

54. [D] 对具有 n 个元素的任意序列采用插入排序法进行排序, 排序趟数为 ()。答案: $n-1$

55. [D] 对链表, 以下叙述中正确的是 ()。答案: 不能随机访问任一结点

56. [D] 队列的删除操作是在 ()。答案: 队前

57. [D] 队列是一种操作受限的线性表, 其限

制是 ()。答案: 仅允许在表的一端进行插入, 而在另一端进行删除操作

58. [D] 对 n 个元素进行冒泡排序, 通常要进行 $n-1$ 趟冒泡, 在第 j 趟冒泡中共要进行 () 次元素间的比较。答案: $n-j$

59. [D] 对数据元素序列(49, 72, 68, 13, 38, 50, 97, 27)进行排序, 前三趟排序结果时的结果依次为第一趟: 49, 72, 68, 13, 38, 50, 97, 27; 第二趟: 49, 68, 72, 13, 38, 50, 97, 27; 第三趟: 13, 49, 68, 72, 38, 50, 97, 27。该排序采用的方法是 ()。答案: 插入排序法

60. [D] 对稀疏矩阵进行压缩存储, 可采用三元组表, 一个10 行8列的稀疏矩阵 A 共有73个零元素, A 的右下角元素为6, 其相应的三元组表中的第7个元素是 ()。答案: (10, 8, 6)

61. [D] 对稀疏矩阵进行压缩存储, 可采用三元组表, 一个10 行8列的稀疏矩阵 A 共有73个零元素, 其相应的三元组表共有 () 个元素。答案: 7

62. [D] 对稀疏矩阵进行压缩存储, 可采用三元组表, 一个10 行8列的稀疏矩阵 A , 其相应的三元组表共有6个元素, 矩阵 A 共有 () 个零元素。答案: 74

63. [D] 对线性表进行二分查找时, 要求线性表必须 ()。答案: 以顺序存储方式, 且数据元素有序

64. [D] 对线性表进行二分查找时, 要求线性表必需 ()。答案: 以顺序方式存储, 且结点按关键字有序排列

65. [D] 对序列(49, 38, 65, 97, 76, 13, 47, 50)采用直接插入排序法进行排序, 要把第七个元素47插入到已排序中, 为寻找插入的合适位置需要进行 () 次元素间的比较。答案: 5

66. [D] 对一个栈顶指针为 top 的链栈进行入栈操作, 通过指针变量 p 生成入栈结点, 并给该结点赋值 a , 则执行: $p = (\text{struct node} *) \text{malloc}(\text{sizeof}(\text{struct node}); p \rightarrow \text{data} = a;$ 和

()。答案: $p \rightarrow next = top; top = p;$

67. [D]对于顺序存储的有序表

{5, 12, 20, 26, 37, 42, 46, 50, 64}, 若采用折半查找, 则查找元素26的比较次数是()。

答案: 4

68. [D]对于一个具有n个顶点和e条边的无向图, 若采用邻接表表示, 则所有顶点邻接表中的结点总数为()。答案: $2e$

69. [D]对于一个具有n个结点的单向链表, 在给定值为x的结点之后插入一个新结点的时间复杂度为()。答案: $O(n)$

70. [D]对于一个链串s, 查找第一个字符值为x的算法的时间复杂度为()。答案: $O(n)$

71. [D]对于一个线性表, 若要求既能进行较快地插入和删除, 又要求存储结构能够反映数据元素之间的逻辑关系, 则应该()。答案: 以链接存储方式

72. [C]@@@ 二叉排序树结点类型定义如下:

```
typedef struct Bnode{    int key;  
    struct Bnode *le...ft;    elsep=p->right  
    if (p==NULL)        break;} return  
(p);} 答案:  $p \rightarrow key != k$ 
```

73. [E]二叉树的按层遍历算法需要使用()。答案: 队列

74. [E]二叉树的深度为k, 则二叉树最多有()个结点。答案: 2^{k-1}

75. [E]二叉树第k层上最多有()个结点。答案: 2^{k-1}

76. [E]二叉树是非线性数据结构, 所以()。答案: 顺序存储结构和链式存储结构都能存储

77. [F]非空的单向循环链表的尾结点满足() (设头指针为head, 指针p指向尾结点)。答案: $p \rightarrow next == head$

78. [G]关于哈希查找的说法正确的是()。答案: 哈希函数的好坏要根据具体情况而定

79. [G]关于栈和队列的说法中, 错误的是

()。答案: 栈是先进先出, 队列是后进先出

80. [G]广义表(a, a, b, d, e, ((i, j), k))的表头是_____。答案: a

81. [G]广义表(a, a, b, d, e, ((i, j), k))的表头是()。答案: a

82. [G]广义表(a, (a, b), d, e, ((i, j), k))的长度和深度分别是()。答案: 5, 3

83. [G]广义表((a))的表尾是()。答案: ()

84. [G]广义表((a))的表尾是()。答案: 0

85. [G]广义表(a, d, e, ((i, j), k))的表尾是()。答案: (d, e, ((i, j), k))

86. [G]广义表的(a, d, e, ((i, j), k))的表尾是_____。答案: (d, e, ((i, j), k))

87. [G]广义表(f, h, (a, b, d, c), d, e, ((i, j), k))的长度是()。答案: 6

88. [J]假定一棵二叉树中, 双分支结点数为15, 单分支结点数为30, 则叶子结点数为()。答案: 16

89. [J]假定一棵二叉树中, 叶子结点数为10, 单分支结点数为30, 则双分支结点数为()。答案: 9

90. [J]假设链队的队首和队尾指针是F和R, 那么队空的条件是()。答案: $R = NULL$

91. [J]假设在有序线性表A[1..20]上进行折半查找, 则比较五次查找成功的结点数为()。答案: 5

92. [J]将含有150个结点的完全二叉树从根这一层开始, 每一层从左到右依次对结点进行编号, 根结点的编号为1, 则编号为69的结点的双亲结点的编号为()。答案: 34

93. [J]将两个各有n个元素的有序表归并成一个有序表, 其最少的比较次数是()。

答案: n

94. [J]就排序算法所用的辅助空间而言, 堆排序、快速排序、归并排序的关系是()。答案: 堆排序 < 快速排序 < 归并排序

95. [K]空串的长度为()。答案: 0

96. [K]空串与空格串()。答案: 不相同

97. [L]利用2、4、5、10这四个值作为叶子结点的权, 生成一棵哈夫曼树, 该树的带权路径长度为()。答案: 38

98. [L]利用3、6、8、12这四个值作为叶子结点的权, 生成一棵哈夫曼树, 该树中所有叶子结点中的最长带权路径长度为()。答案: 18

99. [L]链表不具备的特点是()。答案: 可随机访问任一结点

100. [L]链表不具有的特点是()。答案: 可随机访问任一元素

101. [L]链表所具备的特点是()。答案: 插入删除元素的操作不需要移动元素结点

102. [L]链栈和顺序栈相比, 有一个比较明显的缺点, 即()。答案: 通常不会出现栈满的情况

103. [L]链栈和顺序栈相比, 有一个比较明显的优点, 即()。答案: 通常不会出现栈满的情况

104. [L]两个字符串相等的条件是()。答案: 两串的长度相等, 并且对应位置上的字符相同

105. [L]两个字符串相等的条件是()。答案: 两个串的长度相等且对应位置的字符相同

106. [L]邻接表是图的一种()。答案: 链式存储结构

107. [M]每次把待排序的区间划分为左、右两个子区间, 其中左区间中记录的关键字均小于等于基准记录的关键字, 右区间中记录

的关键字均大于等于基准记录的关键字, 这种排序称为()。答案: 快速排序

108. [M]每个存储结点只存储一个数据元素, 各结点存储在连续的存储空间, 该存储方式是()存储方式。答案: 顺序

109. [M]某串的长度小于一个常数, 则采用()存储方式最节省空间。答案: 顺序

110. [P]排序方法中, 从未排序序列中挑选元素, 并将其依次放入已排序序列(初始为空)的一端的方法, 称为()排序。答案: 选择

111. [P]排序方法中, 从未排序序列中依次取出元素与已排序序列(初始为空)中的元素进行比较(要求比较次数尽量少), 然后将其放入已排序序列的正确位置的方法, 是()排序。答案: 折半插入

112. [P]判断向上增长型的顺序栈空的条件是()。答案: $top == -1$

113. [P]判断一个顺序队列sq(最多元素为m)为空的条件是()。答案: $sq \rightarrow front == sq \rightarrow rear$

114. [P]判断一个顺序队列(最多元素为m)为空的条件是()。答案: $front == rear$

115. [P]判断一个循环队列Q(最多元素为m)为满的条件是()。答案: $Q \rightarrow front == (Q \rightarrow rear + 1) \% m$

116. [P]判断一个循环队列为满的条件是()。答案: $(rear + 1) \% MaxSize == front$

117. [P]判断栈满(元素个数最多n个)的条件是()。答案: $top == n - 1$

118. [Q]权值为{1, 2, 6, 8}的四个结点构成的哈夫曼树的带权路径长度是()。答案: 29

119. [R]任何一棵二叉树的叶结点在先序、中序和后序遍历序列中的()。答案: 相对次序不改变

120. [R]任何一棵二叉树的叶结点在先序、中序和后序遍历序列中的相对次序()。答案: 不发生改变

121. [R]任何一棵无向连通图的最小生成树 ()。答案: 有一棵或多棵

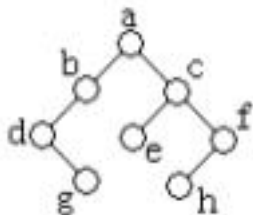
122. [R]如果将给定的一组数据作为叶子数值, 所构造出的二叉树的带权路径长度最小, 则该树称为 ()。答案: 哈夫曼树

123. [R]如果进行串的比较, 下列哪个串最大? () 答案: "BEIJING"

124. [R]如果要求一个线性表既能较快地查找, 又能动态适应变化要求, 可以采用 () 查找方法。答案: 散列

125. [R]如果以链表作为栈的存储结构, 则退栈操作时 ()。答案: 必须判断栈是否空

126. [R]如图所示二叉树的中序遍历序列是



答案:

dgbaechf

127. [R]若Head为一个带头结点的单链表的表头指针, 则该表为空表的条件是 ()。答案: Head->next==NULL

128. [R]若待排序序列在排序前已按关键字递增排列, 则采用 () 方法比较次数最多。答案: 直接插入排序

129. [R]若某线性表最常用的操作是存取任一指定序号的元素和在最后进行插入和删除运算, 则利用 () 存储方式最省时间。答案: 顺序表

130. [R]若让元素1, 2, 3依次进栈, 则出栈顺序不可能为 ()。答案: 3, 1, 2

131. [S]设a, b为一棵二叉树的两个结点, 在后续遍历中, a在b前的条件是 ()。答案: a在b左方

132. [Q]@@@设查找表为:

序号	1	2	3	4	5	6	7
序列	4	12	18	19	37	55	65

用折半查找在该查找表成功查找到元素55需要经过 () 次比较。答案: 2

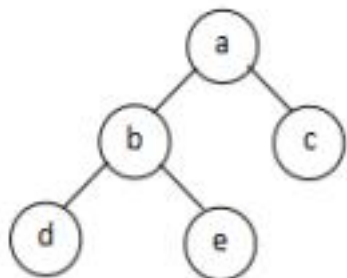
133. [Q]@@@设关键字序列为: (36, 69, 46, 28, 30, 74), 将此序列用快速排序的方法, 以第一个记录为基准得到一趟划分的结果为 ()。答案: 30, 28, 36, 46, 69, 74

134. [S]设广义表类((a, b, c)), 则L的长度和深度分别为 ()。答案: 1和2

135. [S]设head为非空的单向循环链表头指针, p指向链表的尾结点, 则满足逻辑表达式 () 的值为真。答案: p->next==head

136. [S]设链表中的结点是NODE类型的结构体变量, 且有NODE *p; 为了申请一个新结点, 并由p指向该结点, 可用以下语句 ()。答案: p=(NODE*)malloc(sizeof(NODE));

137. [Q]@@@设某二叉树先序遍历为abdec, 中序遍历为dbeac。该二叉树的图形是 ()。答案:



138. [S]设某一二叉树先序遍历为abdec, 中序遍历为dbeac, 则该二叉树后序遍历的顺序是 ()。答案: debca

139. [S]设某一二叉树中序遍历为badce, 后序遍历为bdeca, 则该二叉树先序遍历的顺序是 ()。答案: abcde

140. [Q]@@@设数据序列为:

{53, 30, 37, 12, 45, 24, 96}, 从空二叉树开始逐个插入该数据序列来形成二叉排序树, 若希望高度最小, 应该选择的序列是 ()。答案: 37, 24, 12, 30, 53, 45, 96

141. [S]设顺序存储的线性表长度为n, 对于插入操作, 设插入位置是等概率的, 则插入一个元素平均移动元素的次数为 ()。答案: n/2

142. [S]设top是一个链栈的栈顶指针, 栈中每个结点由一个数据域data和指针域next组成, 设用x接收栈顶元素, 则取栈顶元素的操作为 ()。答案: x=top->data;

143. [S]设头指针为head的非空的单向链表, 指针p指向尾结点, 则通过以下操作 () 可使其成为单向循环链表。答案: p->next=head;

144. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... {printf("%d\n", [[1]];  
        [[2]];  
        }while[[3]];  
};
```

1答案: p->data

145. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... {printf("%d\n", [[1]];  
        [[2]];  
        }while[[3]];  
};
```

2答案: p=p->next

146. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... {printf("%d\n", [[1]];  
        [[2]];  
        }while[[3]];  
};
```

3答案: p!=NULL

147. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... f("%d\n", __ (1) __;  
        __ (2) __;  
        }while __ (3) __;  
};选项1答案: p->data
```

148. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... f("%d\n", __ (1) __;  
        __ (2) __;  
        }while __ (3) __;  
};选项2答案: p=p->next
```

149. [S]设线性表以不带头结点的单向链表存储, 链表头指针为head, 以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。

```
#de... f("%d\n", __ (1) __;  
        __ (2) __;  
        }while __ (3) __;  
};选项3答案: p!=NULL
```

150. [Q]@@@设线性表以不带头结点的单向链表存储, 链表头指针为head。以下程序的功能是输出链表中各结点中的数据域data, 完成程序中空格部分。 ... { printf("%d\n", p->data); p=p->next; } while (____); } 答案: p!=NULL

151. [S]设一棵采用链式存储的二叉树, 除叶结点外每个结点数都为2, 该树结点中共有20个指针域为空, 则该树有 () 个叶结点。答案: 10

152. [S] 设一棵哈夫曼树共有 $2n+1$ 个结点, 则该树有 () 个非叶结点。答案: n

153. [S] 设一棵哈夫曼树共有 n 个非叶结点, 则该树有 () 个叶结点。答案: $n+1$

154. [S] 设已有 m 个元素有序, 在未排好序的序列中挑选第 $m+1$ 个元素, 并且只经过一次元素间的交换, 就使第 $m+1$ 个元素排序到位, 该方法是 ()。答案: 简单选择排序

155. [S] 设有1000个无序的元素, 希望用最快的速度挑选出其中前10个最大的元素, 最好选用 () 排序法。答案: 堆排序

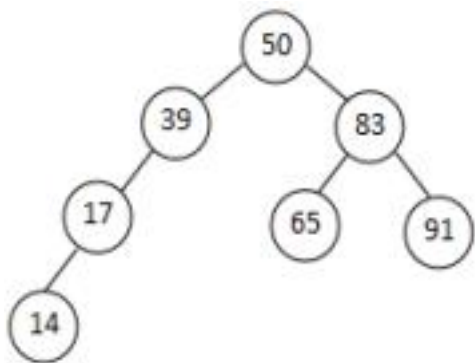
156. [S] 设有两个长度为 n 的单向链表, 结点类型相同, 分别是循环链表和非循环链表, 则 ()。答案: 对于两个链表来说, 删除最后一个结点的操作, 其时间复杂度都是 $O(n)$

157. [S] 设有两个串 p 和 q , 其中 q 是 p 的子串, q 在 p 中首次出现的位置的算法称为 ()。答案: 匹配

158. [S] 设有如下遗产继承规则: 丈夫和妻子可以互相继承遗产, 子女可以继承父亲和母亲的遗产, 子女间不能相互继承, 则表示该遗产继承关系最合适的数据结构应该是 () 结构。答案: 图状

159. [Q]@@@ 设有数据集合 $\{50, 39, 17, 83, 91, 14, 65\}$, 此二叉排序树的 () 遍历是有序序列。答案: 中序

160. [Q]@@@ 设有数据集合 $\{50, 39, 17, 83, 91, 14, 65\}$, 依次取集合中各数据构造一棵二叉排序树, 是如下的 ()。答案:



161. [S] 设有一个10阶的对称矩阵 A , 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), 则矩阵中元素 $A_{8,5}$ 在一维数组 B 中的下标是 ()。答案: 33

162. [S] 设有一个18阶的对称矩阵 A , 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), 则矩阵中元素 $A_{10,8}$ 在一维数组 B 中的下标是 ()。答案: 53

163. [S] 设有一个18阶的对称矩阵 A , 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), 则数组中第53号元素对应于矩阵中的元素是 ()。答案: $a_{10,8}$

164. [S] 设有一个20阶的对称矩阵 A (第一个元素为 $a_{1,1}$), 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), 则矩阵元素 $a_{6,2}$ 在一维数组 B 中的下标是 ()。答案: 17

165. [S] 设有一个长度为 n 的顺序表, 要删除第 i 个元素移动元素的个数为 ()。答案: $n-i$

166. [S] 设有一个长度为 n 的顺序表, 要删除第 i 个元素, 则需移动元素的个数为 ()。答案: $n-i$

167. [S] 设有一个长度为 n 的顺序表, 要在第 i 个元素之前 (也就是插入元素作为新表的第 i

个元素), 插入一个元素, 则移动元素个数为 ()。答案: $n-i+1$

168. [S] 设有一个带头结点的链队列, 队列中每个结点由一个数据域 $data$ 和指针域 $next$ 组成, $front$ 和 $rear$ 分别为链队列的头指针和尾指针。设 p 指向要入队的新结点 (该结点已被赋值), 则入队操作为 ()。答案: $rear \rightarrow next = p; rear = p;$

169. [S] 设有一个带头结点的链队列, 队列中每个结点由一个数据域 $data$ 和指针域 $next$ 组成, $front$ 和 $rear$ 分别为链队列的头指针和尾指针, 要执行出队操作, 用 x 保存出队元素的值, p 为指向结点类型的指针, 可执行如下操作: $p = front \rightarrow next; x = p \rightarrow data$; 然后执行 ()。答案: $front \rightarrow next = p \rightarrow next;$

170. [S] 设有一个对称矩阵 A , 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), B 数组共有55个元素, 则该矩阵是 () 阶的对称矩阵。答案: 10

171. [S] 设有一个对称矩阵 A , 采用压缩存储的方式, 将其下三角部分以行序为主序存储到一维数组 B 中 (数组下标从1开始), B 数组共有55个元素, 则矩阵是 () 阶的对称矩阵。答案: 10

172. [S] 设有一个广义表 $A(a)$, 其表尾为 ()。答案: ()

173. [S] 设有一个头指针为 $head$ 的不带头结点单向链表, p 、 q 是指向链表中结点类型的指针变量, p 指向链表中结点 a , (设链表中没有结点的数据域与结点 $a \dots$; $p = p \rightarrow next$; while($p \rightarrow data \neq x$))

```
{ q=p;
__ (2) __
}
```

$s \rightarrow next = p;$

__ (3) __; 选项1 答案: $q \rightarrow next \neq NULL$

174. [S] 设有一个头指针为 $head$ 的不带头结点单向链表, p 、 q 是指向链表中结点类型的

指针变量, p 指向链表中结点 a , (设链表中没有结点的数据域与结点 $a \dots$; $p = p \rightarrow next$; while($p \rightarrow data \neq x$))

```
{ q=p;
__ (2) __
}
```

$s \rightarrow next = p;$

__ (3) __; 选项2 答案: $p = p \rightarrow next$

175. [S] 设有一个头指针为 $head$ 的不带头结点单向链表, p 、 q 是指向链表中结点类型的指针变量, p 指向链表中结点 a , (设链表中没有结点的数据域与结点 $a \dots$; $p = p \rightarrow next$; while($p \rightarrow data \neq x$))

```
{ q=p;
__ (2) __
}
```

$s \rightarrow next = p;$

__ (3) __; 选项3 答案: $q \rightarrow next = s$

176. [S] 设有一个头指针为 $head$ 的不带头结点单向链表, p 、 q 是指向链表中结点类型的指针变量, p 指向链表中结点 a , (设链表中没有结点的数据域与结点 $a \dots$;

$q = p; p = p \rightarrow next;$

while($p \rightarrow data \neq x$))

```
{ q=p;
[[1]]
}
```

$s \rightarrow next = p;$

[[2]];

1 答案: $p = p \rightarrow next$

177. [S] 设有一个头指针为 $head$ 的不带头结点单向链表, p 、 q 是指向链表中结点类型的指针变量, p 指向链表中结点 a , (设链表中没有结点的数据域与结点 $a \dots$;

$q = p; p = p \rightarrow next;$

while($p \rightarrow data \neq x$))

```
{ q=p;
[[1]]
}
```



```
s->next=p;
[[2]];
2答案: q->next=s
```

178. [S] 设有一个头指针为head的不带头结点单向链表, p、q是指向链表中结点类型的指针变量, p指向链表中结点a, (设链表中没有结点的数据域与结点a...;

```
q=p; p=p->next;
while(p->data!=x)
{ q=p;
  [[1]]
}
```

```
s->next=p;
[[2]];
3答案: q->next!=NULL
```

179. [0]@@@设有一个头指针为head的单向链表中(结点类型为NODE), p为指向该链表中某个结点的指针。以下程序段为插入一个指针为s的结点, 使它成...选项。 NODE
*q;q=head; while(q->next!=p)
_____;s->next=p;q->next=s;答案: q=q->next

180. [S] 设主串为“ABcCDABcdEFaBc”, 以下模式串能与主串成功匹配的是()。答案: Bcd

181. [S] 使用折半查找法时, 要求查找表中各元素的键值必须是()排列的。答案: 递增或递减

182. [S] 树的()没有前驱结点, 其他结点有且仅有一个直接前驱结点。答案: 根结点

183. [S] 数据的存储结构包括数据元素的表示和()。答案: 数据元素间关系的表示

184. [S] 数据的存储结构包括数据元素的表示和()。答案: 数据元素间的关系的表示

185. [S] 数据结构是一门研究计算机中()对象及其关系的科学。答案: 非数值运算

186. [S] 数据结构中, 与所使用的计算机无关的是数据的()。答案: 逻辑结构

187. [S] 树形结构中数据元素之间的关系是()。答案: 一对多

188. [S] 树中所有结点的度等于所有结点数加()。答案: -1

189. [S] 数组a经初始化char a[]=“English”;a[1]中存放的是()。答案: 字符n

190. [S] 数组a经初始化char a[]=“English”;a[7]中存放的是()。答案: 字符串的结束符

191. [S] 树最适合于用来表示()。答案: 元素之间有包含和层次关系的数据

192. [S] 顺序队列中, 队首元素位置为5, 则队首指针位置为()。答案: 4

193. [S] 算法的时间复杂度与()有关。答案: 算法本身

194. [S] 算法的时间复杂度与()有关。答案: 算法本身

195. [S] 算法分析的两个主要方面是()。答案: 时间复杂性和空间复杂性

196. [T] 通常的使用顺序栈或者链栈实现递归算法, 下面哪个说法正确()。答案: 顺序栈和链栈性能基本相同

197. [T] 头指针为head的带头结点的单向链表为空的判定条件是()为真。答案: head->next==NULL

198. [T] 图的深度优先遍历算法类似于二叉树的()遍历。答案: 先序

199. [T] 图状结构中数据元素的位置之间存在()的关系。答案: 多对多

200. [W] 无向图的邻接矩阵是一个()。答案: 对称矩阵

201. [X] 稀疏矩阵采用压缩存储的目的主要是()。答案: 减少不必要的存储空间的开销

202. [X] 稀疏矩阵的压缩存储方式通常有两种, 即()。答案: 三元组和十字链表

203. [X] 下列的叙述中, 不属于算法特性的是()。答案: 可读性

204. [X] 下列关于串的叙述中, 不正确的是()。答案: 空串是由空格构成的串

205. [X] 下列广义表中的线性表是()。答案: E(a,b)

206. [X] 下列是”abcd321ABCD”的子串的选项是()。答案: ”21ABC”

207. [X] 下列说法不正确的是()。答案: 数据项可由若干个数据元素构成

208. [X] 下列说法不正确的是()。答案: 串不是线性结构

209. [X] 下列说法中, 不正确的是()。答案: 数据项可由若干个数据元素构成

210. [X] 下列有关二叉树的说法正确的是()。答案: 二叉树中度为0的结点的个数等于度为2的结点的个数加1

211. [X] 下列有关图遍历的说法不正确的是()。答案: 非连通图不能用深度优先搜索法

212. [X] 下面程序段的时间复杂度是()。

```
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++){
    c[i][j]=0;
    for(k=1;k<=n;k++){
      c[i][j]=c[i][j]+a[i][k]*b[k][j];
    }
  }
} 答案: O(n³)
```

213. [X] 下面的操作不是栈基本运算的是()。答案: 排序操作

214. [X] 下面的应用中, 不符合栈的后进先出特点的是()。答案: 算数运算、逻辑运算和关系运算

215. [X] 下面关于串的叙述中, 不正确的是()。答案: 空串是由空格构成的串

216. [X] 下面关于串的叙述中, 正确的是()。答案: 模式匹配是串的一种重要运算

217. [X] 下面关于二叉树的结论正确的是()。答案: 二叉树中, 度为0的结点个数等于度为2的结点个数加1

218. [X] 下面关于栈的基本运算算法中, 复杂度最高的是()。答案: 链栈清空运算

219. [X] 下面结论中不正确的是()。答案: 一个图按广度优先搜索法遍历的结果是唯一的

220. [X] 下面说法不正确的是()。答案: 图的深度遍历不适用于有向图

221. [X] 下述几种排序方法中, 要求内存量最大的是()。答案: 归并排序

222. [X] 线性表的顺序存储比链式存储最与利于进行()操作。答案: 表尾插入或删除

223. [X] 线性表的顺序结构中, ()。答案: 逻辑上相邻的元素在物理位置上也相邻

224. [X] 线性表在链式存储中各结点之间的地址()。答案: 连续与否无所谓

225. [X] 线性表只有以()方式存储, 才能进行折半查找。答案: 顺序

226. [X] 线性结构中数据元素之间的关系是()。答案: 一对一

227. [X] 向顺序栈中压入新元素时, 应当()。答案: 先移动栈顶指针, 再存入元素

228. [X] 向顺序栈中压入新元素时, 应当()。答案: 应当先移动栈顶指针, 再存入元素

229. [X] 向一个有127个元素的顺序表中插入一个新元素, 并保持原来的顺序不变, 平均要移动()个元素。答案: 5

230. [X] 向一个有127个元素的顺序表中插入一个新元素, 并保持原来的顺序不变, 平均要移动()个元素。答案: 63.5

231. [X] 序列状态为()时, 快速排序达到最好的时间复杂度。答案: 序列无序

232. [X] 循环队列A[m] 存放其元素, 用front和rear分别表示队头及队尾, 则循环队列满的条件是()。答案: (rear+1)%m=front

233. [Y]研究数据结构就是研究（ ）。答案：数据的逻辑结构和存储结构以及其数据在运算上的实现

234. [Y]依次将每两个相邻的有序表合并成一个有序表的排序方法称为（ ）。答案：归并排序

235. [Y]一棵采用链式存储的二叉树中有n个指针域为空,该二叉树共有（ ）个结点。答案：n-1

236. [Y]一棵二叉树采用链式存储, n个结点的二叉树共有（ ）个指针域为空。答案：n+1

237. [Y]一棵非空的二叉树, 先序遍历与后续遍历正好相反, 则该二叉树满足（ ）。答案：只有一个叶子结点

238. [Y]一棵 结点数 $31 < n < 40$ 的完全二叉树, 最后一层有4个结点, 则该树有（ ）个叶结点。答案：18

239. [Y]一棵具有16个结点的完全二叉树, 共有（ ）层。(设根结点在第一层) 答案：5

240. [Y]一棵完全二叉树共有5层, 且第5层上有六个结点, 该树共有（ ）个结点。答案：21

241. [Y]一棵完全二叉树共有6层, 且第6层上有6个结点, 该树共有（ ）个结点。答案：37

242. [Y]一棵有n个结点采用链式存储的二叉树中, 共有（ ）个指针域为空。答案：n+1

243. [Y]一维数组A采用顺序存储结构, 每个元素占用6个字节, 第6个元素的存储地址为100, 则该数组的首地址是（ ）。答案：70

244. [Y]一组记录的关键字序列为(25, 48, 16, 35, 79, 82, 23, 40, 36, 72), 其中, 含有5个长度为2的有序表, 按归并排序的方法对该序列进行一趟归并后的结果为（ ）。答案：16, 25, 35, 48, 23, 40, 79, 82, 36, 72

245. [Y]一组记录的关键字序列为(25, 50, 15, 35, 80, 85, 20, 40, 36, 70), 其中含有5个长度为2的有序表, 按归并排序的方法对该序列进行一趟归并后的结果为（ ）。答案：(15, 25, 35, 50, 20, 40, 80, 85, 36, 70)

246. [Y]一组记录的关键字序列为(26, 59, 36, 18, 20, 25), 利用堆排序的方法建立的初始小根堆为（ ）。答案：18, 20, 25, 59, 26, 36

247. [Y]一组记录的关键字序列为(36, 69, 46, 28, 30, 84), 利用快速排序, 以第一个关键字为分割元素, 经一次划分后结果为（ ）。答案：30, 28, 36, 46, 69, 74

248. [Y]一组记录的关键字序列为(37, 70, 47, 29, 31, 85), 利用快速排序, 以第一个关键字为分割元素, 经过一次划分后结果为（ ）。答案：31, 29, 37, 47, 77, 85

249. [Y]一组记录的关键字序列为(46, 20, 30, 79, 56, 38, 40, 84, 90, 110), 利用快速排序, 以第一个关键字为分割元素, 经过一次划分后结果为（ ）。答案：40, 20, 30, 38, 46, 56, 79, 84, 90, 110

250. [Y]一组记录的关键字序列为(46, 79, 56, 38, 40, 45, 62), 利用堆排序(堆顶元素是最小元素)的方法建立的初始堆为（ ）。答案：38, 40, 45, 79, 46, 56, 62

251. [Y]一组记录的关键字序列为(46, 79, 56, 38, 40, 45), 利用堆排序(堆顶元素是最小元素)的方法建立的初始堆为（ ）。答案：38, 40, 45, 79, 46, 56

252. [Y]一组记录的关键字序列为(46, 79, 56, 38, 40, 84), 利用快速排序, 以第一个关键字为分割元素, 经过一次划分后结果为（ ）。答案：40, 38, 46, 56, 79, 84

253. [Y]一组记录的关键字序列为(47, 80,

57, 39, 41, 46), 利用堆排序(堆顶元素是最小元素)的方法建立的初始化堆为（ ）。答案：39, 41, 46, 80, 47, 57

254. [Y]一组记录的关键字序列为(80, 57, 41, 39, 46, 47), 利用堆排序(堆顶元素是最小元素)的方法建立的初始堆为（ ）。答案：39, 46, 41, 57, 80, 47

255. [Y]一个存储结点存储一个（ ）。答案：数据元素

256. [Y]一个递归算法必须包括（ ）。答案：终止条件和递归部分

257. [Y]一个队列的入队序列是1, 2, 3, 4。则队列的输出序列是（ ）。答案：1, 2, 3, 4

258. [Y]一个队列的入队序列是2, 4, 6, 8, 则队列的输出序列是（ ）。答案：2, 4, 6, 8

259. [Y]一个队列的入队序列是a, b, c, d, 按该队列的可能输出序列使各元素依次入栈, 该栈的可能输出序列是（ ）。(进栈出栈可以交替进行)。答案：d, c, b, a

260. [Y]一个非空广义表的表头（ ）。答案：可以是子表或原子

261. [Y]一个具有n个顶点的无向完全图包含（ ）条边。答案： $n(n-1)/2$

262. [Y]一个具有n个顶点的有向完全图包含（ ）条边。答案： $n(n-1)$

263. [Y]一个顺序表第一个元素的存储地址是90, 每个元素的长度为2, 则第6个元素的地址是（ ）。答案：100

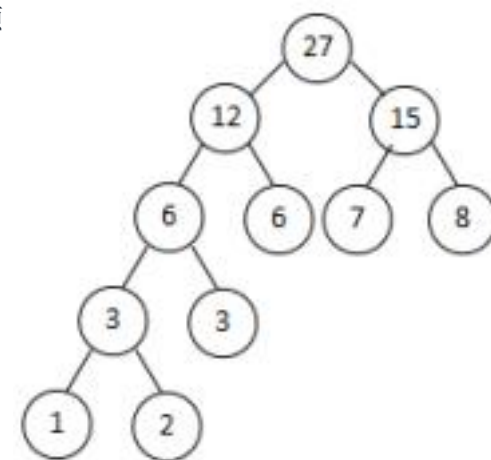
264. [Y]一个顺序存储线性表的第一个元素的存储地址是90, 每个元素的长度是2, 则第6个元素的存储地址是（ ）。答案：100

265. [Y]一个顺序栈一旦被声明, 其占用空间的大小（ ）。答案：已固定

266. [Y]一个栈的进栈序列是5, 6, 7, 8, 则栈的不可能的出栈序列是（ ）。(进

出栈操作可以交替进行) 答案：5, 8, 6, 7

267. [Q]@@@以1, 2, 3, 6, 7, 8作为叶结点的权, 构造一棵哈夫曼树是如下哪个图?（ ） 答案：



268. [Y]以下表中可以随机访问的是（ ）。答案：顺序表

269. [Y]以下陈述中正确的是（ ）。答案：串是一种特殊的线性表

270. [Q]@@@以下程序段的结果是：c的值为（ ）
char a[]=" abcdefghj"; int
*p=a, c=0; While (*p++) c++; 答案：9

271. [Y]以下程序是后序遍历二叉树的递归算法的程序, 完成程序中空格部分(树结构中左、右指针域分别为left和right, 数据域data为字符型, BT指向根结点)。完成程序中空格部

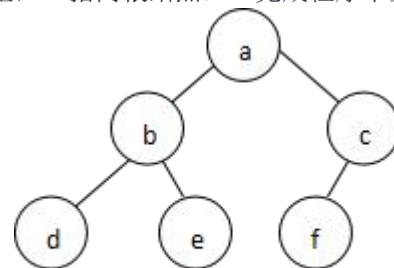


图 3

分。

void

```
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (2) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (3) __；

;选项1答案: Inorder(BT-> right)

272. [Y] 以下程序是后序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为字符型，BT指向根结点）。完成程序中空格部分。

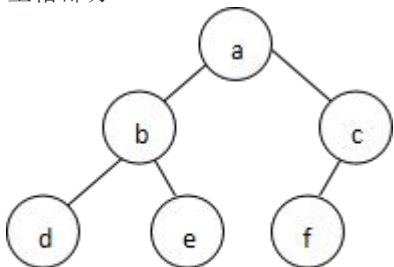


图 3

```
void
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (2) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (3) __；

;选项2答案: printf("%c" ,BT->data)

273. [Y] 以下程序是后序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为字符型，BT指向根结点）。完成程序中空格部分。

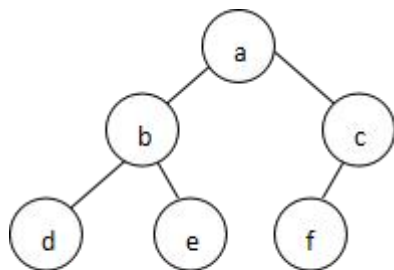


图 3

```
void
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (2) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (3) __；

;选项3答案: d, e, b, f, c, a

274. [Y] 以下程序是后序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为字符型，BT指向根结点）。完成程序中空格部分。

```
void
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (1) __
    __ (3) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (2) __；

1答案: Inorder(BT-> right)

275. [Y] 以下程序是后序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为字符型，BT指向根结点）。完成程序中空格部分。

```
void
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (1) __
}
```

```
__ (3) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (2) __；

2答案: d, e, b, f, c, a

276. [Y] 以下程序是后序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为字符型，BT指向根结点）。完成程序中空格部分。

```
void
Inorder (struct BTreeNode *BT)
{
    if( B...    __ (1) __
    __ (3) __
}
```

利用上述程序对左图进行后序遍历，结果是__ (2) __；

3答案: printf("%c" ,BT->data)

277. [0] 以下程序是先序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为left和right，数据域data为... ）。完成程序中空格部分。

```
void
Preorder (struct BTreeNode *BT)
{
    Preorder(BT->left);
    Preorder(BT->right);
    __ (1) __
}
```

278. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... - -)

```
__ (5) __=a[k];
a[j+1]=a[0];
}
```

;选项1答案: n

279. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... - -)

```
__ (5) __=a[k];
a[j+1]=a[0];
}
```

```
__ (3) __
}
```

;选项2答案: (s+j)/2

280. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... - -)

```
__ (5) __=a[k];
a[j+1]=a[0];
}
```

;选项3答案: j=m-1

281. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... - -)

```
__ (5) __=a[k];
a[j+1]=a[0];
}
```

;选项4答案: s=m+1

282. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... - -)

```
__ (5) __=a[k];
a[j+1]=a[0];
}
```

;选项5答案: a[k+1]

283. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... 1;k- -)

```
__ (3) __=a[k];
a[j+1]=a[0];
}
```

;选项1答案: (s+j)/2

284. [Y] 以下程序是折半插入排序的算法

设待排序的记录序列存放在a[1],...a[n]中，以a[0]作为辅助工作单元，程序是要把a[i] 插入到... 1;k- -)


```

        [[3]]=a[k];
        a[j+1]=a[0];
    }
};

```

2答案: j=m-1

285. [Y] 以下程序是折半插入排序的算法
 设待排序的记录序列存放在a[1], ..., a[n]
 中, 以a[0]作为辅助工作单元, 程序是要把
 a[i] 插入到...1;k- -)

```

        [[3]]=a[k];
        a[j+1]=a[0];
    }
};

```

3答案: a[k+1]

286. [Y] 以下程序是折半插入排序的算法
 设待排序的记录序列存放在a[1], ..., a[n]
 中, 以a[0]作为辅助工作单元, 程序是要把
 a[i] 插入到...1;k- -)

```

        [[3]]=a[k];
        a[j+1]=a[0];
    }
};

```

4答案: n

287. [Y] 以下程序是折半插入排序的算法
 设待排序的记录序列存放在a[1], ..., a[n]
 中, 以a[0]作为辅助工作单元, 程序是要把
 a[i] 插入到...1;k- -)

```

        [[3]]=a[k];
        a[j+1]=a[0];
    }
};

```

5答案: s=m+1

288. [Y] 以下程序是中序遍历二叉树的递归算
 法的程序, 完成程序中空格部分 (树结构中
 左、右指针域分别为left和right, 数据域
 data为字符型, BT指向根结点)。

```

void Inorder (struct BTreeNode *BT)

```

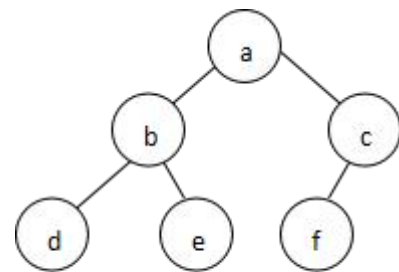


图 4

```

{
    LL) {
        Inorder(BT->left);
        __ (1) __;
        __ (2) __;
    }
}

```

利用上述程序对右图进行中序遍
 历, 结果是__ (3) __;

;选项1答案: printf(“%c”, BT->data)

289. [Y] 以下程序是中序遍历二叉树的递归算
 法的程序, 完成程序中空格部分 (树结构中
 左、右指针域分别为left和right, 数据域
 data为字符型, BT指向根结点)。

```

void Inorder (struct BTreeNode *BT)

```

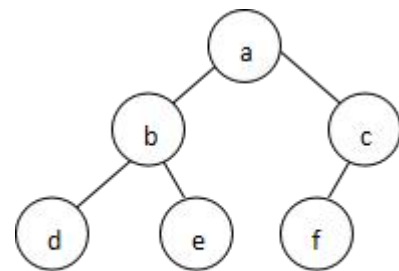


图 4

```

{
    LL) {
        Inorder(BT->left);
        __ (1) __;
    }
}

```

```

        __ (2) __;
    }
}

```

利用上述程序对右图进行中序遍
 历, 结果是__ (3) __;

;选项2答案: Inorder(BT->right)

290. [Y] 以下程序是中序遍历二叉树的递归
 算法的程序, 完成程序中空格部分 (树结构
 中左、右指针域分别为left和right, 数据域
 data为字符型, BT指向根结点)。

```

void Inorder (struct BTreeNode *BT)

```

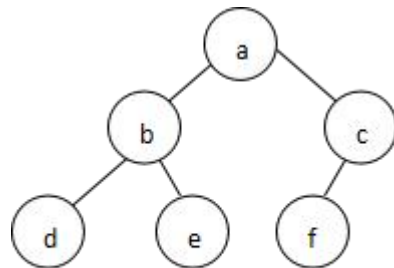


图 4

```

{
    LL) {
        Inorder(BT->left);
        __ (1) __;
        __ (2) __;
    }
}

```

利用上述程序对右图进行中序遍
 历, 结果是__ (3) __;

;选项3答案: d, b, e, a, f, c

291. [Y] 以下程序是中序遍历二叉树的递归
 算法的程序, 完成程序中空格部分 (树结构
 中左、右指针域分别为left和right, 数据域
 data为字符型, BT指... [[2]];

```

        [[3]];
```

利用上述程序对右图进行中序遍

历, 结果是[[1]];;

1答案: d, b, e, a, f, c

292. [Y] 以下程序是中序遍历二叉树的递归算法
 的程序, 完成程序中空格部分 (树结构中左、
 右指针域分别为left和right, 数据域data为字
 符型, BT指... [[2]];

```

        [[3]];
```

利用上述程序对右图进行中序遍历,
 结果是[[1]];;

2答案: printf(“%c”, BT->data)

293. [Y] 以下程序是中序遍历二叉树的递归算法
 的程序, 完成程序中空格部分 (树结构中左、
 右指针域分别为left和right, 数据域data为字
 符型, BT指... [[2]];

```

        [[3]];
```

利用上述程序对右图进行中序遍历,
 结果是[[1]];;

3答案: Inorder(BT->right)

294. [Y] 以下函数是二叉排序树的查找算法, 若
 二叉树为空, 则返回根结点的指针, 否则, 返
 回值是指向树结点的结构指针p (查找成功p指
 向查到的树结点, 不成功p...

```

    }
    return(__ (5) __;
}

```

;选项1答案: NULL

295. [Y] 以下函数是二叉排序树的查找算法, 若
 二叉树为空, 则返回根结点的指针, 否则, 返
 回值是指向树结点的结构指针p (查找成功p指
 向查到的树结点, 不成功p...

```

    }
    return(__ (5) __;
}

```

;选项2答案: k

296. [Y] 以下函数是二叉排序树的查找算法, 若
 二叉树为空, 则返回根结点的指针, 否则, 返
 回值是指向树结点的结构指针p (查找成功p指

向查到的树结点，不成功p...

```
    }  
    return(__ (5)__);  
}
```

;选项3答案: p=p->left

297. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功指向查到的树结点，不成功p...

```
    }  
    return(__ (5)__);  
}
```

;选项4答案: p=p->right

298. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功指向查到的树结点，不成功p...

```
    }  
    return(__ (5)__);  
}
```

;选项5答案: p

299. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功指向查到的树结点，不成功p...ak;

```
    }  
    return([[4]]);  
}
```

;

1答案: p=p->left

300. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功指向查到的树结点，不成功p...ak;

```
    }  
    return([[4]]);  
}
```

;

2答案: NULL

301. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功p指向查到的树结点，不成功p...ak;

```
    }  
    return([[4]]);  
}
```

3答案: p=p->right

302. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功p指向查到的树结点，不成功p...ak;

```
    }  
    return([[4]]);  
}
```

;

4答案: p

303. [Y] 以下函数是二叉排序树的查找算法，若二叉树为空，则返回根结点的指针，否则，返回值是指向树结点的结构指针p（查找成功p指向查到的树结点，不成功p...ak;

```
    }  
    return([[4]]);  
}
```

;

5答案: k

304. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= __ (3)__;  
};选项1答案: (sizeof (struct  
node))
```

305. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= __ (3)__;
```

```
};选项2答案: rear->next=p
```

306. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= __ (3)__;  
};选项3答案: p
```

307. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= [[2]];  
};
```

1答案: rear->next=p

308. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= [[2]];  
};
```

2答案: p

309. [Y] 以下函数为链队列的入队操作，x为要入队的结点的数据域的值，front、rear分别链队列的队头、队尾指针

```
struct node  
{ ... rear= [[2]];  
};
```

3答案: (sizeof (struct node))

310. [Y] 以下函数为链栈的进栈操作，x是要进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... __ (3)__;  
};选项1答案: sizeof (struct node)
```

311. [Y] 以下函数为链栈的进栈操作，x是要

进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... __ (3)__;  
};选项2答案: p->next=top
```

312. [Y] 以下函数为链栈的进栈操作，x是要进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... __ (3)__;  
};选项3答案: top=p
```

313. [Y] 以下函数为链栈的进栈操作，x是要进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... [[2]];  
};
```

1答案: A.sizeof (struct node)

314. [Y] 以下函数为链栈的进栈操作，x是要进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... [[2]];  
};
```

2答案: top=p

315. [Y] 以下函数为链栈的进栈操作，x是要进栈的结点的数据域，top为栈顶指针

```
struct node  
{ ElemType data;  
st... [[2]];  
};
```

3答案: p->next=top

316. [Y] 以下函数在a[0]到a[n-1]中，用折半查找算法查找关键字等于k的记录，查找成功返回

该记录的下标，失败时返回-1，完成程序中的空格选项。... mid; else if

(a[mid].key<k) low=mid+1; else high=mid-1; } return -1 } 答案: low<=high

317. [Q]@@@以下是冒泡排序算法对存放在a[1],a[2],...,a[n]中序列按关键字key由小到大排序，完成程序中空格部分。... temp=a[i]; a[i]=a[i+1];a[i+1]=temp; } if (flag==0) break;}} 答案: a[i].key>a[i+1].key

318. [Q]@@@以下是直接插入排序算法对存放在a[0],a[1],...,a[n-1]中，长度为n的记录序列按关键字key由小到大排序，完成程序中空格部...or (i=1;i=0&&temp.key<a[j].key){ a[j+1]=a[j]; _____; } a[j+1]=temp;}} 答案: j--

319. [Y]以下数据结构中()是线性结构。答案: 栈

320. [Y]以下说法正确的是()。答案: 栈的特点是先进后出，队列的特点是先进先出

321. [Y]以下说法正确的是()。答案: 答案: abcefd

二叉树中任一结点的值均大于其左孩子的值，小于其右孩子的值。则它是一棵二叉排序树。

322. [Y]以下说法中不正确的是()。答案: 已知单向链表中任一结点的指针就能访问到链表中每个结点

323. [Y]以下四个串中最小的是()。答案: "ABADF"

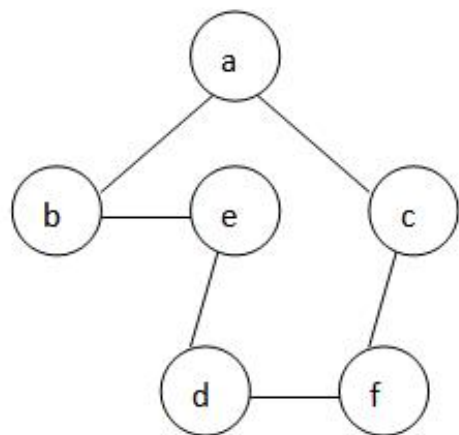
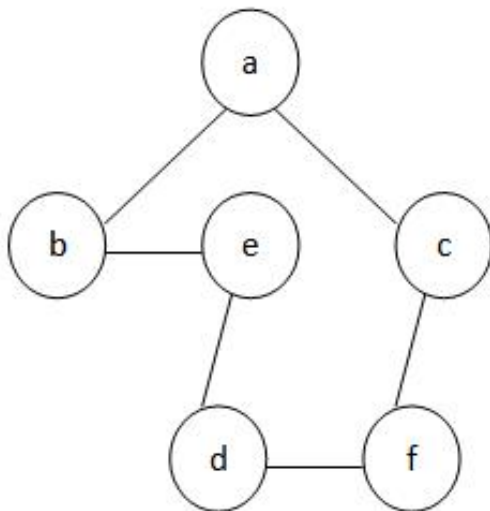
324. [Y]以下为求二叉树深度的算法，完成程序中空格部分。int BTreeDepth(BTreeNode* BT){ if (BT==... /* 计算右子树的深度 */ if () return dep1+1; else return dep2+1; } } 答案: dep1>dep2

325. [Y]已知10个数据元素为(54, 28, 16, 34, 73, 62, 95, 60, 26, 43)，对该数列从小到大排序，经过一趟冒泡排序后的序列为()。答案: 28, 16, 34, 54, 62, 73, 60, 26, 43, 95

326. [Y]已知某二叉树的后序遍历序列是

dabec，中序遍历是debac，则它的先序遍历序列是()。答案: cedba

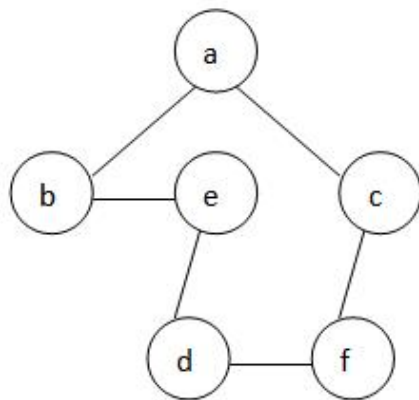
327. [Y]已知如图2所示的一个图，若从顶点a出发，按广度优先搜索法进行遍历，则可能得到的一种顶点序列为()。



案: aedfcb

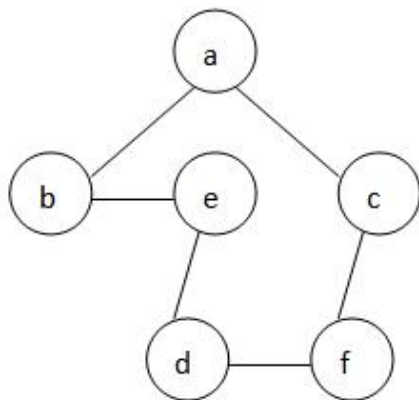
329. [Y]已知如图4所示的一个图，若从顶点a

出发，按深度优先搜索法进行遍历，则可能得到的一种顶点序列为()。



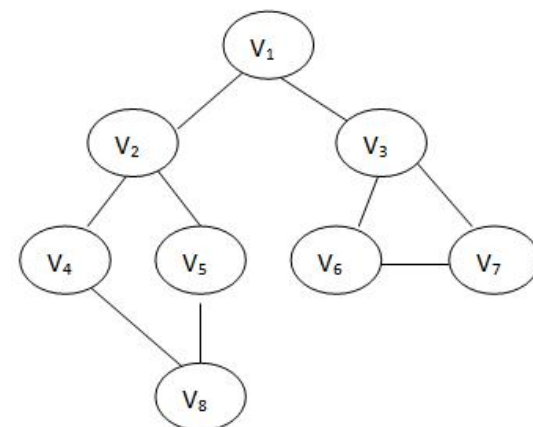
aedfcb

330. [Y]已知如图5所示的一个图，若从顶点a出发，按广度优先搜索法进行遍历，则可能得到的一种顶点序列为()。



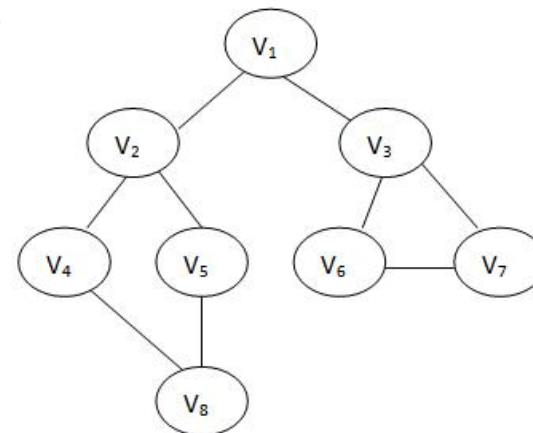
abcefd

331. [Y]已知如图6所示的一个图，若从顶点V₁出发，按深度优先搜索法进行遍历，则可能得到的一种顶点序列为()。



答案: V₁V₂V₄V₈V₅V₃V₆V₇

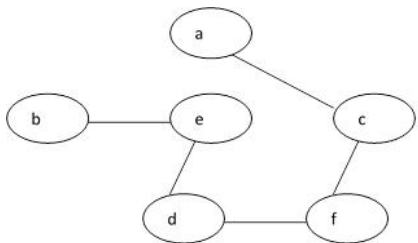
332. [Y]已知如图7所示的一个图，若从顶点V₁出发，按深度优先搜索法进行遍历，则可能得到的一种顶点序列为()。



答案: 答案: V₁V₂V₃V₄V₅V₆V₇V₈

333. [Y]已知如下图所示的一个图，若从顶点a出发，按广度优先搜索法进行遍历，则可能得到的一种顶点序列为()。

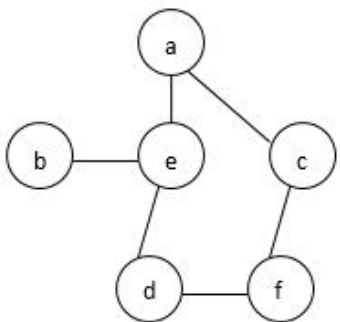
答



答案:

aecbdf

334. [Y] 已知下图所示的一个图, 若从顶点 V_1 出发, 按深度优先搜索法进行遍历, 则可能得到的一种顶点序列为 ()。



答案:

$V_1V_2V_4V_8V_5V_3V_6V_7$

335. [Y] 已知一个图的边数为 m , 则该图的所有顶点的度数之和为 ()。答案: $2m$

336. [Y] 已知一个有序表为 {11, 22, 33, 44, 55, 66, 77, 88, 99}, 则顺序查找元素55需要比较 () 次。答案: 5

337. [Y] 一般情况下, 将递归算法转换成等价的非递归算法应该设置 ()。答案: 栈

338. [Y] 用单链表表示的链式队列的队头在链表的 () 位置。答案: 链头

339. [Y] 用链表表示线性表的优点是 ()。答案: 便于插入和删除

340. [Y] 用某种排序方法对线性表 (25, 84, 21, 47, 15, 27, 68, 35, 20) 进行排序时, 元素序列的变化情况如下:

(1) 25, 84, 21, ... 20, 21, 25, 35, 27, 47, 68, 84

(4) 15, 20, 21, 25, 27, 35, 47, 68, 84 则采用的排序方法是 ()。答案: 快速排序

341. [Y] 有关线性表的正确说法是 ()。答案: 除了一个和最后一个元素外, 其余元素都有一个且仅有一个直接前驱和一个直接后继

342. [() () 有两个指针域, 分别指向直接前驱和直接后继, 可以实现从前向后和从后向前查找。答案: 双向循环链表

343. [Y] 有 n 个叶子结点的哈夫曼树的结点总数为 ()。答案: $2n-1$

344. [Y] 有数据 {53, 30, 37, 12, 45, 24, 96}, 从空二叉树开始逐个插入数据来形成二叉排序树, 若希望高度最小, 应该选择的序列是 ()。答案: 37, 24, 12, 30, 53, 45, 96

345. [Y] 有序表为 {2, 4, 10, 13, 33, 42, 46, 64, 76, 79, 85, 95, 120}, 用折半查找值为85的结点时, 经 () 次比较后成功查到。答案: 2

346. [Y] 有一个长度为10的有序表, 按折半查找对该表进行查找, 在等概率情况下查找成功的平均比较次数为 ()。答案: 29/10

347. [Y] 有一个长度为12的有序表, 按折半查找对该表进行查找, 在等概率情况下查找成功的平均比较次数为 ()。答案: 37/12

348. [Y] 与顺序表相比, 链表的优势是 ()。答案: 插入数据元素较快

349. [Z] 在C语言中, 存储字符串“ABCD”需要占用 () 字节。答案: 5

350. [Z] 在C语言中, 存储字符串“ABCD”需占用 () 字节。答案: 5

351. [Z] 在长度为 n ($n>1$) 的 () 上, 删除第一个元素, 其算法的时间复杂度为 $O(n)$ 。答案: 只有首结点指针 h 的不带头结点的单向循环链表

352. [Z] 在单链表中, 若 p 不是尾结点, 在其后插入 s 结点的操作是 ()。答案: $s->$

$next=p->next;p->next=s;$

353. [Z] 在二叉树的第4层最多含有 () 个结点。答案: 8

354. [Z] 在二维数组 $A[8][10]$ 中, 每一个数组元素 $A[i][j]$ 占用3个存储空间, 所有数组元素相继存放于一个连续的存储空间中, 则存放该数组至少需要的存储空间是 ()。答案: 240

355. [Z] 在非空双向循环链表的 $*p$ 结点之前插入 $*q$ 结点的操作是 ()。答案: $q->next=p;q->prior=p->prior;p->prior->next=q;p->prior=q;$

356. [Z] 在解决计算机主机与打印机之间速度不匹配问题时通常设置一个打印数据缓冲区, 主机将要输出的数据依次写入缓冲区中, 而打印机则从缓冲区中取出数据打印, 该缓冲区应该是一个 () 结构。答案: 队列

357. [Z] 在链队列中, f 和 r 分别为队头和队尾指针, 要把 s 所指结点入队, 应执行 ()。答案: $r->next=s;r=s;$

358. [Z] 在实际应用中, 要输入多个字符串, 且长度无法预定。则应该采用 () 存储比较合适 ()。答案: 链式

359. [Z] 在数据结构中, 从逻辑上可以把数据结构分成 ()。答案: 线性结构和非线性结构

360. [Z] 在双向循环双链表中, 删除 $*p$ 结点需要 ()。答案: $p->prior->next=p->next;p->next->prior=p->prior;$

361. [Z] 在所有的排序方法中, 关键字比较的次数与记录初始排列秩序无关的是 ()。答案: 直接选择排序

362. [Z] 在下面空格处填写一条语句, 以使下面的出栈算法完整。 ElemType Pop(struct SeqStack*s, ElemType x) { ... printf(“栈下溢错误! \n”); exit(1); } $x=s->data[s->top];$ _____ return x; } 答案: $s->top--;$

363. [0] @@@ 在下面空格处填写一条语句, 以使下面的串比较算法完整。 int strcmp(char *s1, char *s2) { int i; ... if (s1[i] != '\0') return 1; else return -1; } 答案: return 0;

364. [0] @@@ 在下面空格处填写一条语句, 以使下面的串连接算法完整。 char *strcat(char *s1, char *s2) { char *p... } { *p=*s2; p++; _____ } *p='\0'; return s1; } 答案: s2++;

365. [0] @@@ 在下面空格处填写一条语句, 以使下面的进栈算法完整。 void Push(struct SeqStack*s, ElemType x) { ... f(“栈满溢出错误! \n”); exit(1); } _____ s->data[s->top]=x; } 答案: s->top++;

366. [0] @@@ 在下面空格处填写一条语句, 以使下面的顺序队列出队算法完整。 ElemType OutQueue(struct SeqQueue *sq) ... 操作! \n”); exit(1); } _____ return sq->data[sq->front-1]; } 答案: sq->front++;

367. [0] @@@ 在下面空格处填写一条语句, 以使下面的顺序队列入队算法完整。 void InQueue(struct SeqQueue *sq, int ... int f(“队列已满! \n”); exit(1); } _____ sq->rear++; } 答案: sq->data[sq->rear]=x;

368. [Z] 在下面空格处填写一条语句, 以使下面的循环队列出队算法完整。 ElemType OutQueue(struct SeqQueue *sq) { ... \n”); exit(1); } _____ sq->front=(sq->front+1)%MaxSize; return x; } 答案: x=sq->data[sq->front];

369. [Z] 在线性表的顺序结构中, 以下说法正确的是 ()。答案: 逻辑上相邻的元素在物理位置上也相邻

370. [Z] 在一非空二叉树的中序遍历序列中, 根

结点的右边 ()。答案: 只有右子树上的所有结点

371. [Z]在一棵度具有5层的满二叉树中结点总数为 ()。答案: 31

372. [Z]在一棵二叉树中,若编号为i的结点存在右孩子,则右孩子的顺序编号为 ()。答案: $2i+1$

373. [Z]在一棵二叉树中,若编号为i的结点存在左孩子,则左孩子结点的顺序编号为 ()。答案: $2i$

374. [Z]在一棵二叉树中,若编号为i的结点是其双亲结点的右孩子,则双亲结点的顺序编号为 ()。答案: $i/2$ 向下取整

375. [Z]在一棵具有35个结点的完全二叉树中,该树的深度为 ()。答案: 6

376. [Z]在一棵具有n个结点的二叉树的第i层上,最多具有 () 个结点。答案: 2^{i-1}

377. [Z]在一棵树中, () 没有前驱结点。答案: 树根结点

378. [Z]在一个不带头结点的单循环链表中,p、q分别指向表中第一个结点和尾结点,现要删除第一个结点,且p、q仍然分别指向新表中第一个结点和尾结点。可用的语句是 $p=p->next$;和 ()。答案: $q->next=p$

379. [Z]在一个长度为n的顺序表中为了删除第5个元素,由第6个元素开始从后到前依次移动了15个元素。则原顺序表的长度为 ()。答案: 20

380. [Z]在一个长度为n的顺序存储线性表中,向第i个元素 ($1 \leq i \leq n$) 之前插入一个新元素时,需要依次后移 () 个元素。答案: $n-i+1$

381. [Z]在一个带头结点的单向链表中,若要在指针q所指结点后插入p指针所指结点,则执行 ()。答案: $p->next=q->next$; $q->next=p$;

382. [Z]在一个单链表中,p、q分别指向表中两个相邻的结点,且q所指结点是p所指结点的

直接后继,现要删除q所指结点,可用语句 ()。答案: $p->next=q->next$

383. [Z]在一个单链表中p所指结点之后插入一个s所指的结点时,可执行 ()。答案: $s->next=p->next$; $p->next=s$;

384. [Z]在一个具有n个顶点的无向图中,要连通全部顶点至少需要 () 边。答案: $n-1$

385. [Z]在一个链队中,假设f和r分别为队头和队尾指针,则插入s所指结点的运算为 ()。答案: $r->next=s$; $r=s$;

386. [Z]在一个链队中,假设f和r分别为队头和队尾指针,则删除一个结点的运算为 ()。答案: $f=f->next$;

387. [Z]在一个链队中,设front和rear分别为队首和队尾指针,则插入p所指结点时,应执行 ()。答案: $rear->next=p$; $rear=p$;

388. [Z]在一个图G中,所有顶点的度数之和等于所有边数之和的 () 倍。答案: 2

389. [Z]在一个栈顶指针为top的链栈中,将一个p指针所指的结点入栈,应执行 ()。答案: $p->next=top$; $top=p$;

390. [Z]在一个栈顶指针为top的链栈中删除一个结点时,用x保存被删结点的值,则执行 ()。答案: $x=top->data$; $top=top->next$;

391. [Z]在有向图的邻接表中,每个顶点邻接表链接着该顶点所有 () 邻接点。答案: 出边

392. [Z]在有序表{10, 14, 34, 43, 47, 64, 75, 80, 90}中,用折半查找法查找值80时,经 () 次比较后查找成功。答案: 3

393. [Z]在有序表{1, 3, 8, 13, 33, 42, 46, 63, 76, 78, 86, 97, 100}中,用折半查找值86时,经 () 次比较后查找成功。答案: 4

394. [Z]栈的操作特性决定了它是一种 () 的线性表。答案: 先进后出

395. [Z]栈的插入删除操作在 () 进行。答案: 栈顶

396. [Z]栈的基本运算包括 () 答案: 取栈顶元素

397. [Z]栈和队列的相同点是 ()。答案: 逻辑结构与线性表相同,都是操作规则受到限制的线性表

398. [Z]字符串 $a_1="AEIJING"$, $a_2="AEI"$, $a_3="AEFANG"$, $a_4="AEFI"$ 中最大的是 ()。答案: a_1

399. [Z]字符串 "abcd321ABCD" 的子串是 ()。答案: "21ABC"

400. [Z]组成数据的基本单位是 ()。答案: 数据元素

判断题(177)微信号: zyd_z_9527

1. AOV网是一个带权的有向图。答案: ×

2. [A]按照二叉树的递归定义,对二叉树遍历的常用算法有深度优先遍历和深度优先遍历两种方法。答案: 错

3. [A]按照一定规则,在二叉排序树上插入、删除结点,仍能保持二叉排序树的性质。答案: √

4. [C]采用分块查找时,数据的组织方式是把数据分成若干块,块内数据不必有序,但块间必需有序,每块内最大(或最小)的数据组成索引表。答案: √

5. [C]采用分块查找时,数据的组织方式为把数据分成若干块,每块内数据有序,每块内最大(或最小)的数据组成索引表。答案: ×

6. [C]采用顺序查找法对长度为n(n为偶数)的线性表进行查找,采用从前向后的方向查找。在等概率条件下成功查找到前n/2个元素的平均查找长度为 $(n+2)/4$ 。答案: √

7. [C]采用顺序查找方法查找长度为n的线性表时,每个元素的平均查找长度为 $n/2$ 。答案: ×

8. [C]长度为0的线性表称为空表。答案: √

9. [C]程序一定是算法 ()。答案: 错

10. [C]串的两种最基本的存储方式是顺序和链接。答案: √

11. [C]串中的元素只可能是字母。答案: ×

12. [C]串中字符的个数称为串的长度。答案: √

13. [C]存储图的邻接矩阵中,邻接矩阵的大小不但与图的顶点个数有关,而且与图的边数也有关。答案: ×

14. [D]待排序的序列为8, 3, 4, 1, 2, 5, 9, 采用直接选择排序算法,当进行了两趟选择后,结果序列为1, 2, 8, 3, 4, 5, 9。答案: ×

15. [D]递归的算法简单、易懂、容易编写,而且执行效率也高。答案: ×

16. [D]递归算法可读性差,但是效率高答案: ×

17. [D]递归算法执行时,每次递归可将原问题的规模缩小。答案: √

18. [D]对连通图进行深度优先遍历可以访问到该图中的所有顶点。答案: √

19. [D]队列的特性是先进后出。答案: ×

20. [D]对稀疏矩阵进行压缩存储,矩阵中每个非零元素对应的三元组包括该元素的行号、列号和元素值三项信息。答案: √

21. [D]对稀疏矩阵进行压缩存储,矩阵中每个非零元素对应的三元组包括该元素的行下标、列下标、和非零元素值三项信息。答案: 对

22. [D]对稀疏矩阵进行压缩存储,可采用三元组表,一个6行7列的稀疏矩阵A相应的三元组表共有8个元素,则矩阵A共有34个零元素。答案: 对

23. [D]对于一棵具有n个结点的二叉树,其相应的链式存储结构中n-1个指针域空。答案: 错

24. [D]对于一棵深度为4的满三叉树,其结点数为40。答案: √

25. [D]对于一棵深度为h,度为3的树最多有 $(3h-1)/2$ 个结点。答案: ×

26. [D] 对于一个具有 n 个结点的单链表，在 $*p$ 结点后插入一个新结点的时间复杂度是 $O(n)$ 。
答案：×
27. [D] 对于一个无向图，每个顶点的入度等于出度。
答案：√
28. [E] 二叉排序树中某一结点的左儿子一定小于树中任一结点的右儿子。
答案：×
29. [E] 二叉树的遍历就是按照一定次序访问树中所有结点，并且每个结点的值仅被访问一次的过程。
答案：√
30. [E] 二叉树的根结点值大于其左子树结点的值，小于右子树结点的值，则它是一棵二叉排序树。
答案：×
31. [E] 二叉树中任一结点的值均大于其左孩子的值，小于其右孩子的值，则它是一棵二叉排序树。
答案：×
32. [E] 二分查找是一种最简单的查找方法。
答案：×
33. [F] 分块查找分为两个步骤：第一步是要对索引表进行查找；第二步是在块中查找。这两步查找都可以采用折半查找或者顺序查找方法。
答案：×
34. [F] 父亲李贵有两个儿子李万胜和李万利，李万胜又有三个儿子李建新、李建中和李建国，这个家庭可以用树结构来描述。
答案：√
35. [G] 各种链表只需定义有两个域的结点。
答案：×
36. [G] 根据搜索方法的不同，图的遍历有：先序；中序；后序三种方法。
答案：错
37. [G] 根据图的存储结构进行某种次序的遍历，得到的顶点序列是唯一的。
答案：×
38. [H] 哈夫曼树叶结点数比非叶结点数多1。
答案：√
39. [H] 哈夫曼树一定是完全二叉树或满二叉树。
答案：×
40. [H] 哈夫曼树只存在着双支结点，不存在单支结点。
答案：√
41. [J] 计算机所处理的数据一般具有某种关系，这是指数据元素与数据元素之间存在的某种关系。
答案：√
42. [J] 健壮的算法不会因非法的输入数据而出现莫名其妙的状态。
答案：√
43. [J] 具有10个结点的完全二叉树有5个叶子。
答案：√
44. [J] 具有12个结点的完全二叉树的深度为4。
答案：√
45. [J] 具有 n 个结点的二叉树，采用二叉链表存储，共有 $n-1$ 个空链域。
答案：×
46. [K] 空串的长度是0；空格串的长度是空格字符的个数。
答案：对
47. [L] 链接存储表示中数据元素之间的逻辑关系是由指针表示的。
答案：√
48. [L] 链栈通常不会出现栈满的状态。
答案：√
49. [L] 两个字符串比较时，较长的串比较短的串大。
答案：×
50. [L] 邻接表只能用于存储有向图，而邻接矩阵则可存储有向图和无向图。
答案：×
51. [M] 满二叉树中没有度为1的结点。
答案：√
52. [M] 冒泡排序是一种比较简单的插入排序方法。
答案：×
53. [M] 冒泡排序是一种比较简单的交换排序方法。
答案：√
54. [R] 如果结点A有3个兄弟3个孩子，而且B是A的双亲，则A的度是3。
答案：×
55. [R] 如果一个叶子结点是某二叉树中序遍历序列的最后一个结点，那么它也是该二叉树的先序遍历序列的最后一个结点。
答案：√
56. [R] 若让元素1, 2, 3依次进栈，则出栈次序1, 3, 2是不可能出现的。
答案：×
57. [R] 若让元素a, b, c依次进栈，则出栈次序c, a, b是不可能出现的。
答案：√
58. [R] 若树中各结点的子树是按照一定的次序从左向右安排的，则称之为有序树。
答案：√
59. [R] 若一个强连通图有 n 个顶点，则该强连通图中至少含有 n 条有向边。
答案：√
60. [S] 散列技术中的冲突指的是两个元素具有相同的序号。
答案：×
61. [S] 森林是 m ($m \geq 0$) 棵互不相交的树的集合。
答案：√
62. [S] 删除顺序表的最后一个元素，需要移动的元素最多。
答案：×
63. [S] 设广义表 $L = ((), (()))$ ，则其表头是 $(())$ 。
答案：×
64. [S] 设广义表 $L = ((), (()))$ ，则其表尾是 $()$ 。
答案：×
65. [S] 设某棵二叉树的中序遍历序列为ABCD，前序遍历序列为CABD，则后序遍历该二叉树得到序列为BCDA。
答案：×
66. [S] 设一棵完全二叉树，其最高层上最右边的叶结点的编号为奇数，该叶结点的双亲结点的编号为10，该完全二叉树一共有21个结点。
答案：对
67. [S] 设一棵完全二叉树，其最高层上最右边的叶结点的编号为偶数，该叶结点的双亲结点的编号为9，该完全二叉树一共有19个结点。
答案：错
68. [S] 设有6个结点的无向图，该图至少应有6条边才能确保是一个连通图。
答案：×
69. [S] 设有 n 阶对称矩阵A，用一维数组s压缩存储A的下三角元素，s的下标从零开始，元素 $s[26]$ 相应于A中的元素为 $a_{7,5}$ 。
答案：错
70. [S] 设有一个不带头结点的单向循环链表，结点的指针域为next，指针p指向尾结点，现要使p指向第一个结点，可用语句 $p = p \rightarrow next$ ；。
答案：对
71. [S] 设有一个不带头结点的单向循环链表，结点的指针域为next，指针p指向尾结点，现要使p指向第一个结点，可用语句 $p = p \rightarrow next$ ；。
答案：√
72. [S] 设有一个长度为40的顺序表，要删除第8个元素需移动元素的个数为33。
答案：×
73. [S] 设有一个单向链表，结点的指针域为next，头指针为head，p指向尾结点，为了使该单向链表改为单向循环链表，可用语句 $p \rightarrow next = head$ 。
答案：对
74. [S] 设有一个单向链表，结点的指针域为next，头指针为head，p指向尾结点，为了使该单向链表改为单向循环链表，可用语句 $p \rightarrow next = head$ 。
答案：√
75. [S] 设有一个单向循环链表，结点的指针域为next，头指针为head，指针p指向表中某结点，若逻辑表达式 $p \rightarrow next == head$ ；的结果为真，则p所指结点为尾结点。
答案：对
76. [S] 设有一个单向循环链表，结点的指针域为next，头指针为head，指针p指向表中某结点，若逻辑表达式 $p \rightarrow next == head$ ；的结果为真，则p所指结点为尾结点。
答案：√
77. [S] 设有一个单向循环链表，头指针为head，链表中结点的指针域为next，p指向尾结点的直接前驱结点，若要删除尾结点，得到一个新的单向循环链表，可执行操作 $p \rightarrow next = head$ ；。
答案：对
78. [S] 设有一个非空的链栈，栈顶指针为hs，要进行出栈操作，用x保存出栈结点的值，栈结点的指针域为next，数据域为data，则可执行 $hs = hs \rightarrow next$ ； $x = hs \rightarrow data$ ；
答案：错
79. [S] 设有一个非空的链栈，栈顶指针为hs，要进行出栈操作，用x保存出栈结点的值，栈结点的指针域为next，则可执行 $hs = hs \rightarrow next$ ； $x = hs \rightarrow data$ ；
答案：错
80. [S] 设有一个链栈，栈顶指针为hs，现有一个s所指向的结点要入栈，则可执行操作。
 $hs = s$ ；
 $s \rightarrow next = hs$ ；
答案：错
81. [S] 深度为k的完全二叉树至少有 2^{k-1} 个结点。
答案：×

82. [S]使用邻接矩阵存储图的时候, 占用空间大小与图的结点个数没有关系。答案: ×

83. [S]使用三元组表示稀疏矩阵中的非零元素能节省存储空间。答案: √

84. [S]树的所有结点有且只有一个前驱结点。答案: ×

85. [S]数据的逻辑结构是与存储该结构的计算机相关的。答案: 错

86. [S]数据的逻辑结构是与存储该结构的计算机相关的。答案: ×

87. [S]数据的逻辑结构是指各数据元素之间的逻辑关系, 是用户根据应用需要建立的。答案: √

88. [S]数据的逻辑结构是指数据的各数据项之间的逻辑关系 ()。答案: 错

89. [S]数据的逻辑结构在计算机中的表示称为逻辑结构。答案: 错

90. [S]数据的物理结构是指数据在计算机内的实际存储形式 ()。答案: 对

91. [S]数据的物理结构是指数据在计算机内的实际存储形式。答案: √

92. [S]数据结构中评价算法的两个重要指标是时间复杂度和空间复杂度 ()。答案: 对

93. [S]数据结构中, 元素之间存在多对多的关系称为树状结构。答案: 错

94. [S]数据结构中, 元素之间存在多对多的关系称为树状结构。答案: ×

95. [S]数据项是数据的最小单位。答案: √

96. [S]数据元素可以有一个或多个数据项组成。答案: 对

97. [S]数据元素是数据的最小单位 ()。答案: 错

98. [S]数据元素之间的抽象关系称为物理结构。答案: 错

99. [S]树是一种线性结构。答案: ×

100. [S]树是一种重要的非线性数据结构。答案: √

101. [S]树型结构的元素间存在多对多的关

系。答案: ×

102. [S]数组通常具有的操作是顺序存取。答案: ×

103. [S]树最适合表示元素之间具有层次关系的数据。答案: √

104. [S]顺序查找是一种最简单的查找方法。答案: √

105. [S]顺序存储方式的有点是存储密度大, 且插入、删除运算效率高。 () 答案: 错

106. [S]顺序存储方式只能用于存储线性结构。 () 答案: 错

107. [S]顺序存取的线性表乐意随机存取 ()。答案: 对

108. [S]顺序队列的入队算法是先检查队列是否为满, 若不满则将新元素值赋给队头指针所指向的数据单元, 再将队头指针加1。答案: ×

109. [S]算法的优劣与算法描述语言无关, 但与所用计算机有关 ()。答案: 错

110. [S]算法可以用不同的语言描述, 如果用C语言等高级语言来描述, 则算法实际上就是程序了 ()。答案: 错

111. [S]算法是在数据结构的基础上对特定问题求解步骤的一种描述, 也是若干条指令组成的优先序列 ()。答案: 对

112. [T]通常可以把某城市中各公交站点间的线路图抽象成树型结构。答案: 错

113. [T]通常可以把一本含有不同章节的书的目录结构抽象成线性结构。答案: 错

114. [T]图的广度优先搜索序列是惟一的。答案: ×

115. [T]图的连通分量是无向图的极大连通子图。答案: √

116. [T]图的深度优先搜索序列和广度优先搜索序列不是惟一的。答案: √

117. [T]图的最小生成树只有一棵。答案: ×

118. [W]完全二叉树中没有度为1的结点。答案: ×

119. [X]线性表的链式存储结构优于顺序存储结构 ()。答案: 错

120. [X]线性表的顺序存储是利用数组来实现的。答案: √

121. [X]线性表是一个有限序列, 不可以为空。答案: ×

122. [X]线性表用顺序方式存储可以随机访问。答案: √

123. [X]线性表中的元素可以是各种各样的, 但同一线性表中的数据元具有相同的特性, 因此是属于同一数据对象 ()。答案: 对

124. [X]向一个长度为n的顺序表中的第i个元素 ($1 \leq i \leq n$) 之前插入一个元素时, 需向后移动n-i个元素。答案: ×

125. [X]序列15, 13, 16, 14, 19, 17, 采用冒泡排序算法(升序), 经一趟冒泡后, 结果序列是13, 15, 14, 16, 17, 19。答案: √

126. [X]序列3, 1, 7, 18, 6, 9, 13, 12经一趟归并排序的结果为1, 3, 7, 18, 6, 9, 13, 12。答案: ×

127. [X]循环队列的队头指针为f, 队尾指针为r, 当 $r = f$ 时表明队列已满。答案: 错

128. [X]循环队列的引入, 目的是为了克服假上溢。答案: 对

129. [X]循环队列的最大存储空间为MaxSize=6, 采用少用一个元素空间以有效的判断栈空或栈满, 若队头指针front=4, 当队尾指针rear=3时队满。答案: 对

130. [X]循环队列的最大存储空间为MaxSize=6, 采用少用一个元素空间以有效的判断栈空或栈满, 若队头指针front=4, 队尾指针rear=3时, 队列中共有5个元素。答案: 对

131. [X]循环队列的最大存储空间为MaxSize, 队头指针为f, 队尾指针为r, 当 $(r+1) \% \text{MaxSize} = f$ 时表明队列已满。答案: 对

132. [X]循环队列队头指针在队尾指针后一个位置, 队列是“满”状态。答案: √

133. [X]循环队列是将队列想象成一个首尾相接的圆环。答案: √

134. [Y]要在一个带头结点的单向循环链表中删除头结点, 得到一个新的不带头结点的单向循环链表, 若结点的指针域为next, 头指针为head, 尾指针为p, 则可执行head=head->next; p->next=head; 。答案: 对

135. [Y]要在一个单向链表中p所指向的结点之后插入一个s所指向的新结点, 若链表中结点的指针域为next, 可执行 p->next=s; s->next=p->next; 的操作。答案: 错

136. [Y]要在一个单向链表中删除p所指向的结点, 已知q指向p所指结点的直接前驱结点, 若链表中结点的指针域为next, 则可执行q->next= p->next; 答案: 对

137. [Y]要在一个单向链表中删除p所指向的结点, 已知q指向p所指结点的直接前驱结点, 若链表中结点的指针域为next, 则可执行q->next= p->next。答案: √

138. [Y]一棵二叉树的叶结点(终端结点)数为5, 单分支结点数为2, 该树共有11个结点。答案: 对

139. [Y]一棵二叉树每一层的结点数都达到最大值, 则这个二叉树是完全二叉树。答案: ×

140. [Y]一棵二叉树有6个叶结点, 则该树总共有11个结点。答案: 错

141. [Y]一棵有14个结点的完全二叉树, 则它的最高层上有7个结点。答案: 对

142. [Y]一棵有7个叶结点的二叉树, 其1度结点个数为2, 则该树共有15个结点。答案: 对

143. [Y]一棵有8个权重值构造的哈夫曼数, 共有17个结点。答案: 错

144. [Y]一个广义表的表头总是一个广义表。答案: ×

145. [Y]一个空格的串的长度是0。答案: ×

146. [Y]一个有向图的邻接表和逆邻接表中的节

点个数一定相等。答案：✓

147. [Y] 已知一棵树的先序序列和后序序列，一定能构造出该树。答案：✗

148. [Y] 用邻接矩阵存储图的时候，占用空间大小不但与图的结点个数有关还与图的边数有关。答案：✗

149. [Y] 用数组实现顺序栈，栈底可以是数组空间的任何一端。答案：✓

150. [Y] 用字符数组存储长度为n的字符串，数组长度至少为n+1。答案：✓

151. [Y] 由一个具有n个顶点的连通图生成的最小生成树中，具有n-1条边。答案：✓

152. [Y] 由于顺序存储要求连续的存储区域，所以在存储管理上不够灵活（ ）。答案：对

153. [Y] 有一个链栈，栈顶指针为h，现有一个p所指向的结点要入栈，则可执行操作p->next=h；和h=p；答案：对

154. [Z] 在长度为n的顺序表L中查找指定元素值的元素，其时间复杂度为O(n)。答案：✓

155. [Z] 在单链表中，任何两个元素的存储位置之间都有固定的联系，因为可以从头结点进行查找任何一个元素（ ）。答案：错

156. [Z] 在单链表中，要取得某个元素，只要知道该元素的指针机可，因此单链表是随机存储的存储结构。（ ）答案：错

157. [Z] 在单链表中，要删除某一指定的结点，必须找到该结点的直接前驱结点。答案：✓

158. [Z] 在对10个记录的序列(14, 30, 10, 7, 22, 13, 66, 85, 47, 58)进行直接插入排序时，当把第6个记录13 插入到有序表时，为寻找插入位置，需比较3次。答案：✗

159. [Z] 在队列的顺序存储结构中，当插入一个新的队列元素时，尾指针后移，当删除一个元素队列时，头指针后移。答案：✓

160. [Z] 在各种查找方法中，平均查找长度与结点个数n无关的查找方法是哈希表查找。答案：✓

案：✓

161. [Z] 在归并排序中，在第3趟归并中，是把长度为4的有序表归并为长度为8的有序表。答案：✓

162. [Z] 在双向循环链表上，删除最后一个结点，其算法的时间复杂度为O(1)。答案：✓

163. [Z] 在顺序存储结构中，有时也存储数据结构中元素之间的关系（ ）。答案：错

164. [Z] 在线性表的顺序存储结构中，插入和删除元素时，移动元素的个数与该元素的位置有关（ ）。答案：对

165. [Z] 在线性表的顺序存储结构中，逻辑上相邻的两个元素但是在物理上位置并不一定是相邻的（ ）。答案：错

166. [Z] 在线性表的顺序存储中，元素之间的逻辑关系是通过物理存储位置决定的；在线性表的链式存储中，元素之间的逻辑关系是通过链域的指针值决定的。答案：✓

167. [Z] 在一个不带头结点的非空链队中，f和r分别为队头和队尾指针，队结点的数据域为data，指针域为next，若要进行出队操作，并用变量x存放出队元素的数据值，则相关操作为x=f->data；f=f->next；答案：对

168. [Z] 在一个查找表中，能够唯一地确定一个记录的关键字称为主关键字。答案：✓

169. [Z] 在一个链队中，f和r分别为队头和队尾指针，队结点的指针域为next，s指向一个要入队的结点，则入队操作为r=s；r->next=s；答案：错

170. [Z] 在一个链队中，f和r分别为队头和队尾指针，队结点的指针域为next，则插入所指结点的操作为r->next=s；r=s；答案：对

171. [Z] 在一个顺序存储的循环队列中，队头指针指向队头元素的后一个位置。答案：✗

172. [Z] 在有序表A[1...18]中，采用二分查找算法查找元素值等于A[17]的元素，所比较过的元素的下标依次是9、14、16、17。答案：✓

173. [Z] 在有序顺序存储的线性表中查找一个元素，用折半查找速度一定比顺序查找快。答案：✗

174. [Z] 栈是限定在表的两端进行插入和删除操作的线性表，又称为先进先出表。答案：✗

175. [Z] 折半查找方法运用在升序序列比降序序列效率更高，所以降序序列最好先转换为升序序列。答案：✗

176. [Z] 字符串a1="heijing"，a2="hen"，a3="heifang"，a4="heni"，其中最小的是a2。答案：✗

177. [Z] 字符串属于线性的数据结构。答案：✓

复合题(33)微信号：zydz_9527

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
18.
19.
20.
21.
22.
23.

24.
25.
26.
27.
28.
29.
30.
31.
32.

33. 假设队列顺序存储结构为：struct SeqQueue {...

1.

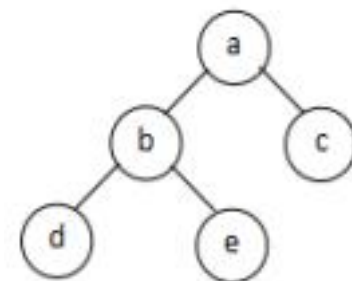
1. [Y] 以下是冒泡排序算法对存放在a[1], a[2], ..., a[n]中序列按关键字key由小到大排序，完成程序中空格部分。

```
...+1];  
a[i+1]=temp;  
}  
if (flag==0) break;  
}
```

}答案：a[i].key>a[i+1].key

2.

1. [S] 设某二叉树先序遍历为abdec，中序遍历为dbeac。该二叉树的图形是（ ）。答案：



3.

1. [Y] 以下程序段的结果是：c的值为（ ）
char a[]=" abcdefghj" ;
int *p=a, c=0;
While (*p++) c++;答案：9
4.

1. [E] 二叉排序树结点类型定义如下:

```
typedef struct Bnode
{
    int key;
    struct Bnode *lef...p->right;
    if (p==NULL)
        break;
}
return (p);
```

} 答案: p->key!=k

5.

1. [S] 设数据序列为:

{53, 30, 37, 12, 45, 24, 96}, 从空二叉树开始逐个插入该数据序列来形成二叉排序树, 若希望高度最小, 应该选择的序列是 ()。答案: 37, 24, 12, 30, 53, 45, 96

6.

1. [Z] 在下面空格处填写一条语句, 以使下面的进栈算法完整。

```
void Push(struct SeqStack*s, ElemType x)
{
    i...溢出错误! \n");
    exit(1);
}
```

s->data[s->top]=x;

} 答案: s->top++;

7.

1. [Y] 以下程序是先序遍历二叉树的递归算法的程序, 完成程序中空格部分 (树结构中左、右指针域分别为left和right, 数据域data为字符...);

```
Preorder(BT-- >left);
Preorder(BT-- >right);
```

} 答案: printf("%c", BT->data)

8.

1. [Z] 在下面空格处填写一条语句, 以使下面的出栈算法完整。

```
ElemType Pop(struct SeqStack*s, ElemType x)
```

```
{... (1);
}
x=s->data[s->top];
```

return x;

} 答案: s->top--;

9.

1. [Z] 在下面空格处填写一条语句, 以使下面的循环队列出队算法完整。

```
ElemType OutQueue(struct SeqQueue *sq)
{...
```

```
sq->front=(sq->front+1)%MaxSize;
return x;
```

} 答案: x=sq->data[sq->front];

10.

1. [Y] 以下函数在a[0]到a[n-1]中, 用折半查找算法查找关键字等于k的记录, 查找成功返回该记录的下标, 失败时返回-1, 完成程序中的空格选项。

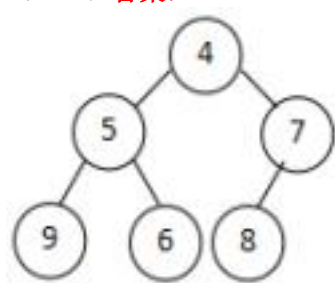
```
...;
else
    high=mid-1;
}
return -1
```

} 答案: low<=high

11.

1. [Y] 一组记录的关键字序列为 (6, 9, 7, 4, 5, 8), 利用堆排序 (堆顶元素是最小元素) 的方法建立初始堆是如下哪个图?

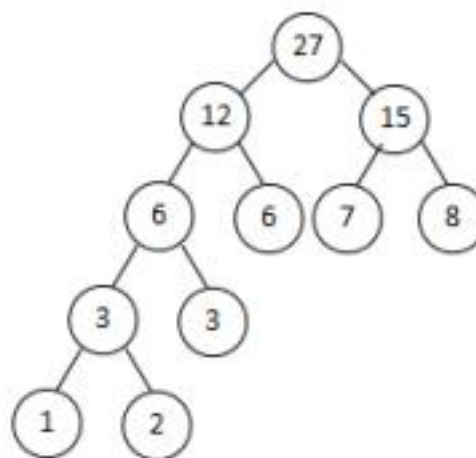
() 答案:



12.

1. [Y] 以1, 2, 3, 6, 7, 8作为叶结点的权, 构造一棵哈夫曼树是如下哪个图?

() 答案:



13.

1. [Z] 在下面空格处填写一条语句, 以使下面的顺序队列入队算法完整。

```
void InQueue(struct SeqQueue *sq, int x)... “队列已满! \n”);
exit(1);
```

sq->rear++;

} 答案: sq->data[sq->rear]=x;

14.

1. [Y] 以下程序是快速排序的算法, 完成程序中空格部分。

设待排序的记录序列存放在a[start], ...a[end]中, 按记录的关键字进行快速排序...i]=mid;

```
quicksort(a, i+1, end);
```

} 答案: quicksort(a, start, i-1);

15.

1. [S] 设查找表为:

序号	1	2	3	4	5	6	7	8
序列	4	12	18	19	37	55	65	77

用折半查找在该查找表成功查找到元素55需要经过 () 次比较。答案: 2

16.

1. [S] 设查找表为(16, 15, 20, 53, 64, 7), 用冒泡法对该表进行排序, 在排序后的有序表的基础上进行折半查找, 在等概率条件下, 成功查找的平均查找长度为 ()。答案: 14/6

17.

1. [S] 设有一个头指针为head的单向链表中 (结点类型为NODE), p为指向该链表中某个结点的指针。以下程序段为插入一个指针为s的结点, 使它成为p结...NODE *q;
q=head;
while(q->next!=p)

_____;

s->next=p;

q->next=s; 答案: q=q->next

18.

1. [S] 设有一个头指针为head的不带头结点单向链表中 (结点类型为NODE), p为指向该链表中某个结点的指针。以下程序段为插入一个指针为s的结点, 使...NODE *q;
q=head;
while(q->next!=p)

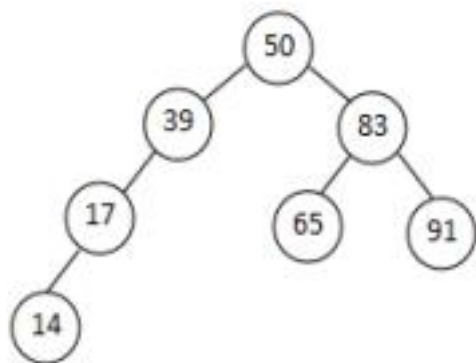
q=q->next;

s->next=p;

_____; 答案: q->next=s

19.

1. [S] 设有数据集{50, 39, 17, 83, 91, 14, 65}, 依次取集合中各数据构造一棵二叉排序树, 是如下的 ()。答案:



20.

1. [Y] 以下是直接插入排序算法对存放在 $a[0], a[1], \dots, a[n-1]$ 中, 长度为 n 的记录序列按关键字 key 由小到大排序, 完成程序中空格部分。

```
... { a[j+1]=a[j];
      _____;
    }
    a[j+1]=temp;
    }
```

答案: $j--$

21.

1. [Z] 在下面空格处填写一条语句, 以使下面的顺序队列出队算法完整。

```
ElemType OutQueue(struct SeqQueue *sq)
{... exit(1);
  }
  _____
  return sq->data[sq->front-1];
```

答案: $sq->front++$;

22.

1. [S] 设有数据集 $\{50, 39, 17, 83, 91, 14, 65\}$, 此二叉排序树的 () 遍历是有序序列。 **答案:** 中序

23.

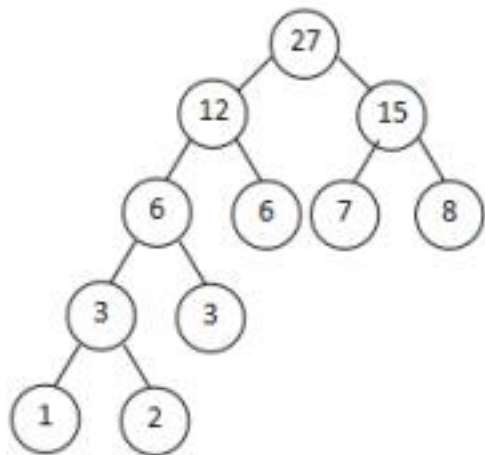
1. [S] 设关键字序列为: $(36, 69, 46, 28, 30, 74)$, 将此序列用快速排序的方法, 以第一个记录为基准得到的一趟划分的结果为

()。 **答案:** $30, 28, 36, 46, 69, 74$

24.

1. [Y] 以 $1, 2, 3, 6, 7, 8$ 作为叶结点的权, 构造一棵哈夫曼树是如下哪个图?

() **答案:**



25.

1. [Z] 在下面空格处填写一条语句, 以使下面的串连接算法完整。

```
char *strcat(char *s1, char *s2)
{ char *p=...2;
  { char *p=...2;
    p++;
    _____
  }
  *p='\0';
  return s1;
```

答案: $s2++$;

26.

1. [S] 设线性表以不带头结点的单向链表存储, 链表头指针为 $head$ 。以下程序的功能是输出链表中各结点中的数据域 $data$, 完成程序中空格部分。

```
#d...rintf(“%d\n”, p->data);
    p=p->next;
  } while(_____);
```

答案: $p!=NULL$

27.

1. [Y] 一组记录的关键字序列为 $(36, 69, 46, 28, 30, 84)$, 对该序列进行直接选择排序 (每次选择最小关键字), 第二趟排序后的结果序列为 ()。 **答案:** $28, 30, 46, 36, 69, 84$

28.

1. [Y] 已知某带权图的邻接矩阵如下所示:

$$\begin{bmatrix} 0 & 6 & 1 & 5 & \infty & \infty \\ 6 & 0 & 5 & \infty & 3 & \infty \\ 1 & 5 & 0 & 5 & 6 & 4 \\ 5 & \infty & 5 & 0 & \infty & 2 \\ \infty & 3 & 6 & \infty & 0 & 6 \\ \infty & \infty & 4 & 2 & 6 & 0 \end{bmatrix}$$

从顶点1出

发的广度优先搜索序列为 ()。 **答案:**

$1, 2, 3, 4, 5, 6$

29.

1. [X] 写出下列程序段执行后的结果

```
SeqQueue Q;
InitQueue(Q);
int i,
a[4]={5,8,1...Q, OutQueue(Q)+10);
while(!QueueEmpty(Q))
printf(“%d ”, OutQueue(Q)); 答案: 12
15 5 30 18
```

30.

1. [Y] 一组记录的关键字序列为 $(46, 79, 56, 38, 40, 45, 62)$, 利用堆排序 (堆顶元素是最小元素) 的方法建立的初始堆为 ()。 **答案:** $38, 40, 45, 79, 46, 56, 62$

31.

1. [Y] 以下为求二叉树深度的算法, 完成程序中空格部分。

```
int BTreeDepth(BTreeNode* BT)
```

```
{
    if (BT==... return dep1+1;
    else
    return dep2+!;
```

答案: $dep1>dep2$

32.

1. [Z] 在下面空格处填写一条语句, 以使下面的串比较算法完整。

```
int strcmp(char *s1, char *s2)
{ int i;
  ...s1[i]!='\0')
    return 1;
  else
    return -1;
```

答案: $return 0$;

33. [J] 假设队列顺序存储结构为:

```
struct SeqQueue {
    ElemType data[MaxSize];
    int front... (1);
    }
    2
    return sq->data[sq->front-1];
}
```

1. [Q] 其中, 1和2处应该补充的代码是 () **答案:** $sq->front==sq->rear, sq->front++$

【单选题】数据结构中，与所使用的计算机无关的是数据的（ ）。
A. 存储结构 B. 物理和存储结构 C. 物理结构 D. 逻辑结构

【答案】 D 【知识点编号】1

【单选题】在数据结构中，从逻辑上可以把数据结构分为（ ）。
A. 动态结构和静态结构 B. 紧凑结构和非紧凑结构
C. 内部结构和外部结构 D. 线性结构和非线性结构
【答案】 D 【知识点编号】1

【单选题】线性结构中数据元素之间的关系是（ ）。
A. 一对一 B. 一对多 C. 多对一 D. 多对多
【答案】 A 【知识点编号】1

【单选题】树形结构中数据元素之间的关系是（ ）。
A. 一对一 B. 一对多 C. 多对一 D. 多对多
【答案】 B 【知识点编号】1

【单选题】数据的存储结构包括数据元素的表示和（ ）。
A. 数据处理的方法 B. 相关算法 C. 数据元素的类型 D. 数据元素间的关系的表示
【答案】 D 【知识点编号】1

【单选题】每个存储结点只存储一个数据元素，各结点存储在连续的存储空间，该存储方式是（ ）存储方式。
A. 顺序 B. 链接 C. 索引 D. 散列
【答案】 A 【知识点编号】2

【单选题】线性表中（ ）称为线性表的长度。
A. 数据最大值 B. 数据最小值 C. 数据元素个数 D. 表的行数
【答案】 C 【知识点编号】2

【单选题】设有一个长度为n的顺序表，要在第i个元素之前（也就是插入元素作为新表的第i个元素），插入一个元素，则移动元素个数为（ ）。
A. n-i B. n-i-1 C. n-i+1 D. i
【答案】 C 【知识点编号】2

【单选题】设有一个长度为n的顺序表，要删除第i个元素，则需移动元素的个数为（ ）。
A. i B. n-i-1 C. n-i D. n-i+1
【答案】 C 【知识点编号】2

【单选题】有关线性表的正确说法是（ ）。
A. 线性表至少要求一个元素
B. 每个元素都有一个直接前驱和一个直接后继
C. 表中的元素必须按由小到大或由大到下排序
D. 除了一个和最后一个元素外，其余元素都有一个且仅有一个直接前驱和一个直接后继
【答案】 D 【知识点编号】2

【单选题】在线性表的顺序结构中，以下说法正确的是（ ）。

A. 逻辑上相邻的元素在物理位置上不一定相邻
B. 数据元素是不能随机访问的
C. 进行数据元素的插入、删除效率较高
D. 逻辑上相邻的元素在物理位置上也相邻
【答案】 D 【知识点编号】2

【单选题】在非空双向循环链表的*p结点之前插入*q结点的操作是（ ）。
A. p->prior=q;q->next=p;p->prior->next=q;q->prior=p->prior;
B. p->prior=q;p->prior->next=q;q->next=p;q->prior=p->prior;
C. q->next=p;q->prior=p->prior;p->prior=q;p->prior->next=q;
D. q->next=p;q->prior=p->prior;p->prior->next=q;p->prior=q;
【答案】 D 【知识点编号】3

【单选题】对链表，以下叙述中正确的是（ ）。
A. 不能随机访问任一结点
B. 插入删除元素的操作一定要移动结点
C. 结点占用的存储空间是连续的
D. 可以通过下标对链表进行直接访问
【答案】 A 【知识点编号】3

【单选题】非空的单向循环链表的尾结点满足（ ）（设头指针为head，指针p指向尾结点）。
A. p->next==head B. p==NULL
C. p== head D. p->next==NULL
【答案】 A 【知识点编号】3

【单选题】设头指针为head的非空的单向链表，指针p指向尾结点，则通过以下操作（ ）可使其成为单向循环链表。
A. head = p;
B. p=head;
C. p->next = NULL;
D. p->next=head;
【答案】 D 【知识点编号】3

【单选题】链表不具有的特点是（ ）。
A. 不必事先估计存储空间
B. 可随机访问任一元素
C. 逻辑上相邻的元素在物理位置上不一定相邻
D. 插入删除不需要移动元素
【答案】 B 【知识点编号】3

【单选题】在一个单链表Head中，若要向表头插入一个由指针p指向的结点，则执行（ ）。
A. Head=p;p->next=Head;
B. p->next=Head;Head=p;
C. p->next=Head;p=Head;
D. p->next=Head->next;Head->next=p;

【答案】 A 【知识点编号】 3

【单选题】在一个单链表中 p 所指结点之后插入一个 s 所指的结点时，可执行（ ）。

- A. p->next=s; s->next=p->next;
- B. p->next=s->next;
- C. p=s->next;
- D. s->next=p->next; p->next=s;

【答案】 D 【知识点编号】 3

【单选题】对于一个线性表，若要求既能进行较快地插入和删除，又要求存储结构能够反映数据元素之间的逻辑关系，则应该（ ）。

- A. 以顺序存储方式
- B. 以链接存储方式
- C. 以索引存储方式
- D. 以散列存储方式

【答案】 B 【知识点编号】 3

【单选题】若 Head 为一个带头结点的单链表的表头指针，则该表为空表的条件是（ ）。

- A. Head==NULL
- B. Head->next==NULL
- C. Head->next==Head
- D. Head!=NULL

【答案】 B 【知识点编号】 3

【单选题】每个存储结点不仅含有一个数据元素，还包含一组指针，该存储方式是（ ）存储方式。

- A. 顺序
- B. 链接
- C. 索引
- D. 散列

【答案】 B 【知识点编号】 3

【单选题】非空的单向循环链表 L 的尾结点（由 p 所指向）满足（ ）。

- A. p==NULL
- B. p->next==NULL
- C. p=L
- D. p->next=L

【答案】 D 【知识点编号】 3

【单选题】在双向循环双链表中，删除*p 结点需要（ ）。

- A. p->next->prior=p->prior; p->prior->next=p->next;
- B. p->prior->next=p->next; p->next->prior=p->prior;
- C. p->prior->next=p; p->prior=p->prior->prior;
- D. p->prior=p->next->next; p->next=p->prior->prior;

【答案】 B 【知识点编号】 3

【单选题】链表所具备的特点是（ ）。

- A. 可以随机访问任一结点
- B. 占用连续的存储空间
- C. 插入删除元素的操作不需要移动元素结点
- D. 可以通过下标对链表进行直接访问

【答案】 C 【知识点编号】 3

【单选题】带头结点的双向循环链表 L 为空表的条件是（ ）。

- A. L=NULL

B. L->next->prior=NULL

C. L->next=L

D. L->prior=NULL

【答案】 C 【知识点编号】 3

【单选题】在一个带头结点的单向链表中，若要在指针 q 所指结点后插入 p 指针所指结点，则执行（ ）。

- A. p->next=q->next; q->next=p;
- B. q->next=p->next; p=q;
- C. p->next=q->next; p->next=q;
- D. q->next=p->next; p->next=q;

【答案】 A 【知识点编号】 3

【单选题】若某表最常用的操作是在最后一个结点之后插入一个结点，则采用_____最节省运算时间。

- A. 单链表
- B. 给出表头指针的单向循环链表
- C. 双链表
- D. 带头结点的双向循环链表

【答案】 D 【知识点编号】 3

【单选题】设有两个长度为 n 的单向链表，结点类型相同，分别是循环链表和非循环链表，则（ ）。

- A. 对于两个链表来说，删除第一个结点的操作，其时间复杂度都是 O(1)
- B. 对于两个链表来说，删除最后一个结点的操作，其时间复杂度都是 O(n)
- C. 循环链表要比非循环链表占用更多的内存空间
- D. 循环链表与非循环链表占用相同的内存空间

【答案】 B 【知识点编号】 3

【单选题】单向线性链表的结点包含 data 域和（ ）域。

- A. next
- B. right
- C. left
- D. head

【答案】 A 【知识点编号】 3

【单选题】带头结点的单向链表为空的判断条件是（ ）（设头指针为 head）。

- A. head = =NULL
- B. head!=NULL
- C. head->next= =head
- D. head->next= =NULL

【答案】 D 【知识点编号】 3

【单选题】当利用大小为 N 的数组顺序存储一个栈时，假定用 top== -1 表示栈空，则入栈应该执行（ ）语句修改 top 指针。

- A. top++
- B. top--
- C. top=0
- D. ! top

【答案】 A 【知识点编号】 4

【单选题】从顺序栈中删除新元素时，应当（ ）。

- A. 先移动栈顶指针，再存入元素
- B. 先读取元素，再移动栈顶指针
- C. 先后次序无关紧要
- D. 同时进行

【答案】 B 【知识点编号】 4

【单选题】（ ）的一个重要应用是在程序设计中实现递归调用。

A. 双向链表 B. 循环链表 C. 栈 D. 队列

【答案】 C 【知识点编号】 4

【单选题】栈是一种操作受限的线性表，其限制是（ ）。

A. 仅允许在表的一端进行插入和删除操作

B. 仅允许进行插入操作

C. 仅允许进行删除操作

D. 仅允许在表的一端进行插入，而在另一端进行删除操作

【答案】 A 【知识点编号】 4

【单选题】表达式 $3*(x+y)/(2-x)$ 的后缀表达式是（ ）。

A. $3\ x\ y\ +\ 2\ * \ 2\ x\ -\ /\$

B. $3\ x\ * \ y\ +\ 2\ x\ -\ /\$

C. $3\ x\ y\ 2\ x\ * \ + \ /\ -$

D. $3\ x\ y\ + \ * \ 2\ x\ -\ /\$

【答案】 D 【知识点编号】 4

【单选题】当利用大小为 N 的数组顺序存储一个栈时，假定用 $top=N$ 表示栈空，则入栈应该执行（ ）语句修改 top 指针。

A. $top++$ B. $top--$ C. $top=0$ D. $! \ top$

【答案】 B 【知识点编号】 4

【单选题】在一个栈顶指针为 top 的链栈中删除一个结点时，用 x 保存被删结点的值，则执行（ ）。

A. $x=top; top=top->next;$

B. $x=top->data;$

C. $top=top->next; x=top->data;$

D. $x=top->data; top=top->next;$

【答案】 D 【知识点编号】 4

【单选题】关于单链表实现的链栈，下面说法正确的是（ ）。

A. 表头为栈顶效率高 B. 表尾为栈顶效率高

C. 表中为栈顶效率高 D. 以上答案均不对

【答案】 A 【知识点编号】 4

【单选题】表达式 $8/5+4$ 的后缀表达式是（ ）。

A. $8\ 5\ 4\ /\ +$ B. $8\ 5\ /\ +\ 4$

C. $8\ 4\ +\ 5\ /\$ D. $8\ 5\ /\ 4\ +$

【答案】 D 【知识点编号】 4

【单选题】栈顶指针通常命名为（ ）

A. $next$ B. top C. $rear$ D. $front$

【答案】 B 【知识点编号】 4

【单选题】元素 4, 6, 8, 10 按顺序依次进栈，按该栈的可能输出序列依次入队列，该队列的可能输出序列是（ ）（进栈出栈可以交替进行）。

A. 10, 8, 4, 6 B. 10, 6, 4, 8

C. 8, 4, 6, 10 D. 10, 8, 6, 4

【答案】 D 【知识点编号】 4

【单选题】一般情况下，将递归算法转换成等价的非递归算法应该设置（ ）。

A. 栈 B. 队列 C. 堆栈或队列 D. 数组

【答案】 A 【知识点编号】 4

【单选题】栈的基本运算包括（ ）

A. 求栈长 B. 修改栈元素 C. 取栈底元素 D. 取栈顶元素

【答案】 D 【知识点编号】 4

【单选题】若让元素 a, b, c 依次进栈，则出栈顺序不可能为（ ）。

A. c, b, a B. b, a, c C. c, a, b D. a, c, b

【答案】 C 【知识点编号】 4

【单选题】通常的使用顺序栈或者链栈实现递归算法，下面哪个说法正确（ ）。

A. 顺序栈效率高 B. 链栈效率高 C. 顺序栈和链栈性能基本相同 D. 视情况而定

【答案】 C 【知识点编号】 4

【单选题】一个栈的进栈序列是 10, 20, 30, 40, 50，则栈的不可能输出序列是（ ）（进栈出栈可以交替进行）。

A. 10, 20, 30, 40, 50 B. 40, 30, 50, 10, 20

C. 40, 50, 30, 20, 10 D. 50, 40, 30, 20, 10

【答案】 B 【知识点编号】 4

【单选题】向顺序栈中压入新元素时，应当（ ）。

A. 先移动栈顶指针，再存入元素 B. 先存入元素，再移动栈顶指针

C. 先后次序无关紧要 D. 同时进行

【答案】 A 【知识点编号】 4

【单选题】链栈和顺序栈相比，有一个比较明显的优点，即（ ）。

A. 插入操作更加方便 B. 通常不会出现栈满的情况

C. 不会出现栈空的情况 D. 删除操作更加方便

【答案】 B 【知识点编号】 4

【单选题】以下数据结构中（ ）是线性结构。

A. 有向图 B. 堆 C. 完全二叉树 D. 栈

【答案】 D 【知识点编号】 4

【单选题】表达式 $a*(b+c)-d$ 的后缀表达式是（ ）。

A. $abcd*+-$ B. $abc+*d-$ C. $abc*++d-$ D. $-+*abcd$

【答案】 B 【知识点编号】 4

【单选题】在一个栈顶指针为 top 的链栈中，将一个 p 指针所指的结点入栈，应执行（ ）。

A. top->next=p;
B. p->next=top->next; top->next=p;
C. p->next=top; top=p;
D. p->next=top->next; top=top->next

【答案】 C 【知识点编号】 4

【单选题】一个队列的入队序列是 1, 2, 3, 4。则队列的输出序列是（ ）。

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

【答案】 B 【知识点编号】 5

【单选题】判断一个顺序队列 sq（最多元素为 m）为空的条件是（ ）。

A. sq->rear-sq->front==m
B. sq->rear-sq->front-1==m
C. sq->front==sq->rear
D. sq->front==sq->rear+1

【答案】 C 【知识点编号】 5

【单选题】以下（ ）不是队列的基本运算。

A. 向队尾插入一个新元素 B. 判断一个队列是否为空
C. 从队列中删除第 i 个元素 D. 读取队首元素的值

【答案】 C 【知识点编号】 5

【单选题】假设链队的队首和队尾指针是 F 和 R，那么队空的条件是（ ）。

A. F==R B. F!=NULL C. R=NULL D. F!=R

【答案】 C 【知识点编号】 5

【单选题】队列是一种操作受限的线性表，其限制是（ ）。

A. 仅允许在表的一端进行插入和删除操作
B. 仅允许进行插入操作
C. 仅允许进行删除操作
D. 仅允许在表的一端进行插入，而在另一端进行删除操作

【答案】 D 【知识点编号】 5

【单选题】顺序队列中，队首元素位置为 5，则队首指针位置为（ ）。

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

【答案】 B 【知识点编号】 5

【单选题】（ ）的一个重要应用是解决主机和打印机之间速度不匹配的问题。

A. 双向链表 B. 循环链表 C. 栈 D. 队列

【答案】 D 【知识点编号】 5

【单选题】下面关于串的叙述中，正确的是（ ）。

A. 串其实是字母序列 B. 空串是由空格构成的串
C. 模式匹配是串的一种重要运算 D. 串只能采用顺序存储

【答案】 C 【知识点编号】 6

【单选题】空串与空格串（ ）。

A. 相同 B. 不相同 C. 可能相同 D. 无法确定

【答案】 B 【知识点编号】 6

【单选题】下列是”abcd321ABCD”的子串的选项是（ ）。

A. ”21ABC” B. ”abcABCD” C. “abcd” D. “321a”

【答案】 A 【知识点编号】 6

【单选题】串是什么？（ ）。

A. 多个字母的序列 B. 任意个字母的序列
C. 有限个字符的序列 D. 无数个字符的序列

【答案】 C 【知识点编号】 6

【单选题】以下陈述中正确的是（ ）。

A. 串是一种特殊的线性表 B. 串的长度必须大于零
C. 串中元素只能是字母 D. 空串就是空白串

【答案】 A 【知识点编号】 6

【单选题】下列说法不正确的是（ ）。

A. 串不是线性结构
B. 串中元素可能是字母、数字或其他字符
C. 空串和空白串不一样
D. 串的长度可能等于零

【答案】 A 【知识点编号】 6

【单选题】两个字符串相等的条件是（ ）。

A. 串的长度相等
B. 含有相同的字符集
C. 都是非空串
D. 两个串的长度相等且对应位置的字符相同

【答案】 D 【知识点编号】 6

【单选题】以下四个串中最小的是（ ）。

A. ”ABADF” B. ”ABAFD” C. ”ABADFA” D. ”ABAF”

【答案】 A 【知识点编号】 7

【单选题】串函数 Strcat（a,b）的功能是进行串（ ）。

A. 比较 B. 复制 C. 赋值 D. 连接

【答案】 D 【知识点编号】 7

【单选题】字符串处理函数 Strcmp（a,b）的功能是进行串（ ）。

A. 连接 B. 比较 C. 复制 D. 模式匹配

【答案】 B 【知识点编号】 7

【单选题】设有两个串 p 和 q，其中 q 是 p 的子串，q 在 p 中首次出现的位置的算法称为（ ）。

A. 求子串 B. 连接 C. 模式匹配 D. 求串长

【答案】 C 【知识点编号】 7

【单选题】两个字符串相等的条件是（ ）。

- A. 两串的长度相等
B. 两串包含的字符相同
C. 两串的长度相等，并且两串包含的字符相同
D. 两串的长度相等，并且对应位置上的字符相同

【答案】 D 【知识点编号】7

【单选题】某串的长度小于一个常数，则采用（ ）存储方式最节省空间。

- A. 链式 B. 顺序 C. 堆结构 D. 无法确定

【答案】 B 【知识点编号】7

【单选题】如果进行串的比较，下列哪个串最大？（ ）

- A. “BEIJING” B. “BEI” C. “BEFANG” D. “BEFI”

【答案】 A 【知识点编号】7

【单选题】广义表(f, h, (a, b, d, c), d, e, ((i, j), k))的长度是()。

- A. 6 B. 10 C. 8 D. 4

【答案】 A 【知识点编号】8

【单选题】一个非空广义表的表头（ ）。

- A. 不可能是原子 B. 只能是子表 C. 只能是原子 D. 可以是子表或原子

【答案】 D 【知识点编号】8

【单选题】下列广义表中的线性表是（ ）。

- A. E(a, (b, c)) B. E(a, E)

- C. E(a, b) D. E(a, L())

【答案】 C 【知识点编号】8

【单选题】广义表(a, (d, a, b), h, (e, ((i, j), k)))深度是()。

- A. 6 B. 10 C. 8 D. 4

【答案】 D 【知识点编号】8

【单选题】广义表(a, a, b, d, e, ((i, j), k))的表头是()。

- A. (a) B. a C. a, (a, b) D. (a, a, b)

【答案】 B 【知识点编号】8

【单选题】设有一个广义表A(a)，其表尾为()。

- A. a B. (()) C. () D. (a)

【答案】 C 【知识点编号】8

【单选题】深度为5的二叉树至多有()个结点。

- A. 16 B. 32 C. 31 D. 10

【答案】 C 【知识点编号】9

【单选题】在一棵二叉树中，若编号为8的结点存在右孩子，则右孩子的顺序编号为()。

- A. 18 B. 16 C. 15 D. 17

【答案】 D 【知识点编号】9

【单选题】在二叉树的第4层最多含有()个结点。

- A. 8 B. 15 C. 16 D. 17

【答案】 A 【知识点编号】9

【单选题】假定一棵二叉树中，叶子结点数为10，单分支结点数为30，则双分支结点数为()。

- A. 7 B. 8 C. 9 D. 19

【答案】 C 【知识点编号】9

【单选题】在一棵二叉树上，第5层的结点数最多为()。

- A. 8 B. 15 C. 16 D. 32

【答案】 C 【知识点编号】9

【单选题】一棵高度为4的二叉树，最多含有()个结点。

- A. 8 B. 15 C. 16 D. 17

【答案】 B 【知识点编号】9

【单选题】在一棵树中，度为0的结点称作()。

- A. 叶子结点 B. 分支结点 C. 孩子结点 D. 双亲结点

【答案】 A 【知识点编号】9

【单选题】树中所有结点的度等于所有结点数加()。

- A. 1 B. 0 C. 2 D. -1

【答案】 D 【知识点编号】9

【单选题】在一棵度具有5层的满二叉树中结点总数为()。

- A. 31 B. 32 C. 33 D. 16

【答案】 A 【知识点编号】9

【单选题】二叉树的按层遍历算法需要使用()。

- A. 队列 B. 栈 C. 广义表 D. 二维数组

【答案】 A 【知识点编号】9

【单选题】对一棵二叉树中顺序编号为i的结点，若它存在左孩子，则左孩子结点的编号为()。

- A. 2i B. 2i+1 C. 2i-1 D. i/2

【答案】 A 【知识点编号】9

【单选题】权值为{1, 2, 6, 8}的四个结点构成的哈夫曼树的带权路径长度是()。

- A. 18 B. 28 C. 19 D. 29

【答案】 D 【知识点编号】10

【单选题】利用2、4、5、10这四个值作为叶子结点的权，生成一棵哈夫曼树，该树的带权路径长度为()。

- A. 18 B. 16 C. 38 D. 30

【答案】 C 【知识点编号】10

【单选题】哈夫曼树只有（ ）的结点的二叉树。

A. 度为 0 B. 度为 2 C. 度为 2 和度为 0 D. 度为 2 或度为 0

【答案】 C 【知识点编号】10

【单选题】设 a, b 为一棵二叉树的两个结点，在后续遍历中，a 在 b 前的条件是（ ）。

A. a 在 b 上方 B. a 在 b 下方 C. a 在 b 左方 D. a 在 b 右方

【答案】 B 【知识点编号】10

【单选题】由六个叶子结点 a、b、c、d、e、f 构造的哈夫曼树（ ）。

A. 唯一 B. 不唯一 C. 不确定 D. 以上都不对

【答案】 B 【知识点编号】10

【单选题】设一棵哈夫曼树有 20 个叶子结点，该树共有（ ）个非叶子结点。

A. 19 B. 20 C. 39 D. 40

【答案】 A 【知识点编号】10

【单选题】无向图的邻接矩阵是一个（ ）。

A. 对称矩阵 B. 零矩阵 C. 上三角矩阵 D. 对角矩阵

【答案】 A 【知识点编号】11

【单选题】在无向图中定义顶点 v_i 与 v_j 之间的路径为从 v_i 到 v_j 的一个（ ）。

A. 顶点序列 B. 边序列 C. 权值总和 D. 边的条数

【答案】 A 【知识点编号】11

【单选题】在一个图 G 中，所有顶点的度数之和等于所有边数之和的（ ）倍。

A. 1/2 B. 1 C. 2 D. 4

【答案】 C 【知识点编号】11

【单选题】一个具有 n 个顶点的有向完全图包含（ ）条边。

A. $n(n-1)$ B. $n(n+1)$ C. $n(n-1)/2$ D. $n(n+1)/2$

【答案】 A 【知识点编号】11

【单选题】在一个无向图中，若两顶点之间的路径长度为 k，则该路径上的顶点数为（ ）。

A. k B. k+1 C. k+2 D. 2k

【答案】 B 【知识点编号】11

【单选题】邻接表是图的一种（ ）。

A. 顺序存储结构 B. 链式存储结构 C. 索引存储结构 D. 散列存储结构

【答案】 B 【知识点编号】11

【单选题】在有向图的邻接表中，每个顶点邻接表链接着该顶点所有（ ）邻接点。

A. 入边 B. 出边 C. 入边和出边 D. 不是入边也不是出边

【答案】 B 【知识点编号】11

【单选题】对有 18 个元素的有序表作二分查找，则查找 A[3] 的比较序列的下标可能为（ ）。

A. 1、2、3 B. 9、5、2、3 C. 9、5、3 D. 9、4、2、3

【答案】 D 【知识点编号】12

【单选题】已知一个有序表为 {11, 22, 33, 44, 55, 66, 77, 88, 99}，则顺序查找元素 55 需要比较（ ）次。

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

【答案】 C 【知识点编号】12

【单选题】有一个长度为 12 的有序表，按折半查找对该表进行查找，在等概率情况下查找成功的平均比较次数为（ ）。

A. 37/12 B. 39/12 C. 41/12 D. 35/12

【答案】 A 【知识点编号】12

【单选题】有一个长度为 10 的有序表，按折半查找对该表进行查找，在等概率情况下查找成功的平均比较次数为（ ）。

A. 29/10 B. 31/10 C. 26/10 D. 29/9

【答案】 A 【知识点编号】12

【单选题】对线性表进行二分查找时，要求线性表必须（ ）。

A. 以顺序存储方式

B. 以链接存储方式

C. 以顺序存储方式，且数据元素有序

D. 以链接存储方式，且数据元素有序

【答案】 C 【知识点编号】12

【单选题】对于顺序存储的有序表 {5, 12, 20, 26, 37, 42, 46, 50, 64}，若采用折半查找，则查找元素 26 的比较次数是（ ）。

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

【答案】 B 【知识点编号】12

【单选题】一组记录的关键字序列为 {80, 57, 41, 39, 46, 47}，利用堆排序（堆顶元素是最小元素）的方法建立的初始堆为（ ）。

A. 39, 47, 46, 80, 41, 57

B. 41, 39, 46, 47, 57, 80

C. 39, 46, 41, 57, 80, 47

D. 39, 80, 46, 47, 41, 57

【答案】 C 【知识点编号】13

【单选题】每次把待排序的区间划分为左、右两个子区间，其中左区间中记录的关键字均小于等于基准记录的关键字，右区间中记录的关键字均大于等于基准记录的关键字，这种排序称为（ ）。

A. 插入排序 B. 快速排序 C. 堆排序 D. 归并排序

【答案】 B 【知识点编号】13

【单选题】设有 2000 个无序的元素，希望用最快的速度挑选出其中前 10 个最大的元素，最好选用（ ）排序法。

A. 快速排序 B. 基数排序 C. 冒泡排序 D. 堆排序
【答案】 D 【知识点编号】 13

【单选题】一组记录的关键字序列为（26，59，36，18，20，25），利用堆排序的方法建立的初始小根堆为（ ）。

A. 18，20，36，59，26，25
B. 18，20，25，59，26，36
C. 26，18，59，20，36，25
D. 26，59，36，18，20，25
【答案】 B 【知识点编号】 13

【单选题】对数据元素序列（49，72，68，13，38，50，97，27）进行排序，前三趟排序结果时的结果依次为第一趟：49，72，68，13，38，50，97，27；第二趟：49，68，72，13，38，50，97，27；第三趟：13，49，68，72，38，50，97，27。该排序采用的方法是（ ）。

A. 插入排序法 B. 选择排序法 C. 冒泡排序法 D. 堆排序法
【答案】 A 【知识点编号】 13

【单选题】就排序算法所用的辅助空间而言，堆排序、快速排序、归并排序的关系是（ ）。

A. 堆排序> 快速排序> 归并排序 B. 堆排序< 快速排序< 归并排序
C. 堆排序< 归并排序< 快速排序 D. 堆排序> 归并排序> 快速排序
【答案】 B 【知识点编号】 13

【单选题】从未排序序列中依次取出元素与已经排好序的序列中的元素作比较。将其放入已排序序列的正确的位置上，此方法称为（ ）。

A. 插入排序 B. 交换排序 C. 选择排序 D. 归并排序
【答案】 A 【知识点编号】 13

【单选题】依次将每两个相邻的有序表合并成一个有序表的排序方法称为（ ）。

A. 插入排序 B. 交换排序 C. 选择排序 D. 归并排序
【答案】 D 【知识点编号】 13

【单选题】一组记录的关键字序列为（60，47，80，57，39，41，46，30），利用归并排序的方法，第一趟归并后的结果为（ ）。

A. 47，57，60，80，30，39，41，46
B. 30，39，41，46，47，57，60，80
C. 30，47，80，57，39，41，46，60
D. 47，60，57，80，39，41，30，46
【答案】 D 【知识点编号】 13

【单选题】将两个各有 n 个元素的有序表归并成一个有序表，其最少的比较次数是（ ）。

A. 2n-1 B. n-1 C. 2n D. n
【答案】 D 【知识点编号】 13

【单选题】已知 10 个数据元素为（54，28，16，34，73，62，95，60，26，43），对该数列从小到大排序，经过一趟冒泡排序后的序列为（ ）。

A. 16，28，34，54，73，62，60，26，43，95
B. 28，16，34，54，62，73，60，26，43，95

C. 16，28，34，54，62，60，73，26，43，95
D. 28，16，34，54，62，60，73，26，43，95
【答案】 B 【知识点编号】 13

【判断题】通常可以把某城市中各公交站点间的线路图抽象成树型结构。
【答案】 × 【知识点编号】 14

【判断题】数据结构的基本操作的设置最重要的准则是实现应用程序与存储结构的独立。
【答案】 √ 【知识点编号】 14

【判断题】数据的逻辑结构与数据元素本身的内容和形式无关。（ ）
【答案】 √ 【知识点编号】 14

【判断题】数据结构中,元素之间存在多对多的关系称为树状结构。
【答案】 × 【知识点编号】 14

【判断题】只有用面向对象的计算机语言才能描述数据结构算法。
【答案】 × 【知识点编号】 14

【判断题】算法可以用不同的语言描述,如果用 C 语言或 PASCAL 语言等高级语言来描述,则算法实际上就是程序了。
【答案】 × 【知识点编号】 14

【判断题】数据元素可以由一个或多个数据项组成。
【答案】 √ 【知识点编号】 14

【判断题】数据元素之间的抽象关系称为物理结构。（ ）
【答案】 × 【知识点编号】 14

【判断题】数据的物理结构是指数据在计算机内的实际存储形式。
【答案】 √ 【知识点编号】 14

【判断题】健壮算法不会因非法的输入数据而出现莫名其妙的状态。
【答案】 √ 【知识点编号】 14

【判断题】数据元素是数据的最小单位。
【答案】 × 【知识点编号】 14

【判断题】通常可以把一本含有不同章节的书的目录结构抽象成线性结构。
【答案】 × 【知识点编号】 14

【判断题】数据元素是对数据操作的基本单位。
【答案】 √ 【知识点编号】 14

【判断题】类 C 语言是对 C 语言的简化和扩展，强化了 C 语言的表达能力。
【答案】 √ 【知识点编号】 14

【判断题】数据的逻辑结构说明数据元素之间的顺序关系,它依赖于计算机的储存结构。
【答案】 × 【知识点编号】14

【判断题】顺序表属于逻辑结构。
【答案】 × 【知识点编号】14

【判断题】算法和程序都应具有下面一些特征:有输入,有输出,确定性,有穷性,有效性。
【答案】 × 【知识点编号】14

【判断题】顺序存储方式的优点是存储密度大,且插入、删除运算效率高。
【答案】 × 【知识点编号】14

【判断题】数据的逻辑结构是指数据的各数据项之间的逻辑关系。
【答案】 × 【知识点编号】14

【判断题】数据对象是性质相同的数据元素构成的集合。
【答案】 √ 【知识点编号】14

【判断题】线性表是一个有限序列,不可以为空。
【答案】 × 【知识点编号】15

【判断题】线性结构采取链式存储时,其元素地址一定是连续的。
【答案】 × 【知识点编号】15

【判断题】求线性表各元素的平均值是线性表的基本运算之一。
【答案】 × 【知识点编号】15

【判断题】向一个长度为 n 的顺序表中的第 i 个元素 ($1 \leq i \leq n$) 之前插入一个元素时,需向后移动 $n-i$ 个元素。
【答案】 × 【知识点编号】15

【判断题】若频繁地对一个线性表进行插入和删除操作,则使用顺序表比较好。
【答案】 × 【知识点编号】15

【判断题】用顺序结构存储的线性表称为顺序表。
【答案】 √ 【知识点编号】15

【判断题】长度为 0 的线性表称为空表。
【答案】 √ 【知识点编号】15

【判断题】对于一个线性表,既要求能够较快地进行插入和删除,又要求存储结构能够反映数据元素之间的逻辑关系,则应采用链式存储方式。
【答案】 √ 【知识点编号】15

【判断题】在线性表的顺序存储中,元素之间的逻辑关系是通过物理存储位置决定的;在线性表的链式存储中,元素之间的逻辑关系是通过链域的指针值决定的。
【答案】 √ 【知识点编号】15

【判断题】一个新结点插入链表中只需要修改一个指针域即可,而不需要移动数据元素。
【答案】 × 【知识点编号】16

【判断题】在单链表中,要删除某一指定的结点,必须找到该结点的直接前驱结点。
【答案】 √ 【知识点编号】16

【判断题】设有一个不带头结点的单向循环链表,结点的指针域为 next,指针 p 指向尾结点,现要使 p 指向第一个结点,可用语句 $p=p \rightarrow next$ ；。
【答案】 √ 【知识点编号】16

【判断题】要在一个带头结点的单向循环链表中删除头结点,得到一个新的不带头结点的单向循环链表,若结点的指针域为 next,头指针为 head,尾指针为 p,则可执行 $head=head \rightarrow next$; $p \rightarrow next=head$; 。
【答案】 √ 【知识点编号】16

【判断题】如果不知道单向链表的头指针,就无法访问该链表的任意结点。
【答案】 √ 【知识点编号】16

【判断题】各种链表只需定义有两个域的结点。
【答案】 × 【知识点编号】16

【判断题】在双向循环链表上,删除最后一个结点,其算法的时间复杂度为 $O(1)$ 。
【答案】 √ 【知识点编号】16

【判断题】设有一个单向循环链表,结点的指针域为 next,头指针为 head,指针 p 指向表中某结点,若逻辑表达式 $p \rightarrow next == head$;的结果为真,则 p 所指结点为尾结点。
【答案】 √ 【知识点编号】16

【判断题】访问单链表中的结点,必须沿着指针链依次进行。
【答案】 √ 【知识点编号】16

【判断题】要在一个单向链表中 p 所指向的结点之后插入一个 s 所指向的新结点,若链表中结点的指针域为 next,可执行 $p \rightarrow next=s$; $s \rightarrow next=p \rightarrow next$; 的操作。
【答案】 × 【知识点编号】16

【判断题】要在一个单向链表中删除 p 所指向的结点,已知 q 指向 p 所指结点的直接前驱结点,若链表中结点的指针域为 next,则可执行 $q \rightarrow next=p \rightarrow next$ 。
【答案】 √ 【知识点编号】16

【判断题】采用链式存储的线性表称作链表。
【答案】 √ 【知识点编号】16

【判断题】设有一个单向链表,结点的指针域为 next,头指针为 head, p 指向尾结点,为了使该单向链表改为单向循环链表,可用语句 $p \rightarrow next=head$ 。
【答案】 √ 【知识点编号】16

【判断题】链式栈与顺序栈相比，一个明显的优点是通常不会出现栈满的情况。
【答案】 √ 【知识点编号】17

【判断题】递归的算法简单、易懂、容易编写，而且执行效率也高。
【答案】 × 【知识点编号】17

【判断题】进栈运算是栈的基本运算之一。
【答案】 √ 【知识点编号】17

【判断题】若让元素 a, b, c 依次进栈，则出栈次序 c, a, b 是不可能出现的情况。
【答案】 √ 【知识点编号】17

【判断题】递归算法可读性差，但是效率高
【答案】 × 【知识点编号】17

【判断题】往栈中插入元素的操作方式是：先写入元素，后移动栈顶指针。
【答案】 × 【知识点编号】17

【判断题】栈是限定在表的一端进行插入和删除操作的线性表，又称为先进后出表。
【答案】 √ 【知识点编号】17

【判断题】栈是限定在表的两端进行插入和删除操作的线性表，又称为先进先出表。
【答案】 × 【知识点编号】17

【判断题】不考虑栈空，顺序栈删除元素操作是，先读出元素，再移动栈顶指针
【答案】 √ 【知识点编号】17

【判断题】顺序栈永远不会出现栈满的状态
【答案】 × 【知识点编号】17

【判断题】链栈永远不会出现栈空的状态
【答案】 × 【知识点编号】17

【判断题】递归定义的数据结构通常用递归算法来实现对它的操作。
【答案】 √ 【知识点编号】17

【判断题】用数组实现顺序栈，栈底可以是数组空间的任何一端
【答案】 √ 【知识点编号】17

【判断题】链栈通常不会出现栈满的状态
【答案】 √ 【知识点编号】17

【判断题】队列的特性是先进后出。
【答案】 × 【知识点编号】18

【判断题】双向循环链表构建的队列，可以只设立队首指针，也可以只设队尾指针
【答案】 √ 【知识点编号】18

【判断题】在队列的顺序存储结构中，当插入一个新的队列元素时，尾指针后移，当删除一个元素队列时，头指针后移。
【答案】 √ 【知识点编号】18

【判断题】将新元素插入到队列任意位置是队列的基本运算之一。
【答案】 × 【知识点编号】18

【判断题】队列允许删除的一端称为队尾，允许插入的一端称为队头。
【答案】 × 【知识点编号】18

【判断题】顺序队列的入队算法是先检查队列是否为满，若不满则将新元素值赋给队头指针所指向的数据单元，再将队头指针加 1。
【答案】 × 【知识点编号】18

【判断题】在一个顺序存储的循环队列中，队头指针指向队头元素的后一个位置。
【答案】 × 【知识点编号】18

【判断题】串中的元素只可能是字母。
【答案】 × 【知识点编号】19

【判断题】字符串属于线性的数据结构
【答案】 √ 【知识点编号】19

【判断题】串函数 StrCmp（“ABCd”，“ABCD”）的值为-1。
【答案】 × 【知识点编号】19

【判断题】长度为 0 字符串称为空白串。
【答案】 × 【知识点编号】19

【判断题】两个字符串比较时，较长的串比较短的串大
【答案】 × 【知识点编号】19

【判断题】用字符数组存储长度为 n 的字符串，数组长度至少为 n+1。
【答案】 √ 【知识点编号】19

【判断题】串即可以采用顺序存储，也可以采用链式存储
【答案】 √ 【知识点编号】19

【判断题】一个空格的串的长度是 0。
【答案】 × 【知识点编号】19

【判断题】空串的长度是 1。
【答案】 × 【知识点编号】19

【判断题】一个广义表的表尾总是一个表。
【答案】 √ 【知识点编号】20

【判断题】一个广义表的表头总是一个广义表。

【答案】 × 【知识点编号】20

【判断题】数组通常具有的操作是顺序存取。

【答案】 × 【知识点编号】20

【判断题】广义表的表头总是一个广义表。

【答案】 × 【知识点编号】20

【判断题】递归算法执行时，每次递归可将原问题的规模缩小。

【答案】 √ 【知识点编号】20

【判断题】对稀疏矩阵进行压缩存储，矩阵中每个非零元素对应的三元组包括该元素的行号、列号和元素值三项信息。

【答案】 √ 【知识点编号】20

【判断题】广义表 A ((a, b, c), (d, e, f)) 的表尾为 ((d, e, f)) 。

【答案】 √ 【知识点编号】20

【判断题】一个广义表 ((a), ((b), c), (((d)))) 的长度为 3，深度为 4。

【答案】 √ 【知识点编号】20

【判断题】设广义表 L= ((), ())，则其表头是 (()) 。

【答案】 × 【知识点编号】20

【判断题】树是一种重要的非线性数据结构。

【答案】 √ 【知识点编号】21

【判断题】对完全二叉树按从上到下、从左到右的顺序依次编号 1, 2, ..., n，则有当 $2i \leq n$ 时，结点 i 的左孩子编号为 $2i$ ，否则无左孩子。

【答案】 √ 【知识点编号】21

【判断题】在二叉树的链接存储中，每个结点设置三个域：值域、左指针域和右指针域。

【答案】 √ 【知识点编号】21

【判断题】对于一棵深度为 h，度为 3 的树最多有 $(3h-1)/2$ 个结点。

【答案】 × 【知识点编号】21

【判断题】一棵完全二叉树深度为 5，最少有 16 个结点。

【答案】 √ 【知识点编号】21

【判断题】完全二叉树中没有度为 1 的结点。

【答案】 × 【知识点编号】21

【判断题】若树的度为 2 时，该数为二叉树。

【答案】 × 【知识点编号】21

【判断题】二叉树的存储结构有两种，分别为顺序存储和链式存储。

【答案】 √ 【知识点编号】21

【判断题】森林是 m ($m \geq 0$) 棵互不相交的树的集合。

【答案】 √ 【知识点编号】21

【判断题】树的所有结点有且只有一个前驱结点。

【答案】 × 【知识点编号】21

【判断题】树中全部结点的度均大于 0。

【答案】 × 【知识点编号】21

【判断题】深度为 5 的二叉树最多有 3 层。

【答案】 × 【知识点编号】21

【判断题】父亲李贵有两个儿子李万胜和李万利，李万胜又有三个儿子李建新、李建中和李建国，这个家庭可以用树结构来描述。

【答案】 √ 【知识点编号】21

【判断题】如果结点 A 有 3 个兄弟 3 个孩子，而且 B 是 A 的双亲，则 A 的度是 3。

【答案】 × 【知识点编号】21

【判断题】具有 12 个结点的完全二叉树的深度为 4。

【答案】 √ 【知识点编号】21

【判断题】一棵二叉树每一层的结点数都达到最大值，则这个二叉树是完全二叉树。

【答案】 × 【知识点编号】21

【判断题】完全二叉树和满二叉树比较适合采用顺序存储。

【答案】 √ 【知识点编号】21

【判断题】具有 32 个叶子结点的满二叉树共有 63 个结点。

【答案】 √ 【知识点编号】21

【判断题】二叉树只能采用二叉链表来存储。

【答案】 × 【知识点编号】21

【判断题】如果结点 A 有 3 个兄弟，而且 B 是 A 的双亲，则 B 的度是 4。

【答案】 √ 【知识点编号】21

【判断题】对于一棵深度为 4 的满三叉树，其结点数为 40。

【答案】 √ 【知识点编号】21

【判断题】哈夫曼树只存在着双支结点，不存在单支结点。

【答案】 √ 【知识点编号】22

【判断题】如果一个叶子结点是某二叉树中序遍历序列的最后一个结点，那么它也是该二叉树的先序遍历序列的最后一个结点。

【答案】 √ 【知识点编号】22

【判断题】哈夫曼树一定是完全二叉树或满二叉树。

【答案】 × 【知识点编号】22

【判断题】二叉树的遍历就是按照一定次序访问树中所有结点，并且每个结点的值仅被访问一次的过程。

【答案】 √ 【知识点编号】22

【判断题】已知一棵树的先序序列和后序序列，一定能构造出该树。

【答案】 × 【知识点编号】22

【判断题】图的连通分量是无向图的极大连通子图。

【答案】 √ 【知识点编号】23

【判断题】在一个具有 n 个顶点和 e 条边的无向图的邻接表中，边结点的个数为 e。

【答案】 × 【知识点编号】23

【判断题】由一个具有 n 个顶点的连通图生成的最小生成树中，具有 n-1 条边。

【答案】 √ 【知识点编号】23

【判断题】强连通分量是有向图的极大连通子图。

【答案】 √ 【知识点编号】23

【判断题】有向图用邻接矩阵表示后，顶点 i 的出度等于第 i 行中非 0 且非无穷的元素个数。

【答案】 √ 【知识点编号】23

【判断题】对于一个无向图，每个顶点的入度等于出度。

【答案】 √ 【知识点编号】23

【判断题】对任意一个图从它的某个顶点出发进行一次深度优先或广度优先搜索遍历可访问到该图的每个顶点。

【答案】 × 【知识点编号】23

【判断题】一个有向图的邻接表和逆邻接表中的节点个数一定相等。

【答案】 √ 【知识点编号】23

【判断题】有 n 个结点的无向图中，若边数大于 n-1，则该图是连通的。

【答案】 √ 【知识点编号】23

【判断题】无向图的邻接矩阵一定是对称的。

【答案】 √ 【知识点编号】23

【判断题】对连通图进行深度优先遍历可以访问到该图中的所有顶点。

【答案】 √ 【知识点编号】23

【判断题】在一个连通图中存在着 1 个连通分量。

【答案】 √ 【知识点编号】23

【判断题】邻接表只能用于存储有向图，而邻接矩阵则可存储有向图和无向图。

【答案】 × 【知识点编号】23

【判断题】有向图的邻接矩阵一定是非对称的。

【答案】 × 【知识点编号】23

【判断题】图的广度优先搜索序列是惟一的。

【答案】 × 【知识点编号】23

【判断题】采用邻接表存储的图的广度优先遍历方法类似于二叉树的按层次遍历方法。

【答案】 √ 【知识点编号】23

【判断题】图的最小生成树只有一棵。

【答案】 × 【知识点编号】23

【判断题】根据图的存储结构进行某种次序的遍历，得到的顶点序列是唯一的。

【答案】 × 【知识点编号】23

【判断题】使用邻接矩阵存储图的时候，占用空间大小与图的结点数没有关系。

【答案】 × 【知识点编号】23

【判断题】用邻接矩阵存储图的时候，占用空间大小不但与图的结点数有关还与图的边数有关。

【答案】 × 【知识点编号】23

【判断题】假设在有序线性表 A[1..20]上进行折半查找，则比较五次查找成功的结点数为 5。

【答案】 √ 【知识点编号】24

【判断题】在一个查找表中，能够唯一地确定一个记录的关键字称为主关键字。

【答案】 √ 【知识点编号】24

【判断题】折半查找只适用于顺序存储结构的有序表。

【答案】 √ 【知识点编号】24

【判断题】采用顺序查找法对长度为 n（n 为偶数）的线性表进行查找，采用从前向后的方向查找。在等概率条件下成功查找到前 n/2 个元素的平均查找长度为 (n+2)/4。

【答案】 √ 【知识点编号】24

【判断题】在顺序查找、折半查找、哈希表查找 3 种方法中，平均查找长度与结点数 n 无关的查找方法是折半查找。

【答案】 × 【知识点编号】24

【判断题】二叉排序树的建立过程实际上是从空树逐次插入的过程。

【答案】 √ 【知识点编号】24

【判断题】二叉排序树中某一结点的左儿子一定小于树中任一结点的右儿子。

【答案】 × 【知识点编号】24

【判断题】根据无序序列构造二叉排序树的过程，也是对无序序列排序的过程。

【答案】 √ 【知识点编号】24

【判断题】使用折半查找算法的前提条件是，查找表中记录相应的关键字值必须按升序或降序排列。

【答案】 √ 【知识点编号】24

【判断题】二叉树为二叉排序树的充分必要条件是，任一个分支结点的值都大于其左孩子的值，小于右孩子的值。

【答案】 × 【知识点编号】24

【判断题】在各种查找方法中，平均查找长度与结点个数 n 无关的查找方法是哈希表查找。

【答案】 √ 【知识点编号】24

【判断题】理想情况下，哈希表查找等概率查找成功的时间复杂度是 $O(1)$ 。

【答案】 √ 【知识点编号】24

【判断题】采用分块查找时，数据的组织方式是把数据分成若干块，块内数据不必有序，但块间必需有序，每块内最大（或最小）的数据组成索引表。

【答案】 √ 【知识点编号】24

【判断题】折半查找的前提条件是，查找表中记录相应的关键字值必须有序或者部分有序。

【答案】 × 【知识点编号】24

【判断题】折半查找方法运用在升序序列比降序序列效率更高，所以降序序列最好先转换为升序序列。

【答案】 × 【知识点编号】24

【判断题】一个好的哈希函数，应该使哈希地址均匀地分布在整个哈希表的地址区间中，完全避免冲突的发生。

【答案】 × 【知识点编号】24

【判断题】二叉排序树中任一棵子树都是二叉排序树。

【答案】 √ 【知识点编号】24

【判断题】分块查找是一种介于顺序查找和折半查找之间的查找方法。

【答案】 √ 【知识点编号】24

【判断题】按 {18, 42, 10, 86, 52, 20} 的顺序构成的二叉排序树，其根结点为 18。

【答案】 √ 【知识点编号】24

【判断题】线性表用关键字的顺序方式存储，可以用二分法排序。

【答案】 × 【知识点编号】24

【判断题】对二叉排序树进行后序遍历，可以使遍历所得到的序列是有序序列。

【答案】 × 【知识点编号】24

【判断题】二叉排序树在呈单支二叉树时，查找效率最低。

【答案】 √ 【知识点编号】24

【判断题】采用分块查找时，若线性表中共有 324 个元素，查找每个元素的概率相同，假设采用顺序查找来确定结点所在的块，每块应分 12 个结点最佳。

【答案】 × 【知识点编号】24

【判断题】按照一定规则，在二叉排序树上插入、删除结点，仍能保持二叉排序树的性质。

【答案】 √ 【知识点编号】24

【判断题】分块查找分为两个步骤：第一步是要对索引表进行查找；第二步是在块中查找。这两步查找都可以采用折半查找或者顺序查找方法。

【答案】 × 【知识点编号】24

【判断题】序列 3, 1, 7, 18, 6, 9, 13, 12 经一趟归并排序的结果为 1, 3, 7, 18, 6, 9, 13, 12。

【答案】 × 【知识点编号】25

【判断题】在归并排序中，在第 3 趟归并中，是把长度为 4 的有序表归并为长度为 8 的有序表。

【答案】 √ 【知识点编号】25

【判断题】对记录序列排序是指按记录的某个关键字排序，记录序列按主关键字排序结果是唯一的。

【答案】 √ 【知识点编号】25

【判断题】对 16 个元素的序列用冒泡排法进行排序，最多需要进行 15 趟冒泡。

【答案】 √ 【知识点编号】25

【判断题】待排序的序列为 8, 3, 4, 1, 2, 5, 9，采用直接选择排序算法，当进行了两趟选择后，结果序列为 1, 2, 8, 3, 4, 5, 9。

【答案】 × 【知识点编号】25

【判断题】在对一组记录（50，40，95，20，15，70，60，45，80）进行直接插入排序时，当把第 7 个记录 60 插入到有序表时，为寻找插入位置需要比较 2 次。

【答案】 × 【知识点编号】25

【判断题】冒泡排序是一种比较简单的交换排序方法。

【答案】 √ 【知识点编号】25

【判断题】8 个元素进行冒泡法排序，至多需要进行 7 趟冒泡，其中第 5 趟冒泡共需要进行 3 次元素间的比较。

【答案】 √ 【知识点编号】25

【判断题】在堆排序和快速排序中，若原始记录接近正序和反序，则最好选用快速排序。

【答案】 × 【知识点编号】25

【判断题】简单插入排序是不稳定的排序算法。

【答案】 × 【知识点编号】25

【判断题】序列 15, 13, 16, 14, 19, 17，采用冒泡排序算法(升序)，经一趟冒泡后，结果序列是 13, 15, 14, 16, 17, 19。

【答案】 √ 【知识点编号】25

【综合题】设 SeqStack 为顺序栈，写出下列程序段执行后的结果。

```
SeqStack S;
InitStack(S);
Push(S, 3);
Push(S, 4);
Push(S, 5);
int x=Pop(S)+2*Pop(S);
Push(S, x);
int i, a[4]={5, 8, 12, 15};
for (i=0; i<4; i++) Push(S, a[i]);
while(!StackEmpty(S)) Printf(“%d ”, Pop(S));
```

A. 15 12 8 5 13 3

B. 3 5 8 12 13 15

C. 15 13 12 8 5 3

D. 15 12 13 3 8 5

【答案】 A 【知识点编号】26

【综合题】设有一个不带头结点的单向链表，头指针为 head，p、prep 是指向结点类型的指针，该链表在输入信息时不慎把相邻两个结点的信息重复输入，以下程序段是在该单向链表中查找这相邻两个结点，把该结点的数据域 data 打印出来，并把其中之一从链表中删除，填写程序中的空格。

```
prep=head;
p=prep->next;
while(p->data!=prep->data)
{ prep=p;
  ____①____;
}
printf(“%d”, p->data);
prep->next=____②____;
```

A. ①p=p->next ②p->next

B. ①prep->data ②p->next

C. ①p->next ②p=p->next

D. ①p->next ②p->data

【答案】 A 【知识点编号】26

【综合题】设有一个头指针为 head 的单向链表中（结点类型为 NODE），p 为指向该链表中某个结点的指针。以下程序段为插入一个指针为 s 的结点，使它成为 p 结点的直接前驱，请选择

其中空格的选项。

```
NODE *q;
q=head;
while(q->next!=p)
  ____;
s->next=p;
q->next=s;
A. p=p->next B. q=q->next
C. s=s->next D. head=head->next
【答案】 B 【知识点编号】26
```

【综合题】设有一个头指针为 head 的不带头结点单向链表中（结点类型为 NODE），p 为指向该链表中某个结点的指针。以下程序段为插入一个指针为 s 的结点，使它成为 p 结点的直接前驱，请把合适选项填写到空行处。

```
NODE *q;
q=head;
while(q->next!=p)
  q=q->next;
s->next=p;
____;
A. p->next=q B. p->next=s
C. q->next=s D. q->next=p
【答案】 C 【知识点编号】26
```

【综合题】设线性表以不带头结点的单向链表存储，链表头指针为 head。以下程序的功能是输出链表中各结点中的数据域 data，完成程序中空格部分。

```
#define NULL 0
void main( )
{ NODE *head ,*p ;
  p=head; /*p 为工作指针*/
  do
  { printf(“%d\n”, p->data);
    p=p->next;
  } while(____);
}
```

A. p=NULL B. p!=NULL C. p!=head D. p==head

【答案】 B 【知识点编号】26

【综合题】假设队列顺序存储结构为：

```
struct SeqQueue {
  ElemType data[MaxSize];
  int front, rear;
};
struct SeqQueue *sq;
请补充下面出队算法（不考虑空间循环使用）。
void OutQueue(struct SeqQueue * sq, ElemType x){
  if( 1 ){
```

```

        printf(“队列已空，不能出队\n”);
        exit(1);
    }
    2
    return sq->data[sq->front-1];
}

```

其中，1 和 2 处应该补充的代码是（ ）。
 A. sq->front==sq->rear , sq->front++
 B. sq->rear==sq->front , sq->rear++;
 C. sq->front=sq->rear , sq->front++
 D. sq->rear=sq->front , sq->rear++;
【答案】 A 【知识点编号】 27

【综合题】 以下程序段执行后，c 的值为（ ）。
 char *a[5]={“12378”,“1237”,“1236789”,“1237”,“123708”}
 int i,c=0
 for(i=0;i<5;i++)
 if (strcmp(a[i],“1237”)==0) c++;
 A.2 B.5 C.0 D.1237
【答案】 A 【知识点编号】 27

【综合题】 假设链表结点存储结构为：
 struct node {
 int data ;
 struct node *next;
 };
 struct node *front,*next;
 InitQueue()、InQueue()、OutQueue()、QueueEmpty() 分别是链队的初始化、入队、出队、判空操作。
 下面程序执行后，运行结果是（ ）。
 int i;
 InitQueue();
 for(i=0;i<6;i++) InQueue(i++);
 while(!QueueEmpty())
 printf(“%d”,OutQueue());
 printf(“\n”);
 A.0 1 2 3 4 5 B.1 2 3 4 5 6
 C.0 1 4 9 16 25 D. 以上答案均不对
【答案】 B 【知识点编号】 27

【综合题】 在下面空格处填写一条语句，以使下面的顺序队列入队算法完整。
 void InQueue(struct SeqQueue *sq, int x)
 { if (sq->rear==MaxSize)
 { printf(“队列已满!\n”);
 exit(1);
 }

```

        sq->rear++;
    }
    A. x=sq->data[sq->rear];
    B. x=sq->rear[sq->data];
    C. sq->rear[sq->data]=x;
    D. sq->data[sq->rear]=x;
【答案】 D 【知识点编号】 27

```

【综合题】 假设队列顺序存储结构为：
 struct SeqQueue {
 ElemType data[MaxSize];
 int front,rear;
 };
 struct SeqQueue *sq;
 请补充下面入队算法（不考虑空间循环使用）。
 void InQueue(struct SeqQueue * sq,ElemType x){
 if(1){
 printf(“队列已满\n”);
 exit(1);
 }
 sq->data[sq->rear]=x;
 2
 }

其中，1 和 2 处应该补充的代码是（ ）。
 A. sq->front==Maxsize, sq->rear++;
 B. sq->rear==Maxsize, sq->rear++;
 C. sq->front==Maxsize, sq->front++;
 D. sq->rear==Maxsize, sq->front++;
【答案】 B 【知识点编号】 27

【综合题】 写出下列程序段执行后的结果
 SeqQueue Q;
 InitQueue(Q);
 int i, a[4]={5,8,12,15};
 for(i=0;i<4;i++) InQueue(Q,a[i]);
 InQueue(Q,OutQueue(Q));
 InQueue(Q,30);
 InQueue(Q,OutQueue(Q)+10);
 while(!QueueEmpty(Q)) printf(“%d ”,OutQueue(Q));
 A.5 8 12 15 30
 B.12 15 5 30 18
 C.8 12 15 30 18
 D.12 15 5 18 30
【答案】 B 【知识点编号】 27

【综合题】 在下面空格处填写一条语句，以使下面的循环队列出队算法完整。
 ElemType OutQueue(struct SeqQueue *sq)


```

{ if (sq->rear==sq->front)
{ printf(“队列已空，不能进行出队操作!\n”);
    exit(1);
}
}

```

```

sq->front=(sq->front+1)%MaxSize;
return x;
}
A.x=sq->data[sq->front-1];
B.x=sq->data[sq->front+1];
C.x=sq->data[sq->front];
D.x=sq->data[sq->rear];

```

【答案】 C 【知识点编号】 27

【综合题】在下面空格处填写一条语句，以使下面的串比较算法完整。

```

int strcmp(char *s1, char *s2)
{ int i;
  for(i=0;s1[i]!='\0' && s2[i]!='\0';i++)
    if(s1[i]>s2[i])
      return 1;
  else
    if(s1[i]<s2[i])
      return -1;
  if(s1[i]=='\0' && s2[i]!='\0')
    _____
  else
    if(s1[i]!='\0')
      return 1;
    else
      return -1;
}

```

A.return 0; B.return -1;
C.return 1; D.return i;

【答案】 A 【知识点编号】 27

【综合题】以下程序段的结果是：c 的值为（ ）

```

char a[]="abcdefghj" ;
int *p=a, c=0;
While (*p++) c++;

```

A.8 B.9 C.10 D.11

【答案】 B 【知识点编号】 27

【综合题】设查找表为：

用折半查找在该查找表成功查找到元素 55 需要经过（ ）次比较。

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

【答案】 B 【知识点编号】 28

【综合题】设查找表为(16, 15, 20, 53, 64, 7)，用冒泡法对该表进行排序，在排序后的有序表的基础上进行折半查找，在等概率条件下，成功查找的平均查找长度为（ ）。

A.3.5 B.3 C.19/6 D.14/6

【答案】 D 【知识点编号】 28

【综合题】设查找表为(1, 10, 11, 14, 23, 27, 29, 55, 68)，对上述查找表进行折半查找，为了成功查找到元素 14，需要依次与元素（ ）进行比较。

A.23, 10, 1, 14 B.23, 29, 27, 14

C.23, 10, 11, 14 D.23, 29, 55, 14

【答案】 C 【知识点编号】 28

【综合题】设有数据集合{50, 39, 17, 83, 91, 14, 65}，此二叉排序树的（ ）遍历是有序序列。

A.先序 B.中序 C.后序 D.按层

【答案】 B 【知识点编号】 28

【综合题】设有查找表为(5, 14, 2, 6, 18, 7, 4, 16, 3)，依次取表中数据，构造一棵二叉排序树，对该二叉树进行后序遍历的结果序列为（ ）。

A.3, 4, 2, 7, 6, 16, 18, 14, 5

B.2, 3, 4, 5, 6, 7, 14, 16, 18

C.5, 2, 14, 4, 6, 18, 3, 7, 16

D.5, 2, 4, 3, 14, 6, 7, 18, 16

【答案】 A 【知识点编号】 28

【综合题】以下为求二叉树深度的算法，完成程序中空格部分。

```

int BTreeDepth(BTreeNode* BT)
{ if (BT==NULL)
  return 0;
  else
  { int dep1=BTreeDepth(BT->left); /* 计算左子树的深度 */
    int dep2=BTreeDepth(BT->right); /* 计算右子树的深度 */
    if (_____)
      return dep1+1;
    else
      return dep2+1;
  }
}

```

A.dep1>dep2 B.dep1<dep2

C.BT->left==NULL D.BT->right==NULL

【答案】 A 【知识点编号】 29

【综合题】以下程序是先序遍历二叉树的递归算法的程序，完成程序中空格部分（树结构中左、右指针域分别为 left 和 right，数据域 data 为字符型，BT 指向根结点）。

```

void Preorder (struct BTreeNode *BT)
{ if (BT!=NULL)
{ _____;
  Preorder(BT->left);
}
}

```

```

        Preorder(BT->right);
    }
}
A.printf(“%c”,BT->left)
B.printf(“%c”,BT->right)
C.printf(“%c”,BT->data)
D.printf(“%d”,BT->data)

```

【答案】 C 【知识点编号】 29

【综合题】设数据序列为：{53, 30, 37, 12, 45, 24, 96}，从空二叉树开始逐个插入该数据序列来形成二叉排序树，若希望高度最小，应该选择的序列是（ ）。

- A. 45, 24, 53, 12, 37, 96, 30
- B. 37, 24, 12, 30, 53, 45, 96
- C. 12, 24, 30, 37, 45, 53, 96
- D. 30, 24, 12, 37, 45, 96, 53

【答案】 B 【知识点编号】 30

【综合题】设一组记录的关键字序列为（49, 83, 59, 41, 43, 47），采用堆排序算法建立初始小根堆，在该小根堆上逐次取走堆顶元素后，经调整得到的 4 个元素的堆为（ ）。

- A. 47, 49, 59, 83
- B. 47, 59, 49, 83
- C. 47, 83, 49, 59
- D. 49, 59, 47, 83

【答案】 A 【知识点编号】 30

【综合题】以下是直接插入排序算法对存放在 a[0], a[1], …, a[n-1] 中，长度为 n 的记录序列按关键字 key 由小到大排序，完成程序中空格部分。

```

void disort (NODE a[ ], int n)
{ int i, j;
  NODE temp;
  for (i=1; i<n; i++)
  { temp=a[i];
    j=i-1;
    while ( j>=0&&temp.key<a[j].key)
    { a[j+1]=a[j];
      _____;
    }
    a[j+1]=temp;
  }
}

```

- A. j++
- B. i++
- C. j--
- D. i--

【答案】 C 【知识点编号】 30

【综合题】一组记录的关键字序列为（36, 69, 46, 28, 30, 84），对该序列进行直接选择排序（每次选择最小关键字），第二趟排序后的结果序列为（ ）。

- A. 28, 69, 46, 36, 30, 84
- B. 28, 30, 46, 36, 69, 84
- C. 36, 46, 69, 28, 30, 84
- D. 28, 30, 36, 69, 46, 84

【答案】 B 【知识点编号】 30

【综合题】设关键字序列为：（36, 69, 46, 28, 30, 74），将此序列用快速排序的方法，以第一个记录为基准得到的一趟划分的结果为（ ）。

- A. 30, 28, 46, 36, 69, 74
- B. 28, 30, 36, 46, 69, 74
- C. 28, 30, 46, 36, 69, 74
- D. 30, 28, 36, 46, 69, 74

【答案】 D 【知识点编号】 30

【综合题】以下程序是快速排序的算法，完成程序中空格部分。

设待排序的记录序列存放在 a[start], …, a[end] 中，按记录的关键字进行快速排序，先进行一次划分，再分别进行递归调用。

```

void quicksort(NODE a[], int start, int end)
{ int i, j;
  NODE mid;
  if (start>=end)
    return;
  i=start;
  j=end;
  mid=a[i];
  while (i<j)
  { while(i<j && a[j].key>mid.key)
    j--;
    if(i<j)
    { a[i]=a[j];
      i++;
    }
    while(i<j && a[i].key<=mid.key)
      i++;
    if(i<j)
    { a[j]=a[i];
      j--;
    }
  }
  a[i]=mid;
  _____;
  quicksort(a, i+1, end);
}

```

- A. quicksort(a, start, i-1);
- B. quicksort(a, start, i+1);
- C. quicksort(a, start, j-1);
- D. quicksort(a, start, j+1);

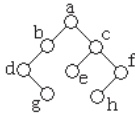
【答案】 A 【知识点编号】 30

【综合题】以下是冒泡排序算法对存放在 a[1], a[2], …, a[n] 中序列按关键字 key 由小到大排序，完成程序中空格部分。

```
void bsort (NODE a[ ], int n)
{ int i,j,flag;
NODE temp;
for (j=1;j<=n-1;j++)
{ flag=0;
for (i=1;i<=n-j;i++)
if ( )
{ flag=1;
temp=a[i];
a[i]=a[i+1];
a[i+1]=temp;
}
if (flag==0) break;
}
}
```

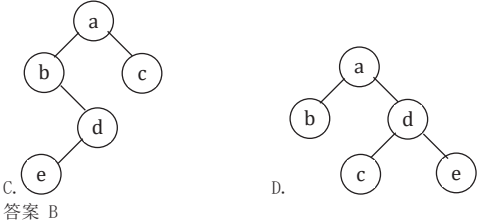
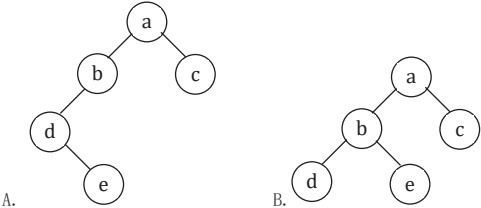
- A. a[i].key>a[i+1].key
 - B. a[i].key<a[i+1].key
 - C. a[j].key>a[j+1].key
 - D. a[j].key<a[j+1].key
- 【答案】 A 【知识点编号】 30

如图所示二叉树的中序遍历序列是（ ）。



- A. abdgcefh
 - B. dgbaechf
 - C. gdbehfca
 - D. abcdefgh
- 答案 B

设某二叉树先序遍历为 abdec，中序遍历为 dbeac。该二叉树的图形是（ ）。



设某二叉树先序遍历为 abdec，中序遍历为 dbeac。写出该二叉树后序遍历的结果（ ）。
A. edbca B. debca C. ebdca D. bceda

答案 B

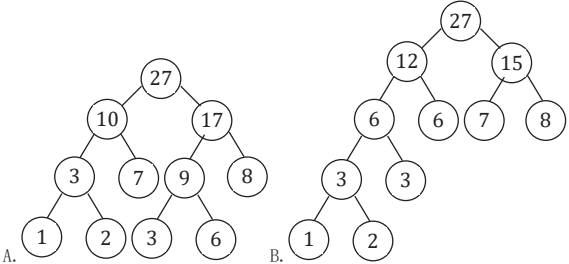
已知某带权图的邻接矩阵如下所示：

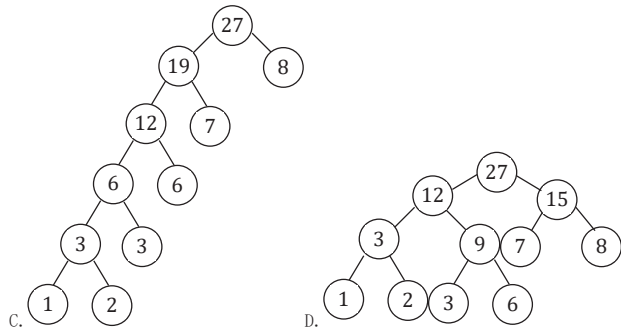
0	6	1	5	∞	∞
6	0	5	∞	3	∞
1	5	0	5	6	4
5	∞	5	0	∞	2
∞	3	6	∞	0	6
∞	∞	4	2	6	0

从顶点 1 出发的广度优先搜索序列为（ ）。
A. 1， 2， 3， 4， 5， 6 B. 1， 4， 3， 2， 6， 5
C. 1， 3， 2， 4， 6， 5 D. 1， 2， 4， 3， 5， 6

答案 A

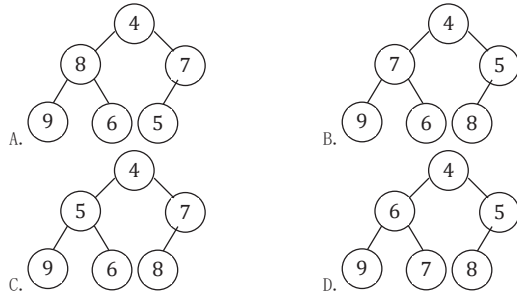
以 1， 2， 3， 6， 7， 8 作为叶结点的权，构造一棵哈夫曼树是如下哪个图？（ ）





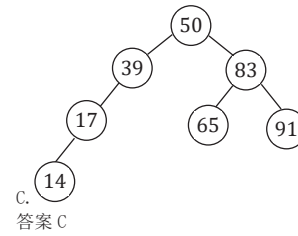
答案 B

一组记录的关键字序列为（6，9，7，4，5，8），利用堆排序（堆顶元素是最小元素）的方法建立初始堆是如下哪个图？（ ）

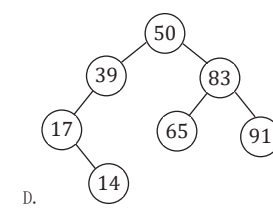


答案 C

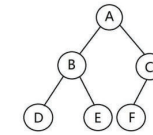
设有数据集合 {50，39，17，83，91，14，65}，依次取集合中各数据构造一棵二叉排序树，是如下的（ ）。



答案 C



D.



中序遍历序列（ ）

A. ABDECF

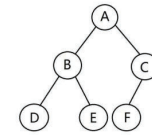
B. DBEAF C

C. ABCDEF

D. DEBFCA

答案 B

由如图所示的二叉树，回答以下问题：



后序遍历是（ ）。

A. ABDECF

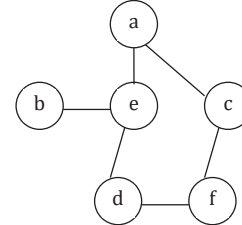
B. DBEAF C

C. ABCDEF

D. DEBFCA

答案 D

下图中，若从顶点 a 出发，按深度优先搜索法进行遍历，则可能得到的一种顶点序列为（ ）。



A. abecdf

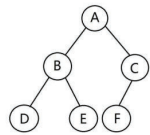
B. acfebd

C. aedfcb

D. aebcfd

答案 C

由如图所示的二叉树，回答以下问题：



其先序遍历序列（ ）括号内增加空格

A. ABDECF

B. DBEAF C

C. ABCDEF

D. DEBFCA

答案 A

下面程序段的时间复杂度是（ ）。

```
for (i=1; i<=n; i++)
    for (j=1; j<=n; j++) {
        c[i][j]=0;
        for (k=1; k<=n; k++)
            c[i][j]=c[i][j]+a[i][k]*b[k][j];
    }
```

A. $O(1)$

B. $O(\log n)$

C. $O(n)$

D. $O(n^3)$

答案 D

二叉树第 k 层上最多有（ ）个结点。

A. 2^k

B. 2^{k-1}

C. 2^k-1

D. $2k-1$

答案 B

对于具有 n 个顶点的图，若采用邻接矩阵表示，则该矩阵的大小为（ ）。

A. n

B. n^2

C. $n-1$

D. $(n-1)^2$

答案 B

采用折半查找方法查找长度为 n 的线性表时，其算法的时间复杂度为（ ）。

A. $O(n^2)$

B. $O(n \log_2 n)$

C. $O(n)$

D. $O(\log_2 n)$

答案 D

深度为 k 的完全二叉树至少有 2^k-1 个结点。

答案 ×

对于一个具有 n 个元素的线性表，建立其单向链表的时间复杂度为（ ）。

A. $O(\log_2 n)$

B. $O(1)$

C. $O(n^2)$

D. $O(n)$

答案 D

对具有 n 个元素的任意序列采用插入排序法进行排序，排序趟数为（ ）。

A. $n-1$

B. n

C. $n+1$

D. $\log_2 n$

答案 A