

一. 单项选择题

1. B

概念题，荣政 P16，王道课后题 1.2.3 第一题。

2. A

考研范围内，顺序表相对于链表的优势就是，更便于随机存取以及存储密度更大。

3. C

第 i 个位置要插入元素，则前 $i-1$ 个元素不会被移动，所以需要移动的元素有 $n - (i-1) = n-i+1$ 。

4. B

另外注意插入过程是从后向前依次移动一个元素。

5. B

荣政 P99，链栈没有头结点。

6. C

一棵二叉树总结点数为 $N_0 + N_1 + N_2$ ，总指针数为 $N_1 + 2N_2$ ，而一棵二叉树中，除了根结点外，每个结点都有一个指针指向它，所以总结点数要比总指针数多 1，也就是多了根结点。

即： $N_0 + N_1 + N_2 = N_1 + 2N_2 + 1$ ，解得 $N_0 = N_2 + 1$ 。

注：多叉树计算方法一样，先求总结点数，再求总指针数，根据总结点数比总指针数多 1，即可求出等量关系。

7. A

后序: CAEBD

中序: DACBE

① 后序遍历序列最后一个结点即为二叉树的根结点。

② 故此题中二叉树根结点为 D。

然后在中序遍历序列中找到根结点，其左边即为左子树上的结点，右边即为右子树上的结点。

显然，此二叉树根结点 D 没有左子树。

ACBE 均为右子树上的结点。

③ 后序序列第二个结点 B，因二叉树递归定义，根据① B 为 D 右子树的根结点。

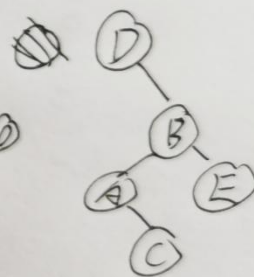
④ 在中序序列中找到 B，可知，AC 为 B 左子树上的结点，E 为 B 右子树上的结点。

⑤ 根据①，A 为 B 左子树的根结点。

⑥ 在中序序列中找到 A，则 C 为 A 右子树上的结点。

二叉树如右图。

则先序遍历序列为 DBACE



8. C

概念题，荣政 P167。

9. A

n 个顶点的无向图，成为一个连通图，至少需要 $n-1$ 条边，此时这些顶点构成一棵树；若在**任何情况下**都保证是连通图，则至少需要 $\frac{(n-1)(n-2)}{2}+1$ 条边，也就是让其中 $n-1$ 个顶点构成完全无向图，然后把剩余的一个结点连接上，加一条边。

n 个顶点的有向图，成为一个连通图，至少需要 n 条边，此时这些顶点构成一个有向环；若在**任何情况下**都保证是连通图，则至少需要 $(n-1)(n-2)+2$ 条边，也就是让其中 $n-1$ 个顶点构成完全有向图，然后把剩余的一个结点连接上，加两条边。

10. A

荣政 P188，同时注意建立邻接矩阵的时间复杂度为 $O(n^2)$ 。

11. D

根据题目描述，可知每个顶点都和除自身外的所有顶点之间都有边，所以是完全图，有向完全图或者无向完全图均可；C 选项，若边的权重为 1，也符合题目描述，但它是有权图。

12. B

不可直接删除，假设利用开放地址法中的线性探查法，因为删除元素后，该位置为空，查找被删除元素之后的元素时，如果利用散列函数定位到被删元素之前，继续比较的过程中会遇到被删元素的空位

置，此时会判定为查找失败。也就是说，直接删除元素，会破坏散列表的查找路径。若是拉链法，则可以直接删除。

13. D

荣政 P231，习题 7-8 原题。Hash 函数应尽可能减少冲突的发生，冲突可以理解为关键字在表中“扎堆”，也就是关键字经过 Hash 函数处理后落在表中各个位置上的概率不相等，为了减少冲突，应该使得函数值以等概率取其值域中的每个值。注意这里所说的值域，如 Hash 函数为 $H(\text{key}) = \text{key} \bmod 13$ ，则值域为 0 到 12，也就是表中的 13 个位置。

14. C

荣政 P242。

快速排序，待排序列越接近无序，算法效率越高；待排序列越接近有序，算法效率越低。快速排序的运行时间与划分是否对称有关，若待排序列基本有序，且每次选取第 n 个元素为基准，则划分区间不均匀，算法效率低。

15. B

荣政 P24。

假设 n 为 7，第一次两两归并，形成三个有序二元组和一个一元组；第二次两两归并，形成一个有序四元组和一个有序三元组；第三次归

并，形成最终有序序列。归并次数为 $\lceil \lg 7 \rceil = 3$ 。注：k 路归并排序的趟数为 $\lceil \log_k n \rceil$ 。

二. 判断题

1. 错误

线性表中第一个元素没有前驱，最后一个元素没有后继。

2. 错误

栈和队列既有顺序存储方式，又有链式存储方式。

3. 正确

4. 正确

后序遍历顺序是“左右中”，这里的“中”就是树根（无论是整棵树的根节点还是子树的根节点），所以后序遍历序列中，根节点肯定在其孩子结点的后面。

5. 正确

设 n 个结点的完全二叉树高度为 h .

① 前 $h-1$ 层总结点数为 $2^{h-1} - 1$

② 前 h 层总结点数最多为 $2^h - 1$

③ 则有不等关系: $2^{h-1} - 1 < n \leq 2^h - 1$
可以写成 $2^{h-1} \leq n < 2^h$

④ 两边取对数, 有 $h-1 \leq \log_2 n < h$

由于 h 为整数, 故 $\log_2 n$ 向下取整,

有 $\lfloor \log_2 n \rfloor = h-1$, 即 $h = \lfloor \log_2 n \rfloor + 1$

另解: 根据 ③, $2^{h-1} - 1 < n \leq 2^h - 1$, 两边与中间同时加 1,
有 $2^{h-1} < n+1 \leq 2^h$
取对数, 有 $h-1 < \log_2(n+1) \leq h$
因 h 为整数, 故对 $\log_2(n+1)$ 向上取整, 有 $\lceil \log_2(n+1) \rceil = h$.

6. 正确

荣政 P177。

二叉排序树的插入过程: 插入一个关键字首先要找到插入位置, 对于一个不存在于二叉排序树中的关键字, 其**查找不成功的位置**即为该关键字的插入位置, 该位置一定在**空指针域**上, 因此在空指针域上连接的一个新结点必为叶子结点。

7. 错误

对于无向非连通图, 一次深度优先遍历只能访问到起点所在的连通分量; 对于有向非强连通图, 起点不一定能访问到所有顶点。

8. 错误

荣政 P194。

最小生成树唯一的图：所有边权值不相等，或者有权值相等的边，但是在构造最小生成树的过程中权值相等的边都被并入生成树的图，其最小生成树唯一。

9. 正确

无向图无论是否有权，其邻接矩阵一定是对称的。所以若邻接矩阵不对称，该图一定是有向图。

10. 错误

a 与 b 之间可能存在一条路径，但并不一定存在一条弧。

11. 错误

荣政 P233。散列函数一对一只是一种理想状态，实际情况中，散列函数设定很灵活，只要使任何关键字的散列函数值都出现在表长允许的范围即可。

12. 错误

荣政 P230。

装填因子越大，说明表越满，再插入新元素时发生冲突的可能性就越大；但装填因子过小，空间浪费就会过多。

13. 正确

概念题，荣政 P218。

14. 正确

第一次划分无论选择哪个元素作为基准，其他 $n-1$ 个元素都需要与基准元素进行比较。

15. 正确

因为归并排序需要转存整个待排序列，故空间复杂度为 $O(n)$ 。

三. 填空题

1. 顺序存储结构；链式存储结构。

2. 相邻。

3. $n \rightarrow \text{prior} = m$

4. (b,c,e,d,a)

5. 长度

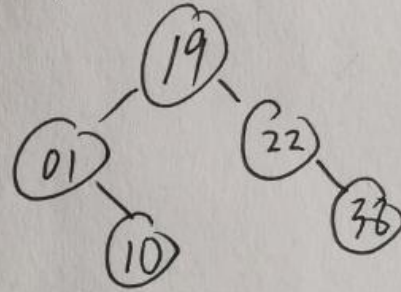
6. $2n-1$

二叉树中，叶子结点数比双分支结点数多 1，而哈夫曼树只有度为 0 和度为 2 的两种结点，故总结点数为 $n+(n-1)=2n-1$ 。

7. $p \rightarrow \text{lchild} == \text{NULL} \ \&\& \ p \rightarrow \text{rchild} == \text{NULL}$

8. 3

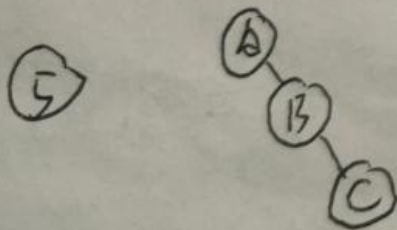
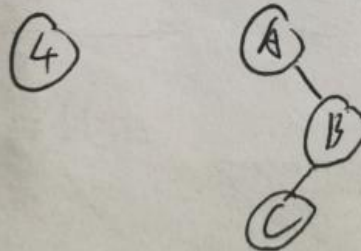
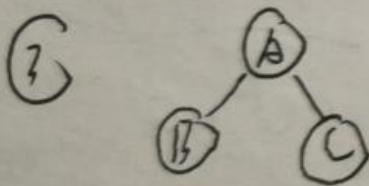
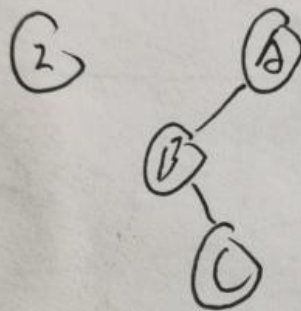
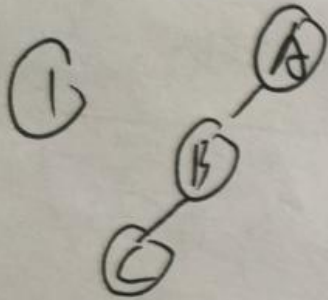
(19, 22, 01, 38, 10)



9. 5

前序: ABC.

前序序列第一个结点必为根结点:



10. 开放地址法; 拉链法

荣政 P225。

11. 2e

在边表中, 每个结点都出现了两次。比如结点 i 和结点 j 之间存在

一条边，则在 i 的邻接表中保存了 j 结点，在 j 的邻接表中保存了 i 结点，就是说，每条边连接的两个结点都出现在了边表中，故边表中有 $2e$ 个结点。

12. $n(n-1)$

13. $\frac{d}{2}$

无向图中每条边连接着两个顶点，这两个顶点因这条边而各有一个度，所以图中各顶点的度数是总边数的 2 倍。

14. $O(n \lg n)$

荣政 P242。根据“文件”以及荣政课本涉及到的几种排序算法描述，推理出此题考查快速排序。

15. $n-1$

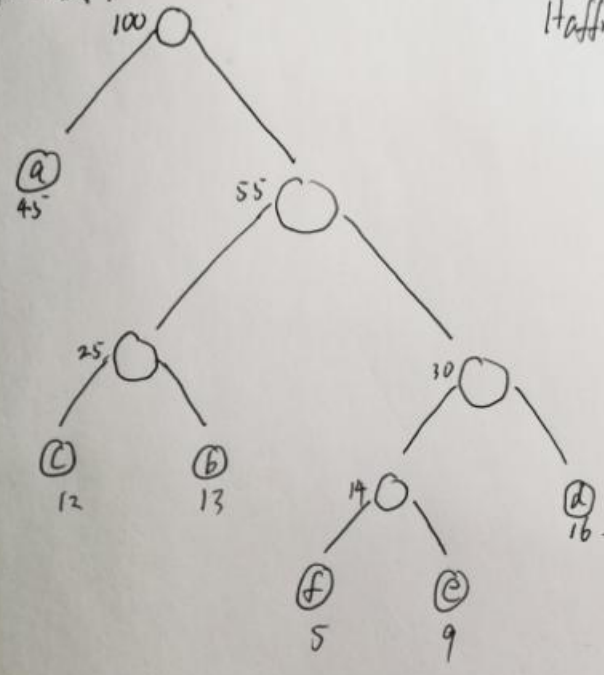
待排序列已经有序的情况下，只需要进行 $n-1$ 次比较即可；对应到冒泡排序代码中，就是内层循环执行 $n-1$ 次后算法结束。

四. 问题求解题

1. 荣政 P168 哈夫曼树构造方法。

(1).

Huffman 树:



Huffman 编码:

a: 0
b: 101
c: 100
d: 111
e: 1101
f: 1100

(2).

$$WPL = 45 \times 1 + 13 \times 3 + 12 \times 3 + 16 \times 3 + 9 \times 4 + 5 \times 4$$
$$= 224$$

2. 荣政 P148, 完全二叉树的顺序存储。

演算：对结点依次编号：

A	B	C	D	E
1	2	3	4	5

在完全二叉树中，结点编号关系。

结点 i ：①若 $i \neq 1$ ，则其双亲结点编号为 $\lfloor i/2 \rfloor$

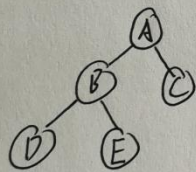
②若 $2i \leq n$ ，则其左孩子编号为 $2i$ ；

若 $2i > n$ ，则 i 没有左孩子。

③若 $2i+1 \leq n$ ，则其右孩子编号为 $2i+1$ ；

若 $2i+1 > n$ ，则 i 没有右孩子。

正式答案：



前序序列：A, B, D, E, C

中序序列：D, B, E, A, C

后序序列：D, E, B, C, A.

索引地址	0	1	2	3	4	5	6	7	8	9	10	11	12
关键字	78		15	03		57	45	20	31		23	36	12
比较次数	1		1	1		1	1	1	4		1	2	1

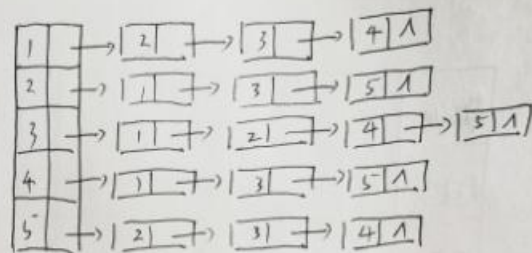
$$ASL = (1+1+1+1+1+1+1+4+1+2+1)/10 = 1.4$$

4. 荣政 P185+P191

(1) 邻接矩阵:

$$\begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

邻接表:



(2).

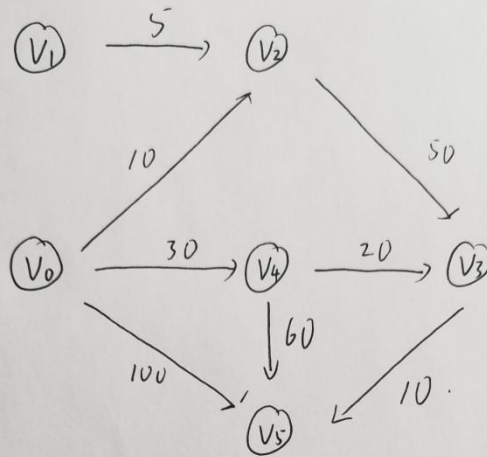
2, 1, 3, 5, 4

(3).

数组, 队列.

5. 荣政 P200

(1).



(2).

步骤	最短路径点集	选择的顶点	源点到各顶点的长度					
			D[0]	D[1]	D[2]	D[3]	D[4]	D[5]
初始	{0}	-	0	∞	10	∞	30	100
第一步	{0, 2}	2	0	∞	10	60	30	100
第二步	{0, 2, 4}	4	0	∞	10	50	30	90
第三步	{0, 2, 4, 3}	3	0	∞	10	50	30	60
第四步	{0, 2, 4, 3, 5}	5	0	∞	10	50	30	60
第五步	{0, 2, 4, 3, 5, 1}	1	0	∞	10	50	30	60

(1).

首先计算每种物品的单位重量的价值 v_i/w_i ；根据贪心选择策略，将尽可能多的单位重量价值高的物品装入背包；若将这种物品全部装入背包后，背包内的物品总重量未超过 C ，则选择单位重量价值次高的物品并尽可能多地装入背包；依此类推，直到背包装满为止。

(2).

大米单位重量价值： $\frac{50}{5} = 10$ 元/公斤

面粉单位重量价值： $\frac{80}{10} = 8$ 元/公斤

大豆单位重量价值： $\frac{45}{15} = 3$ 元/公斤

将 5 公斤大米全部携带，还可携带 20 公斤其他物品；

将 10 公斤面粉全部携带，还可携带 10 公斤其他物品；

最后携带 10 公斤大豆，此时不可继续携带其他物品。

最优方案为：携带 5 公斤大米，10 公斤面粉，10 公斤大豆。

五. 算法题

1. 王道 2.3.7 算法第 7 题

```
void delete(LinkList *head, int max, int min)
{
    LinkList *p, *q; //q为工作指针, p为其前驱
    p = head;
    q = head -> next;
    while(q != NULL)
    {
        if(q -> data < max && q -> data > min)
        {
            p -> next = q -> next;
            free(q);
            q = p -> next;
        }
        else
        {
            p = q;
            q = q -> next;
        }
    }
}
```

2.

```

ElemType FindMax(BTreeNode *BST)
{
    BTreeNode *t;
    if(BST == NULL)
    {
        printf("不能在空树上查找最小值！\n");
        return;
    }
    t = BST;
    while(t -> left != NULL)
        t = t -> left;
    return t -> data;
}

```

3.

```

ga[i].link -> adjvex == j

p = ga[i].link

p != NULL

visited[p -> adjvex] == 0

exist_path_DFS(ga, p -> adjvex, j) == 1

```