2011年390真题答案.

一. 选择题.

1. B

→ 队列"先进先出"

2. A

3. D (同2016-13)

4. B

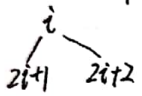
→ 数据基本单位是数据元素，最小单位是数据项.

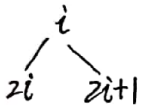
5. A

→ 考查了森林与二叉树转化

6. B

→ 区分从0与从1开始的编号区别.

→ 从0开始 → 

从1开始 → 

7. B

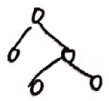
→ 第i行非0元素个数之和表示该行编号元素的出度. 同理，第i列即表示入度.

8. A

→ 考查了希尔排序的做法.

9. C

10. B

→ 这类题都可以举例，如  ，m=3，容纳的顶点个数为2m.

11. D

→

K	7	34	55	25	64	46	20	10
H(K)=K%9	7	7	1	7	1	1	2	1

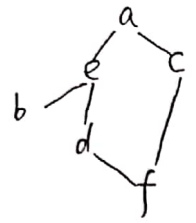
12. B

13. B

→ 考查了连通分量的概念：极大连通子图

→ 极大连通子图通俗的言表是极尽可能的把顶点与边加入子图中直到再加入该子图变为非连通图为止

14. C

→ 

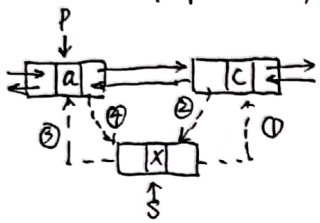
→ ① aedfcb
② aebdfc
③ acfdeb

15. B

→ 这类题是非常经典且难区分的题，有一定难度，下面数大约1分钟所当该题。

→ 考查了双向链表插入，主要判断是否断链

a. 情况1：在p结点之后插入S结点



① s→next = p→next;
② p→next → prior = s;
③ s→prior = p;
④ p→next = s;

→ 技巧：若在p结点之后插入，则关注 p→next，p→next必须在引用之后才能改变，不然会发生断链问题（即找不到原来的p→next）。若①，②中的 p→next 皆为引用，④中的 p→next=s 即直接改变了 p→next，发生断链

b. 情况2：在p结点前插入S结点

→ 将a情况关注点 p→next 换成 p→prior.

→ 所题（将每个选项语句编号为①，②，③，④，方便讲解）

A项：① 在引用②，④之前，错误。

B项：② 在引用①之前之后，满足

C项：根本没有发生断链，说明 p→next 还是原来的，插入失败。

D项：③ 在引用④之前，错误。

→ 注：为什么区分是引用还是本尊呢？只有 p→next 单独出现在等式左边才是本尊，其他皆为引用。本尊要在引用之后改变。

二. 填空题

1. 冒泡排序

→ 有序情况下：①冒泡排序 O(n) ②堆排序 O(nlog₂n) ③快排 (O(n²))。

2. 13 27 38 49 76 97 65 50

3. O(n) O(n²)

4. top==0 或 top==base top=stacksize+1

5. 分裂 1

6. 8/3

7. 6 （总分支数 = 总结点数 −1，3×2+2×1+1×2 = 2+1+2+n₀−1 ⇒ n₀=6）

三. 判断题.

1. f （链表上也可以顺序查找）
2. f （判满条件 (rear+1)%m = front）
3. f （ $\frac{n(n+1)}{2}$ 个单元空间）
4. f
5. f （快排，希尔排序，堆排序，选择排序皆为不稳定）.
6. f
7. t
8. t
9. t
10. f

四. 简答题

1.

简单选择排序：

每一趟排序都能确定一个元素的最终位置（最大元素或最小元素）

且在最坏情况下，时间复杂度为 $O(n^2)$，平均情况下时间复杂度为 $O(n^2)$

冒泡排序：

每一趟排序都能确定一个元素的最终位置，平均情况下时间复杂度为 $O(n^2)$

堆排序：

每一趟能确定一个最终位置，平均时间复杂度为 $O(n\log n)$

归并排序：

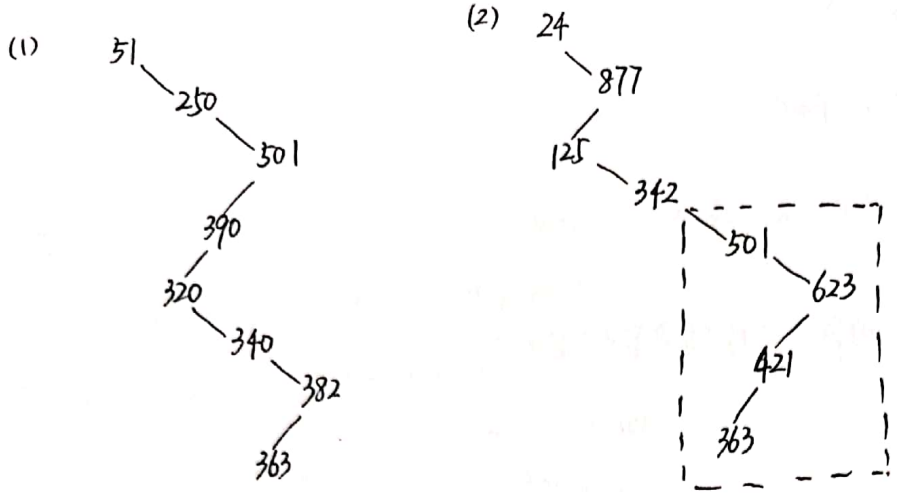
在未排序完成之前，无法确定元素的最终位置

综上所述：选堆排序满足题意.

→ 考查各种排序过程及性质. 诸如此类的还有考查不同排序算法的关键字比较次数 稳定性等.

2.

解：由二叉排序树的关键字序列(1)，(2)可得下图



由图可知，关键字序列(2)不满足二叉排序树的定义（ 左子树全部元素值 < root(根元素值) < 右子树全部元素值.

图中 501 > 421，501 > 363 不满足二叉排序树定义

→ 查查了二叉排序树定义：

二叉排序树或者是空树，或者是满足以下性质的二叉树：

① 若它的左子树不空，则左子树上所有关键字的值均小于根关键字的值.

② 若它的右子树不空，则右子树上所有关键字的值均大于根关键字的值

③ 左右子树又各是一棵二叉排序树.

3.

(1) 最小深度： $\lfloor \log_2 n \rfloor + 1$ 或 $\lceil \log_2 n + 1 \rceil$

最大深度： n

(2) $n - \lfloor n/2 \rfloor$ ， $n - \lfloor n/2 \rfloor - 1$

(3)

当 $n_1 = 1$ 时 $n_0 + n_1 + n_2 = n_0 + 1 + n_0 - 1 = 2n_0$

当 $n_1 = 0$ 时 $n_0 + n_1 + n_2 = n_0 + n_0 - 1 = 2n_0 - 1$

（或从另一个角度，由(2)中知 $n_0 = n - \lfloor n/2 \rfloor \rightarrow n = ?$ ）

→ 查查了二叉排序树与完全二叉树的性质：

(1) 最小深度 → 当二叉树为完全二叉树时，最大深度 → 单支树情况

(2) 完全二叉树中最后一个非叶结点编号为 $\lfloor n/2 \rfloor$（从1开始），即非叶结点是有 $\lfloor n/2 \rfloor$，$n - \lfloor n/2 \rfloor$ 即为叶结点个 → 从0开始的话编号为 $\lfloor n/2 \rfloor - 1$，但非叶个数不变

(3) 题给了两种思考角度，第3种暂时不知怎么解.

→ 这类题相对简单，要对一些性质结论比较熟悉即可！

2017-4

4.

答： sky

→ 过程如下： X进栈，S={'y'} → y进栈，S={'y','s'} → S出栈一个元素给变量X，x='s'，S={'y'}

→进栈'k'，S={'y','k'}→ 进栈变量x='s' → S={'y','k','s'}

→整个栈进一出栈 → sky.

→ 注意变量y与字符'y'区别就行

→ 栈：先进后出特性.

5. 解：

设置两个矩阵A与path，A用于记录当前已经求得的任意两个顶点最短路径的长度，path用来记录当前两顶点间最短路径要经过的中间结点，这里就规定-1为无中间结点的情况

① 初始情况下：

$$A_0=\begin{bmatrix}0&4&11\\6&0&2\\3&\infty&0\end{bmatrix}\qquad path_0=\begin{bmatrix}-1&-1&-1\\-1&-1&-1\\-1&-1&-1\end{bmatrix}$$

② 以1为中间点，检测所有顶点对，若A[i][j] > A[i][1]+A[1][j]，则将A[i][j]改为A[i][1]+A[1][j]的值.
并且path[i][j]置为1，下面步骤同理

$$A_1=\begin{bmatrix}0&4&11\\6&0&2\\3&7&0\end{bmatrix}\qquad path_1=\begin{bmatrix}-1&-1&-1\\-1&-1&-1\\-1&1&-1\end{bmatrix}$$

③ 以2为中间点

$$A_2=\begin{bmatrix}0&4&6\\6&0&2\\3&7&0\end{bmatrix}\qquad path_2=\begin{bmatrix}-1&-1&2\\-1&-1&-1\\-1&1&-1\end{bmatrix}$$

④ 以3为中间点

$$A_3=\begin{bmatrix}0&4&6\\5&0&2\\3&7&0\end{bmatrix}\qquad path_3=\begin{bmatrix}-1&-1&2\\3&-1&-1\\-1&1&-1\end{bmatrix}$$

综上所述：

<1,2> 最短路径为 1→2，长度为4

<1,3> 最短路径为 1→2→3，长度为6

<2,3> 最短路径为 2→3，长度为2

<2,1> 最短路径为 2→3→1，长度为5

<3,1> 最短路径为 3→1，长度为3

<3,2> 最短路径为 3→1→2，长度为7.

→具体解释Floyd见2012-5

2017-5

6.

答：不唯一

当带权无向图存在权值相同的情况下最小生成树可能不唯一.（见2013-4）

五. 算法填空

1.

① 0

② j++

③ S->next = p->next

④ p->next = S

→ 素材来源于王版<<数据结构>> P29-30 的算法 2.9

2.

⑤ indegree[i]==0 // 入度为0的顶点入栈S

⑥ !StackEmpty(S)

⑦ Pop(S,i)

⑧ p

⑨ Push(S,k)

⑩ Count<G.vexnum

→ 素材来源于王版<<数据结构>> P187 的算法 7.13. 只是将"建零入度顶点栈S"实现了一下，其他基本一致

)

六. 编写算法

1. 见 2012-1 (同类型题第一道)，不写具体的解释

2. 见 2011-2 (同类型第二道)