(1) Given the following three tree traversal combinations, tell which combination results in a unique binary tree. If it corresponds to a unique binary tree, show the tree and list the level order traversal. Otherwise, explain why more than one binary tree is produced, or there does not exist such a binary tree.

(a) preorder: J C B A D E F I G H (5%)
inorder: A B C E D F J G I H

(b) preorder: A B D G C E H I F (5%)
postorder: G D B H I E F C A

(c) postorder: F E C H G D B A (5%)
inorder: F C E A B H D G

(2) Given the following data structures with n elements, please specify how to delete the largest and the smallest elements, and list the complexity.

(a) max heap (10%)

(b) 2-3-4 tree (10%)

(3) XML (eXtensible Markup Language) is a universal format for representing documents. The following shows part of a large-scale XML document collection:

```
<article>
    <author> M.F. Porter </author>
    <title> An Algorithm for Suffix Stripping </title>
    <abstract> The automatic removal of suffixes from words … </abstract>
    <pages> 313-316 </pages>
</article>
<article>
    <author> D.K. Harman </author>
    <title> User-Friendly Systems Instead of User-Friendly Front-Ends </title>
    <abstract> Most systems are not designed to service end user … </abstract>
    <pages> 413-423 </pages>
</article>
…
```

Users usually submit queries like "find papers written by a specific author", "find papers about a specific topic", "find papers consisting of some keywords", and so on.

(a) Propose data structures to support the above queries efficiently. (8 %)

(b) Explain how to find papers written by some specific author (e.g., M.F. Porter) and concerning some specific topic (e.g., suffix removal) based on the data structures in (a). (7 %)

接背面

(4) Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers. (20%)

(5) Argue that the solution to the recurrence $T(n) = T(\frac{n}{4}) + T(\frac{3n}{4}) + n$ is $\Omega(n \log n)$ by appealing to a recursion tree. (15%)

(6) The ***square*** of a directed graph $G = (V, E)$ is the graph $G^2 = (V, E^2)$ such that $(u, w) \in E^2$ if and only if for some $v \in V$, both $(u, v) \in E$ and $(v, w) \in E$. That is, G^2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe an efficient algorithm for computing G^2 from G for the adjacency-matrix representation of G. Analyze the running time of your algorithm. (15%)