Answer all the questions. Give a correctness proof and run time analysis for all the algorithms.
Problem 1: [10pts] Given two sorted lists $A$ and $B$, each represented as an array, and a positive integer $k$, describe an $O(\log k)$ time algorithm for computing the $k$-th smallest element in $A \cup B$. 
Problem 2: [15pts] Let \( S = (x_1, \ldots, x_n) \) be a sequence of \( n \) real numbers. Given a pair \( i, j \) such that \( 1 \leq i \leq j \leq n \), the range-max query asks to return \( \max_{i \leq k \leq j} x_k \).

Suppose a rooted tree \( T \) with \( n \) nodes can be preprocessed into a data structure of size \( O(n) \) so that for any two nodes of \( T \), their lowest common ancestor can be computed in \( O(\alpha(n)) \) time, where \( \alpha(n) \) is the inverse Ackermann function.

Using this data structure, show that \( S \) can be preprocessed into a data structure of size \( O(n) \) so that a range-max query can be answered in \( O(\alpha(n)) \) time.
**Problem 3:** [20pts] For any edge $e$ in any graph $G = (V, E)$, let $G \setminus e$ denote the graph obtained by deleting $e$ from $G$. Let $|V| = n$ and $|E| = m$.

Suppose you are given a directed graph $G$, in which the shortest path from vertex $u$ to vertex $v$ passes through all vertices in $G$. Give an $O(m \log n)$-time algorithm to compute the shortest path from $u$ to $v$ in $G \setminus e$, for every edge $e$ of $G$. The algorithm should output a set of $|E|$ shortest-path distances, one for each edge of the input graph. All edge weights are non-negative. (**Hint:** If an edge of the original shortest path is deleted, how do the old and new shortest paths overlap?)

**Extra credit** [5pts]: Assuming that the shortest path from $u$ to $v$ is given, show that the above problem can be solved in $O(m)$ time.
Problem 4: [17+8pts]

(a) Suppose we are given a set $L$ of $n$ line segments in the plane, where each segment has one endpoint on the line $y = 0$ and one endpoint on the line $y = 1$, and all $2n$ endpoints are distinct. Describe an $O(n^2)$-time algorithm to compute the largest subset of $L$ in which no pair of segments intersects. (Hint: Use dynamic programming.)

(b) Now suppose the endpoints of segments in $L$ lie on the unit circle $x^2 + y^2 = 1$, and all $2n$ endpoints are distinct. Modify your previous algorithm to compute in $O(n^2)$ time the largest subset of $L$ in which no pair of segments intersects.
Problem 5: [20+10pts] Let $G = (V, E)$ be a directed graph, with $|V| = n$ and $|E| = m$. The minimum equivalent graph of $G$ is a smallest subgraph $H = (V, E')$ of $G$ such that for any two vertices $u, v \in V$, there is a path from $u$ to $v$ in $H$ if and only if there is a path from $u$ to $v$ in $G$.

(a) If $G$ is acyclic, then show that the minimum equivalent graph of $G$ is unique and can be computed in $O(mn)$ time.

(b) Is there a polynomial-time algorithm to compute a minimum equivalent graph if $G$ has cycles?