Due Date: 11:59 PM, February 14, 2013.

Problem 1: Given a sequence of n distinct numbers a_1, ... a_n, we define a *2-inversion* to be a pair (i,j), such that $i < j$ and $a_i > 2a_j$. Give an $O(n \log n)$ algorithm to count the number of 2-inversions in this sequence.

Problem 2: A graph (V, E) is called *bipartite* if the vertices V can be partitioned into two subsets L and R, such that every edge has one vertex in L and the other in R.

(a) Prove that every tree is a bipartite graph.

(b) Give an $O(|V| + |E|)$-time algorithm to determine whether a given undirected graph is bipartite.

Problem 3: 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 we delete an edge of the original shortest path, how do the old and new shortest paths overlap?*)

Problem 4: Given a DAG (directed acyclic graph) $G = (V, E)$, and two vertices $u, v \in V$, describe an $O(|V| + |E|)$-time algorithm that outputs the number of different directed paths from u to v in G.

Problem 5: Given a directed graph, find the length of the shortest cycle in the graph. (If the graph is acyclic, return 0). All edge lengths of the graph are positive, and the running time of the algorithm should be at most $O(V^3)$.