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

Problem 1: You are at an NCAA March Madness basketball convention with n players, each player belonging to exactly one team/university. Being new to the game, you don't know which team any player belongs to. However, you can determine whether two players belong to the same team or not, by introducing them to each other—players of the same team always greet each other with a hug, while players of different teams greet each other with insults and stares.

You can only introduce the players to one another, and for various reasons, cannot ask which team they belong to. Neither do they have any other objects that can help you identify them.

a) Suppose *more* than half the players are from the same team. Describe an efficient algorithm that identifies a player (*any player*) of the majority team.

b) Suppose exactly k teams are present at the convention, and one team has more players than any other team. Give an algorithm to pick a player (*any player*) from this majority team, as optimally as possible. (Minimum introductions).

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 $|E| = m$.

(a) 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?*)

(b) Let u and v now be arbitrary vertices in an arbitrary directed graph G. Describe an algorithm to compute the shortest path distance from u to v in $G \setminus e$, for every edge $e \in E$, in $O(m \log n)$ time. Again, all edge weights are non-negative.

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: Let $G = (V, E)$ be a weighted undirected graph such that the weight of each edge is positive, and let $s \in V$ be a vertex such that the length of the shortest path (called *shortest distance*) from s to every vertex of G is $O(n)$. Modify Djikstra's algorithm so that it can compute the shortest distance from s to every vertex in $O(|V| + |E|)$ time.