COMP 271 Design and Analysis of Algorithms
2003 Spring Semester
Questions for Sixth Tutorial – March 28, 2003.

1. Let G be a connected undirected graph with weights on the edges. Assume that all the edge weights are distinct. Prove that the edge with the smallest weight must be included in any minimum spanning tree of G. *You have to prove this from first principles, i.e., you are not allowed to use the Lemmas proven in class or assume the correctness of Kuskal's or Prim's algorithm.*

2. Let $G = (V, E)$ be a connected undirected graph in which all edges have weight either 1 or 2. Give an $O(|V|+|E|)$ algorithm to compute a minimum spanning tree of G. Justify the running time of your algorithm. (*Note:* You may either present a new algorithm or just show how to modify an algorithm taught in class.)

3. Let $G = (V, E)$ be a weighted acyclic directed graph (DAG) with source vertex s (that is, s has indegree 0). Give an $O(|V| + |E|)$ algorithm for finding all vertices that can be reached from s and a shortest path tree that has shorstey paths from s to those vertices. It is assumed that the graph is given using an adjacency list.

   Section 24.2 in CLRS gives such an algorithm. In this problem we use dynamic programming to develop a slightly different one.

   Before starting the algorithm we first use $O(|V| + |E|)$ time to topologically sort the vertices. After this we assume that the vertices are given as $v_1, v_2, \ldots v_{|E|}$ in topological order.

   Next we run through the graph and create an *in-adjacency* list. That is, for each vertex v we create a list of all vertices $\{u : (u, v) \in E\}$. Show that this can be done in $(|V| + |E|)$ time.

   Let $D(u)$ be the length of a shortest path from s to u.

   Now prove the following statement:

$$\forall v \in V - \{s\}, \quad D(v) = \min_{u\,:\,(u,v)\in E} \left(D(u) + w(u, v)\right).$$

   Next, use this statement and a dynamic programming approach to find the shortest path tree in $O(|V| + |E|)$ time.