1. Let $G = (V, E)$ be an undirected graph. Design and analyze an $O(|V|)$-time algorithm to determine if $G$ is a tree.

2. Suppose the cost of a path is defined to be the maximum cost of any edge along the path, rather than the sum of the edge costs. Show how the Floyd-Warshall algorithm can be modified to handle this situation.

3. Let $G = (V, E)$ be a DAG with weighted edges, where the cost associated with each edge may be an arbitrary real number (possibly negative). Show how to find a path with maximum cost in $G$ in time $O(|V| + |E|)$.

4. Consider the following divide-and-conquer algorithm that purports to find a minimum spanning tree for a weighted, undirected graph $G = (V, E)$. The algorithm splits the vertex set $V$ into two disjoint sets $V_1$ and $V_2$, that differ in size by at most 1. Let $E_1$ be the set of edges in $E$ that connect vertices in $V_1$, and let $E_2$ be the set of edges in $E$ that connect vertices in $V_2$. The algorithm recursively solves a minimum-spanning-tree problem for the graphs $(V_1, E_1)$ and $(V_2, E_2)$. Finally, among all the edges that connect a vertex in $V_1$ to a vertex in $V_2$, it selects an edge of minimum weight, and uses this to unite the two subtrees found in the recursive step.

Either argue that the algorithm correctly computes a minimum spanning tree, or provide an example input for which the algorithm produces an incorrect result.