

Due at the beginning of class on Feb 28.

Undergraduate students: please do problems 1–5. Graduate students: please do problems 2–6.

If you cannot come up with algorithms that run in the required time, then provide (correct) slower algorithms for partial credit. Write your answers using *pseudo-code* in the same style as the textbook. These make the algorithm description precise, and easy to read (as opposed to code in C or some other language).

1. You are given a directed acyclic graph $G = (V, E)$ in adjacency list format. Give a linear-time algorithm to decide if G has a path that contains all the vertices, and to output such a path if it is present.
2. Suppose G is a strongly connected directed graph. We choose an arbitrary DFS tree T of G , and remove all the forward edges of T from G . Show that G still remains strongly connected.
3. Do problem 22.5-6 in the textbook, page 557. Analyze the running time of your algorithm (maximum credit for a linear-time algorithm), and also prove that the algorithm is correct.
4. Let $G = (V, E)$ be a connected undirected graph with a weight specified for each edge, and let T be a minimum spanning tree (MST) of G . We now increase the weight of each edge by the same value α (which can be negative, zero, or positive). Is it true that T still remains an MST? Prove or give a counterexample.
5. Let $G = (V, E)$ be a connected undirected graph, with a weight specified for each edge. We are also given a set F of edges in G , where F is a forest. Develop as efficient an algorithm as you can to find a spanning tree in G that contains F , and has minimum weight among all spanning trees that contain F . Prove that the algorithm is correct.
6. We are given a directed graph $G = (V, E)$ with a weight for each edge, and a vertex $r \in V$. A *directed spanning tree T with root r* is a spanning tree in G which is rooted at r , and is such that all edges of T are directed away from the root r . The *bottleneck value* of such a tree T is the weight of the maximum-weight edge in T . Design as efficient an algorithm as you can to find a directed spanning tree T with root r that has the minimum bottleneck value. Analyze the running time of your algorithm, and prove that the algorithm is correct.