CMSC 451: Mid-Term Exam, Spring 2002
11AM – 12:15PM, March 14th 2002

If you cannot come up with complete solutions but have some partial idea that looks promising, please write that down clearly. Write your answers using pseudo-code in the same style as the textbook.

Undergraduate students: please do problems 1, 2, 3. Graduate students: please do problems 1, 2, 4.

1. Let \( G = (V, E) \) be an undirected graph represented in adjacency matrix format. Give as fast an implementation as you can, of depth-first search. (If you prefer, you can just say how to modify the adjacency list version given in the textbook.) Spell out, with proof, the worst-case running time of your algorithm. (10 points)

2. Show that there is a directed graph \( G = (V, E) \), vertices \( s, t \in V \), and a weight for each edge in \( E \), such that the following holds:

   There exists a path \( P \) such that \( P \) is now a shortest path from \( s \) to \( t \) in \( G \). However, there is some value \( \alpha \) such that if we increase the weight of each edge by the same value \( \alpha \), then \( P \) is no longer a shortest path from \( s \) to \( t \) in \( G \).

   [Hint: There is such a graph \( G \) with very few vertices.] (5 points)

3. (For undergraduate students only.) You are given an undirected graph \( G = (V, E) \) in adjacency list format. You are also given that \( G \) is a simple cycle: i.e., we can order the vertices as \( v_0, v_1, \ldots, v_{n-1} \) (where \( n = |V| \)) such that the set of edges \( E \) is exactly the set

\[
\{(v_0, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n), (v_n, v_0)\}.
\]

Given a weight for each edge of \( G \), give an \( O(V) \) time algorithm to find a minimum spanning tree in \( G \). [Hint: Make use of the fact that \( G \) is a simple cycle.] (5 points)

4. (For graduate students only.) You are given an undirected binary tree \( T = (V, E) \) in adjacency list format; each edge has a given weight. The distance between any two vertices is the total weight of the unique path that connects them. Give an \( O(V^2) \) time algorithm to print out the distance between every pair of vertices in \( V \). Justify why the running time is \( O(V^2) \). (10 points)