CompSci 404.1 Name: ______________________________ **Homework 6**

Directed Graphs

1. Given a DAG and two vertices v and w, describe an algorithm to find the lowest common ancestor (LCA) of v and w. The LCA of v and w is an ancestor of v and w that has no descendants that are also ancestors of v and w.

   Computing the LCA is useful in multiple inheritance in programming languages, analysis of genealogical data (find degree of inbreeding in a pedigree graph), and other applications.

   Hint: determine a way to find the height of vertices v and w as a distance from a root. Use that to find the lowest common ancestor of v and w.

Topological Sort

2. Describe a method that checks whether or not a given permutation of a DAG's vertices is a topological ordering of that DAG.

3. A colleague suggests an alternate method to find the topological ordering of a DAG: run BFS, and label the vertices by increasing distance to their respective source. Explain why your colleague's algorithm won't necessarily always produce a true topological ordering.

4. Design an algorithm to determine whether a directed graph has a unique topological ordering.

   Hint: If the directed graph has multiple topological orderings, then a second topological order can be obtained by swapping a pair of consecutive vertices.

Strongly Connected Components

5. Describe what the strongly connected components of a directed acyclic graph (DAG) are.

. . . .

. . . .

. . . .

. . . .

6. A colleague doesn't believe that the the strongly connected components in G^R are the same as in G. How can you prove your colleague wrong?

. . . .

. . . .

. . . .

. . . .

7. Why do we need Kosaraju's algorithm? Why can't we simply use DFS or BFS on a DAG to determine strongly connected componets like we do in undirected graphs?

. . . .

. . . .

. . . .

. . . .

8. The reverse postorder of a graph's reverse is the same as the postorder of the graph. Is there any truth to this statement? Why or why not?

. . . .

. . . .

. . . .

. . . .

9. Why is it necessary to first transpose the graph (reverse the edges) in Kosaraju's algorithm? Could the algorithm have been designed to not require the transpose?

. . . .

. . . .

. . . .

. . . .