Department of Computer Science **University of San Francisco**

Computer Science 673
Fall 2016
Homework 8: Graphs I
Due Friday, October 28th

All problem / exercise numbers are from the **3nd edition** of Introduction to Algorithms (the 1st and 2nd editions are different!)

1. (6 points) Exercise 22.1-6 When an adjacency-matrix representation is used, most graph algorithms require time $\Omega(|V|^2)$, but there are some exceptions. Show that determining whether a directed graph G contains a universal sink – a vertex with in-degree $|V|-1$ and out-degree 0, can be determined in time $O(|V|)$, given an adjaceny matrix for G. That is, you should 1. Describe the main idea behind your algorithm, and why it works, and 2. Give pseudocode.

2. (6 points) Exercuse 22.5-3 Professor Bacon claims that the algorithm for strongly connected components would be simplier if it used the original (instead of the transpose) of the graph in the second depth-first search, and scanned the verticies in *increasing* finishing times. Does this simplier algorithm always produce correct results?

3. (6 points) Exercise 22.4-2 Give a linear-time $(O(V+E))$ algorithm that takes as input a directed acyclic graph $G=(V,E)$ and two vertices s and t, and returns the number of paths from s to t in G. (See text for examples) (this one is a little tricky). As with the other problems, explain how your algorithm works, **and** give pseudocode.

4. (6 points) Exercise 22.4-3 Give an algorithm that determines whether or not a given undirected graph $G=(V,E)$ contains a cycle. Your algorithm should run in $O(|V|)$ time, independent of $|E|$. You may assume that you have an adjacency list representation of G, and that G is a well-formed graph (though you cannot assume that G is necessarily connected.) Be careful! Give pseudocode, and make sure that your algorithm works in all cases. I want something exact for this one.