EECS 311: Data Structures and Data Management Homework 6
Assigned: 11/07/08 Due: 11/14/08

1 Reading.

Chapter 9, Sections 0–3.2.

2 Problems.

This homework is shorter than usual and will count for 10 points (instead of the usual 20). The late penalty is 2.5 points.

1. Give an algorithm to determine whether an undirected graph contains an odd-length cycle. An odd-length cycle is one with an odd number of edges in it. What is the runtime of your algorithm?

2. We discussed in class the following high-level algorithm for *topological sort* on a directed graph $G = (V, E)$.

   Repeat until no vertices are left:

   (a) Find a vertex v with no incoming edges.

   (b) Process v.

   (c) Remove v from graph.

   These high-level steps are correct, though it is not exactly clear how one implements Step 2a and Step 2c efficiently. Naively, Step 2a would take $O(n^2)$ time with both *adjacency matrix* and *adjacency list* representations of G. Thus, the most obvious runtime analysis of the above high-level algorithm would given an $O(n^3)$ bound on the topological sort runtime. This is a bit ridiculous! Fortunately, with some auxiliary data it is possible to perform a topological sort in $O(n + m)$ time.

   Assume that the graph G is given via the *adjacency list* representation. Give detailed pseudo-code specifying precisely how each step is performed. Your pseudo-code should be at the level of accessing array entries and accessing linked-lists (you can assume we all know how to search, add, and remove from a linked list; i.e., do not bother describing pointer manipulations). If you choose to maintain any auxiliary data, be sure to specify what it represents (in English), specify how it is initialized (in your pseudo-code), and how it is maintained (in your pseudo-code). Your runtime should be $O(n + m)$.