CS 180, Fall 2015 Homework 4

The following homework is due on Wednesday, Oct 28th at the beginning of lecture.

When submitting your homework, please include your name at the top of each page. If you submit multiple pages, please staple them together. Late submissions are not accepted.

1. Give an iterative algorithm for the celebrity problem.
2. Recall that we can use DFS to assign post-order numbers to the nodes of a directed graph; the post-order number of a node is assigned when it becomes black.
   (a) Prove that the post-order numbers generated by running DFS on a DAG gives a reverse topological ordering of the DAG. (A reverse topological ordering of the nodes of a DAG is a numbering of vertices such that for every directed edge (u, v) the number assigned to u is higher than the number assigned to v.)
   (b) Let A be a strongly connected component of a directed graph G. Prove that the post-order number of the first vertex of A that is visited by DFS is higher than the post-order number of any other node in the A.
   (c) Consider a directed graph G, and let A and B be two strongly connected components in G. Prove that if there is an edge from a node in A to a node in B, then the maximum post-order number assigned in A is larger than the maximum post-order number assigned in B.
3. Consider the following variation of DFS that uses two colors *blue* and *red* to color the nodes of the graph as it traverses them. The first node in DFS is colored blue, and the algorithm assigns red color to every unvisited neighbor discovered from a blue node, and it assigns blue color to every unvisited neighbor discovered from a red node.
   (a) Prove that a directed strongly connected graph contains an odd cycle if and only if the DFS finds an edge from a red node to an already visited red node or an edge from a blue node to a visited blue node.
   (b) Prove that the above is not true in general directed graphs that are not strongly connected. In other words, show that if the DFS finds an edge from a colored node to an already visited node of the same color, this does not imply that the graph has an odd cycle. Show this by giving an example.
   (c) Give an algorithm that finds an odd cycle in a directed graph (The graph may or may not be strongly connected).
4. Imagine you have a Tesla (If you do good on CS180 and go to Google you just might...), and you want to drive from Los Angeles to San Fransisco. A fully-charged Tesla can drive for 200 miles without the need of charge. There are many ways to go from LA to SF, and some cities on the way have charging stations and some don't. The underlying map of the cities and the roads between them can be viewed as a weighted undirected graph, and the weight of each edge is the distance between endpoints.
   (a) Assume that you drive a constant (high) speed and when stop at a charging station the car gets fully charged instantaneously. Also, you are given a procedure that calculates the shortest path between any two charing stations. Design an algorithm to find a path to get to SF from LA as fast as possible. You need to make sure your Tesla doesn't run out of power in the middle of your trip. Therefore, any path that doesn't satisfy this requirement is not a feasible path. The goal is to find the shortest path in a restricted set of paths that are feasible.

(b) Let's change the assumptions of the problem a little and assume that charging is extremely slow, and the time spent in a charging station is much more than the time spent traveling between different cities. Thus, we can assume that all distances between cities are negligible (or simply zero). With these assumptions the best path is the path that minimize the number of stops at charing stations. Design an algorithm that finds a path from LA to SF with the minimum number of stops at charging station.