

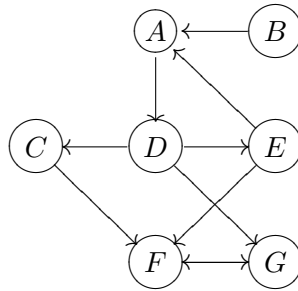
# Homework 3

CS157 - Spring 2009

Due: Tuesday March 10, 2009 10:30am

## Problem 1: Strongly Connected Components

List the vertices in each strongly connected component of the following graph, then draw its *meta-graph*, as defined on page 92 of the text. Label each node of the meta-graph with the node from the corresponding connected component whose label occurs first in the alphabet.



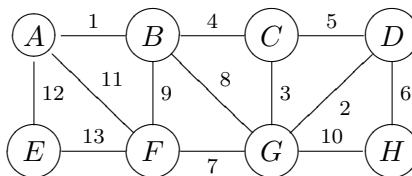
## Problem 2: Union-Find

After each of the following sets of Union-Find operations, display the forest of nodes in the disjoint-set data structure such that each of the root nodes and every level of each tree is ordered lexicographically (A-Z). Assume that both union by rank and path compression are in use. If there is ambiguity in the union operation, always choose the node with lowest lexicographic order as the root node (A instead of Z).

- (a) `makeset(A)`, `makeset(B)`, `makeset(C)`, `makeset(D)`, `makeset(E)`, `makeset(F)`, `makeset(G)`
- (b) `union(A, B)`, `union(B, C)`, `union(D, E)`, `union(D, F)`
- (c) `union(C, E)`
- (d) `union(G, E)`

## Problem 3: Prim vs. Kruskal

Suppose that Prim's algorithm is executed on the following graph starting with node A. Display the resulting MST, labeling each edge in the order it is added (you do not need to show the edge weights). Do the same using Kruskal's algorithm.



## Problem 4: Heap-based Priority Queue

Suppose that we use a binary heap as our priority queue implementation in the execution of Prim's algorithm in Problem 3. For each of the following successive states of the heap, display the heap with each node labeled with its corresponding vertex and its current weight. Please refer to the algorithm as described on page 139 of the textbook. Assume that the nodes are initially loaded into the tree from top to bottom and left to right in lexicographic order, and that at each iteration we loop through the edges in increasing order of their weights.

- (a) The algorithm has completed the first iteration of the `while` loop, so that `deletemin` has been called once and `decreasekey` has been called for each of the edges adjacent to the start vertex ( $A$ ).
- (b) The algorithm has completed another call to `deletemin`.
- (c) The algorithm has completed another set of `decreasekey` calls.