CSCI 4041, Spring 2019, Quiz 6 (30 minutes, 20 points)

Name:

x500:

Discussion Start Time (**circle one**): 3:35 4:40 5:45 6:50 7:55 other:

---

1. (1 points each) True/False - Circle one. Note that when asking about the properties of an algorithm, we specifically mean the version of that algorithm discussed in lecture.

True **False** Depth First Search will never find the shortest path from the starting node to each node reachable from the start.

While DFS is not guaranteed to find the shortest path from start to each node for all graphs, it will in some cases.

True False Depth First Search uses a stack or recursion.

DFS uses a stack to keep track or recursion to keep track of which nodes it has discovered, but not yet expanded (that is, it hasn't yet explored the neighbors of those nodes)

True False If you run DFS and BFS on an undirected line of nodes of length 101, starting in the middle, BFS will use less memory than DFS.

In this situation BFS will only need to have two nodes on the queue at any given time, one in the left direction, and one in the right. DFS will need to keep up to half of the nodes on the stack.

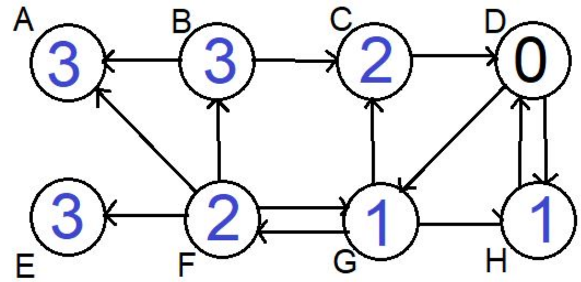
True False The Bellman-Ford algorithm runs in O(VE) time.

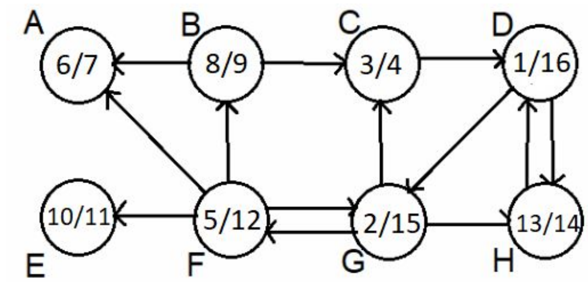
The Bellman-Ford algorithm consists of two nested loops: the outer one loops a number of times equal to the vertices -1, and the inner loops through all of the edges.

True False Dijkstra's algorithm is optimal so long as there are no negative weighted edges in the graph.

Dijkstra's algorithm does find the optimal shortest path in any scenario where there are no weighted edges in the graph.

2. The following questions concern the graphs shown on the right:





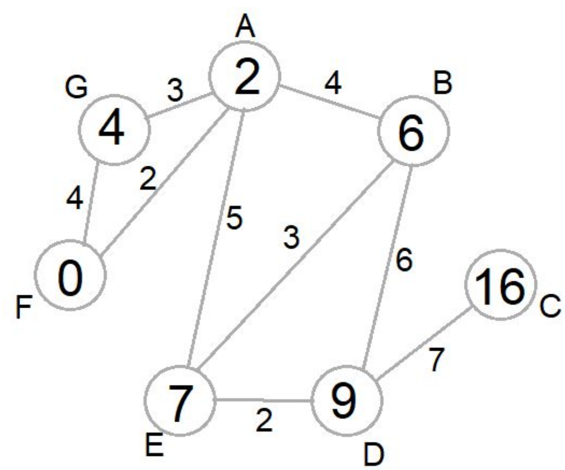
a. (1 point) What is the out-degree of node G?
3

b. (1 points) Which node(s) are sinks, if any?
A and E

c. (2 points) Suppose we ran BFS starting at node D. Write the depth of each node within the circle representing the node on the upper graph

d. (2 points) Suppose we ran DFS starting at node D, and the adjacency list of each node was in alphabetical order. Write the discovery time and finish time for each node (in the format node.d/node.f) within the circle representing the node on the lower graph.

3. (4 points) Compute the minimum weight path from node F to every other node in the graph to the left, using any method. Write the final distance value to each node within the circle representing the node.



4. Suppose you are given a set $\{x_1, x_2, x_3, \ldots x_n\}$ of points on the real line (that is, floating point numbers). Consider the problem of determining the smallest set of unit-length closed intervals that contains all of the given points.

   For example, using the set {1.9, 7.0, 8.6, 8.8, 7.7}, all of the following sets of closed intervals contain all of the given points:
   { [1.2, 2.2], [6.1, 7.1], [7.3, 8.3], [8.6, 9.6] }
   { [1.9, 2.9], [6.5, 7.5], [7.7, 8.7], [8.8, 9.8] }
   { [1.5, 2.5], [7.0, 8.0], [8.5, 9.5] }
   but only the last set is a solution, because 3 is the minimum number of intervals required.

   a. (2 points) Describe, in English or pseudocode, a greedy algorithm that solves the above problem optimally.

```
coverPoints(X)
        MergeSort(X)
        i = 1
        covered = -inf
        Inters = []
        while i <= X.length
                if X[i] > covered
                        Inters.append([X[i], X[i]+1])
                        covered = X[i] + 1
                i += 1
        return Inters
```

Basically, find the minimum point X[i] not already covered by an interval, and add greedy choice interval [X[i], X[i]+1] to the solution set. Repeat until every point is covered.

   b. (3 points) Prove that the greedy choice is contained in at least one minimum size set of unit-length closed intervals.

Let $X = \{x_1, x_2, x_3, \ldots x_n\}$ be a set of points on the real line, and let Inters = $\{i_1, i_2 \ldots i_k\}$ be a minimum size set of intervals covering all the points. Let x_{min} be the smallest value in X, and let i_{greedy} be the greedy choice of interval $[x_{min}, x_{min}+1]$. If i_{greedy} is in Inters, then we're done. Otherwise, let i_k be the interval in Inters that covers x_{min}. $i_k = [n, n+1]$ for some $n \le x_{min}$, since i_k covers x_{min}. Therefore, i_{greedy} covers at least as many elements of X that are greater than x_{min} as i_k does, since $x_{min} + 1 >= n+1$, and there aren't any elements of X less than x_{min} by assumption, so i_{greedy} covers all of the elements of X that i_k does. Therefore, Inters - $\{i_k\}$ U $\{i_{greedy}\}$ is also a minimum size set of intervals covering all of the points in X. Thus, there exists at least one minimum size set of intervals covering all of the points in X that contains the greedy choice i_{greedy}, so the greedy choice property holds.