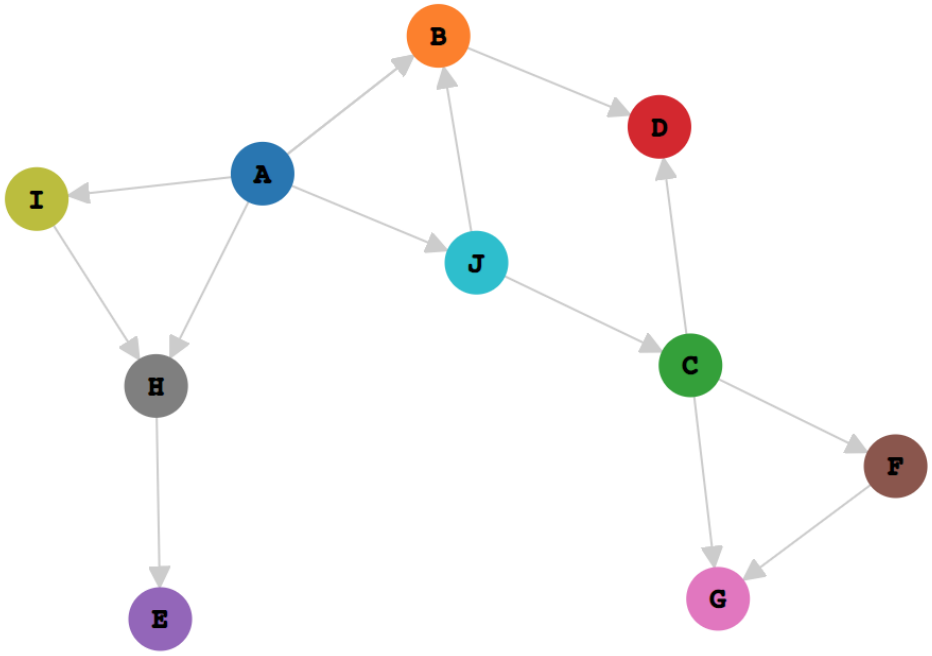
Name(s)_____ _____

Algorithms, Assignment 03: Graph Search, Topological Orderings, and Representations

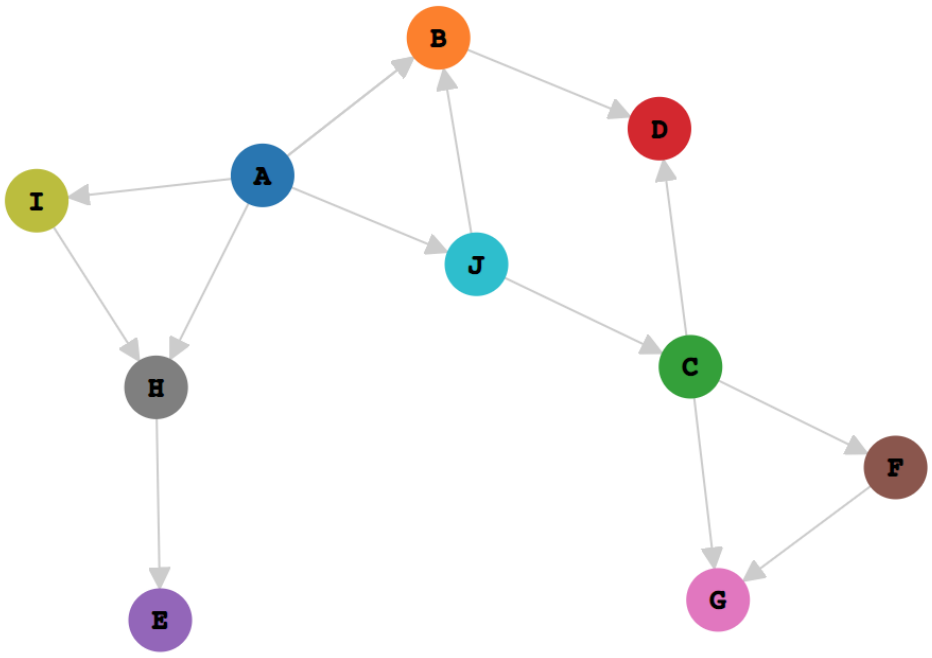


1. Starting at vertex A, in what order are the nodes visited by a **breadth**-first search? Show your work stepping through the code. *Note: you should choose edges in alphabetical order (based on the destination vertex).*

   While stepping through a BFS you must also note the "layer" (the number of hops away from the A) for each vertex.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
BFS Order	1									
Layer	0									

Name(s)_____ _____



2. Starting at vertex A, in what order are the nodes visited by a **depth**-first search? Show your work stepping through the code. *Note: you should choose edges in alphabetical order (based on the destination vertex).*

   While stepping through a DFS you must also compute a topological ordering.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
DFS Order	1									
f-values										

Name(s)_____ _____

3. Consider graphs with the following properties:
   i. connected (all vertices have a path to one another),
   ii. unweighted, and
   iii. undirected.

   The distance between two vertices in such a graph is just the minimum number of edges on a path between the vertices. The **diameter** of the graph is the maximum distance among all pairs of vertices.

   (a) Describe an algorithm for computing the diameter. The description can be in English or pseudocode (with comments if appropriate).

   (b) What is the running time of your algorithm? Explain.

   (c) In a few sentences, sketch a proof of the correctness of your algorithm. Your proof does not need to be formal.

Name(s)_____ _____

4. Given an adjacency list representation of a directed graph, where each vertex maintains an array of its outgoing edges (but **not** its incoming edges), how long does it take, in the worst case, to compute the in-degree (the number of edges pointing to the vertex) of a given vertex? Let n and m represent the number of vertices and edges, respectively, and let k denote the maximum in-degree of a vertex. Use Big-O notation.

5. Consider the following problem: given an undirected graph G, does there exist at least one path between any two specified vertices?

   If G is given in its adjacency list representation, the problem can be solved in O(m + n) time using BFS or DFS (make sure you see why this is true).

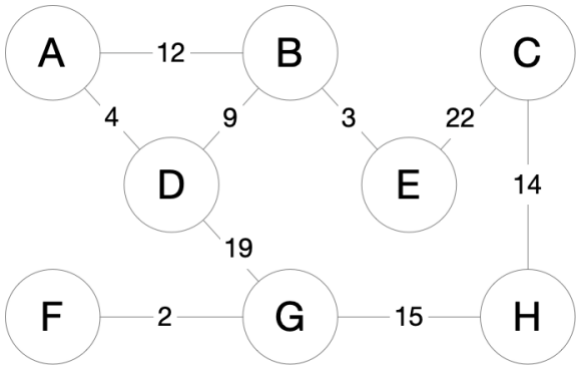
   Suppose instead that G is given in its adjacency **matrix** representation. What running time is required, in the worst case, to solve the computational problem stated above?

   Another way to think of this problem is: what is the running time of BFS or DFS when using an adjacency matrix?

   Please provide an explanation with your answer.

Name(s)_____ _____

6. Compute the shortest path from **D** to all other vertices.



A	B	C	**D**	E	F	G	H
4			0				

You must show your work below to receive full credit. Specifically, show your candidate edges (alphabetically) for each iteration of Dijkstra's Shortest Path Algorithm.

Iter	**V**	**vOther**	**weight**	**length**
1	D	A	4	4
		B	9	9
		G	19	19
2	A	B	12	16
	D	B	9	9
		G	19	19
3				

Name(s)_____________________________________ ____