- Please note that one of the main goals of this course is to design efficient algorithms. So, there are points for efficiency, even though we may not explicitly state this in the question.
- Unless otherwise mentioned, assume that graphs are given in adjacency list representation.
- In the lectures, we have discussed an $O(V + E)$ algorithm for finding all the SCCs of a directed graph. We can extend this algorithm to design an algorithm `CreateMetaGraph(`G`)` that outputs the meta-graph of a given directed graph in $O(V + E)$ time. You may use `CreateMetaGraph(`G`)` as a sub-routine for this homework.
- The other instructions are the same as in Homework-1.

There are 6 questions for a total of 82 points.

---

1. A tree is defined as an undirected graph containing no cycles. An undirected graph is said to be connected iff, for every pair of vertices, there is a path between them. For this question, you have to show the following statement:

   Any connected undirected graph with n vertices and $(n-1)$ edges is a tree.

   We will prove the statement using Mathematical Induction. The first step in such proof is to define the propositional function. Fortunately, this is already given in the statement of the claim for this problem.

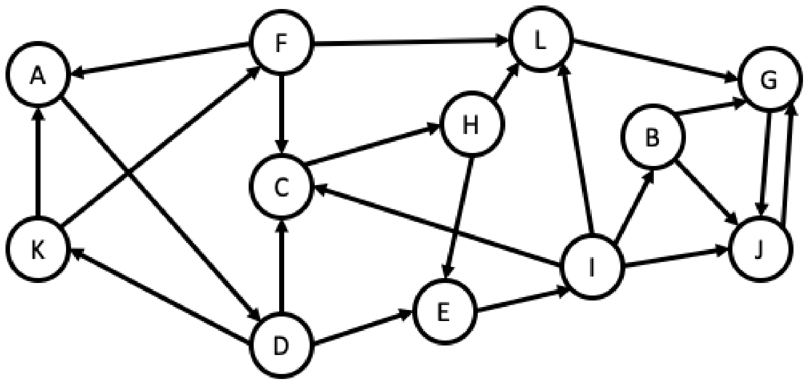
   $P(n)$: Any connected undirected graph with n vertices and $(n-1)$ edges is a tree.

   The base case is simple. $P(1)$ holds since any graph with 1 node and 0 edges is indeed a tree. For the inductive step, we assume that $P(1), P(2), ..., P(k)$ holds for an arbitrary $k \geq 1$, and then we will show that $P(k+1)$ holds. Consider any connected graph G with $(k+1)$ nodes and k edges. You are asked to complete the argument by doing the following:

   (a) (3 points) Show that G has a node v with degree 1.

   (b) (2 points) Consider the graph G' obtained from G by removing vertex v and its edge. Now use the induction assumption on G' to conclude that G is a tree.

2. Consider the following directed graph and answer the questions that follow:



   (a) ($1/2$ point) Is the graph a DAG?

   (b) (1 point) How many SCCs does this graph have?

   (c) ($1/2$ point) How many source SCCs does this graph have?

(d) (1 point) Suppose we run the DFS algorithm on the graph exploring nodes alphabetically. Given this, what is the pre-number of vertex F?

(e) (1 point) Suppose we run the DFS algorithm on the graph exploring nodes alphabetically. Given this, what is the post-number of vertex J?

(f) (1 point) Is it possible to add a single edge to this graph so that the graph becomes a strongly connected graph? If so, which edge would you add?

3. You are given a directed graph $G = (V, E)$ in which each node $u \in V$ has an associated *price*, denoted by $price(u)$, which is a positive integer. The *cost* of a node u, denoted by $cost(u)$, is defined to be the price of the most expensive node reachable from u (including u itself). Your goal is to compute the cost of every node in the graph (this can be produced as an array *cost* of size $|V|$).

   (a) (9 points) Give a linear time algorithm that works for DAG's. (*Hint: Handle the vertices in a particular order*)

   (b) (9 points) Extend this to a linear time algorithm that works for any directed graph. (*Hint: Consider making use of the meta-graph of the given graph.*)

   Give running time analysis and proof of correctness for both parts.

4. Given a directed graph $G = (V, E)$ that is not a strongly connected graph, you have to determine if there exists a pair of vertices $u, v \in V$ such that the graph $G' = (V, E \cup \{(u, v)\})$ is strongly connected. In other words, you must determine if there exists a pair of vertices $u, v \in V$ such that adding a directed edge from u to v in G converts it into a strongly connected graph. Design an algorithm for this problem. Your algorithm should output "yes" if such an edge exists and "no" otherwise.

   (a) (9 points) Give a linear time algorithm for DAGs.

   (b) (9 points) Extend this to a linear time algorithm that works for any directed graph. (*Hint: Consider making use of the meta-graph of the given graph.*)

   Give running time analysis and proof of correctness for both parts.

5. (18 points) A directed graph $G = (V, E)$ is called one-way-connected if, for all pair of vertices u and v, there is a path from vertex u to v **or** there is a path from vertex v to u. Note that the "or" in the previous statement is a logical OR, not XOR. Design an algorithm to check if a given graph is one-way-connected. Give running time analysis and proof of correctness.

6. (18 points) Design an algorithm that given a directed acyclic graph $G = (V, E)$ and a vertex $u \in V$, outputs all the nodes that have a simple path to u with a length that is a multiple of 3. *(Recall that a simple path is a path with all distinct vertices.)*

   Give running time analysis and proof of correctness.