Problem Set 5

Assigned: June 22
Due: June 29

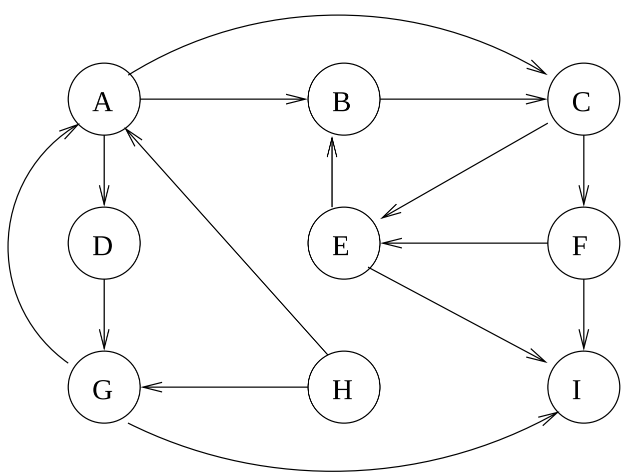
Problem 1

(CLR&S Ex. 23.1-6) When an adjacency matrix representation is used, most graph algorithms require time $\Omega(N^2)$ but there are some exceptions. Show that determining whether a directed graph contains a **sink** — a vertex with in-degree $|V|-1$ and out-degree 0 — can be determined in time $O(|V|)$ even if an adjacency matrix representation is used.

Problem 2

Execute a depth-first search on the graph shown below. Assume that the top-level routine DFS(G) searches the vertices in alphabetical order and that the recursive routine DFS-VISIT(U) enumerates the out-edges from U in alphabetical order. Your answer should specify:

- The preorder sequence found
- The postorder sequence found.
- The classification of every edge (tree edge, forward edge, cross edge, or back edge.)



Problem 3

Delete edges B $\rightarrow$ C, and D $\rightarrow$ G from the figure in problem 2. The figure is now a DAG.

A. Compute the topological sort of the graph using the DFS-based routine in the text. As in problem 2, assume that vertices are examined in alphabetical order.

B. Compute the topological sort of the graph using the following algorithm, discussed in class. Again, assume that vertices are examined in alphabetical order.

```
L = [];
for (i = 1 to |V|) {
  W = a node with in-degree 0;
  add W to the end of L;
  delete W and all its outarcs from the graph;
  }
```

Problem 4

Explain how the algorithm in problem 3.B can be implemented to run in time $\Theta(|V| + |E|)$.

Problem 5

A. Let G be a DAG. Show that there is an ordering of the vertices in the top level DFS(G) in which every edge is a cross-edge.

B. Show that there exists a DAG G such that any DFS of G produces at least one cross-edge, no matter how the vertices are ordered. (Hint: you need only three vertices.)