1. Prove that if a graph $G = (V, E)$ is connected, then $\lg|E| = \Theta(\lg|V|)$.

2. The *incidence matrix* of a directed graph $G = (V, E)$ with no-self loops is a $|V| \times |E|$ matrix $B = (b_{ij})$ such that

$$b_{ij} = \begin{cases} -1 & \text{if edge } j \text{ leaves vertex } i, \\ 1 & \text{if edge } j \text{ enters vertex } i, \\ 0 & \text{otherwise.} \end{cases}$$

Describe what the entries of the matrix product BB^T represent, where B^T is the transpose of B.

3. In an undirected graph, the *degree* $d(v)$ of a vertex v is the number of neighbors v has, or equivalently, the number of edges incident upon it. In a directed graph, we distinguish between the *indegree* $d_{\text{in}}(v)$, which is the number of edges into v, and the *outdegree* $d_{\text{out}}(v)$, the number of edges leaving v.

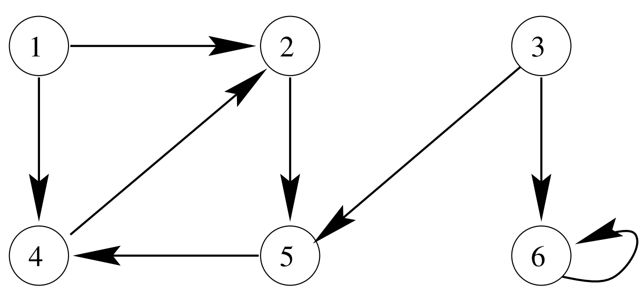
(a) Show that in an undirected graph $G = (V, E)$, $\sum_{v \in V} d(v) = 2|E|$.

(b) Use part (a) to show that in an undirected graph, there must have an even number of vertices whose degree is odd.

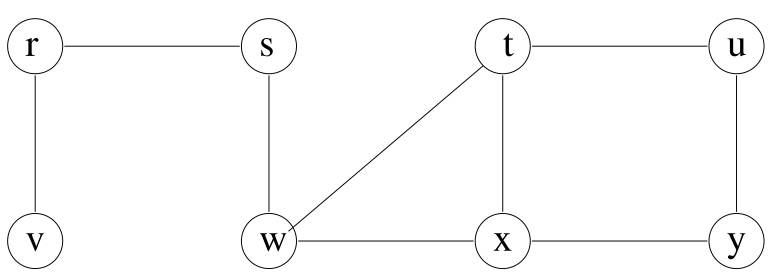
(c) Does a similar statement hold for th number of vertices with odd indegree in a directed graph?

4. Give an algorithm that determining whether a directed graph contains a *sink*, namely a vertex with in-degree $|V| - 1$ and out-degree 0, in $O(|V|)$ time.

5. Show the result of running BFS on the following directed graph, using vertex **3** as the source. *Make sure to show show all intermediate results.*



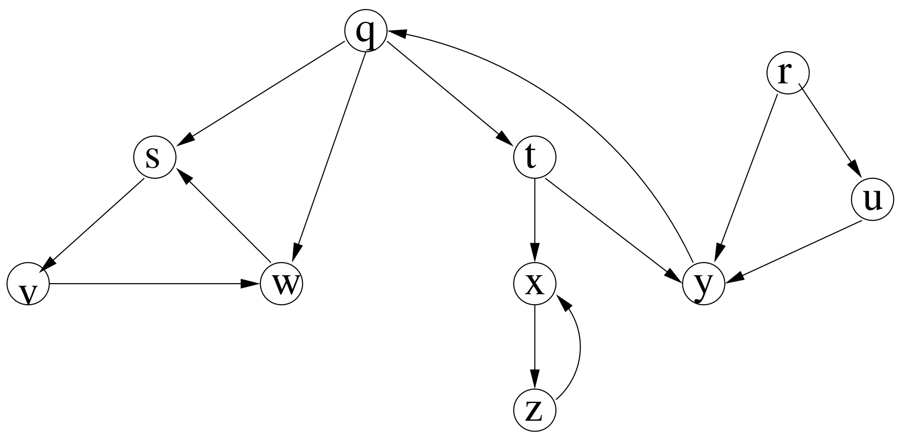
6. Show the result of running BFS on the following undirected graph, using vertex **u** as the source. *Make sure to show show all intermediate results.*



7. Show how DFS works on the following graph. Assume that the for-loop of the DFS procedure considers the vertices in alphabetical order, and assume that each adjacency list is ordered alphabetically.

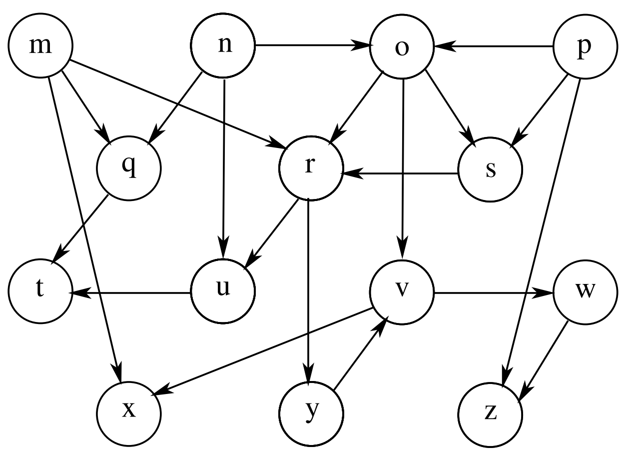
    (1) Show the discovery and finishing times for each vertex.

    (2) Mark the classification of each edge.



8. Show that a DFS of an undirected graph G can be used to identify the **connected component** of G, and that the DFS contains as many trees as G has connected components. More precisely, show how to modify DFS so that each vertex v is assigned an integer label $cc[v]$ between 1 to k, where k is the number of connected components of G, such that $cc[u] = cc[v]$ if and only if u and v are in the same connected component.

9. Show the ordering of vertices proceduced by Topological-Sort when it is run in the following dag, where it is assumed that the for-loop of the DFS procedure considers the vertices in alphabetical order, and assume that each adjacency list is ordered alphabetically.



10. Give an algorithm that determines whether or not a given undirected graph $G = (V, E)$ contains a cycle. Your algorithm should run in $O(|V|)$ time, independent of $|E|$.