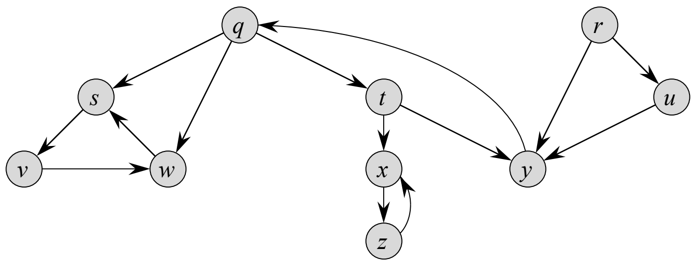
# Exercise Session 10

**Exercise 1.**
(CLRS 22.1-3) The transpose of a directed graph $G = (V, E)$ is the graph $G^T = (V, E^T)$, where $E^T = \{(v, u) \in V \times V : (u, v) \in E\}$. Describe efficient algorithms for computing $G^T$ from $G$, for both adjacency-list and adjacency-matrix representations of $G$. Analyse the running times of your algorithms.

**Exercise 2.**
The diameter of a tree $T = (V, E)$ is defined as $\max_{u,v \in V} \delta(u, v)$, that is, the largest of all shortest-path distances in the tree. Give an efficient algorithm to compute the diameter of a tree, and analyse the running time of your algorithm.

**Exercise 3.**
Consider the graph $G$ depicted below.




(a) Write the intervals for the discovery time and finishing time of each vertex in the graph obtained by performing a depth-first search visit of $G$.

(b) Write the corresponding "parenthesization" of the vertices in the sense of Theorem 22.7 in CLRS

(c) Assign with each edge a label $T$ (tree edge), $B$ (back edge), $F$ (forward edge), $C$ (cross edge) corresponding to the classification of edges induced by the DFS visit performed before.

(d) If $G$ admits a topological sorting, then show the result of TOPOLOGICAL-SORT($G$).

**Exercise 4.**
(CLRS 22.4-5) Another way to perform topological sorting on a directed acyclic graph $G = (V, E)$ is to repeatedly find a vertex of in-degree 0, output it, and remove it and all of its outgoing edges from the graph. Explain how to implement this idea so that it runs in time $O(|V| + |E|)$. What happens to this algorithm if $G$ has cycles?

**Exercise 5.**
Assume $G$ is a directed acyclic graph. Give an efficient algorithm to compute the graph of strongly connected components of $G$, and analyse the running time of your algorithm.

**Exercise 6.**
(CLRS 22.5-1) How can the number of strongly connected components of a graph change if a new edge is added?