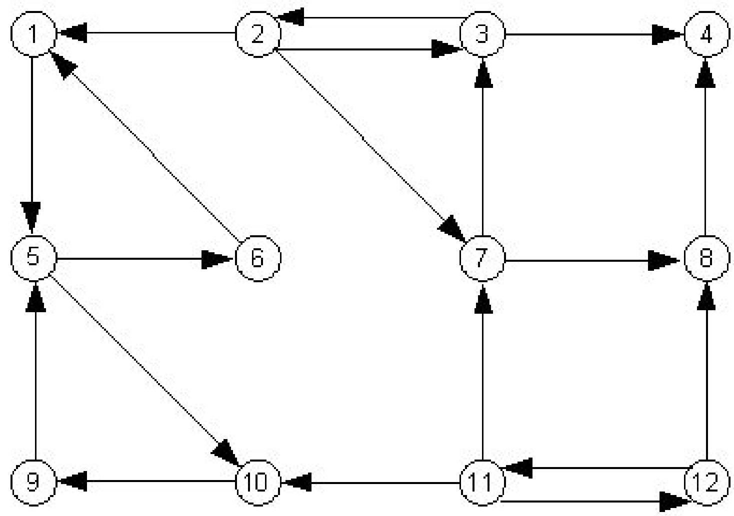
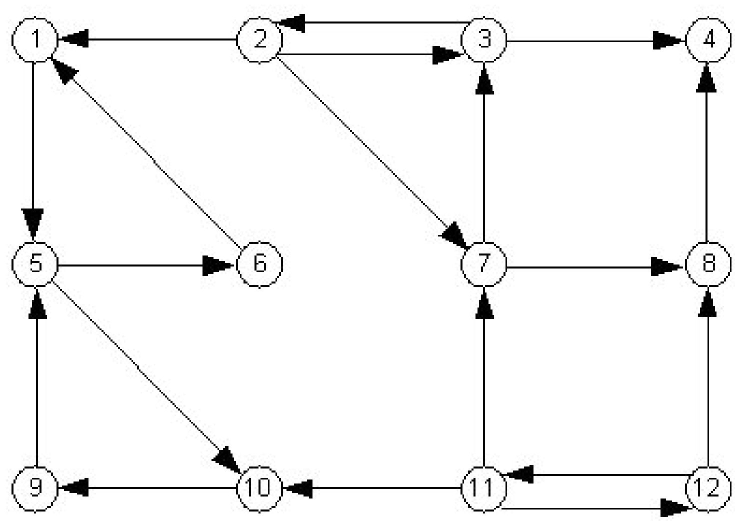
Algorithms and Data Structures
Spring 2018
Exercises for Units 19 and 20

1. An easy exercise: Execute Sharir's double DFS algorithm to find the strongly connected components in the following graph (which for convenience is given twice).





2. In general graphs it is very difficult to find the longest simple path between two given nodes s and t, i.e. the simple path with the most edges. If one could find such path easily it would be possible to efficiently solve the Hamiltonian Path Problem, which is known to be NP-complete.

   Show that in a directed acyclic graph the longest directed path between two nodes can be found in linear time, i.e. time $O(m + n)$.

3. Let $G = (V, E)$ be a connected undirected graph. An *articulation point* of G is a vertex whose removal disconnects G. Similarly, a *bridge* is an edge whose removal disconnects G. A *biconnected component* of G is a maximal set of edges such that any two edges in the set lie on a common simple cycle.

   () Draw an appropriate example graph and illustrate to yourself these three concepts.

   Let $G_\pi = (V, E_\pi)$ be a DFS tree of G.

   (a) Give a test for telling whether the root of G_π is an articulation point.

   (b) Let v be a non-root vertex of G_π. Prove v is an articulation point iff there is no back edge (u, w) such that in G_π the vertex u is a descendant of v and w is a proper ancestor of v.

   (c) Let $low[v]$ be the minimum of $d[v]$ and $\min\{d[w] : (u, w)$ is a back edge for some descendant u of $v\}$. Show how to compute $low[v]$ for all vertices $v \in V$ in $O(|E|)$ time.

   (d) Show how to compute all articulation points in $O(|E|)$ time.

   (e) Prove that an edge of G is a bridge iff it does not lie on any simple cycle of G.

   (f) Show how to compute all bridges of G in $O(|E|)$ time.

   (g) Prove that the biconnected components partition the non-bridge edges of G.

   (h) Give a linear time algorithm to label each edge e of G with a positive integer $bcc[e]$ such that $bcc[e] = bcc[e']$ iff e and e' are in the same biconnected component.

4. Assume all edge lengths in a graph are integers in the range 1 through K. Show that the single source shortest path problem for such a graph can be solved in time $O(m + n \log K)$. (Of course n is the number of nodes in the graph, and m is the number of edges.)

   Hint: Can you arrange things so that the priority queue used in Dijkstra's algorithm contains at all times only $O(K)$ distinct keys?

5. Let G be a directed graph, where each edge is colored either red or blue. Let u and v be two vertices of G.

   (a) Design an efficient method for deciding whether there exists a directed path from u to v that contains more red edges than blue edges.

   (b) Design an efficient method for finding a path from u to v that contains as few red edges as possible.