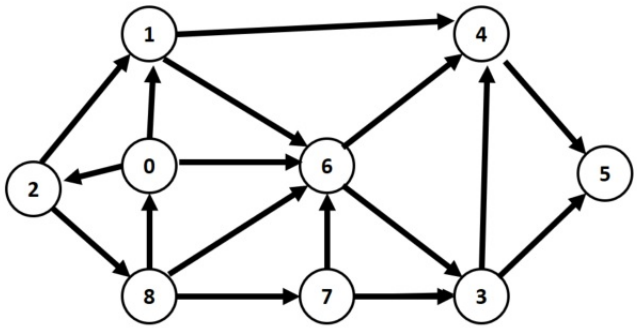
PRINCETON UNIVERSITY

COS 226–Algorithms and Data Structures

Week 8: *Graph Algorithms (Algs. §4.2 & videos §14.C,14.D,15.C)*

Version: November 9, 2017

Exercise 1 – Graph Traversal



Consider the digraph given above. Assume the adjacency lists are in sorted order: for example, when iterating through the edges pointing from 0, consider the edge $0 \to 1$ before $0 \to 2$ or $0 \to 6$ in that order.

A. Run depth-first search on the digraph, starting from vertex 0. List the vertices in postorder.

B. Run breadth-first search on the same digraph, starting from vertex 0. List the vertices in the order they are visited.

©Fall 2017 by COS 226 Staff.

Exercise 2 – Undirected Graphs

Conisder the code below where G is an undirected graph whose vertices are marked as integers from $0...V-1$. The integer s is in the range $0...V-1$

```
1  private void foo(Graph G, int s) {
2   Queue<Integer> queue = new Queue<Integer>();
3   for (int v = 0; v < G.V(); v++){
4      distTo[v] = Integer.MAX_VALUE;
5      marked[v] = false;
6   }
7   distTo[s] = 0;
8   marked[s] = true;
9   q.enqueue(s);
10  while (!q.isEmpty()) {
11    int v = q.dequeue();
12    for (int w : G.adj(v)) {
13      if (!marked[w]) {
14        edgeTo[w] = v;
15        distTo[w] = distTo[v] + 1; <=== line 1
16        marked[w] = true;          <=== line 2
17        q.enqueue(w);
18      }
19    }
20  }
21 }
```

A. Briefly describe the purpose of the foo method?

B. What is the purpose of the code marked line 1?

C. What if marked array is not updated (i.e code marked line 2 is removed)?

D. Determine the order of growth of the foo method in terms of V and E?

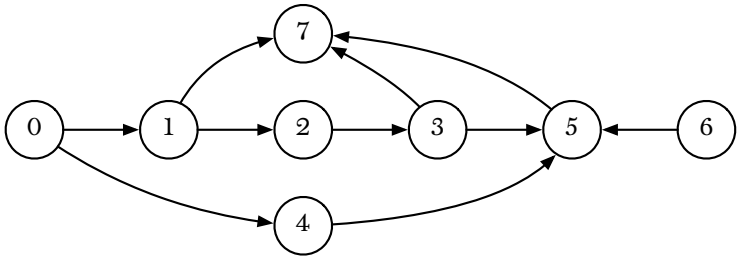
Exercise 3 – Shortest Common Ancestor

The following exercise looks at the algorithm required to compute the shortest common ancestor in a directed acyclic graph, which is an essential component of the WordNet assignment's `ShortestCommonAncestor` class.

In a directed graph, a vertex x is an *ancestor* of v if there exists a directed path from v to x. Given two vertices v and w in a rooted directed acyclic graph (DAG), a *shortest common ancestor* $\mathrm{sca}(v, w)$ is a vertex x which:

- is an ancestor of both v and w;
- minimizes the sum of the distances from v to x and w to x (this path, which goes from v to x to w, is the *shortest ancestral path* between v and w).

A. Is the following digraph a DAG? If so, is it a rooted DAG? Device an algorithm to identify a rooted DAG.



B. In the digraph given in Part A, find the shortest common ancestor of vertices 1 and 4, and give the sum of the path lengths from these vertices to the shortest common ancestor.

C. Describe an algorithm for calculating the shortest common ancestor of two vertices v and w in a rooted DAG. Your algorithm should run in linear time (at most $V + E$).

[See other side for part D]

D. A shortest ancestral path of two subsets of vertices A and B is a shortest ancestral path over all pairs of vertices v and w, with v in A and w in B. How would your algorithm differ if A and B were two sets of vertices, rather than single vertices? In the example below $A = \{3, 10\}$ and $B = \{9, 12, 15\}$. Recall that the shortest path from a set of vertices A to another vertex w (not in A) is the minimum of the shortest paths from any of the vertices $v \in A$ to w.