

# 13.8 Exercises CS 5343: HW #3

For source code and help with exercises, please visit [java.datastructures.net](http://java.datastructures.net).

Finish by Sat., March 24

Reinforcement Do the circled problems:

R-13.1 Draw a simple undirected graph  $G$  that has 12 vertices, 18 edges, and 3 connected components. Why would it be impossible to draw  $G$  with 3 connected components if  $G$  had 66 edges?

R-13.2 Let  $G$  be a simple connected graph with  $n$  vertices and  $m$  edges. Explain why  $O(\log m)$  is  $O(\log n)$ .

R-13.3 Draw an adjacency list and adjacency matrix representation of the undirected graph shown in Figure 13.1.

R-13.4 Draw a simple connected directed graph with 8 vertices and 16 edges such that the in-degree and out-degree of each vertex is 2. Show that there is a single (nonsimple) cycle that includes all the edges of your graph, that is, you can trace all the edges in their respective directions without ever lifting your pencil. (Such a cycle is called an *Euler tour*.)

R-13.5 Repeat the previous problem and then remove one edge from the graph. Show that now there is a single (nonsimple) path that includes all the edges of your graph. (Such a path is called an *Euler path*.)

R-13.6 Bob loves foreign languages and wants to plan his course schedule for the following years. He is interested in the following nine language courses: LA15, LA16, LA22, LA31, LA32, LA126, LA127, LA141, and LA169.

The course prerequisites are:

- LA15: (none)
- LA16: LA15
- LA22: (none)
- LA31: LA15
- LA32: LA16, LA31
- LA126: LA22, LA32
- LA127: LA16
- LA141: LA22, LA16
- LA169: LA32.

Find the sequence of courses that allows Bob to satisfy all the prerequisites.

R-13.7 Suppose we represent a graph  $G$  having  $n$  vertices and  $m$  edges with the edge list structure. Why, in this case, does the insertVertex method run in  $O(1)$  time while the removeVertex method runs in  $O(m)$  time?

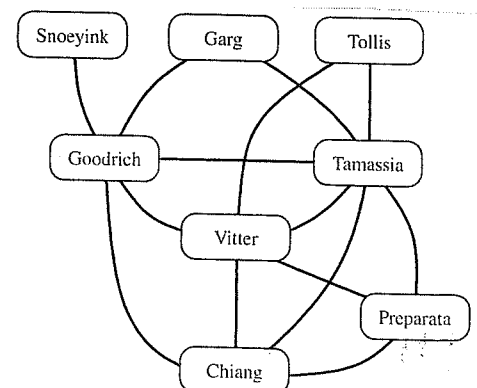
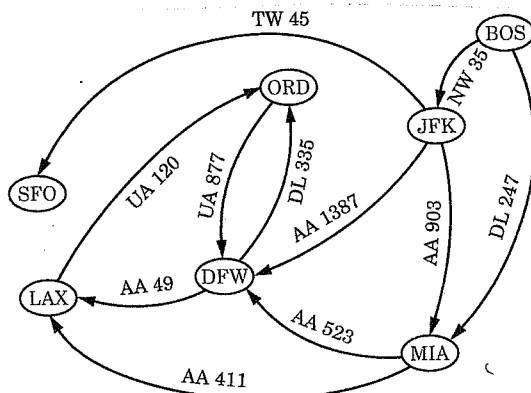


Figure 13.2: Example of a directed graph representing a flight network. The endpoints of edge UA 120 are LAX and ORD; hence, LAX and ORD are adjacent. Figure 13.1: Graph of coauthorship among some au

R-13.8 Let  $G$  be a graph whose vertices are the integers 1 through 8, and let the adjacent vertices of each vertex be given by the table below:

vertex	adjacent vertices
1	(2, 3, 4)
2	(1, 3, 4)
3	(1, 2, 4)
4	(1, 2, 3, 6)
5	(6, 7, 8)
6	(4, 5, 7)
7	(5, 6, 8)
8	(5, 7)

Assume that, in a traversal of  $G$ , the adjacent vertices of a given vertex are returned in the same order as they are listed in the table above.

- Draw  $G$ .
- Give the sequence of vertices of  $G$  visited using a DFS traversal starting at vertex 1.
- Give the sequence of vertices visited using a BFS traversal starting at vertex 1.

R-13.9 Would you use the adjacency list structure or the adjacency matrix structure in each of the following cases? Justify your choice.

- The graph has 10,000 vertices and 20,000 edges, and it is important to use as little space as possible.
- The graph has 10,000 vertices and 20,000,000 edges, and it is important to use as little space as possible.
- You need to answer the query `areAdjacent` as fast as possible, no matter how much space you use.

R-13.10 Explain why the DFS traversal runs in  $O(n^2)$  time on an  $n$ -vertex simple graph that is represented with the adjacency matrix structure.

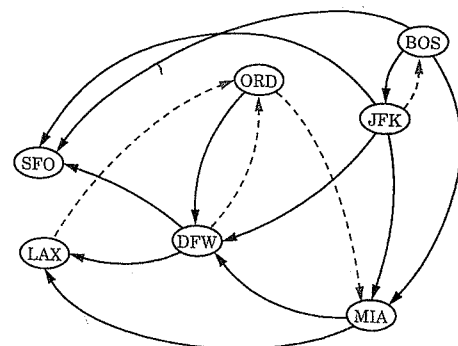
R-13.11 Draw the transitive closure of the directed graph shown in Figure 13.2.

R-13.12 Compute a topological ordering for the directed graph drawn with solid edges in Figure 13.8d.

R-13.13 Can we use a queue instead of a stack as an auxiliary data structure in the topological sorting algorithm shown in Code Fragment 13.13? Why or why not?

R-13.14 Draw a simple, connected, weighted graph with 8 vertices and 16 edges, each with unique edge weights. Identify one vertex as a "start" vertex and illustrate a running of Dijkstra's algorithm on this graph.

R-13.15 Show how to modify the pseudo-code for Dijkstra's algorithm for the case when the graph may contain parallel edges and self-loops.



13.8 (d)