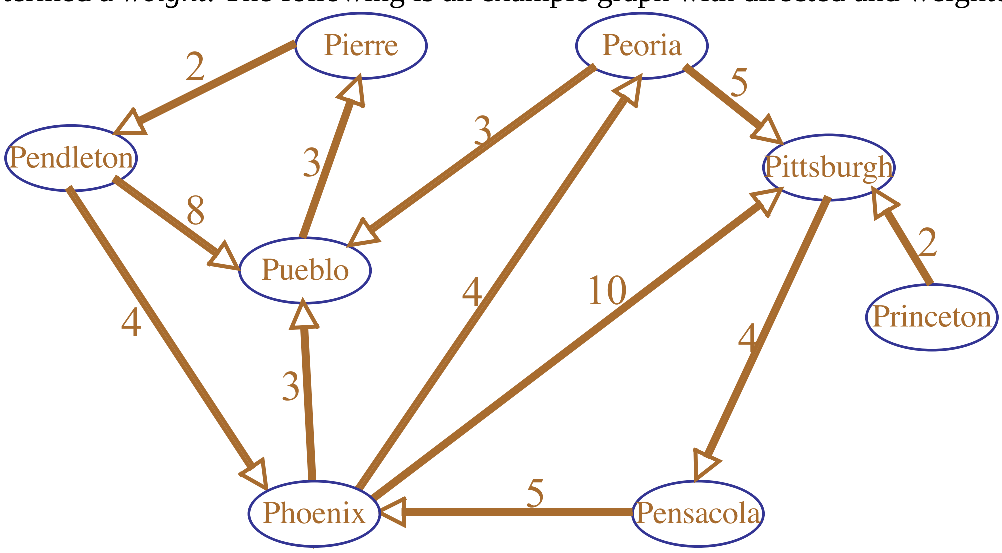
Worksheet 40: Graph Representations

In preparation: Reach Chapter 13 to learn more about graphs and graph algorithms.

As you learned in Chapter 13, a graph is composed of *vertices*, connected by *edges*. Vertices can carry information, often just a name. Edges can either be directed, which indicates an asymmetric relationship, or undirected. In addition, edges can carry a value, termed a *weight*. The following is an example graph with directed and weighted edges.



There are two common ways of representing a graph as a data structure. These are as an *adjacency matrix*, and as an *edge list*. To form an *adjacency matrix* the vertices are assigned a number. For example, we could number the vertices of the graph above by listing the cities in alphabetical order: 0-Pendleton, 1-Pensacola, 2-Peoria, 3-Phoenix, 4-Pierre, 5-Pittsburgh, 6-Princeton, and 7-Pueblo. An 8 by 8 matrix is then constructed. Each (i,j) entry describes the association between vertex i and vertex j. In an unweighted graph this entry is either 0 if there is no connection, or 1 if there is a connection between the two cities. In a weighted matrix the value is the weight on the arc linking the two cities, or the value infinity if there is no connection. In the space at right enter the edge list representation for the graph shown above. An adjacency matrix requires $O(v^2)$ space, where v is the number of vertices. This is true regardless of the number of edges.

	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

The alternative representation, the *edge list*, stores only the edges of a graph. This is advantageous if the graph is relatively sparse. Fundamentally, the edge list uses a map indexed by the vertex (or vertex label). For an unweighted graph the value is a set of vertices that represent the neighbors of the key vertex. In a weighted graph the value is itself represented by another map. In this map the key is the neighboring vertex, and the value is the weight of the arc that connects the two vertices. Complete the following edge list representation of the earlier graph. One entry has been made for you already.

Pendleton: {Pueblo: 8, Phoenix: 4}

Pensacola: { Phoenix: 5 }

Peoria: { Pueblo: 3, Pittsburgh: 5 }

Phoenix: { Pueblo: 3, Peoria: 4, Pittsburgh: 10 }

Pierre: { Pendleton: 2 }

Pittsburgh: { Pensacola: 4 }

Princeton: { Pittsburgh: 2 }

Pueblo: { Pierre: 3 }

Let v represent the number of vertices in a graph, and e the number of edges. Assuming that e > v, the edge list representation requires O(e) storage.