**Graphs 2**

**Weighted Adjacency Matrix: Floyd's Algorithm**

A *weighted* graph contains weight (or distance or cost) information about the edges, as shown below. Then the typical question to ask concerns the shortest (or cheapest) path.

For simplicity, we will deal with graphs that do not have loops or negative cycles. This will allow us to say that the cost to get from a vertex v to the same vertex v is 0.

2

3

8

3

3

10

5

5

4

2

**5** Pittsburgh

**6** Princeton

**1** Pensacola

**2** Peoria

**7** Pueblo

**3** Phoenix

**4** Pierre

**0** Pendleton

In the matrix we use 0 to represent zero cost, the weights as given above, and some impossibly large value to represent that no path exists. The file is called citymatrixweighted.txt

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0.** | **1.** | **2.** | **3.** | **4.** | **5.** | **6.** | **7.** |
| **0. Pendleton** | 0 | 9999 | 9999 | 9999 | 9999 | 9999 | 9999 | 8 |
| **1. Pensacola** | 9999 | 0 | 9999 | 5 | 9999 | 9999 | 9999 | 9999 |
| **2. Peoria** | 9999 | 9999 | 0 | 9999 | 9999 | 5 | 9999 | 3 |
| **3. Phoenix** | 9999 | 9999 | 9999 | 0 | 9999 | 10 | 9999 | 3 |
| **4. Pierre** | 2 | 9999 | 9999 | 9999 | 0 | 9999 | 9999 | 9999 |
| **5. Pittsburgh** | 9999 | 4 | 9999 | 10 | 9999 | 0 | 9999 | 9999 |
| **6. Princeton** | 9999 | 9999 | 9999 | 9999 | 9999 | 2 | 0 | 9999 |
| **7. Pueblo** | 8 | 9999 | 9999 | 9999 | 3 | 9999 | 9999 | 0 |

What is the lowest cost from Phoenix to Pittsburgh? \_\_\_

What is the lowest cost from Pittsburgh to Phoenix? \_\_\_

What is the lowest cost from Pueblo to Pendleton?\_\_

What is the lowest cost from Pendleton to Phoenix?\_\_\_

Floyd's Algorithm is very similar to Warshall's Algorithm (and some books call it the Floyd-Warshall Algorithm). While Warshall's Algorithm solved the *all pairs reachability problem*, Floyd's algorithm solves the *all pairs lowest-cost problem*. The disadvantage is that both algorithms run in O(V3), which may make them too large to run in practice.

**Assignment**

In your AdjMat implement the Floyd interface.

You will probably also have to fix the isEdge that was in the AdjacencyMatrix interface.

Write Floyd’s Algorithm. Call it allPairsWeighted

FloydDriver has been written for you. You will turn in the improved AdjMat.

**Sample Run**

Floyd's Algorithm! Enter file of names: cities  
Enter file of the matrix: citymatrixweighted

Adjacency Matrix  
 0 9999 9999 9999 9999 9999 9999 8   
 9999 0 9999 5 9999 9999 9999 9999   
 9999 9999 0 9999 9999 5 9999 3   
 9999 9999 9999 0 9999 10 9999 3   
 2 9999 9999 9999 0 9999 9999 9999   
 9999 4 9999 10 9999 0 9999 9999   
 9999 9999 9999 9999 9999 2 0 9999   
 8 9999 9999 9999 3 9999 9999 0

Number of Edges: 12

0-Pendleton  
1-Pensacola  
2-Peoria  
3-Phoenix  
4-Pierre  
5-Pittsburgh  
6-Princeton  
7-Pueblo

Cost Matrix  
 0 9999 9999 9999 11 9999 9999 8   
 13 0 9999 5 11 15 9999 8   
 8 9 0 14 6 5 9999 3   
 8 14 9999 0 6 10 9999 3   
 2 9999 9999 9999 0 9999 9999 10   
 17 4 9999 9 15 0 9999 12   
 19 6 9999 11 17 2 0 14   
 5 9999 9999 9999 3 9999 9999 0

Number of Edges: 33

What is the cost? Enter start city (-1 to exit): Pittsburgh  
 Enter end city: Phoenix  
9

What is the cost? Enter start city (-1 to exit): Pendleton  
 Enter end city: Phoenix  
9999

What is the cost? Enter start city (-1 to exit): -1