(CSCI 463 VA) Project Five: BFS and DFS Algorithms

The purpose for this project is to reinforce the students' knowledge on graph. The students will implement the Breadth First Search and Depth First Search for the graph.

The graph will be a direct weighted graph represented in adjunct matrix form. We assume that all weights are positive integers. The graph will be initialized from keyboard. When the students input the graph of n vertices, the students actually input an n by n matrix.

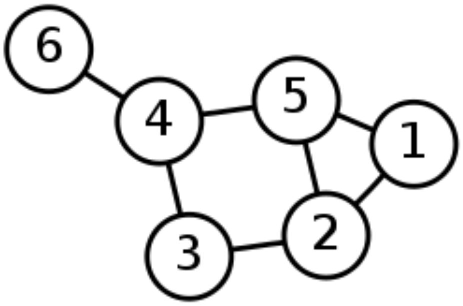
The matrix represents a simple direct weighted graph. There is no loop at vertex. No more than one edge from one vertex to another vertex. The vertices are numbered as 1, 2, ..., n. The graph with n vertices is represented by an (n+1) by (n+1) matrix with row 0 and column 0 unused.

The entry on row i and column j is the weight of edge from vertex i to vertex j. For instance, if there is an edge of weight 8 from vertex i to vertex j (i != j), then entry on row i column j of the matrix will be 8. We assume that all weights are positive integers. If there is no edge from vertex i to vertex j (i != j), then the user will **input 0** for the entry on row i column j of the matrix. However, the given code will automatically change that entry to be Integer.MAX_VALUE.

Two files are given. The one named CSCI463ProjFive.java cannot be modified. However, the students should understand it. The one name MyGraph needs to be implemented. In fact, there are only two methods in this file need to be implemented. The students shall read the description of each method and implement the method accordingly. No methods declaration can be changed. No other public interface can be added.

The students may run the executable jar file to see how the finished project should run. In this implementation, when there are more than one choice of the vertices, it choose the one marked with lower number first.

Example Run:
Suppose the graph is following



For this graph, the student can view it as directed weighted graph. Each edge has two directions. On each direction, the weight is 1. So the array input will be:

```
0 1 0 0 1 0
1 0 1 0 1 0
0 1 0 1 0 0
0 0 1 0 1 1
1 1 0 1 0 0
0 0 0 1 0 0
```

If we run the BFS algorithm with startVertex 6, then the output should be 6, 4, 3, 5, 2, 1,
If we run the DFS algorithm with startVertex 6, then the output should be 6, 4, 3, 2, 1, 5,

Due date will be announced on blackboard.