COMP 410 - Spring 2017
Programming Assignment 5

Due back by **8:00 am on April 17th**

For this assignment, you are to

1) Read in a directed graph from an input text file,
2) Represent this graph in adjacency list form, and
3) Use the topological sort algorithm to either flag the graph as containing a cycle, or obtain a list of its vertices in topologically sorted order.

The input file. Here is an example text-file representation of a graph:

```
5
Dallas  Houston Austin Amarillo Lubbock
Dallas Austin 100
Houston Lubbock 100
Austin Lubbock 200
Dallas Amarillo 125
Lubbock Dallas 10
Austin Amarillo 220
```

This lists a graph with five vertices named Dallas, Houston, Austin, Amarillo, and Lubbock, and 6 edges (e.g., there is an edge leading from Dallas to Austin, of cost 100).

Reading in the graph. You are to write a public class `ReadGraph` that supports the following public methods:

- `public static String[ ] readVertices(Scanner fileIn).`
  This method returns a sorted list of vertex names (and retains a copy for future use).
- `public static AdjListNode[ ] readEdgesAdjList(Scanner fileIn)`
  This method returns an adjacency list representation of the graph.

The Adjacency list representation. For each edge (i, j), there should be an entry in vertex i's adjacency list denoting that there is an edge (from i) to j, and the cost/ weight of the edge – for this assignment, you may assume that the cost is an integer. Here is the `AdjListNode` class you are to use:

```
public class AdjListNode {
   int v; //The vertex to which the edge leads
   int w; //The weight of the edge
   AdjListNode next;
   public AdjListNode(int i, int j, AdjListNode n){//Constructor
   v = i; w = j;next = n;
   }
}
```

The "main" class. Here is what your `main` class looks like:

```
public class Main {
    static String[] vertices;       // The sorted list of vertex names
    static AdjListNode[] adjList;   // The adjacency list

    /*
     * You are to write the following method. You will need to use a stack
     * or queue of integers here (you may use java.util.{Stack,Queue}
     */
    public static String[] topSort() { ... }

    public static void main(String[] args) {
        if (args.length != 1) {
            System.err.println("Incorrect number of args passed");
            System.exit(-1);
        }

        Scanner fileIn = new Scanner(new File(args[0]));
        vertices = ReadGraph.readVertices(fileIn);
        adjList = ReadGraph.readEdgesAdjList(fileIn);
        String[] sorted = topSort();

        /*
         * At this point, "sorted" contains the vertices in topologically-sorted
         * order (or all NULLs if the graph is not acyclic
         */
    }
}
```

Submission instructions. You should upload the following three files in a `.zip`file to Sakai

1) A file titled `ReadGraph.java` containing the two required public methods (you may have additional private methods and variables);
2) A file titled `Main.java`, completing the Main class shown above; and
3) A pdf file indicating the status of your project – whether it all works (and if not, which parts do and how you have tested these parts, and what the problem appears to be with the rest of the parts).

Note. For this assignment you may use the Java methods `Arrays.sort` and `Arrays.binarySearch`. You may NOT use any other Java library methods or classes EXCEPT Arrays.sort(), Arrays.binarySearch(), java.util.Stack, and java.util.Queue.