COMP202 SPRING 2016

Programming Project 5

Graph Implementation

Submission Location: **F Drive**

Due Date: **Thursday 5 May 2016 23:59**

This project has two phases:

First phase: will be done **individually**

Second phase: Will be a group project. If your group mate is from the other school, (Which we will announce how to collaborate, during upcoming week) you will get 20% bonus.

Second phase will be announced after submission of first one.

Your project submissions will be checked for plagiarism.

First phase

Assignment:

For this project, you will implement the undirected graph based on the adjacency list and adjacency matrix (Figure1). According to the following API, you will implement the required methods for graph data structure. (Note that you are not allowed to use any java built in function/class related to graph).

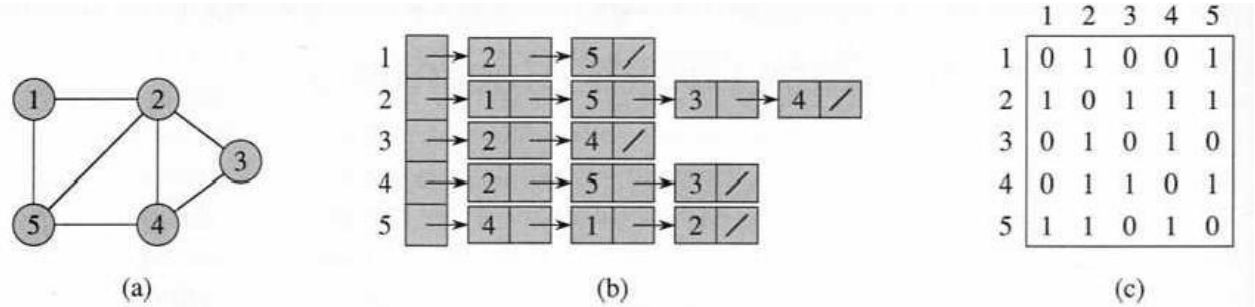

Figure 1 Two representations of an undirected graph. **(a)** An undirected graph G having five vertices and seven edges. **(b)** An adjacency-list representation of G. **(c)** The adjacency-matrix representation of G.

Your task for this project is implementing graph representation according to the adjacency matrix and adjacency list according to following API.

********** **Graph API** **********

```
Graph(int V)
```
// Create an empty graph with V vertices

```
getNumVertices()
```
//Return number of Vertices

```
getNumEdges()
```
//Return number of Edges

```
addVertex()
```
// Add New vertices to the graph

```
addEdge(int v , int w)
```
// Add new edge between node v and w

```
removeEdge(int v , int w)
```
// Remove new edge between node v and w

```
getNeighbors(int v)
```
// Return the List of the vertices who are adjacent to the V

```
List<Integer> degreeSequence()
```
// list of the degrees of the vertices in the graph

The degree sequence of a graph will be sorted (organized in numerical order from largest to smallest, possibly with repetitions) list of the degrees of the vertices in the graph. The degree sequence can tell us a lot about the underlying structure of the graph: Are there hubs, vertices that are adjacent to many others? Are there isolated vertices, vertices that have no neighbors? Is the graph (almost) regular, with all vertices roughly adjacent to the same number of other vertices?

```
verticesWithinDistance2(int v)
```

// Return the list of the vertices which are 2 distance away from vertex V

```
showAsMatrix()
```

// print on console the graph as a matrix.

```
showAsAdjacencyList()
```

// print on console the graph as an array of lists.

Hint:

- First, try to implement graph according to the adjacency matrix.
- Assume that vertices in graphs do not have self-loops.
- For the first phase, we don't provide you input and output for testing, but it will be tested comprehensively via JUnit so ensure that your methods work correctly.
- For list implementation of the verticesWithinDistance2() to find the 2 distance away of node v, first find the neighbors of v, and then expand them out for the two-hop neighbors.
- For list Matrix implementation of the verticesWithinDistance2() you can use matrix multiplication.

Bounce part:

You will receive 10% for implementing following method.

```
removeVertex()
```

// remove vertex and corresponding edges from the graph.

Submission:

1. Does the program compile? (Programs that don't compile will not be graded)
2. Does the program meet all required interfaces?
3. Are comments well organized and concise?
4. If you have reviewed your code and all the criteria on the checklist are acceptable, follow the submission procedure.

Please **do not zip your files**.

Upload it through Novel Login to the

F drive/COURSES/UGRADS/COMP202/PROJECT/YOUR_NAME

Please consider the following format for your projects submission:

Project5 "This should be the only name for your folder"