R·I·T

Rochester Institute of Technology
Golisano College of Computing and Information Sciences
School of Information

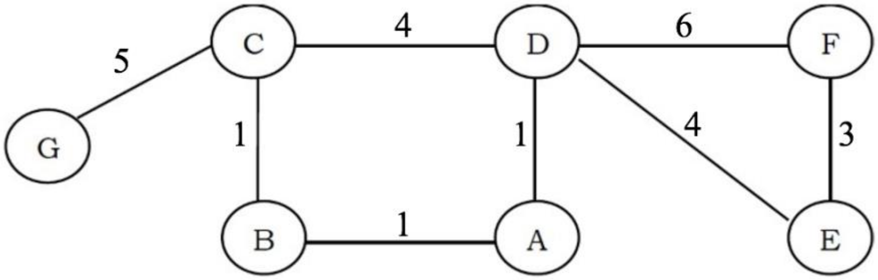
ISTE-222 Assignment 4: Graph Algorithms

Get the Path!

Problem Statement:

The purpose of this assignment is to become deeply familiar with Dijkstra's algorithm and extend the implementation that was provided to you in Dijkstra.java so that it produces more useful output and works in conjunction with a more space efficient graph representation than it does currently.

You will be using Dijkstra's algorithm to find the shortest path from Node G to all other nodes in the graph below.



In addition to finding the cost of the shortest paths, you will also be outputting the entire path for each shortest path.

Requirements/Specifications:

For this assignment, you will be using Dijkstra.java from myCourses as a starting point for your code. Rather than using an adjacency matrix, as the code does currently, you will create an outer class in the same file called WeightedGraph, which will represent an **adjacency list**. Feel free to modify the UnweightedGraph class from GraphTraversal.java to do this.

The dijkstra method in Dijkstra.java currently returns an array indicating the shortest distance (cost) from a source node to all other nodes in the graph. Modify this method so that it returns a 2D array containing two rows. The first row will still contain the shortest distance from a source node to all other nodes in the graph and the second row will contain the "previous node" data, as shown in the Graphs and Graph Algorithms lecture. The previous node data indicates which node we learned about this shortest path from.

Along these same lines, you will modify the printDistances method so that it also outputs a Previous Node column, in addition to the Node and Distance columns that the method already outputs.

Finally, write a method called printPaths that takes in the 2D array returned by the dijkstra method, the array of node names, and the source node and prints out the full paths from the source node to all other nodes. The output should look like this.

```
Shortest path from Node G to Node A is: G, C, B, A
Shortest path from Node G to Node B is: G, C, B
….
Shortest path from Node G to Node F is: …
```

Your code should create the following output using method calls from the main method.

1. Call the toString method of WeightedGraph so that it prints out the adjacency list for the graph after it has been created.
2. After running Dijkstra's algorithm with Node G as the source node, call the printDistances and printPaths methods.

Your code and a PDF document containing your output is due in the Assignment 4 dropbox by the due date specified on myCourses.

ISTE-222 Assignment 4: Graph Algorithms

Item	**Possible points**	**Earned points**
Adjacency List (25%)		
Adjacency list contains all edges and their weights	20	
The toString method neatly outputs all edges and their weights	5	
Dijkstra's Algorithm (40%)		
Correctly determines previous node information	25	
Correctly computes shortest distance from source to all other nodes	10	
Returns 2D array containing distances and previous node information	5	
Output Methods (35%)		
printDistances method also outputs Previous Node column	5	
printPaths method correctly outputs all shortest paths from the source	30	
Total:	100	

Comments: