CSCI 2275 – Data Structures and Algorithms
Instructor: Hoenigman
Assignment 8
Due Sunday, Nov 7, by 11 am

People in Graphville

You are the mayor of Graphville. You have a list of all the people in your town. You also know the relation between them, basically if they know each other. You need to increase the community interaction between your constituents. Now before you do that you need to know a lot of other details like if two people know each other directly or through a friend or in any other way, also how many groups are there in your town.

What your program needs to do:

Build a graph. There is a file on canvas called people.txt that contains the names of people and with them the names of people they know similar to an adjacency list. When the user starts the program, read in the file and build a graph where every person is a vertex, and there is an edge for every person they know.

For this assignment, you are building the graph and implementing functionalities given in the menu.

Use a command-line argument to handle the filename.

Display a menu. Once the graph is built, your program should display a menu with the following options:

1. **Print list of people and their acquaintances**
2. **Print if people know each other**
3. **Print groups**
4. **Find the least number of introductions required**
5. **Quit**

Menu Items and their functionality:

1. **Print list of people.** If the user selects this option, the vertices and adjacent vertices should be displayed.

2. **Print if people know each other.** This option takes two vertices as user inputs and displays True is the vertices are adjacent, and False if they are not adjacent.

   If this is a part of the file:
   Akhil-Paramvir,Aurangzeb

Aurangzeb-Akhil,Kieran
Earl-Paramvir,Kieran,Zack

Then Earl and Zack know each other
Kieran and Aurangzeb know each other
and so on.

3. **Print groups.** If the user selects this option, you need to do a depth-first search (DFS) of the graph to determine the connected people in the graph, and assign those cities the same group ID. The connected people are the vertices that are connected, either directly by an edge, or indirectly through another vertex. For example, in the above example, Akhil is connected to Kieran via Aurangzeb. When assigning group Ids, start at the first vertex and find all vertices connected to that vertex. This is group 1. Next, find the first vertex that is not assigned to group 1. This vertex is the first member of group 2, and you can repeat the depth-first search to find all vertices connected to this vertex. Repeat this process until all vertices have been assigned to a group.

4. **Find the least number of introductions required-** If the user selects this option, they should be prompted for the names of two names. Your code should first determine if they are in the same group. If the people are in different groups, print "No way to introduce them". If the people are in the same group, run a breadth-first search (BFS) that returns the number of vertices to traverse along the shortest path, and the names of the vertices along the path. For example,

For Akhil and Paramvir no one else is required as they are directly connected. But Kieran and Akhil can be introduced via Aurangzeb.
So your output should be something like :
1, Aurangzeb

If there were N people in between then output:
N, Name1-Name2-.....-NameN

Submit it on Canvas - For any questions feel free to email the TA or Professor.