CSCI 2275 – Programming and Data Structures
Instructor: Hoenigman
Assignment 8
Due: Monday, Nov 9, by 6pm

Graphs – simulating connections in a network

You are the mayor of Graphville trying to monitor the spread of an illness that has recently come to your town. You have a list of all the people in your town and their recent contacts and the number of minutes that they spent with each individual. In this program, you are provided with a graph of these individuals that stores a set of vertices and the connected edges as an adjacency list. Your task is to use the graph to simulate the spread of illness in your town.

What your program needs to do:

Build a graph. There are two files on Canvas, one is called people2.txt and the other is called namesOutput.txt. Each file contains the names of people, their contacts, and the minutes they have spent with the person. When your program starts, the file is read in and the graph is built where every person is a vertex, and there is an edge for every person they know with an edge weight that represents the minutes of contact they have had. This part of the program has been done for you.

Use a command-line argument to handle the filename.

Display a menu. Once the graph is built, the program displays a menu with the following options:

1. **Print vertices**
2. **Find groups**
3. **Introduce illness**
4. **Illness spread**
5. **Quit**

Menu Items and their functionality:

1. **Print vertices.** If the user selects this option, the vertices and adjacent vertices should be displayed with the edge weight for each vertex. This is pretty much done for you in displayEdges().
2. **Find 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 people the same group ID. ID is a property in the vertex struct. The connected people are the vertices that are connected, either directly by an edge, or indirectly through another vertex.

   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 DFS search to find all vertices connected to this vertex. Repeat this process until all vertices have been assigned to a group.

3. **Introduce illness -** If the user selects this option, your code should select a person randomly and change their infected state to true. Find all of their adjacent vertices with whom they have had close contact. We'll define close contact as more than 15 minutes spent together. Print the name of the infected person, all of their contacts, and how long they spent together. Your print statements should look something like:

   ```
   Infected: <person>
   Close contacts:
   <person>,<minutes>
   <person>,<minutes>
   …
   ```

6. **Infection spread** – If the user selects this option, they should be prompted to enter an infection rate. Select a person randomly and change their infected state to true. Next, identify all of the individuals with whom the infected individual has had close contact, and use the infection rate to change their infected state to true. For example, if the infection rate is .20, then generate a random number between 0 and 1. If the number is less than .20, the person becomes infected. If it's .20 or greater, the person will not be infected. Once an individual is infected, add them to the queue and repeat the process for their close contacts. The infection rate should be cut in half with each step away from the initial infection, i.e. all individuals with the same parent vertex should have the same infection rate. Repeat until the queue is empty.

   Print the names of all infected individuals and the name of the person who infected them. Your print statement should look something like:

   ```
   Infected: <person> after spending <X> minutes with <person>.
   ```

What to submit:

Submit the Assignment8.cpp, Graph.cpp, and Graph.h files as a zip file to the Assignment8 link on Canvas.