Quarter 3 Project - Graphs (Map)

Advanced Computer Science

Topics: Objects, Classes, Java Components, Map, Set, Generics, Graphs, and Shortest Path (Dijstra's)

Program Description

You will create a map program of a country where the user can get directions from one location to another using the Shortest Path (Dijkstra's) Algorithm. You are to use all of your own **data structure**. I should not see any imports of data structures.

Program Features

1. Select a map from a country of your choice. This country cannot be the same as another student. The map will be displayed through a background image or drawing. The map background should show borders. For example, if you have a country, you should have an outline of its surrounding borders. To get a blank map without labels, you can use the following website (link) (no labels) and link (no labels and roads).

2. You will have at least **20 locations** in your country. Each location will be a point on the map with roads connecting them. **The connections of the locations should not be circular. There should be at least 5 locations with more than 2 connections.** The **location name** and a **3-letter abbreviation** are displayed in this format. name (abv). For example: New York City (nyc) **Roads (e.g. lines) connecting the locations and their distance (a number in miles or km) will be displayed on the map.**

   Optionally: You can show the 3 letter abbreviation only on the map, and have a legend on the side showing the full name of each abbreviation.

3. You will have the option where the user can input the 3-letter abbreviation from one location to another, and the map will display the path of travel utilizing **Djikstra's Shortest Path Algorithm**. The path of travel will be displayed in the following ways.
   a. Display **directions** from start to end including the miles between each location. For example: Traveling from San Francisco to Los Angeles, you go through San Jose and Bakersfield respectively. Your directions should be given in the following format.

      Take San Francisco to San Jose - 50 miles.
      Then take San Jose to Bakersfield - 240 miles.
      Then take Bakersfield to Los Angeles - 110 miles.

   b. Then display the total distance. Given the example above, you will display
      The total distance is 400 miles.

c. Highlight or color the path of travel. The lines connecting the start location to the end location will be a different color or highlighted, so you can clearly see the path of travel.

Graph Class
Your graph data structure will be an adjacency list that will utilize your own MyHashMap, MyHashSet, DLList, etc.. The graph's node (data) will be of the **Location class** that is compatible with a Hash.

Location Class
This class will contain information about a location. At a minimum, it should have the name and 3-letter abbreviations. The hashCode and equals methods will be based on the 3-letter abbreviations. **Do not use (String.hashCode()).** Create your own algorithm for generating a unique number based on the 3 letter abbreviations. Make sure your MyHashMap is sized appropriately. (Optional) You can add more information to this class such as its location on the map.

Screen Class
This class contains your graphical components. It will contain the Graph Data Structure. The maximum screen size is 1920x1080.

Add more classes and methods as needed.

Challenge: Pick One
1. Have a picture for each location on your map. The picture can be right by the location, or when you click on it, it will appear. (If you do the click option, make sure to clearly state it.)

2. Have an animation of a car that travels the path that is given in the directions provided.

3. Have names for each connecting road, and display them on the map next to the road. Your directions would need to include the corresponding road name. For example, the name of the connecting road is Highway 101 between San Francisco to San Jose. Thus, you would say…

Take San Franciso to San Jose <u>via Highway 101</u> - 50 miles

Notes:

Strategy on creating a hash code from 2 letters.

- To convert each char to a number, you can cast it.

```
char myChar = 'a';
int value = (int) myChar; //'a' returns 97, for 'a' to be 0, subtract 97.
a = 0, b = 1, c = 2, etc…
```

- Similar to how we do hex and binary calculations, we can apply concepts for our letters.

```
aa = 0        ab = 1        ac = 2
ba = 26       bb = 27       bc = 28       zz = 675
aa = a(26^1) + a(26^0) = 0(26) + 0(1) = 0
ab = a(26^1) + b(26^0) = 0(26) + 1(1) = 1
ac = a(26^1) + c(26^0) = 0(26) + 2(1) = 2
ba = b(26^1) + a(26^0) = 1(26) + 0(1) = 26
bb = b(26^1) + b(26^0) = 1(26) + 1(1) = 27
bc = b(26^1) + c(26^0) = 1(26) + 2(1) = 28
zz = z(26^1) + z(26^0) = 25(26^1) + 25(26^0) = 675
```

Any late submission or revision will be given a maximum score of 90.

Grading Rubric:
Your project will be graded using the following criteria.

	Weight (of 100)	**4 (100%) Mastery**	**3 (85%) Proficient**	**2 (75%) Emerging**	**1 (60%) Minimal**	**0 (50%) No Effort**
Graphical User Interface (GUI)	10	The program is user-friendly. Buttons, text fields, and/or other components are clearly labeled. Clear instructions are given.	The program is easy to use. One to two functions need labeling or further instructions.	The Progam is not so easy to use but is functional. Two or more functions need labeling or further instructions.	The program is difficult to use. Labels and instructions are unclear.	No attempt.
Data Structure and Code Design	30	The data structure meets all the requirements described in the project description. The code is well organized and has proper indentation.	The data structure meets most of the requirements described in the project description. The code is well organized and has proper indentation.	The data structure lacks some of the requirements described in the project. The code is mostly organized and properly indentated.	The data structure has many missing requirements in its design. The code is poorly organized.	No attempt.
Program Features	50	All required features including the challenge as described in the project description are fully functional.	One feature or challenge is missing or does not work properly.	Two features are lacking or do not work properly.	Three or more features are lacking or do not work properly.	No attempt or nonfunctional code.
Professionalism	10	The student was focused and acted professionally throughout the project process. The milestone and final submission deadline were met. The presentation was professional, academic language was used, and all questions were adequately answered.	Same as level 4 but with one criterion not met.	Same as level 4 but with two criteria not met.	Same as level 4 but with 3 or more criteria not met.	No attempt.

Milestone 1 ~~**Due 3/10**~~ **Due 3/11** - Demo in the class the following. A map with 20 locations. Display locations with names and abbreviations. Display roads connecting your locations. Class participation.

Presentation - You will do a short 2-3 min presentation of your project. You will be asked to explain your code both technically and functionally.

Date Date: See Canvas

Submission: Submit in Canvas.

Test that your program works as follows in the command prompt.

```
javac *.java
java Runner
```

If you are on a local machine:
Store all your files in a folder, and create a zip file (link on how to do this). Rename your zip file to be your **lastname_firstname_labname.zip.** Submit the zip file.

Those that are on Eclipse, Visual Studio, etc.., make sure you can compile with the command line. You will need to move your files out to a folder and compile from there. I can show you how during office hours.