CS 245 - Assignment 3 Fall, 2023

Assignment 2 - International Road Trip (IRoadTrip)

This assignment tests your ability to use Graph and other data structures and the algorithms for handling real world file, web and user inputs.

Background

Béatrice Frison once took a bicycle from Grenoble to Kathmandu. (She talks about the trip briefly, in French, on YouTube.) Fortunately, she planned well and did not encounter too many problems. You will create an implementation of a shortest path trip between two countries for her or someone else who chooses another such trip.

The starter code and data files for this assignment is at https://classroom.github.com/a/KK49Ito0.

Requirements

There are several requirements for this assignment. These requirements are interdependent, meaning that a proper implementation requires that all the requirements be implemented and used in a cooperative system fusing each of the individual requirements.

Requirement 1: Navigate to https://classroom.github.com/a/KK49Ito0 and accept the assignment. The repository contains three data files (`borders.txt`, `capdist.csv` and `state_name.tsv`) and one skeleton java source file (`IRoadTrip.java`).

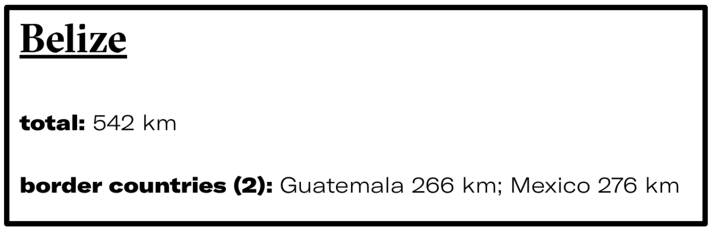
Requirement 2: Read and associate the data files.

There are three data files. Each file contains data you will need, but — because of the sources — each file is in a different format and has different data. The sources for the files are below but appear for reference only. (There has been considerable work to cleanse the data in the files to make your task easier.) The files are as follows:

a) `borders.txt`: This file contains data about country borders from the CIA World Factbook, specifically the “Land boundaries” page — https://www.cia.gov/the-world-factbook/field/land-boundaries/. Each line in the file contains the name of a country, followed by a “=” character and zero, one or more countries and border lengths, with each country separated by a “;” caracter. For example, one of the entries in this file is:

```
Belize = Guatemala 266 km; Mexico 276 km
```

This corresponds to the CIA World Factbook entry shown in Figure 1, indicating that the country of Belize shares land borders with Guatemala and Mexico. The numbers associated with each country (266 for Guatemala) represent the length of the border between these countries.



Belize

total: 542 km

border countries (2): Guatemala 266 km; Mexico 276 km

Figure 1: CIA World Factbook entry for Belize

Some countries have alias names. For example, the country Turkey appears in the CIA World Factbook (and in the borders.txt file) as both as "Turkey" and as "Turkey (Turkiye)". Your implementation must handle both names.

Countries which share no borders with other countries have no entries following the "=" character. The entry for Trinidad and Tobago is an example:

```
Trinidad and Tobago =
```

b) `capdist.csv`: The file capdist.csv comes from Kristian Skrede Gleditsch's site. This file shows the distances between capital cities of many pairs of countries. This file is in CSV (comma-separated value) format, with fields detailed in Table 1. The presence of distances between capital cities does not indicate that the countries are adjacent to each other.

Field #	Field Name	Meaning
1	numa	Unique ID for country A
2	ida	Country code for country A
3	numb	Unique ID for country B
4	idb	Country code for country B
5	kmdist	Distance between capitals of country A and country B in km
6	midist	Distance between capitals of country A and country B in miles

Table 1: Details of capdist.csv file

One of the entries in this file is as follows:

```
451,SIE,580,MAG,7292,4546
```

… indicating that one country ID = SIE (Sierra Leone, with number = 451) is approximately 7292 km (4546 miles) from another country ID = MAG (Madagascar, with number = 580). You may notice that the existence of an entry in this file does not indicate that a path between countries. For example, there is no land path between Sierra Leone and Madagascar. Your implementation must use the distance kmdist (distance in kilometers) field for approximate distances between countries.

c) `state_name.tsv`: This file contains the fields listed in Table 2. This allows you to associate names from the CIA World Factbook file (`borders.txt`) to the distance file (`capdist.csv`). This file may contain multiple entries for the same state / country. For example, one entry in this file is as follows:

```
41    HAI   Haiti 1816-01-01  1915-07-04
41    HAI   Haiti 1934-08-15  2020-12-31
```

… which together indicates that the country of Haiti came into existence twice — once in 1816 and again in 1934[1]. Your implementation must take information corresponding to 2020-12-31, the most recent data contained in this file.

Field #	**Field Name**	**Meaning**
1	statenum	Unique ID for a country
2	stateid	The encoded name for a country (eg. JAM)
3	countryname	The decoded name for a countr (eg. Jamaica)
4	start	The date from which the country was created
5	end	The date after which the country no longer existed

Table 2: Details of capdist.csv file

Requirement 3: Implement the required classes and functions.

Your repository contains the skeleton of one source file, `IRoadTrip.java`, with a number of functions which only return default values. You may use additional classes and files if you choose, but you are required to complete the functions listed below. You may assume that the IRoadTrip class will be called through the main function as follows:

```
java IRoadTrip borders.txt capdist.csv state_name.tsv
```

Input	**Output**	**Explanation**
`a3.getDistance("USF", "My House")`	`-1`	“USF” & “My House” are not countries
`a3.getDistance("France", "Spain")`	`1012`	Paris is ~ 1012 km from Madrid
`a3.getDistance("Canada", "Panama")`	`-1`	Countries do not share borders

Table 3: Sample inputs and outputs to getDistance function

`public IRoadTrip(String [] args)`: This function is the constructor for the IRoadTrip class. The args parameter contains the names of the files as provided to main — i.e. an array (in order): `borders.txt capdist.csv state_name.tsv`. The constructor must read the files and prepare to execute the implementation to respond to requests. The implementation must halt on any failure here.

[1] The United States occupied Haiti from 1915-1934. See wikipedia for details.

`public int getDistance (String country1, String country2)`: This function provides the shortest path distance between the capitals of the two countries passed as arguments. If either of the countries does not exist or if the countries do not share a land border, this function must return a value of `-1`. Examples are as found in Table 3.

`public List<String> findPath (String country1, String country2)`: This function determines and returns the shortest path between the two countries passed as arguments (starting in the capital of country1, ending in the capital of country1, and going through the capitals of each country along the way). This path must start in country1 and end in country 2. If either of the countries does not exist or if there is no path between the countries, the function returns an empty List[2]. Each element of the list must be a String representing one step in a longer path in the format: *starting_country --> ending_country (DISTANCE_IN_KM.)*, eg:

```
Thailand --> Burma (573 km.)
```

`public void acceptUserInput():` This function allows a user to interact with your implementation on the console. Through this function, your implementation is required to receive and validate the names of two countries from a user. The country names must be validated — i.e. your implementation must not accept invalid names. Once two valid country names have been entered by the user, the implementation must print the path between those countries if such a path exists. A sample user interaction is shown below, with user input in bold italics:

```
Enter the name of the first country (type EXIT to quit): CS245
Invalid country name. Please enter a valid country name.
Enter the name of the first country (type EXIT to quit): Yemen
Enter the name of the second country (type EXIT to quit): Jordan
Route from Yemen to Jordan:
* Yemen --> Saudi Arabia (1040 km.)
* Saudi Arabia --> Jordan (1323 km.)
Enter the name of the first country (type EXIT to quit): Paraguay
Enter the name of the second country (type EXIT to quit): Colombia
Route from Paraguay to Colombia:
* Paraguay --> Bolivia (1480 km.)
* Bolivia --> Peru (1069 km.)
* Peru --> Colombia (1880 km.)
Enter the name of the first country (type EXIT to quit): Gabon
Enter the name of the second country (type EXIT to quit): France
Route from Gabon to France:
* Gabon --> Cameroon (405 km.)
* Cameroon --> Nigeria (963 km.)
* Nigeria --> Niger (788 km.)
* Niger --> Algeria (2561 km.)
* Algeria --> Morocco (958 km.)
* Morocco --> Spain (822 km.)
* Spain --> France (1012 km.)
Enter the name of the first country (type EXIT to quit): EXIT
```

[2] An empty List is a List instance of size 0. (Your implementation must not return a null List.)

Submission

You are required to submit your complete implementation for this assignment:

1) The completed IRoadTrip class, along with other Java classes necessary to complete all the requirements above.
2) Optionally, you may submit a PDF or markdown (eg. README.md) document explaining any of the following:
    a) Design choices and OO decomposition of the problem
    b) Data structure choices
    c) Anything else you believe might be useful

On Canvas, submit the link to your github repository containing these items. Any files not found in your repository will not be accepted for grading.

Grading

Grades will be determined as follows:

- 60% = Implementation
- 20% = Design, including use of data structures to create this simulation
- 15% = Efficiency of code and algorithm
- 5% = Style, including consistent indentation, variable naming, etc.

Submit your assignment on or before the due date. The syllabus contains information on policies for late assignments.