Rose Tabares Design CS 245 Assignment 2

---

leastDistanceRoute.java

Inputs:

- Starting city
- Ending city
- List of events or places of interest

Functions:

- public static void main(String[] args)
    - Take in starting, ending city and attraction
    - Store attractions.csv and roads.csv in file variables
    - Create List<String> route and store return value from route(starting_city, ending_city, attraction) function
    - Print List<String> route

- public List<String> route(String starting_city, String ending_city, List<String> attractions)
    - Creates empty list of locations of size of *List<String> attractions* + 2 (for starting and ending city) called *List<String> locationsList*
    - Add *starting_city* in position 0 and *ending_city* in position -1 of *locationsList*
    - attractionsLocations(*attraction*s, *locationsList*)
    - Use Dijkstra's algorithm to calculate shortest distances and add them to *List<String> route* = Dijkstra(*locationsList*)
    - return *List<String> route* representing the route the user should take, ex
        - Grand Canyon AZ, Bemidji MN, New York NY, etc

- public void attractionsLocations(*List<String> attraction*s, *List<String> locationsList*)
    - Loops through *List<String> attractions* and finds the location of that attraction by reading *attractions.csv* and add them to *locationsList*

- Public float calcDistance(String location1, String location2)
    - use *roads.csv* to calculate distances between locations
    - Loop over roads.csv until find row that contains both locations and save the 3rd column to get the miles in between them
    - return number of miles as distance

- Public List<String> Dijkstra(*List<String> locationsList*)
    - Vertex = locationsList
    - Known = start with all as false

- Path = start with all as -1, then leave starting_city or locationsList[0] as -1 path because it will be starting point and all others will go from there, and set this with Known=true and Cost=0
- Cost = start with all cost=-1, then use calcDistance function to get costs:
  float cost = calcDistance(location1, location2)
- Implement Dijkstra's Algorithm until all Known are true and go backwards with Path and Vertex to create route and set to List<String> routeDijkstra
- return routeDijkstra