2XB3 Assignment 2

Department of Computing and Software McMaster University
Instructor: Dr. Reza Samavi

This is an individual assignment. All assignments deemed to be substantially similar to each other will be treated as an academic offence in addition to getting 0 credit.

Posted: March 1st, 2020

Due on: **March 27*, 2020* at 23:59** (strict deadline)

1. Description

You have been hired by a tour planner agency to plan cost effective trips. In this assignment you plan the trip from **Boston to Minneapolis** and you are looking to minimize the cost of the trip and your goal is toimplement algorithms that help determine an optimal route.

2. Setup

In this assignment, there are multiple datasets provided that you should incorporate into your program: "menu.xlsx" which lists menu items of 3 restaurants; "burgerking.csv", "mcdonalds.csv", and "wendys.csv" which list locations of restaurants by longitude and latitude; "connectedCities.csv" which lists unidirectional routes between cities; and "USCities.csv" which lists the latitudes and longitudes of all US cities that are find in connectedCities.

You have to find a path between these two cities first using the routes provided in connectedCities.csv (3.2). Then, you must take into account expenses including a meal at each city (3.4).

The route must be picked following these requirements:

- You must stop at each city along the route and at each stop, you visit one restaurant.
- You do not need to visit a restaurant at your starting city, but you do need to go to a restaurant at your destination city.
- A restaurant is considered to be within a city if it's located within 0.5 degrees of latitude and 0.5 degrees of longitude of the city. If there are multiple restaurants within a city, any of them can be selected.
- You cannot have the same meal in consecutive cities (e.g. You choose to stop by McDonalds to have a Big Mac meal in the first city, you can still stop at a McDonalds in the second city, but you cannot choose a Big Mac meal there. However, at your third city, you can once again have a Big Mac meal).

3. Problems

3.1 Analysis

Identify what type of graph to use to solve this problem. What do the nodes of the graph represent? What do edges represent? Describe how you can use the graph and the algorithms below to solve the given problem.

3.2 Breadth-First Search vs. Depth-First Search

Implement a breadth-first search algorithm and a depth-first search algorithm to find a path from **Boston to Minneapolis** based upon the provided unidirectional routes between cities. The output specification is described below.

Analyze which algorithm found the result in fewer steps? Will this remain true if the number of stops increases? Why or why not?

3.3 JUnit (BFS vs DFS)

Implement a JUnit class to test that all possible routes are examined and the correctness of the routes, i.e. a route includes only cities that are connected and in the correct order.

3.4 Shortest Path

Find the lowest cost route from **Boston to Minneapolis** based upon the given constraints (food expense). Use a shortest path algorithm described in Chapter 4.4 of your textbook to solve this problem. The output specification is described below.

What is the complexity of this problem? Justify your answer. Will the complexity remain the same if the number of cities increases? What if the number of stops is increased? The number of restaurants? Why or why not?

4. Output Specifications

You should have 2 different output files named “a2_out.txt” and “a2_answers.txt”.

The first two lines of “a2_out.txt” should contain the resulting path for BFS and DFS in the following format:

BFS: CityA, CityB, CityC, …

DFS: CityA, CityB, CityC, …

This should be followed by a table which contains your result for 3.4. The table headings are City, Meal Choice, Cost of Meal. Each row in the table should contain a city and the corresponding information.

The “a2_answers.txt” should contain the written answers to 3.1, 3.2, and 3.4.

5. Assignment 2 Marking

Assignment 2 is worth 15% of the course marks. Your grade for this assignment will be determined based on the following rules:

- A submitted solution that does not compile or run gets 0 credit.
- A solution that runs but is partially correct gets partial credit (depending on the progress towards a full solution).

- Providing adequate, concise, and meaningful comments throughout your code is part of the solution grade (i.e., a piece of code that correctly solves a problem without (or with inadequate) comments will score less than a well-commented piece of code that does the same).
- Your implementation should not only be correct but also concise and efficient. Quality aspects of your implementation and programming style particularly preservation of encapsulation and modular programming will be evaluated (refer to pages 96-108 of your textbook).
- Not following the assignment instructions properly for the requested formatting will cost you marks. You may even get 0 credit if the improper formatting will prevent your program from running.
- Every hour after an assignment deadline 2% will be deducted from the assignment mark. After 48 hours the assignment will no longer be accepted and the student will get 0 credit.
- This assignment is individual work. The work you submit must be your own. Both copying assignments and allowing others to copy your assignment are strictly forbidden and will be treated as an academic offence. All assignments deemed to be substantially similar to each other will get 0 credit.
- If you include libraries from any sources other than your own or from the course material (course lecture notes and lab notes/instructions) you must acknowledge them and explicitly give proper credit with meaningful comments inside your code (when using methods from the external libraries). Properly cited external codes can only be included as Java libraries, i.e. you are not allowed to copy full or partial codes from other resources and include them inside your code. The included libraries should not be a substantial part of your assignment. Your work will be checked for plagiarism to account for this.

6. Assignment 2 Submission

Your submission must be an Eclipse project of your A2 implementation. The name of your Eclipse project should be cas2xb3_A2_*lastname_initials*. You should include a txt file named cas2xb3_A2_*lastname_initials*.txt resided in the "data" folder containing the following information (each item in a separate line):

- The course code (COMP SCI 2XB3 or SFWR ENG 2XB3)
- Your full name
- Your student number
- A dated statement that attests to "the fact that the work being submitted by you is your individual work."
- Any design decisions you feel need explanation or attention by the marker (extra methods etc.).

In order to submit your Eclipse project to the relevant lab assignment dropbox in Avenue, you first need to save everything and then export your project.

1. In Eclipse, right-click on the name of the project, select Export->General->Archive File.
2. Ensure that just your project has a check-mark beside it, and select a path and filename to export the project to. Ensure that your export project has a file extension of '.zip'. The name of the zip file should be 2xb3_A2_*lastname_initials*.zip.
   IMPORTANT: You MUST export the FULL Eclipse project. Submitting individual files (e.g. java/class files) will NOT be counted towards your submission. Click 'Finish' to export.

3. Verify the zip file by opening it and ensuring that it has the same folder structure as in Eclipse (it may have some extra files or folder such as 'bin', which is okay).

4. Go to Avenue and upload your zipped project to ' Assignment 2 Submission’ Dropbox.

IMPORTANT: YOU CAN SUBMIT MULTIPLE TIMES, HOWEVER ONLY THE LAST SUBMITTED FILE WILL BE CONSIDERED FOR MARKING AND ANY PREVIOUS SUBMISSION WILL BE AUTOMATICALLY REMOVED FROM THE COURSE WEBSITE - IT IS YOUR RESPONSIBILITY TO CHECK YOUR ZIP FILE BEFORE SUBMITTING.