2XB3_2016 Assignment 3

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

March 8, 2016

This is an individual assignment. All assignments deemed to be substantially similar to each other will get 0 credit.

1. Description

The university is closed for the reading week. After completing piles of assignments and several important exams, students are eager to go on a road trip across the United States. Since many students are concerned about money, you decide to write an application that will help minimize the cost of the trip. You have to factor in the fuel expense of the trip from the departure city to the destination city as well as food cost for each stop along the way.

2. Setup

In this assignment, there are multiple datasets provided that you will need to incorporate into your program: "menu.csv" which lists menu items of 3 restaurants; "burgerking.csv", "mcdonalds.csv", and "wendys.csv" which list locations of restaurants by longitude and latitude; "connectedCities.txt" which lists bi-directional routes between cities and which state they belong to; "USCities.csv" which lists the latitudes and longitudes of all US cities; and "StateGasPrice.csv" which contains the fuel price of each state.

You are to create a Java application (with required analysis) that does the following (See Section 3):

- Read two cities from an input file.
- Find a path between these two cities.
- Calculate the cheapest route between these two cities based upon the given constraints.
- Display the results to an output file.

The cheapest route must be picked following these requirements:

- You must stop at each city along the route and at each stop, you visit one restaurant. 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.
- You can only go to one restaurant at each stop (city).
- At each restaurant, you must have a unique meal (i.e. you cannot repeat a prior meal choice).
- You do not need to stop at a restaurant at your departure city, but you will stop at a restaurant at your final destination.
- Gas prices are different in different states. The prices can be found in StateGasPrice.csv, with the state abbreviation and corresponding price in cents/L.
- The cost of fuel will be calculated based on the aerial distance you are driving between two cities and based on fuel efficiency at 8.2L/100km. In each state you will pay for the amount of fuel that you consume driving inside the state based on the gas price in that state. The cities that fall

approximately on the border between two or three states are tagged with more than one state (e.g., NYC is on the border of NY, CT, and NJ). If there is not a border city between two cities in two different states in a route, you can assume that the fuel cost for that segment is the average fuel price of the two states.

3. Problems

3.1 Analysis (10%)

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

3.2 Breadth-First Search vs. Depth-First Search (30%)

Implement a breadth-first search algorithm and a depth-first search algorithm to find a path from the start city to the destination city based upon the provided routes between cities (connectedCities.txt). The start and destination city MUST be read in from a .txt file called a3_in.txt located in your data folder of the project. The first line of a3_in.txt contains the start city and the second line of a3_in.txt contains the destination city. The output specification is described below. Note that if your application does not generate the required output file you will lose mark for this section too.

Did they return the same result? Which algorithm found the result in fewer steps? Will this remain true if the number of cities increases? Why or why not?

3.3 JUnit (BFS vs DFS) (10%)

Implement a JUnit class to test that all possible routes for your selected departure and destination cities are examined and the correctness of the routes, i.e., a route includes only cities that are connected and in the correct order. For testing, you should pick two cities (as the departure and destination cities) that there exist at least three different routes between them, i.e. at least one stop in each route is different from the others.

3.4 Shortest Path (40%)

Find the lowest cost route from the start city to the destination city based upon the given constraints (food and fuel expenses). Use a shortest path algorithm to solve this problem. The output specification is described below. Note that if your application does not generate the required output file you will lose mark for this section too.

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

4. Output Specifications (10% - Output Formatting, Comments, etc.)

You should have 2 different output files named “a3_out.txt” and “a3_answers.txt”.

The first two lines of “a3_out.txt” should contain the resulting path for BFS and DFS in the following format:

BFS: StartCity, CityA, CityB, …, DestinationCity

DFS: StartCity, CityA, CityB, …, DestinationCity

This should be followed by a table which contains your result for 3.4. The table headings are City, Meal Choice, Cost of Meal, Cost of Fuel (from previous city to this city), Total Cost. Each row in the table should contain a city and the corresponding information. The final row in this table should depict the Total Meal cost, Total Fuel cost, and the Grand Total cost.

The "a3_answers.txt" should contain the written answers to 3.1, 3.2, and 3.4.

Assignment 3 Due Date

March 27 2016, at 22:00.

Assignment 3 Marking

Assignment 3 is worth 10% 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.

Assignment 3 Submission

Your submission will be an Eclipse project of your A3 implementation. The name of your Eclipse project should be cas2xb3_A3_*lastname_initials*. You should include a txt file named 2xb3_A3_*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_A4_*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 3 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