Noah Dela Rosa

CS 201

CTA Project Design

1. The user interface will essentially be a copy of the interface from the previous labs. Thus, the options will be listed, with the user input typing in the number corresponding with the menu option. The options will be the options listed in the project: searching for a specific station, adding, modifying, and deleting stations from the CTA system, find the nearest station to a location, and most importantly, generate a route from one station to another. One key aspect of user interface that will be of focus is on not trying to clutter the screen with too much information, and will strive for a clean design, only showing the important information.
2. Programmer's’ task
3. The input file will be read by inputting the file as a Scanner class into the java program, and the actual input file will be uploaded alongside the java program.
4. The delimiter will be set to commas and the enter sign. The first two lines will be skipped with the nextLine method, as the information on those lines does not seem too pertinent. For each index of the delimiter, the value will be set to a variable, and will be parsed into their respective data types. A CTA Station class will be instantiated from the inputted variables, and under the assumption that each CTA line ArrayList has been initialized, the integer values of the color line will be used in a series of boolean checks. If the integer value is positive for that respective color column, then the CTA Station will be added to that respective color line’s ArrayList. After all the information is processed, the ArrayList of the color stations will be sorted by the color line index with most likely with a bubble sorting algorithm.
5. Each CTA Station will contain its values in the form of a class called CTAStation. And the CTA lines will also be represented as classes, named CTARoute, which will contain an instance variable of a CTAStation ArrayList. This ArrayList will be used to store each individual CTAStation object.
6. I will demonstrate polymorphism with the route forming function. Each CTA Route will have its own route creation method, which will search for both stations on the line. But when the route goes beyond one CTA line, another class, called the CTASystem class (which contains the data of all CTA Routes and extends from CTARoutes), will call the overridden route creation method.
7. I will add, delete and modify data by utilizing the ArrayList methods of the CTARoute class.
8. I am going to search by most likely using the sequential search algorithm, but I will try to implement binary searching. The searching conditional will be based on the equals method of the CTAStation.
9. The classes that I will implement are the client class, CTAStopApp, and then the other classes are GeoLocation, contain location information like latitude and longitude. CTAStation, which will inherit from GeoLocation, represents the each individual value of a station. CTARoute will contain the information of each color line, like a list of all the stations on the respective line. And finally, CTASystem, which will inherit from CTARoute, will contain all of the data of the entire CTA train system, as well as override one of the CTARoute methods.
10. See other page
11. Test Plan
12. After a function is run, I will display the current status of the CTA system as a form of confirmation that the alteration by the function was successful. Every time user input is required, a check will occur that the user is inputting the correct data type with boolean checks and try catch statements. However, String inputs may need extra checks in order to verify that the inputted Strings are valid Strings. String inputs will check if the input matches a set of predefined Strings.
13. Polymorphism will be demonstrated by using the print method to print a unique message for either the super route creation method and the overridden route creation method. So when the route is ran, a specification will be given to which method is being run, but this test also confirms polymorphism by having the same method name being called, but each having a different outcome.