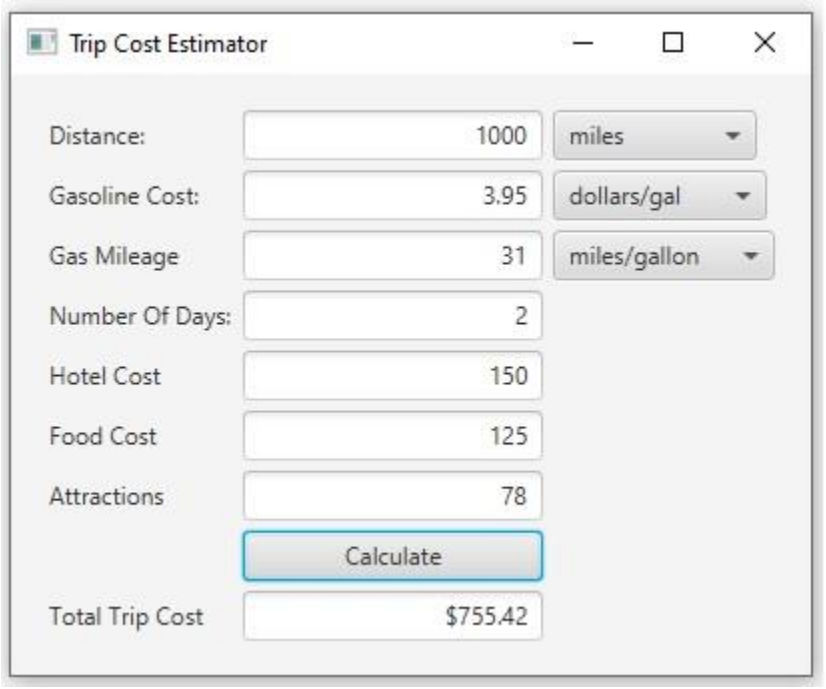
CMSC 215 Intermediate Programming
Programming Project 3

The third programming project involves writing a program that produces a road trip cost estimator with a GUI interface. The required GUI is shown below with an example calculation that uses strictly Imperial units (i.e., miles for distance, dollars per gallon for gasoline cost, and miles per gallon for gas mileage):



The text fields above the *Calculate* button are the input fields and the one below is the output field. The output field should not be editable. Three of the input fields have combo boxes to their right, which allow the data to be input as described below:

- The distance can be entered in either miles or kilometers
- The gasoline cost can be input in either dollars per gallon or dollars per liter
- The gas mileage can be input as either miles per gallon or kilometers per liter

The necessary calculations are as follows;

$$gasoline\ cost = \frac{distance}{gas\ milage} \times gasoline\ cost$$

$$total\ trip\ cost = gasoline\ cost + (hotel\ cost + food\ cost) \times number\ of\ days + attractions$$

Use these constants for your unit conversion calculations. Be sure to declare them as constants rather than embedding the numbers themselves inside your algorithm:

KILOMETERS_PER_MILE = 1.609347

LITERS_PER_GALLON = 3.78541178

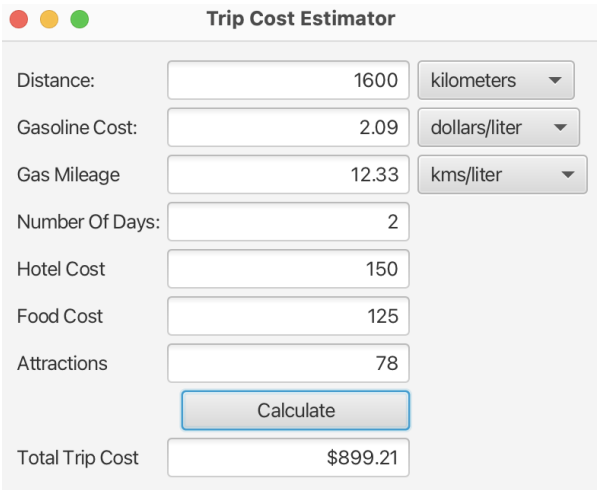
Note also that 1kms/liter = 2.35214 miles/gallon
(i.e. LITERS_PER_GALLON/KILOMETERS_PER_MILE = 2.35214)

Your program should consist of two classes. The first class `Project3` should define the GUI illustrated above. The second class `TripCost` should be an immutable class that has at a minimum a constructor that creates a trip cost object and a method that computes and returns the total trip cost.

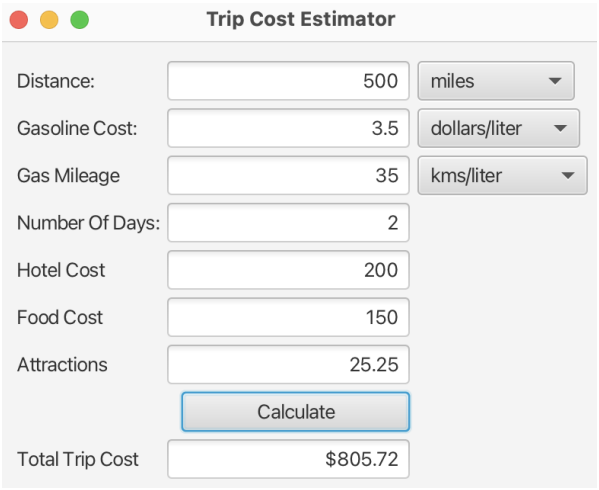
Your solution needs to be able to handle all combinations of unit system choices selected from the combo boxes by the user. Some additional examples are shown below, but they are not exhaustive. Your test plan needs to consider ALL combinations and include test runs for each potential choice. Three combo boxes with two options each result in 2^3 combinations.

Hint: A useful approach is to gather all the unit selections the user makes from the combo boxes and convert each one as necessary to **one of the unit systems**. Then pass the resulting values to the TripCost constructor.

Here is a sample using strictly Metric units:



Here is a sample using a mixture of Imperial and Metric units:



Documentation Requirements:
Make sure your Java program is using the recommended style such as:

- Javadoc comment with your name as author, date, and brief purpose of the program
- Comments for variables and blocks of code to describe major functionality
- Meaningful variable names and prompts
- Class names are written in upper CamelCase
- Variable and method names are written in lower CamelCase
- Constants are written in All Capitals
- Use proper spacing and empty lines to make your source code human readable

You are to submit two files.

1. The first is a `.zip` file that contains all the source code for the project. The `.zip` file should contain only source code and nothing else, **which means only the `.java` files**. If you elect to use a package the `.java` files should be in a folder whose name is the package name. Every outer class should be in a separate `.java` file with the same name as the class name. Each file should include a comment block at the top containing your name, the project name, the date, and a short description of the class contained in that file.
2. The second is a Word document (PDF or RTF is also acceptable) that contains the documentation for the project, which should include the following:
    a. A description of your approach taken in designing and implementing the project.
    b. A UML class diagram that includes all classes you wrote. Do not include predefined classes.
    c. A test plan that includes test cases that you have created indicating what aspects of the program each one is testing. Include the results of your testing with screen captures clearly showing the output for each test case.
    d. A short paragraph on lessons learned from the project.