I approached this code in a similar way as assignment 2. The skeleton was similar with a primary “parent” class being the Routes class for me with all of the other classes extending that class. Interfaces are set up to ensure that the classes have the proper methods(every subclass has a vehicle method and the commuter subclass has an engine method). With the interfaces and parent abstract class created I realized that the constructors needed for each specific class and the ones that aren’t needed made this program far more complex than the assignment 2.

From there I took a step back from the actual programming on the computer and started planning the code out on paper and creating a layout similar to the table you provided in the instructions. This was hopefully so that I would not have to go back and change much of the route classes once I started on the RouteTest class. I created an array that held all of the Route objects. I counted through this with a for each loop and found that I could cast a subclass to the parent class Routes (something I didn’t know you could do until I tried it).

From there I just had to print the values and have the formatting match the example output you provided. This was the most time consuming thing because I had to debug it to make sure the values were right which became very tough because of the number of variables needed by each constructor. I found that even with my outline on paper there were a large number of errors and it just became a game of trial and error for much of the time I spent on the program. Finally I set up all of the return statements so that they rounded to two places after the decimal point (000.00).

Executing this program is very simple, nothing else is needed other than the .java files. The values provided are hard coded in so no input is needed from the user. Compiling all of the files and then running RoutTest is all that is needed.