CSE 271 Lab 12 – Practical Searching and Sorting Algorithm Usage
Spring 2022
Assigned: 4/21/2022
Due: 4/24/2022

Introduction:

In this lab, we will practice utilizing both searching and sorting algorithms in a practical sense. That is, create a new project in your Eclipse IDE called **Lab12** for which we will be creating a class called **Car.java**. This class will be used for creating objects from the data you will read from the provided text file. All methods for sorting, searching, etc. will be contained in this class. Please review the **CarData.txt** file that is provided in the lab assignment download. The data is formatted as follows: "id,carMake,carModel,year". Note that there are no spaces and that it is strictly comma delimited. You may reference the lecture code to help you with this lab assignment. Lastly, to test this class, make a **Tester.java** class, which you will not submit, that will contain the main method. Use this to ensure your methods are working as intended.

Car Class:

Create a class named **Car.java** which implements the interface **Comparable<Car>** and provide the following instance properties and public interface.

- Instance Properties
  - **private int id –** The ID number of the object.
  - **private String carMake –** The make of the car.
  - **private String carModel –** The model of the car.
  - **private int year –** The year the car was manufactured.
- Public Interface
  - **public Car(int id, String carMake, String carModel, int year) –** The workhorse constructor for the **Car.java** class.
  - **public static Car[] readFromFile(String filePath) –** This method reads the data from the provided **CarData.txt** file and returns an array of Car objects.
  - **public boolean equals(Object o) –** This method overrides the default equals() method and determines if this **Car** object and the **Car** object in the parameter are equal. Specifically, it compares the **carMake**, **carModel**, and **year** instance properties.
  - **public int compareTo(Car c) –** This method is implemented from the inclusion of the interface **Comparable<Car>** and compares this **Car** object with the parameter **Car** object. Specifically, it compares both objects based on **carMake** then **carModel** lexicographically, e.g, "Ford" comes before "Honda". That is, if the **carMake** values are the same, then the comparison is done based on the **carModel**, e.g., "Bronco" comes before "Mustang".

- **public static void sort(Car[] cars) –** Sorts the array of **Car** objects. You may use any sorting algorithm you prefer except the built-in Arrays.sort() method. You may utilize additional methods to help perform the sorting. Hint: Instead of comparing using “>” or “<”, use the **compareTo()** method.

- **public static int linearSearch(Car[] cars, Car car) –** An implementation of the Linear Search algorithm. Hint: Instead of using the “==” operator, use the **compareTo()** method.

- **public static int binaryRecursiveSearch(Car[] cars, Car car) –** An implementation of the recursive Binary Search algorithm. Hint: Instead of using the “==” operator, use the **compareTo()** method. Note, you can use an additional private method to do the recursive calls.

- **public String toString() –** This method overrides the default toString() method and returns a formatted String of the **Car** object. For example: “3, Ford, Bronco, 2021” for the **id**, **carMake**, **carModel**, and **year** instance properties, respectively.

Additional Notes:

- As mentioned in the introduction, you may use the lecture code as reference for this lab assignment.
- Make sure you include JavaDoc comments for all methods and the class including parameter and return descriptions.
- Make sure that you name the class correctly.
- Assume all input values in the **CarData.txt** file are correct. That is, you do not need to do any validation checking for this assignment.

Submission Instructions:

After you have completed the lab assignment, locate your source code (**Car.java**) in your workspace and submit it to the corresponding Lab 12 assignment’s CODE plugin.

Rubric:

Task	**Grade**
Car.java	
Workhorse constructor correctly implemented	5
readFromFile() method correctly implemented	5
equals() method correctly implemented	5
compareTo() method correctly implemented	20
sort() method correctly implemented	20
linearSearch() method correctly implemented	20
binaryRecursiveSearch() method correctly implemented	20
toString() method correctly implemented	5
Appropriate JavaDoc included and followed the Miami University coding guidelines (you can only lose points here)	0 (-5)
Total	**100**