**Assignment**

**Search Algorithms**

**Dataset:** We will use cars dataset as part of this project. The dataset is available as part of the assignment (cars.csv and a pdf that describe each of the fields for each record).

**TASK 1**

The idea of this project is to create a text-based application with the following options:

1. **List all cars.** This option will read from the file and when the user selects this option, the program will list all the cars in the file.
2. **Find via Linear search**. This option will allow the user to type a model of the car (ID field in the dataset) and it should print out the information about the car searched, the amount of time it took to find the car using **linear search**. For this option, create a class called SearchMethods and implement the method :

public static <T extends Comparable<T>> int linearSearch(T[] data, int min, int max, T target)

1. **Find via Binary search.** This option will allow the user to type a model of the car (ID field in the dataset) and it should print out the information about the car searched, and the amount of time it took to find the car using **binary search**. For this option create, a class called SearchMethods and implement the method:

public static <T extends Comparable<T>>

int binarySearch(T[] data, int min, int max, T target)

in both 2 and 3 the binary and linear search methods return the position in the array where the elements are found.

**Remember:** To apply binary Search you must sort the array. Measure the time to execute binary search only when calling binary search, not on the sorting part of it.

1. Option to quit the program.

**TASK 2**

**Time Complexity:** Build a table and record the time complexity taken by linear and binary search to look for an element with the following array sizes: 10, 50, 100, 200, 500, 700, 1000. Compare the performance of both algorithms highlighting the time complexity for both algorithms when run on small input size and when run on large input size.

To accomplish this, load a list with the file data and then create a program that selects at random X amount of elements from the original list (where X is a value that the user selects) and place them in an array, then select at random an element in the original list to search in the sampled array using both algorithms. Measure the time it takes for both algorithms on an array of a given input size.

For this part just create a class called Task2Exercise in your project, reuse any code you did for the first part.

**TASK 3**

Using one of the sorting methods, sort the Car ArrayList based on their ‘City mpg’ feature. Justify your choice of the algorithm used.

**HINTS:**

1. You need to create a class called Car that implements the Comparable interface and load the data from the file to a list. Then somehow convert the arraylist to an array to use the implementations of the algorithms
2. To measure the time it took to run, you can use **System.nanoTime() or System.currentTimeMillis** before and after the method is executed (see here: <https://docs.oracle.com/javase/7/docs/api/java/lang/System.html>) and subtract.

**Deliverables:**

1. **A IntelliJ project implementing the tasks**
2. **A report on the second exercise showing a graph of running time vs. input size of both algorithms and your conclusions.**