LAB 6: SEARCH

F27SG – SOFTWARE DEVELOPMENT 3 (5 MARKS)

The topic of this lab is **search**. Download Lab6.zip from Vision and import the project into Eclipse. Your task in the lab is to complete the implementation and Junit tests for this project. Read the source code and make sure that you understand what is going on and what is missing. Before you start, you will need to download the ZIP file and import it into Eclipse:

File -> Import -> Existing Projects into Workspace

Then select the project you downloaded. The project is organized as follows:

- The src directory contains all the source files
    - Entry.java represents entries which you should search over. Each object of this class is meant to represent an entry in a catalogue, containing the name of a person and an associated phone number
    - **ASearch.java** is used to search over arrays. It contains an array called **catalogue** which contains Entry objects.
    - **LSearch.java** is used to search over linked lists
- The test directory contains the unit tests for the project.

1. WRITE UNIT TESTS FOR LSEARCH AND ASEARCH (1 POINT)

In the test directory there are two test files. These are used to test the search methods for arrays and linked lists.

Complete these test methods. Each of the search methods are given a string, which represent the name of a person.

The search method should return the corresponding phone number of the Entry object.

You should first make sure that you are familiar with the code in Entry.java, ASearch.java and LSearch.java. Make sure that the catalogue used to test the binary search method is ordered (on the names).

I would also encourage you to write a **main** method for both **ASearch** and **LSearch** to see the programs working. Here, you could generate some entries, search over them and print the results. Note that this is not required to get the mark.

2. IMPLEMENT LINEAR SEARCH FOR ARRAYS (1 POINT)

In the ASearch.java file there is a **linear search method**

```
public int linearSearch(String name){

      // your code

}
```

Using the **catalogue** array of `Entry` objects, this method is given a (String) name. The method should return the corresponding phone number of the name if it exists. If no such name exists, then return -1. The tests from part 1 should succeed. See hint1 if you are stuck.

3. IMPLEMENT LINEAR SEARCH FOR LINKED LISTS (1 POINT)

In the LSearch.java file there a **linear search method**

```
public int linearSearch(String name){

      // your code

}
```

Using in the previous part, you should complete this search methods, however in this case you will be searching over a linked list. The tests from part 1 should succeed.

4. IMPLEMENT BINARY SEARCH FOR ARRAYS (2 POINTS)

For this part you can assume that the given catalogue is sorted (why is this necessary?).

In the ASearch.java file there is a **binary search method**

```
public int binarySearch(int first,int last,String name){

      // your code

}
```

Your task is to complete this method. As in the other parts you are given the name and should return the corresponding number in the Entry object. The tests from part 1 should succeed. See hint 2 for help.

HINTS

Hint 1! You will need to use the `getName()` and `getNumber()` methods of the Entry objects to get the respective name and number.

To check if two strings are equal, you need to use the equals method for Strings, i.e. `catalogue[i].getName().equals(name)`

Hint 2! *The binarySearch method deviates from the one in lecture 6 by using strings instead of integers.*

To check if one string is smaller than the other (to figure out whether to search on the right or left of middle), use the `compareTo` method, i.e. `(catalogue[middle].getName().compareTo(name) > 0)`