Documentation of CSC1022 Coursework

*Dimo Dimchev Student No:180322572*

## ArrayDirectory

About the code – I have used an array from class Entry (Entry []);

In the **insert** method I add the new element at the end of the array and then I compare its name to the element next to him until the Entry element is sorted in alphabetical order

In the **lookup** method, we use binary search (the array is sorted when it is inserted, so we don’t have to worry about that).

The method **deleteByName** uses lookup to find the element, then it moves it to the end of the array and deletes it (sets it to null in the array)

Delete my number method uses linear search with extension and when it finds the element it puts it in the end and deletes it!

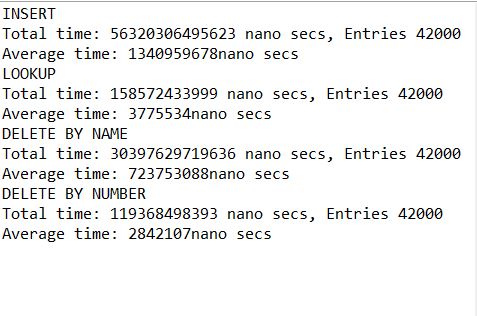
**changeNumber –** uses binary search to find the element by name and then it sets the number of the element to the new one.

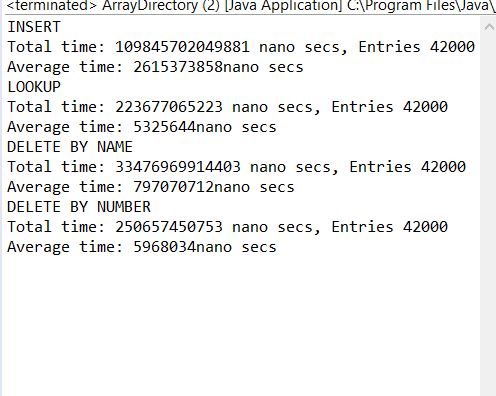
**Print** – prints the whole directory using string builder to create a string and then return it.

**Private interface Comparable** -this is a callback for finding a compatible element of the array -it is passed to the private int find (Comparable c) function, which goes through the array and for each element calls the check function of the interface. The idea is for example, if you want to search for an element by the initials, you just need to add a new function (or to use lambda function) which implements the Comparable interface and passes the find function!

**ad.find(e -> e.initials.equals("N.A."));**

**TestPerformance**  is a callback for performing an function of the Directory interface. It is used to pass to the **TestFromFile**, which calls it and uses the stopwatch to see how much time it takes for a function to be performed.

Those are some testing run times in Nano seconds I will leave the methods commented, so you can see them!

About the GUI embedded in this class!

I have used text fields from the javax.swing library. Text fields are used to input and output values. One of them “Result” can’t be edited, so the user cant input results(really has no reason to)

I have also used buttons for every action that needs to be done! Basically, when the buttons are clicked an action (one of the methods from Array Directory is performed!).

I have also added a list or a text area in which I can print my directory.

I have commented the actions and what they do in the code.

# List Directory

The **insert** method goes through the whole linked list, finds the place for the element and adds it.

The **deleteByname** method does a linear search through the whole linked list comparing the names, and when it finds a certain name it deletes its entry.

The **changeNumber** method uses linear search to find the element by also comparing its name and then it changes its number

The **lookup** method again uses a linear search to find the name and return its number.

*Those are the slowest methods, because they are all with linear search! It Took A lot of time to test.*

## Hash Directory

The **insert** method inserts element into the array of linked lists that are in order – the first linked list has elements starting with ‘A’, second has elements starting with ‘B’ and so on until the last linked list has elements starting with the letter ‘Z’. It finds the right linked list using hash Code. Basically, the hash codes of the alphabet a-z are consecutive numbers and we find the place of the correct linked list by subtracting the value of ‘A’ (hash code) from the first letter of the string that is passed (**name** of the entry).

***int index = e.name.toUpperCase().charAt(0) - 'A';***

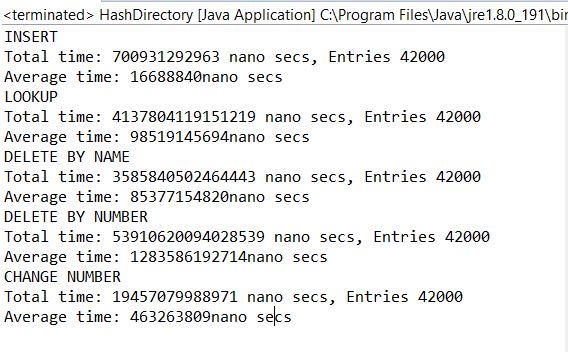
After the right linked list is found the element is sorted and added!

The **deleteByName** finds the way to the correct element using the hash code technique. After that it goes through the whole link list using linear search to find the element with the same name and deletes it.

The **lookup** method uses the same technique used in **deleteByName** to find the element. It first uses hash code technique to find the list and then it does a linear search through the list to find the extension for a certain name and then returns it.

The **deleteByNumber** does a linear search for all the elements in the array of linked lists and when it finds an element with a certain number it removes it!

(Those are not the most efficient methods as you will see in the picture bellow that shows the testing of the performance)



And the **print** method goes through the whole directory and prints the Entries!

# Entry

Basically, a constructor with that gives values to tree public strings – a name, an initials and an extension.

Also has some getters and setters.

# Interface Directory

Pretty straight forward the insert methods take values that are passed from class Entry. And throws exception.

The Delete by name method takes name.

Delete by number takes extension;

Lookup method takes name.

Change number takes the name of the element and the new number.

The print method prints the whole directory!

P.s In every class there is also a Test() method that I included just for testing as well as a private object that I use to test !

IMPORTANT: If you want to run the TestFromFile Method and to check the performance when you call the method. Need to put the directory, where the file I have applied (test.txt) is at. I know that there is a way that takes the file if it is in the same directory, but for some reason it didn’t work for me 😊.

Happy testing!