Lab 7

- Algorithms on Arrays
- Free form (or question) session with TAs

Exercise 1: Execution time (Experiment)

- Recall a java method called linearSearch that we developed in class. It takes as an input a reference to an array of integers and another integer, called a key. The method returns an integer that is the first location in the array the key is found in, or -1 if the key is not found in the array.
- Then recall a java method called binarySearch that solves the same problem but much more efficiently due to the fact that input is a sorted array.
- Open Speed.java file. It contains both methods. In Speed. java you will see first that we randomly generates an array, called numbers, of SIZE=100,000 integers in the range [0 , 5*SIZE)

  and a random integer, key, in range [0, 8*SIZE). Then we sort this array using java.util.Arrays.sort(numbers).

Your job is to estimate/observe/compare the execution times of linearSearch vs BinarySearch. ….. Continue to the next page

Exercise 1: Execution time (Experiment) …continued

You can use the following code template to obtain the execution time in nano seconds:

- long startTime = System.nanoTime();
- perfome the test
- long endTime = System.nanoTime();
- long executionTime = endTime - startTime;

Observe the difference in time taken by LinearSearch and BinarySearch. Play with your code and try to do the same with an array of million integers, and then with 10 million … etc. Observe the times of execution.

Note however, that at some point your program will fail since the OS will not be able to give you enough of memory (eg. for an array of size 100 million)

Exercise 2: Move Zeros Exercise

- Write a Java method that moves all the zeros in an array to the end of the array. For example, if the array is {1, 0, 3, 0, 0, 5, 7} the new array is {1, 3, 5, 7, 0, 0, 0}
- Program 2 different solutions:
  - One should use a second, tmp, array to build the result (easier problem).
  - The other should be moving elements in the same array without creating any additional arrays (so called, in-place, solutions). (harder problem)
  - For both problems the TAs will first discuss algorithmic approaches to solve the problems.

Exercise 3:How do we make a method return more than one value: - need to put the values in an array and return the array

Implement a Java method that takes an array as input, calculates the maximum and the minimum in the given array, and returns as result and array with two elements, the maximum value and the minimum value.