CSCI 241 Data Structures
Laboratory Exercise 3

Objectives

Implementation and application of heaps

Part I Implementation

To access your assignment repository on GitHub, please click https://classroom.github.com/a/ymUwEYeW . You will need to clone the repository to your local disk so you can work and test it. Please refer to the Instruction of Lab 1 for the structure of repository and where the java file is located.

Your task for this part is to implement max heap as a class in java. Your class should be named "HeapMax"; in other words, the file name for this class would be "HeapMax.java". Your max heap will have the following methods as:

- Constructor *HeapMax*: The constructor method should initialize the data storage for your data structure. You need to deal with the scenario when the storage is full if you use arrays or data structures that have inflexible sizes in other methods (e.g., insert).

  public HeapMax() ---- already finished for you

- Insertion *insert*: This method inserts a new element to the max heap. You need to think about the scenario when a user tries to insert more elements than the original designated size (*if you decide to implement the heap using array class*).

  public void insert(int element)

- Get Max *getMax*: This method returns the root element of the max heap. You need to deal with the scenario when there are no elements in the heap while this method gets called.

  public int getMax()

- Remove Max *removeMax*: This method removes and returns the root element of the max heap, and reorganize the heap accordingly to restore its max heap characteristics. You need to deal with the scenario when there are no elements in the heap while this method gets called.

  public int removeMax()

- Display Heap *display*: This method should print all elements in the max heap in the order of levels (e.g., root element first, root's two children second, root's grandchildren third, etc.). Elements on the same level should be separated with one or two white spaces; elements on different levels should be separated with comma.

  public void display()

- Build a Heap *build*: This method should take an array as the only parameter and build a heap based on the array. Note that this method is NOT supposed to be called if a heap is non-empty.

  public void build(int[] arr)

These methods should **NOT** be declared as static. You are free to use any extra instances and helper methods.

Test Your Implementation

Your task for this part is to perform testing of your HeapMax class. You need to create a class named "HeapMaxTest". You need at least three examples covering varied scenarios. In each example, you will need to perform multiple insertions and removing max values. After each operation, you need to call display method to confirm that the operation works as you intended. Here is an example of the printings of a test case:

```
Testing of HeapMax starts.
Insert 5.
Current heap is: 5.
Insert 6.
Current heap is: 6, 5.
Insert 3.
Current heap is: 6, 5  3.
Insert 4.
Current heap is: 6, 5  4, 3.
Remove max.
Current heap is: 5, 4  3.
Testing of HeapMax ends.
```

Application

Apply your HeapMax class to solve the following problem:

In a research project your teammate who is responsible for data collection retrieves the rankings of a set of experiment results, and she chooses to use integers to stand for the success rate of each experiment. The rankings are initialed stored in a n*n matrix where each row and column are sorted in increasing order. Your project manager wants to know the ranking of kth most successful experiment. In other words, your manager wants to find the kth largest number in the matrix. Please help the manager solve the problem.

Please note that it is the kth largest element in the sorted order, not the kth distinct element.

Example:

```
matrix =
[
   [ 1,  5,  9],
   [10, 11, 13],
   [12, 13, 15]
],
k = 2, return 13.
```

This class should be named "HeapMaxApp". The method that solves this problem should be declared static and named as "kthBiggest", which accepts two required parameters:

public static int kthBiggest(int[][] matr, int k)

You should also create a main method under this class, in which at least two test cases need to be given to test kthBiggest method by printing necessary information. For instance, the printed messages of an example are:

```
Testing of kthBiggest starts.
The given matrix is:

1, 5, 9
10, 11, 13
12, 13, 15

The 2nd biggest element is 13.
Testing of kthBiggest ends.
```

Hint: You only need a max heap that always has *matr.length* elements.

Submission and Grading

You should check https://travis-ci.com, and find your lab2 repository. If your code fails to pass the testing cases on Travis-CI, it is not likely you will get a good grade.

Please update the README file, and do not forget to make one last push to GitHub when everything is finished.

Please use comments to explain your program or any sections that are possibly difficult to understand. 10% of the lab points would be deducted if few or no comments can be found in your java files. If your java files fail to compile, you may risk getting a 0.