CS 101, Spring 2020
Homework Assignment 3
Due: 23:55, May 22, 2020

Instructions: Implement your solution using your favorite Java IDE and compress your .java files as a .zip file. Submit your work to Moodle as a single .zip file using the following naming convention: `SS_HW03_Surname_FirstName.zip` where SS is the section number (01 or 02), and Surname is your family name, & FirstName is your first name.

Follow the CS101 Java coding style guidelines provided on Moodle (see FAQ).

Important note: All of your methods must be preceded by a JavaDoc comment that gives a brief description of the method's purpose and includes @param tag to list parameters and @return tag to describe what it returns, where appropriate.

In this homework assignment, you will write a customized class for arrays.

Create a class named `MyArray` that has the following properties and methods:

1. **Instance Data Members**
   - `array` : stores a list of integers (of type `int[]`)

2. **Methods**
   - **Constructors**
     - Write two different types of constructor for initialization.
     - In default constructor, the length of the array should be 10 and the values should be set to -1.
     - The second constructor takes an integer as a parameter, *length,* which is the length of the array. Values in the array should be generated randomly and values should be in the range `[0, length]`.

   - **Accessor/Mutator Methods**
     - Write getter & setter methods. In the setter method, if the lengths of arrays are different, you need to create a new array and that should be our new instance data member. If the array of the parameter is empty, you ignore it and do not change the data member.

   - **Other Methods**
     - `findLucky()`: finds and returns a lucky integer in the array. Let's define a lucky integer. It is an integer which has a frequency in the array equal to its value. If there are multiple lucky integers return the *largest* of them. If there is no lucky integer return **-1**. For the sake of simplicity you may assume the values in the array always lie on `[0, length]`.

- `isBalanced()`: So an array is balanced if there is a place to split the array so that the sum of the numbers on one side is equal to the sum of the numbers on the other side. This method takes an array of integers as a parameter and returns two things: a boolean variable that stores whether the array is balanced and the index of the last element of the first part of the array if balanced. The index should be -1 if the array is not balanced. How can you return two values in Java? *Note that* these values, here, are not even from the same data type.
- `merge()`: takes an array of integers as a parameter and merge with our instance data member. So, the merged one is our new instance data member. The parameter should not be affected.
- `randomize()`: changes the indices of the values randomly in the data member `array`. In other words, it shuffles the values. So, the last version is our new instance data member. *You are expected to do it without using built-in functions except Math.random(). But you may write auxiliary functions if needed.*
- `sort()`: sorts our data member according to the following recipe:

  - ➔ *In first cycle,*
    - *Start by comparing 1st and 2nd element and swap if 1st element is greater.*
    - *After that do the same for the 2nd and the 3rd element.*
    - *Then do the same for the 3rd and the 4th element element, and so on including the last pair of elements in order.*
    - *At the end of the cycle you will get a max element at the end of the list.*
  - ➔ *Now do the same in all subsequent cycles excluding the sorted element(s) at the end of the array.*
  - ➔ *Perform this for (number of elements – 1) times, starting from index 1 each time*
  - ➔ *You will get a sorted list at the end.*

Create a class named TestMyArray by creating `MyArray` objects to ***test your class and methods thoroughly with many cases***. See a sample run below. Note that this shows a small part of your test cases. This is just for you to understand what the methods are supposed to do. Also the reason we are not giving all the cases is because you need to be able to prepare different test cases to test your method. Please prepare and try your all test cases before submitting your homework.

```
-------------  findLucky()  -----------------
Array: [ -1, -1, -1, -1, -1, -1, -1, -1, -1, -1 ]
The array has been set.
Array: [ 1, 2, 2, 3, 3, 3 ]
The lucky integer is 3

-------------  isBalanced() -----------------
The array has been set.
Array: [ 10, 1, 2, 7, 0, 6 ]
The array is balanced and the last index of the first part is 2

-------------  merge()  --------------------
Random array has been created.
Array: [ 4, 3, 0, 6, 2, 9, 5, 2, 7, 7 ]
array to be merged with: [ 99, 65, 45, 21, 31 ]
The arrays have been merged.
Merged array: Array: [ 4, 3, 0, 6, 2, 9, 5, 2, 7, 7, 99, 65, 45, 21, 31 ]

-------------  randomize()  -----------------
Array: [ 4, 3, 0, 6, 2, 9, 5, 2, 7, 7, 99, 65, 45, 21, 31 ]
The array has been randomized.
Array: [ 2, 6, 3, 9, 4, 21, 65, 99, 7, 0, 5, 7, 31, 45, 2 ]

-------------  sort()  ----------------------
Array: [ 2, 6, 3, 9, 4, 21, 65, 99, 7, 0, 5, 7, 31, 45, 2 ]
The array has been sorted.
Array: [ 0, 2, 2, 3, 4, 5, 6, 7, 7, 9, 21, 31, 45, 65, 99 ]
```