CSC/DSCI 2720 Data Structure
Lab 6

Due: 02/18/2024 @ 11:59PM

Requirements:

(Failure to follow the requirements will result in a score of Zero.)

1. You may use whatever IDEs / editors you like, but you must submit your responses on iCollege as **.py files**.
2. Your submissions will work exactly as required.
3. Make sure you submit a file that compiles.
4. Your submission will show an output. Should you recive a Zero for no output shown do not bother to email with "but the logic is perfect".
5. Your program's output must exactly match the specs (design, style) given here for each problem to pass the test cases.
6. Design refers to how well your code is written (i.e., is it clear, efficient, and elegant), while Style refers to the readability of your code (correct indentation, good variable names). Add comments to have necessary explanations for your program.
7. Add a "heading" at the very beginning of your .java files as follow:
   Your Name
   CSc 2720 Lab #N
   Lab time: put your lab time here
   Due time: put the due date here
8. *** If you used any website/online resources as reference, please cite in your comments what website did you use for studying the lab content. Also, you are supposed to make changes to the resource you use to make it your own version. Otherwise, your submission will be considered as plagiarism. ***

In user content generated web-services - let's say YouTube - the process of de- duplication is of serious importance. One straight forward reason is same video by different names is just an extra cost in data-storage. So, getting rid of exact duplicate content makes financial sense. Please be reminded that exact same content with different video qualities may not be candidates for removal via de-duplication.

In today's Lab we will explore on ways to do a de-duplication of videos where video filenames are presented as integers. For the purposes of our task, we will set a very narrow criterion for de-duplication: just the filenames.

Below is how the filenames are represented LST = [50, 11, 33, 21, 40, 50, 40, 40, 21]

Below is the expected output after de-duplication LST = [11, 21, 33, 40, 50]

ATTN: Notice the reduced size of the output list.
Also, take a look again at the input list and be reminded that the input list is not sorted!

Further, please be reminded that you cannot use library functions to either sort and or perform the de-duplication operation. Doing so would straight up result in a score of Zero!

You will solve the problem as:

> [100 points] Implement the function in such a way that your solution solves the problem with O(nlog(n)) time complexity overall but stays within the bounds of O(1) space complexity. Here, n is the length of the list of in- put integers (list). I believe the sorting routine that should be used here is **Quick Sort**. Please state as code comment which sorting routine you are using, sort the array with that algorithm and solve the deduplication problem thereafter. De-duplication part of the solution in itself must adhere to O(n) time bound but there is no constraint for space bound. However, at this stage of the course we will not be considering any memory used by recursion.

Very Very Important:

1. (1) Your code should be well commented which explains all the steps you are performing to solve the problem. A submission without code comments will immediately be deducted 15 points!
2. (2) As a comment in your code, please write your test-cases on how you would test your solution assumptions and hence your code.
   A submission without test cases will immediately be deducted 15 points! Example of cases to be tested for are like: What if the array input which is expected does not exist - that is, input is a null. How should your code handle such a situation? Maybe output some message like" Null input case, so no output"? What if the length of the array is one? ... so on and so forth.

   Please Remember: Although, written as comments - You will address your test cases in the form of code and not prose :)