QUIZ I (Take-Home)

Submission Date : *06.03.2020*
Due Date : *08.03.2020 23:59*
Programming Language: *Java*

Q1. Merge sort is often preferred for sorting a linked list. Implement a natural mergesort for linked lists. (This is the method of choice for sorting linked lists because it uses no extra space and is guaranteed to be linearithmic.) (25 points)

Q2. Suppose that you have an array of numbers having size of m. Further, you are given a number such that n satisfying the condition of (n<m). Your duty is to detect all the items in the array <u>appearing more than m/n times.</u>

Ex: int[] arr_sample = {5, 2, 7, 2, 1, 3, 5, 1, 3, 5, 5, 1, 1, 3};

Let us select n = 3. As the length of our array is 14, we need to find out the elements existing more than 4 ≈(14/3).

The answer will be {5, 1} because they are listed at least 4 times

Here you are expected to propose a solution having the complexity of **O(m.log(m)) + O(m)** (25 points)

*Bonus: Could you do it through the complexity of **O(mn)** ? (20 points plus)*

For this particular question, you must essentially submit one .java code files such as "q2a.java". However, if you want to make also the second version (the bonus version) you can also submit the file "q2b.java". **As a remark, the second bonus solution is not mandatory!** As a rule, those files must involve their own main functions and one sample array. For each .java solution you can use only 1 code file and arbitrary number of functions. Each of the .java code files will run its own algorithm along with hard-coded arr_sample array.

Q3. You are expected to compute the union of sets (arrays) A and B, where n= max (|A|, |B|). In other words, the outcome must be an array of having distinct elements that form the union of the sets, such that they appear exactly once in the union.

(a) Given that A and B are unsorted. Provide an **O(n.log(n))** algorithm for the problem and implement it by also providing two sample A and B integer arrays. (15 points)

(b)Assume that A and B are sorted. Provide an **O(n)** algorithm for the solution and implement it by also providing the A and B arrays given above. (10 points)

The general rule for this question is your submission file must be named "Q3.java" and it should contain the solutions for both of the (a) and (b) together.

Q4. The merge sort iteratively divides the array into 2 parts, sorts it, and finally merges it. The 3-way merge sort, a variant of the merge sort, divides the array into 3 parts instead of dividing it into 2 parts. Write and implement an algorithm that instead of dividing the array into 2 parts at each step, you divide into thirds, sort each third, and combine using a 3-way merge. What is the order of growth of the overall running time of this algorithm? (25 points)

NOTES

***** No late submission is permitted.

*****It is not an obligation but we suggest Eclipse platform since it is free and it gets updates constantly

*****Use submit system that has been configured for you. No other medium except Submit System will be accepted.

*****Use UNDERSTANDABLE names for your variables, functions, and classes (please be sure you are obeying name convention).

*****Write READABLE SOURCE CODE blocks.

*****Use EXPLANATORY COMMENTS in your source codes.

*****Your work must be original. Duplicate or very similar works are both going to be considered as cheating.

*****You will submit your work from https://submit.cs.hacettepe.edu.tr with the file hierarchy as below:

```
Src|--
        |--q1.java
        |--q2a.java
        |--q2b.java (optional)
        |--q3.java
        |--q4.java
```