Offline on Linear Time Sorting

(Submission Deadline for All Subsections: 24-07-2018 07:50am)

***Reference Book*:**
Introduction to Algorithms (Third Edition) by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein

Online pdf link (in case anyone doesn't have it):
https://mcdtu.files.wordpress.com/2017/03/introduction-to-algorithms-3rd-edition-sep-2010.pdf

Task-1: Implement Counting Sort

Ref: Section 8.2 of Reference Book

Problem Description
You will be given some integers. You have to sort them in ascending order & show the output.

Input Description
The first line of input denotes number of elements, *n*. The following line contains *n* space separated integers.

Sample Input - 1
5
10 30 2 15 2

Sample Output - 1
2 2 10 15 30

Sample Input - 2
3
10 -30 4

Sample Output - 2
-30 4 10

Marks Distribution

Sub-tasks	Marks
Implementation for non negative integer elements only	7
Handling negative integer elements without exceeding linear runtime of the algorithm	3

Task-2: Implement Bucket Sort

Ref: Section 8.4 of Reference Book

Problem Description

You will be given some real numbers. You have to sort them in ascending order & show the output.

Input Description

The first line of input denotes number of elements, *n*. The following line contains *n* space separated real numbers.

Sample Input - 1

5
0.25 0.44 0.1134 0.12 0.05

Sample Output - 1

0.05 0.1134 0.12 0.25 0.44

Sample Input - 2

4
0 1.25 -0.44 3.1134

Sample Output - 2

-0.44 0 1.25 3.1134

Marks Distribution

Sub-tasks	Marks
Implementation for elements in [0,1) interval only	7
Handling any real number element not necessarily in [0,1) interval	3

* Assume the number of buckets will always be 5 (Calculate the bucket no. for each value accordingly)

** You can use built-in data structures of the programming language you will use for implementation of the buckets, it is not necessary to implement linked lists. For example, you can use **vector** in *cpp* or **ArrayList** in *java*. However, this is not mandatory.

*** You can use library sort function of your programming language instead of using *insertion sort* in the bucket sort algorithm. This is not mandatory as well.

**** *Hint for handling elements outside [0,1) interval*: You can apply suitable arithmetic operations to bring the elements in [0,1) interval, use bucket sort algorithm to sort them & then revert them back by applying the reverse operations.