Object Oriented Programming (BCS-2A, BCS-2F, BCS-2G) – Assignment 1

Important Instructions:

- Submit only one running .cpp file carefully. After deadline no updated files will be accepted.
- Do not submit compressed files (.zip or .rar etc.).
- **Name of your submission file should be your roll number throughout the semester.**

Problem

You are required to write a program that reads two arrays from an input text file, sorts them, removes duplicate elements from each array and finds all the unique elements in both arrays.

Input File

Important Note: File name in your code should be "Data1.txt", it will be evaluated accordingly.

Data1.txt (Create file and save following data in it.)

```
17
55, 27, 8, 6, 11, 3, 15, 8,5, 55, 83 , 1, 3, 62, 49, 44, 8
15
27, 52, 8, 7, 11, 8,52,52,5, 55, 93 , 52, 45, 44, 8
```

Format of data file is explained below:

```
17 //Size of Array 1
55, 27, 8, 6, 11, 3, 15, 8,5, 55, 83 , 1, 3, 62, 49, 44, 8 //Content of Array 1
15 //Size of Array 2
27, 52, 8, 7, 11, 8,52,52,5, 55, 93 , 52, 45, 44, 8 //Content of Array 2
```

Important Note: *Your code may be evaluated on arrays of different sizes and content and it should work fine in all cases.*

Required Output

Final Output required on console (Your program should NOT take ANYTHING from user):

```
Input Array 1:
55, 27, 8, 6, 11, 3, 15, 8,5, 55, 83 , 1, 3, 62, 49, 44, 8

Sorted Array 1:
 1,3,3,5,6,8,8,8,11,15,27,44,49,55,55,62,83

Distinct Elements in Array 1:
1,3,5,6,8,11,15,27,44,49,55,62,83
```

Input Array 2:
27, 52, 8, 7, 11, 8,52,52,5, 55, 93 , 52, 45, 44, 8

Sorted Array 2:
5,7,8,8,8,11,27,44,45,52,52,52,52,55,93

Distinct Elements in Array 2:
5,7,8,11,27,44,45,52,55,93

Union of Array 1 and Array 2:
1,3,5,6,7,8,11,15,27,44,45,49,52,55,62,83,93

Important Instructions

- Do not consume any single extra byte.
- There should not be any dangling pointer or memory leakage in your code.
- Best use programming practices studied so far.
- Using Global Variables and "goto" in assignment is NOT ALLOWED.

Violation of following may result in **ZERO** credit:

- **Use pointers to scan/traverse the array. Any loop iterating from index zero to size (i.e. using integer iterators to traverse the array) is not allowed.**
- You cannot use subscript operator [] to manipulate arrays in your program.
- There will be marks deduction for every single exception.

Marks Distribution

Your submission will be marked for following:

- User Interface (Printing Input and Output)
- Required functionality
- Program's behavior on list size <= 0 (Display proper error message)
- Dynamic Allocation
- Memory Deallocation (this will be of significant weightage)
- Proper Functions Distribution (Input, Output, Sort etc.)
- Coding Standards / Readability:
    - Variable/Functions Names
    - Comments
    - Other Coding Standards

Help

- Allocate space for array according to its size in the file
- Sorting will be in-place i.e. you do not need to keep extra space for sorting.
- Remove duplicate will be in-place. After compression, you will move the data to array of required size i.e. if original array was of 10 elements and there are only 5 distinct elements then your distinct elements' array will be of size 5 in memory (and previous allocation of size 10 will be de-allocated)

Checklist before Submission

You have to check following before submission:

1. Readability: Proper functions and variable names and indentation. Before every function, in one or two lines properly comment what function does. Extra comments or comments on every line is bad programming. add one line before every loop telling what this loop does
2. Don't use extra variables, no memory leakage (No of news should be equal to No. of deletes), no exceptions, de-allocation carries SIGNIFICANT weightage
3. *(xyzPointer + abcInteger) is offset notation that is not allowed in assignment
4. Proper UI (User Interface)
5. Free a resource as soon as it is not required
6. Passing Pointers by Reference is NOT ALLOWED.