Computer Programming (B&E) – Spring 2018
Assignment 1

Deadline – Wednesday 31st January 2018, 11 a.m.

Submission Folder:
Section B: \\sandata\Xeon\Samin Iftikhar\Computer Programming (B)\Assignment 1 - 1D Array
Section E: \\sandata\Xeon\Samin Iftikhar\Computer Programming (E)\Assignment 1 - 1D Array
Submission Instructions:

- Submit your running files carefully. After deadline no updated files will be accepted.
- Do not submit compressed files (.zip or .rar etc.)

Exercise 1 [Bubble Sort]:

Write a C++ program that takes size of an array and array's elements from "Data1.txt" and displays the sorted list (in ascending order) of elements after applying Bubble Sort.

Important Note: Your assignment will be evaluated on "Data1.txt" provided with assignment.

Required Output:

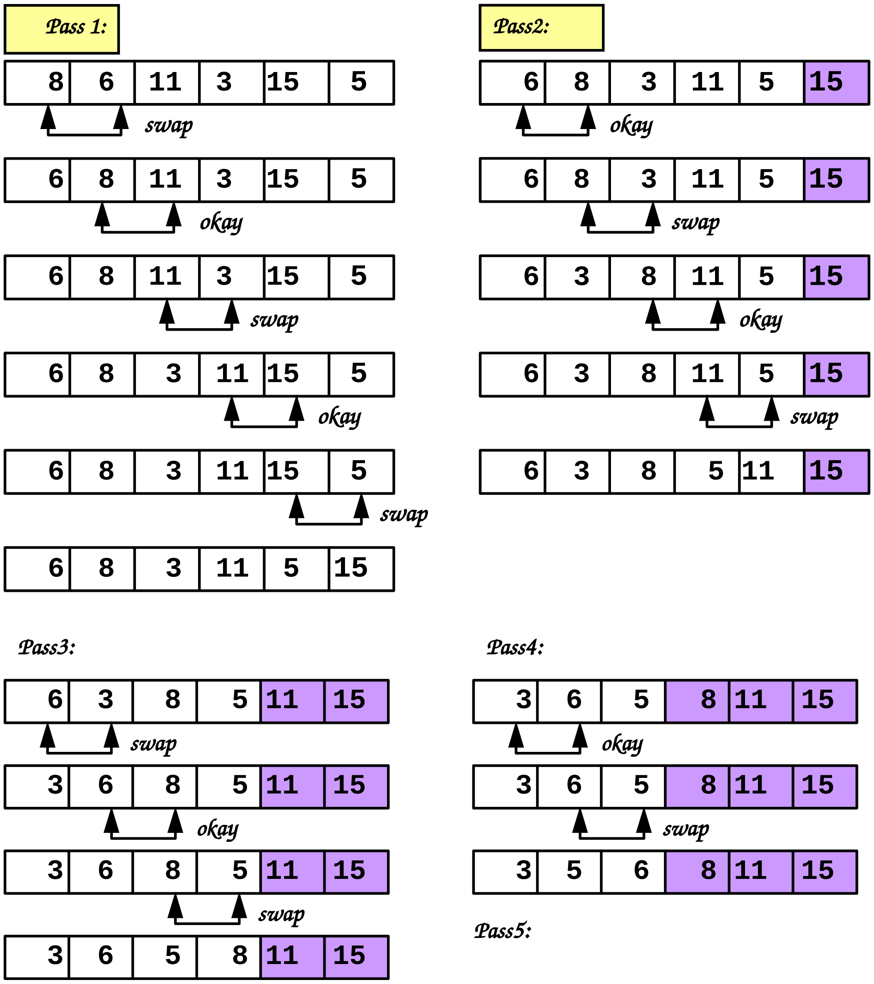
```
Test Case 1:
Array before Sorting: 8, 6, 11, 3, 15, 5
Sorted Array:
3, 5, 6, 8, 11, 15

Test Case 2:
Array before Sorting: 25, 83, 55, 27, 1, 3, 62, 49, 44, 7
Sorted Array:
// Print your sorted array here

Test Case 3:
Error:          Array Size should be greater than zero.
```

Bubble Sort Explanation: A bubble sort starts at the top of the list. Each element is compared to the next. If it is greater than the next element, then swap the two. Pass through the list as many times as necessary to sort it. Usually the number of passes required is equal to (**number of elements –**

1). The smallest value bubbles up to the top of the list while the largest value sinks to the bottom.



Pass 1:

8 | 6 | 11 | 3 | 15 | 5 — *swap*

6 | 8 | 11 | 3 | 15 | 5 — *okay*

6 | 8 | 11 | 3 | 15 | 5 — *swap*

6 | 8 | 3 | 11 | 15 | 5 — *okay*

6 | 8 | 3 | 11 | 15 | 5 — *swap*

6 | 8 | 3 | 11 | 5 | 15

Pass2:

6 | 8 | 3 | 11 | 5 | 15 — *okay*

6 | 8 | 3 | 11 | 5 | 15 — *swap*

6 | 3 | 8 | 11 | 5 | 15 — *okay*

6 | 3 | 8 | 11 | 5 | 15 — *swap*

6 | 3 | 8 | 5 | 11 | 15

Pass3:

6 | 3 | 8 | 5 | 11 | 15 — *swap*

3 | 6 | 8 | 5 | 11 | 15 — *okay*

3 | 6 | 8 | 5 | 11 | 15 — *swap*

3 | 6 | 5 | 8 | 11 | 15

Pass4:

3 | 6 | 5 | 8 | 11 | 15 — *okay*

3 | 6 | 5 | 8 | 11 | 15 — *swap*

3 | 5 | 6 | 8 | 11 | 15

Pass5:

Scan 5 will not do swapping and the algorithm terminates after this pass.

Exercise 2 [Merge Arrays]:

Write a program that takes two sorted arrays and merges the arrays in sorted order. You are not allowed to use any sorting algorithm. Your program will take data from “Data2.txt” (file provided with assignment description).

```
Test Case  1:
Array 1: 2, 5, 9
Array 2: 1, 2, 3, 6

Merged Array:
1, 2, 2, 3, 5, 6, 9


Test  Case 2:
Array 1: 1, 2, 4, 6
Array 2: 2, 5, 8

Merged Array:
1, 2, 2, 4, 5, 6, 8

Test  Case 3:
Array 1: 2, 5, 9, 14, 19
Array 2: 6, 8

Merged Array:
2, 5, 6, 8, 9, 14, 19
```

Important Note:
Violation of following will 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.
- Using Global Variables and “goto” in assignment is **NOT ALLOWED**.

Marks Distribution for Exercise 1 (Tentative):

Requirements	Marks
Executable file (.exe)	2
User Interface (Printing Input and Output)	2 + 2
Sorted List Output	10
Program’s behavior on list size <= 0 (Display proper error message)	2

Dynamic Allocation	2
Memory Deallocation	7
Separate Input Function	5
Separate Output Function	5
Separate Bubble Sort Function	2
Coding Standards / Readability:	
Variable Names	2
Comments	2
Other Coding Standards	2
Total Marks	**45**

Submit both .cpp and .exe file with names YourRollNumber.cpp and Your RollNo.exe, for example, L151234.cpp.

Final Output required on console after implementing functions:

```
---------------------------------------Bubble Sort Result----------------------------------------

Test Case 1:
Array before Sorting: 8, 6, 11, 3, 15, 5
Sorted Array:
3, 5, 6, 8, 11, 15

Test Case 2:
Array before Sorting: 25, 83, 55, 27, 1, 3, 62, 49, 44, 7
Sorted Array:
// Print your sorted array here

Test Case 3:
Error:          Array Size should be greater than zero.

---------------------------------------Merge                                         Arrays
Result---------------------------------------
Test Case  1:
Array 1: 2, 5, 9
Array 2: 1, 2, 3, 6

Merged Array:
1, 2, 2, 3, 5, 6, 9


Test  Case 2:
Array 1: 1, 2, 4, 6
Array 2: 2, 5, 8

Merged Array:
```

```
1, 2, 2, 4, 5, 6, 8

Test  Case 3:
Array 1: 2, 5, 9, 14, 19
Array 2: 6, 8

Merged Array:
2, 5, 6, 8, 9, 14, 19
```

Note: We are not taking ANYTHING from user.