|  |  |
| --- | --- |
| **Name** |  |
| **CMS ID** |  |
| **Date** |  |

**Objectives:**

After completing this Lab students will able to

1. Understand the concept and usage of Bubble sort, selection sort, insertion sort and Quick sort.
2. Implement Bubble sort, selection sort, Quick sort and insertion sort using array and linked lists.
3. Analyze the complexity of Bubble sort, selection sort, Quick sort and insertion sort for different cases.

For revision of the concepts of Bubble sort, selection sort and insertion sort, please refer to the lectures sent. Call me anytime if you need any help. Here are the tasks that you need to perform

1. Implement three functions named BubbleSort, InsSort, QSort and SelSort that takes a float array as parameter and
   1. Sort it using the sorting techniques.
   2. Those functions should also return the value of N where N is the number of comparison done to complete the process.
   3. You should call these functions in main function and get the number of comparison. After getting the value of N for all three sorting algorithms, you should then display the value of N for all algorithms and analyze it manually.
   4. You should analyze the value of N of all algorithms for the best case and worst case as given below  
      **Best Case:**

{1, 2, 3, 4, 5}

**Worst Case:**

{5, 4, 3, 2, 1}

* 1. The txt file named Lab 7 “All permutations.txt” have declaration and initialization of a 2D matrix named “Array\_Permutations” that have all possible permutations of the array given above, you are supposed to modify your code such that it sends each row of Array\_Permutations to all the three implemented functions, calculate catch the value of N for all algorithms and store it into three arrays Complexity\_Bub[120], Complexity\_Sel[120], Complexity\_Qui[120] and Complexity\_Ins[120]. You should then display the value of all three matrix in such format   
     Complexity\_Bub[0] Complexity\_Sel[0] Complexity\_Ins[0] Complexity\_Qui[0]

Complexity\_Bub[1] Complexity\_Sel[1] Complexity\_Ins[1] Complexity\_Qui[1]

Complexity\_Bub[2] Complexity\_Sel[2] Complexity\_Ins[2] Complexity\_Qui[2]

Complexity\_Bub[3] Complexity\_Sel[3] Complexity\_Ins[3] Complexity\_Qui[3]

.  
.  
.

* 1. You should also get the average of Complexity\_Bub, Complexity\_Qui, Complexity\_Sel and Complexity\_Ins and display them at the end.

**Optional task**

Implement Bubble sort, insertion sort and selection sort algorithms using linked list. You don’t need to calculate the complexity this time