

LAB 05

Miss Atika Islam

### ****Basic Implementation of Bubble Sort and Insertion Sort****

* **Problem**: Write two functions, one for bubbleSort and one for insertionSort. Each function should take an array of integers and sort it in ascending order.
* **Input**: An array of integers, e.g., {5, 2, 9, 1, 5, 6}
* **Output**: Sorted array, e.g., {1, 2, 5, 5, 6, 9}
* **Challenge**: Implement both algorithms in-place, without using additional arrays.
* **Problem**: Modify the bubble sort and insertion sort algorithms to sort an array in **descending order** instead of ascending.
* **Input**: An array of integers, e.g., {3, 1, 4, 1, 5, 9}
* **Output**: Sorted array in descending order, e.g., {9, 5, 4, 3, 1, 1}
* **Hint**: In both algorithms, change the comparison condition to reverse the order.
* **Problem**: Modify the bubble sort algorithm to count the number of swaps it performs while sorting an array.
* **Input**: An array of integers, e.g., {5, 3, 2, 1}
* **Output**: Sorted array and the number of swaps, e.g., Sorted array: {1, 2, 3, 5}, Number of swaps: 5
* **Hint**: Increment a counter each time a swap operation occurs.

* **Problem**: Optimize insertion sort by using **binary search** to find the correct position for each element being inserted. Implement this modified insertion sort.
* **Input**: An array of integers, e.g., {7, 3, 8, 2, 9}
* **Output**: Sorted array, e.g., {2, 3, 7, 8, 9}
* **Hint**: Use binary search in the sorted part of the array to find the insertion point, reducing the number of comparisons in each insertion step.