TABLE OF CONTENTS

TEAM INFORMATION

Member 1:

Name: Moaaz Mohamed Salama

ID: 23012085

Member 2:

Name: Moamen Wael Mohammed Mohammed Sultan

ID: 23012104

Member 3:

Name: Noureldin Ashraf Ahmed Ahmed Moustafa

ID: 23012107

COMPARISON

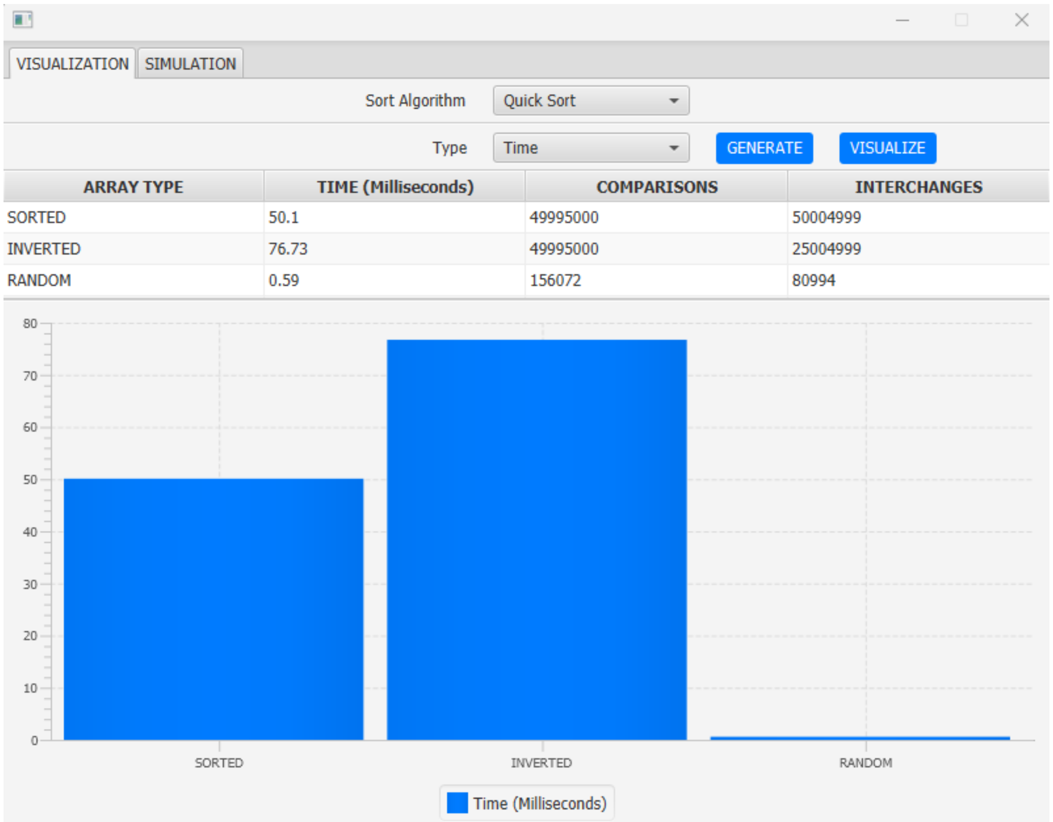
QUICK SORT:

Time Complexity:

SORTED	INVERTED	RANDOM
$O(n^2)$	$O(n^2)$	$O(n\log n)$

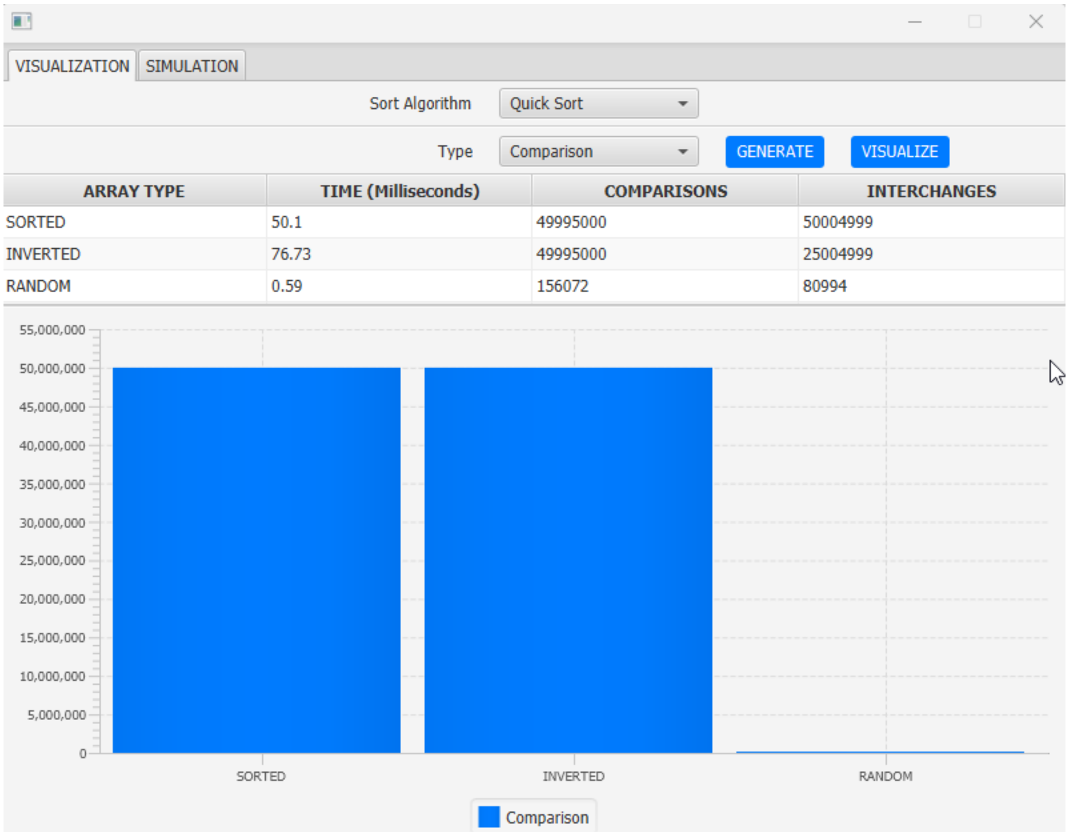
Details of a sample:

(GRAPH OF TIME IN MILLISECONDS)



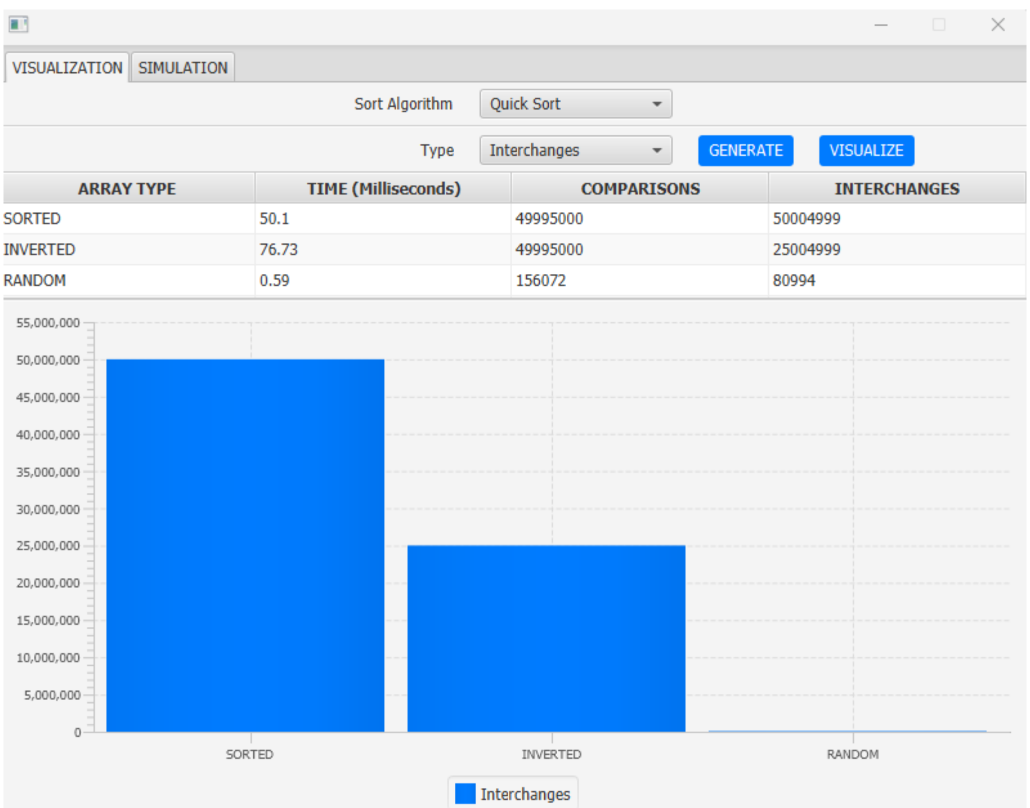
ARRAY TYPE	TIME (Milliseconds)	COMPARISONS	INTERCHANGES
SORTED	50.1	49995000	50004999
INVERTED	76.73	49995000	25004999
RANDOM	0.59	156072	80994

(GRAPH OF COMPARISONS)



ARRAY TYPE	TIME (Milliseconds)	COMPARISONS	INTERCHANGES
SORTED	50.1	49995000	50004999
INVERTED	76.73	49995000	25004999
RANDOM	0.59	156072	80994

(GRAPH OF INTERCHANGES)



ARRAY TYPE	TIME (Milliseconds)	COMPARISONS	INTERCHANGES
SORTED	50.1	49995000	50004999
INVERTED	76.73	49995000	25004999
RANDOM	0.59	156072	80994

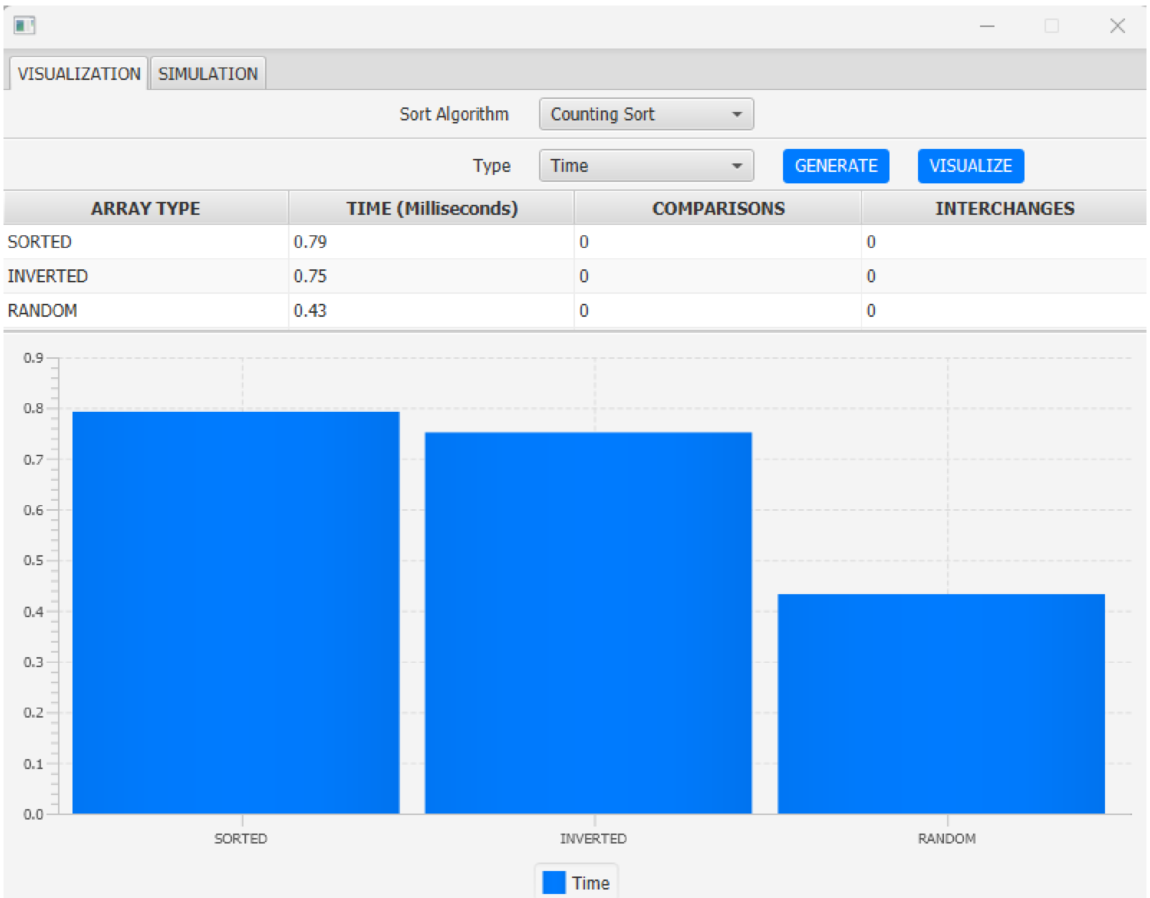
COUNTING SORT:

Time Complexity:

SORTED	INVERTED	RANDOM
$O(n + k)$	$O(n + k)$	$O(n + k)$

Details of a sample: [ONLY TIME GRAPH IS SHOWN, OTHERS DO NOT CONTAIN A GRAPH]

(GRAPH OF TIME IN MILLISECONDS)



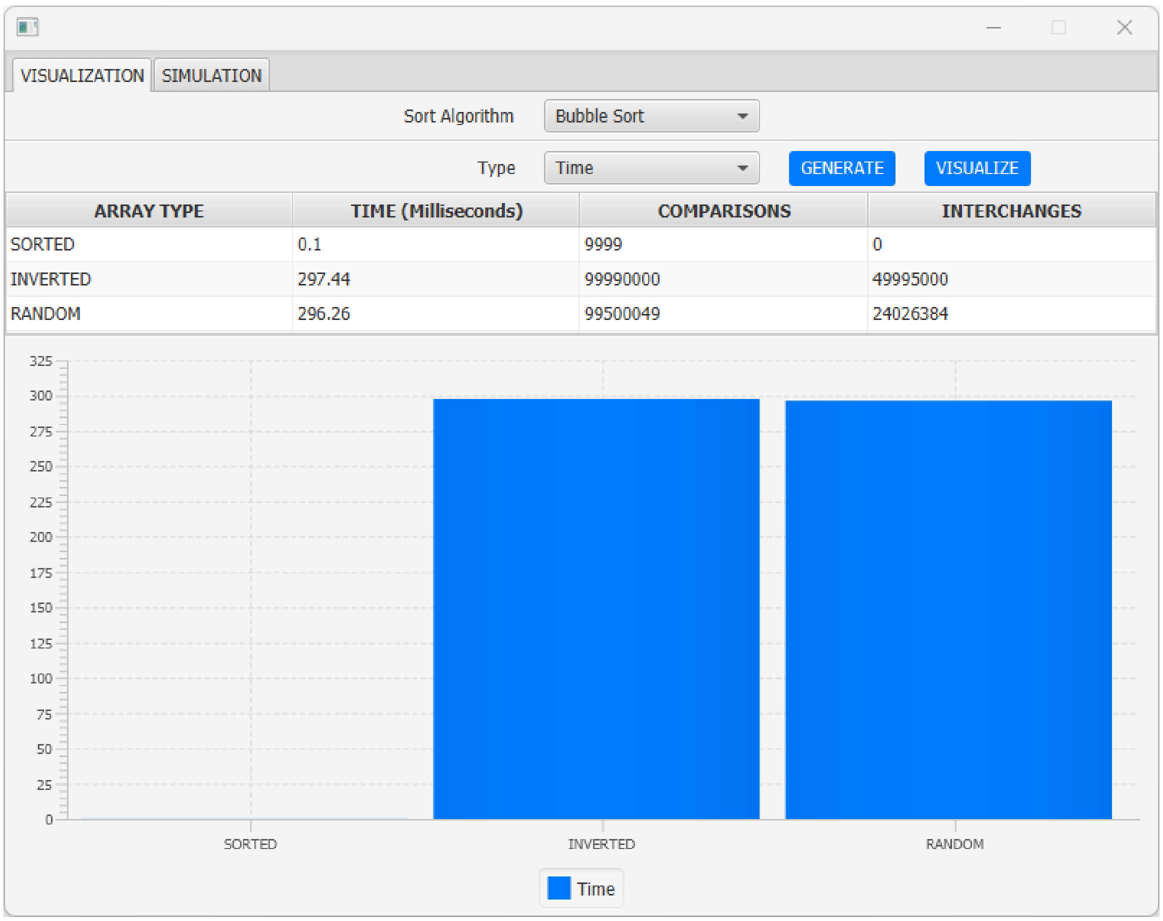
BUBBLE SORT:

Time Complexity:

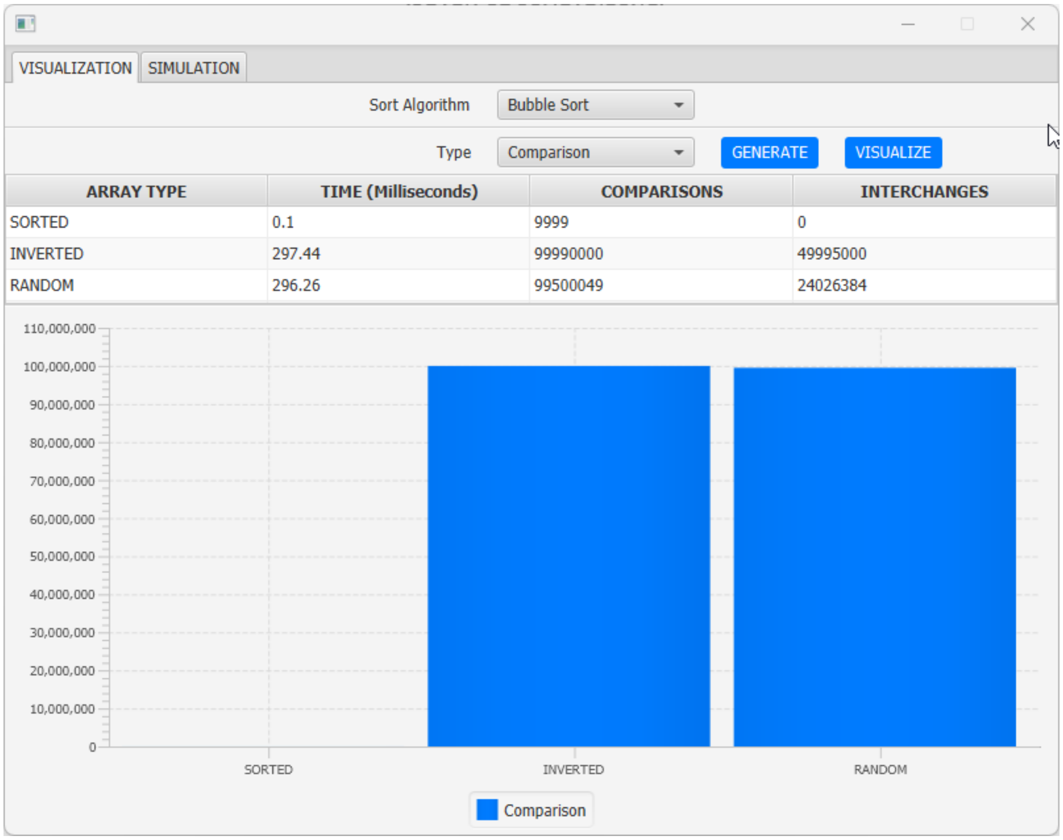
SORTED	INVERTED	RANDOM
$O(n)$	$O(n^2)$	$O(n^2)$

Details of a sample:

(GRAPH OF TIME IN MILLISECONDS)

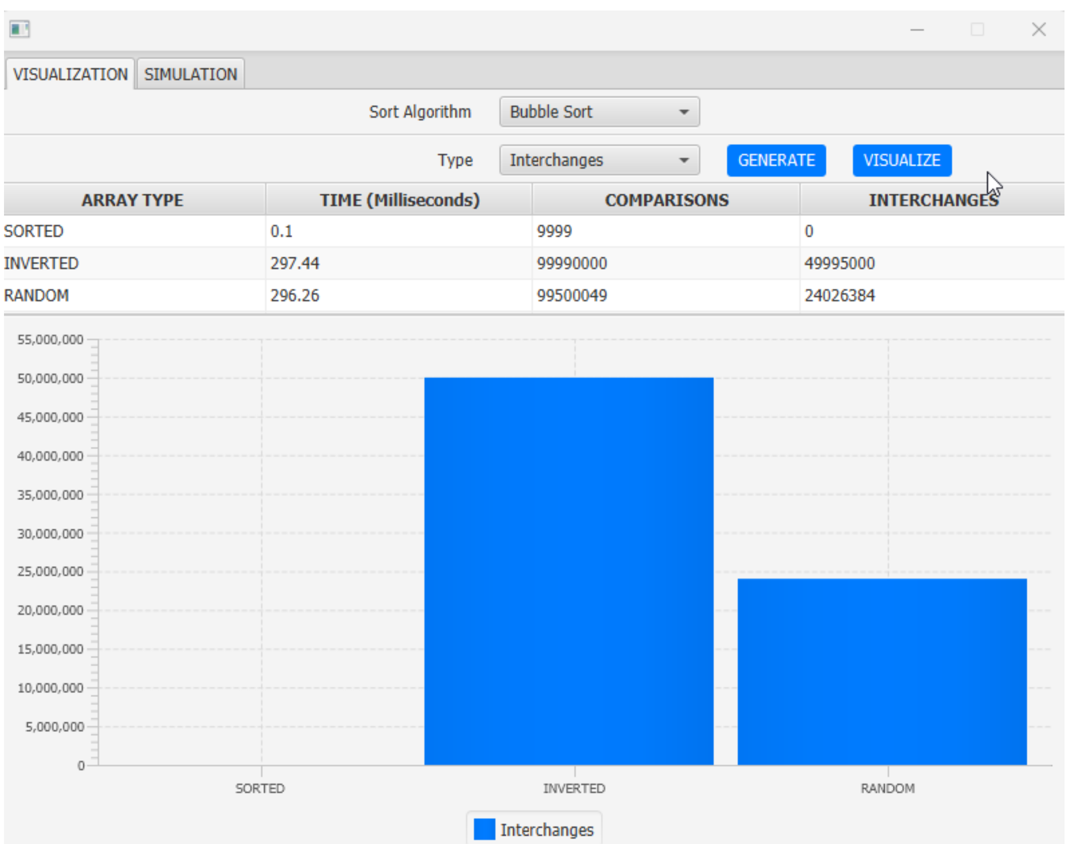


(GRAPH OF COMPARISONS)



ARRAY TYPE	TIME (Milliseconds)	COMPARISONS	INTERCHANGES
SORTED	0.1	9999	0
INVERTED	297.44	99990000	49995000
RANDOM	296.26	99500049	24026384

(GRAPH OF INTERCHANGES)



ARRAY TYPE	TIME (Milliseconds)	COMPARISONS	INTERCHANGES
SORTED	0.1	9999	0
INVERTED	297.44	99990000	49995000
RANDOM	296.26	99500049	24026384

EXPLANATION

QUICK SORT:

While this sorting algorithm is named "quick", it is not the quickest.

Quick Sort performs sorting by selecting a pivot element, partitioning the array, and sorting the subarrays. It operates by finding the correct position for the pivot element and moving elements to their correct positions. Method recursion is used to sort the "subarrays" of the array when partitioned.

Quick sort's best case is when pivot selected is optimal $O(n\log(n))$, while its worst case is when array is sorted or inversely sorted $O(n^2)$.

COUNTING SORT:

Out of the other two sorting algorithms, this one does not involve any swaps/interchanges or comparisons.

Counting sort works by finding the maximum element in the array first, then creating a counter array (all elements initialized by 0) then counting the occurrences of each element then increments it in the counter array in the index equivalent to the element. Next, it iterates through the input array in reverse order, placing each element in its correct sorted position in the output array by using the counts stored in the count array.

It is considered the fastest algorithm between Bubble and Quick sort algorithms, however, only if it's an array of positive integers.

Counting sort's worst and best case is $O(n + k)$, where k is counter array's size.

BUBBLE SORT:

Bubble Sort compares elements next to it and swaps their positions to sort the array. It uses two loops to compare all elements and performs interchanges/swaps if need be.

Bubble sort's best case is when the array is already sorted $O(n^2)$, since it has a condition to stop it from running anymore when first loop doesn't perform any interchanges/swaps.

While Bubble sort's average and worst case is when it's inversely sorted or shuffled/random $O(n^2)$.
