Bubble Sort and Insertion Sort Dry Run

Let's consider a random array: $A = [38, 27, 43, 3, 9, 82, 10, 14]$.

Bubble Sort

Initial Array:	$A = [38, 27, 43, 3, 9, 82, 10, 14]$	
First pass:	$[27, 38, 3, 9, 43, 10, 14, 82]$	Comparisons: 7, Swaps: 5
Second pass:	$[27, 3, 9, 38, 10, 14, 43, 82]$	Comparisons: 6, Swaps: 4
Third pass:	$[3, 9, 27, 10, 14, 38, 43, 82]$	Comparisons: 5, Swaps: 3
Fourth pass:	$[3, 9, 10, 14, 27, 38, 43, 82]$	Comparisons: 4, Swaps: 2
Total Comparisons:	22,	Total Swaps:14

Insertion Sort

Initial Array:	$A = [38, 27, 43, 3, 9, 82, 10, 14]$	
First pass:	$[27, 38, 43, 3, 9, 82, 10, 14]$	Comparisons: 1, Swaps: 1
Second pass:	$[27, 38, 43, 3, 9, 82, 10, 14]$	Comparisons: 0, Swaps: 0
Third pass:	$[3, 27, 38, 43, 9, 82, 10, 14]$	Comparisons: 3, Swaps: 3
Fourth pass:	$[3, 9, 27, 38, 43, 82, 10, 14]$	Comparisons: 3, Swaps: 3
Fifth pass:	$[3, 9, 27, 38, 43, 82, 10, 14]$	Comparisons: 0, Swaps: 0
Sixth pass:	$[3, 9, 10, 27, 38, 43, 82, 14]$	Comparisons: 5, Swaps: 5
Seventh pass:	$[3, 9, 10, 14, 27, 38, 43, 82]$	Comparisons: 6, Swaps: 6
Total Comparisons:	18,	Total Swaps:18

Time and Space Complexity Analysis

Bubble Sort

Time Complexity

Bubble Sort has a worst-case and average-case time complexity of $O(n^2)$, where n is the number of elements in the array. This is evident in the provided example as well, where the algorithm needs to go through multiple passes to sort the array, making a total of 22 comparisons. Each pass through the array requires $O(n)$ time, and since there are n passes, the overall time complexity is $O(n^2)$.

Space Complexity

Bubble Sort is an in-place sorting algorithm, which means it doesn't require any additional memory to be allocated for sorting the array. Hence, the space complexity is $O(1)$.

Insertion Sort

Time Complexity

Insertion Sort also has a worst-case and average-case time complexity of $O(n^2)$. In the given example, Insertion Sort made a total of 18 comparisons. Despite the quadratic time complexity, Insertion Sort often performs better than Bubble Sort in practice for small arrays due to smaller hidden constants and the adaptive nature of the algorithm (it makes fewer comparisons and swaps as the array becomes more sorted).

Space Complexity

Similar to Bubble Sort, Insertion Sort is an in-place sorting algorithm. It only requires a constant amount of additional memory space, resulting in a space complexity of $O(1)$.

Conclusion

Through the dry-run example, we observe the quadratic time complexity characteristic of both Bubble and Insertion Sort as the number of comparisons is proportional to the square of the array size. However, the exact number of comparisons and swaps can vary based on the initial order of the elements in the array. Both algorithms have constant space complexity.