1. *Derive the formulas for (i) Number of comparisons, and (ii) Average-case number of swaps for bubble sort*

(i) There are n-1 passes in the bubble sort algorithm. Then, the first pass is n=1 comparisons, second is n-2 and so on until the last pass has only 1 comparison.

Thus, the total number of comparisons are $(n + 1) + (n + 2) + \dots + 1$ which is a sequence that can be written as $\frac{n(n-1)}{2} \Rightarrow O(n^2)$

(ii) The bubble sort algorithm swaps when two adjacent elements in an array are not in order. Now the occurrence of swap depends on a worst case scenario (two adjacent elements are not in order) or best case scenario (two adjacent elements are in order). Hence the probability of a swap occurring is ½. Now the total number of comparisons is $\frac{n(n-1)}{2}$ $(calculated\ above)$

Thus, the average-case number of swaps for bubble sort is $\frac{n(n-1)}{2} \times \frac{1}{2} = \frac{n(n-1)}{4} \Rightarrow O(n^2)$ complexity

4. *Separately plot the results of #comparisons and #swaps by input size, together with appropriate interpolating functions. Discuss your results: do they match your complexity analysis?*

The number of comparisons and swaps grows quadratically with input size nn, aligning with the O(n^2) complexity analysis. The plot shows that the comparisons are always higher than swaps, with swaps being approximately half the number of comparisons. Thus, these experimental results confirm our complexity analysis from question 1.