Bubble Sort

INTUITION: Here the idea is to sort an unordered list of numbers. It can be achieved by performing two simple actions - Comparing two elements and if they don't fit into your sorting criteria, swap them. This is what we exactly do in bubble sort.

EXPLANATION: Now that we have understood the idea behind it, we will understand how bubble sort is actually performed. We basically compare each element with the next one. If the current element is bigger than the next element, we swap them. If the current element is smaller than the next one, we proceed to the next one.

When we reach the end of the array, we go back to the first element and repeat the process and we continue this cycle until our array has been sorted.

8	5	6	4	8 is bigger than 5 so swap
5	8	6	4	8 is bigger than 6 so swap
5	6	8	4	8 is bigger than 4 so swap

REPEAT

5	6	4	8	5 is smaller than 6 so no swap
5	6	4	8	6 is bigger than 4 so swap

5	4	6	8	6 is smaller than 8 so no swap

REPEAT

5	4	6	8	5 is smaller than 4 so swap
4	5	6	8	Array Sorted!

If we observe carefully, we find out that it takes (**length of the array-1**) turns to complete the entire process. From here we come to the time complexity of the array.

TIME COMPLEXITY: If the length of the array be taken as n and we check every element of the array for n-1 times then the total number of the comparisons in the worst case scenario would be n^2-n. For large numbers we tend to ignore n and say that the bubble sort has a time complexity of $O(n^2)$.

If you have reached this far, I hope you enjoyed the explanation and understood the intricacies of bubble sort :)