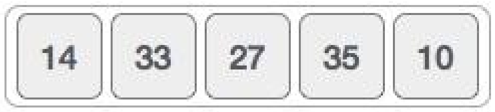
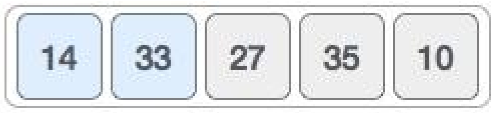
How Bubble Sort Works?

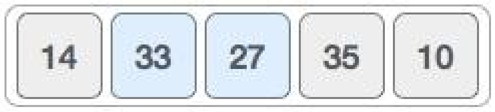
We take an unsorted array for our example. Bubble sort takes $O(n^2)$ time so we're keeping it short and precise.



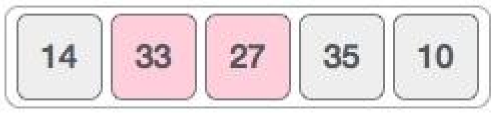
Bubble sort starts with the very first two elements, comparing them to check which one is greater.



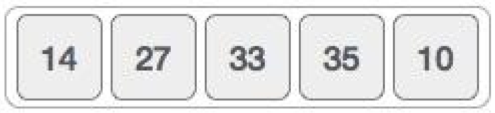
In this case, value 33 is greater than 14, so it is already in sorted locations. Next, we compare 33 with 27.



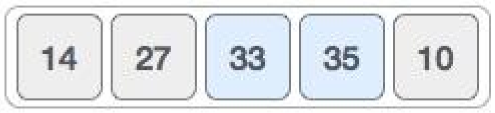
We find that 27 is smaller than 33 and these two values must be swapped.



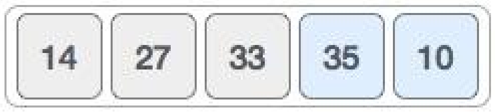
The new array should look like this –



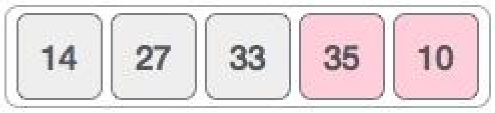
Next we compare 33 and 35. We find that both are in already sorted positions.



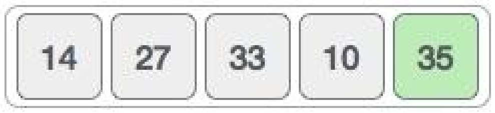
Then we move to the next two values, 35 and 10.



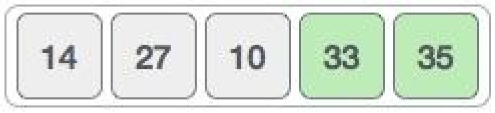
We know then that 10 is smaller 35. Hence they are not sorted.



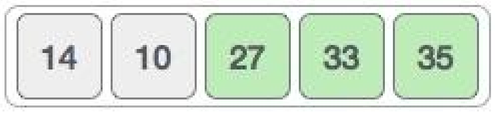
We swap these values. We find that we have reached the end of the array. After one iteration, the array should look like this –



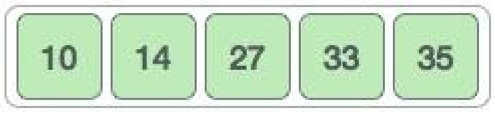
To be precise, we are now showing how an array should look like after each iteration. After the second iteration, it should look like this –



Notice that after each iteration, at least one value moves at the end.



And when there's no swap required, bubble sorts learns that an array is completely sorted.



Now we should look into some practical aspects of bubble sort.

Pseudocode:

```
Function sort(collection)
    Boolean swappableFlag = true;
    Loop as long as swappableFlag is true
        swappableFlag = false
        Loop through the collection where i = currentPosition, n = collection.length - 1
            If value at index i > value at index i+1
            Swap value of i with value of i+1
            swappableFlag = true
            Endif
        End of Loop
    End of Loop
```