## Buble Sort Algorithm

Bubble sort algorithms is one of the most basic sorting algorithms and maybe the least efficient, it has 2 loops the first loop iterations == n and the number of the iterations of the second loop == (n - i ) where is the index of the iteration in the first loop, for example if the size of the array is 10 , then the first loop does 10 iterations and the second nested loop does n -9 , n-8 , n - 7 , n -6 , n-5 , n-4 , n-3, n-2 and n-1 iterations. In case of the ascending , we start by comparing the first element to the second element and if the second element is smaller we swap elements and then we compare second element to third element then third element to fourth element. This guarantees that the largest element will be at the bottom, and the second iteration the second larget item will be in the second element to the bottom. there is an enhancement to the algorithm called the cocktail party, where we once we reach the end of the array , we do a backward pass ( in addition to the foward ) pass to put the smallest item in the top . Also the algorithm check wether the array is sorted and skips if it is sorted see below the pseudo code a la rough garden style

### The Algorithm

**input :** An array of n numbers , in arbitrary order

**output:** An array of the same numbers sorted from smallest to larget

**assumptions :** None

is\_swapped := true, start = 0, end = number\_of\_elements - 1 ,

**while** is\_swapped **do**

**for** forward\_element := start to end **do**

**if** array(forward\_element) > array(forward\_element + 1)

swap forward\_element with foward\_element + 1

is\_swapped := True

**if** is\_swapped := False

reduce end by 1

**for** backward\_element := end -1 to start -1 **do**

**if** array(forward\_element) > array(forward\_element + 1)

swap forward\_element with foward\_element + 1

is\_swapped := True

increase start by 1

### The example

sort 7,9,5,3,4,8,1

**first iteration**

array is 7,9,5,3,4,8,1

the first index of the array is 0 and the last index is 6

in the forward pass the 9 is pushed to the last index of the array, the array becomes 7,5,3,4,8,1,9

the first index of the array is 0 and the last index is 5

in the backward pass the 1 is pushed to the first index of the array the array becomes 1,7,5,3,4,8,9

**Second Iteration**

array is *1*,7,5,3,4,8,*9 , italic element means sorted*

the first index of the array is 1 and the last index is 5

in the forward pass the 7 is pushed to the last index of the array, the array becomes 1,5,3,4,7,8,9

the first index of the array is 1 and the last index is 4

in the backward pass the 3 is pushed to the first index of the array, the array becomes 1,3,5,4,7,8,9

**Third Iteration**

array is *1,3*,5,4,*7,8,9 , italic element means sorted*

the first index of the array is 2 and the last index is 4

in the forward pass the 5 is pushed to the last index of the array, the array becomes 1,3,4,5,7,8,9

the first index of the array is 2 and the last index is 3

nothing to do here

**fourth Iteration**

array is *1,3,4,5,7,8,9 , italic element means sorted*

the first index of the array is 3 and the last index is 3

nothing to do here

swap is set to false

nothing to do here

**We dont enter the fifth Iteration**