# DAY-16

**Sort:**

⇒ Arranging in asc or desc order

**Bubble Sort** → Also known as Sinking Sort or Exchange Sort

**Bubble sort:** Repeatedly swapping the adjacent element if they are in wrong order

**Example:**

1st pass: 3, 1, 5, 4 2

↓

1, 3, 5, 4, 2

↓

1, 3, 4, 5, 2

↓

1, 3, 4, 2, $\boxed{5}$

With first pass through the entire array, the largest element (here, 5) came to the end.

1<3<4

2nd pass: 1, 3, 4, 2, 5

1 2 3

↓

1, 3, 2, $\boxed{4, 5}$ → Need not to compare again

With second pass second largest element comes at second from last index

3rd pass: 1, 3, 2, 4, 5

1 2

↓

$\boxed{1, 2, 3, 4, 5}$ Sorted

**Actual working**

i=0

1st pass

i j

3, 1, 5, 4, 2

0 1 2 3 4

swap ↓

1, 3, 5, 4, 2

i j

swap ↓

1, 3, 4, 5, 2 → Internal Loop [n-1]

j swap

↓

1, 3, 4, 2, 5

0 1 2 3 4

i=1

2nd pass

1, 3, 4, 2, 5

j j j swap

↓

1, 3, 2, $\boxed{4, 5}$

j only check till the elements after that already sorted

i=2

3rd pass

1, 3, 2, 4, 5

j j

Swap

↓

$\boxed{1, 2, 3, 4, 5}$

Sorted array!

Space complexity : $O(1) \rightarrow$ Constant   No extra space required.

Inphase sorting algo.

Time complexity : Best case : $O(N) \rightarrow$ Sorted

Worst case : $O(N^2) \rightarrow$ Sorted in opp

↙

No. of. comparisons

WC: 10,000 ×4 2500

↓ ↓

BC: 100 ×2 50

→ As the size of array is growing, the no of . comparisons also growing.

① Best case → Array is sorted

i=0 first pass

1, 2, 3, 4, 5 → Only once it run, we don't need to check again

i j j j j

NOTE: When j never swaps, arr is sorted.

$$N-1 = \textcircled{N} \Rightarrow O(N) \text{ Best case}$$

② Worst case → Sorting desc to asce order

i=0 1st pass

5, 4, 3, 2, 1

j ↓

4, 5, 3, 2, 1

j ↓

~~4, 5, 2, 4, 1~~

4, 3, 5, 2, 1

↓ j

4, 3, 2, 5, 1

↓ j

Swaps (N-1)

$4, 3, 2, 1, \boxed{5}$

i=1 2nd pass

4, 3, 2, 1, 5

j

3, 4, 2, 1, 5

j

3, 2, 4, 1, 5

Swaps (N-2)

$3, 2, 1, \boxed{4, 5}$

i=2 3rd pass

3, 2, 1, 4, 5

j

2, 3, 1, 4, 5

Swaps(N-3)

j

$2, 1, \boxed{3, 4, 5}$

i=3 4th pass

2, 1, 3, 4, 5

j

Swaps (N-4)

$\boxed{1, 2, 3, 4, 5}$

Array sorted

$$\text{Total} = N-1 + N-2 + N-3 + N-4$$

$$= 4N - (1+2+3+4)$$

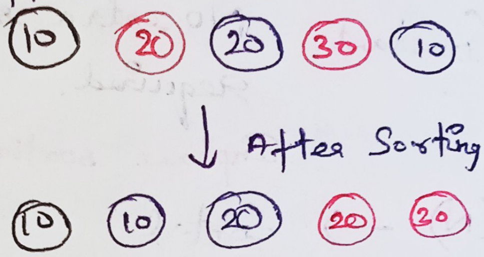
$$= 4N - \left[\frac{N(N+1)}{2}\right] = 4N - \frac{N^2+N}{2}$$

$$= \frac{8N - N^2 + N}{2} = O\left(\frac{7N - N^2}{2}\right) \text{ Const}$$

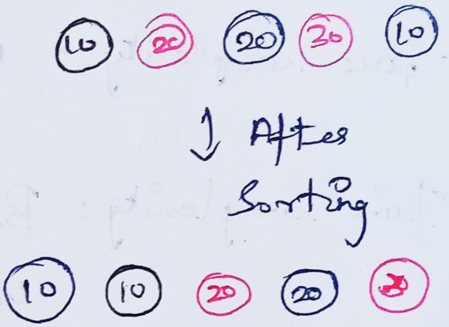
$O(N^2)$ worst case

Diff balls

Stable Sorting Algo:



Unstable sorting Algo:



After sorting the order is maintained.

Order is same when value is same

## Program:

```
import java.util.Arrays;
class bubblesort{
    public static void main(String [] args){
        int [] arr = {3,1,5,4,2}
        bubble(arr);
        System.out.println(Arrays.toString(arr));
    }
    static void bubble(int [] arr){
        boolean swapped;
        for(int i=0; i<arr.length; i++){
            swapped = false;
            for(int j=1; j<arr.length-i; j++){
                if(arr[j] < arr[j-1]){
                    int temp = arr[j];
                    arr[j] = arr[j-1];
                    arr[j-1] = temp;
                    swapped = true;
                }
            }
        }
    }
}
```

o/p:

```
[1,2,3,4,5]
```