17th Feb

16 February 2022 21:01

Bubble Sort :-

T.C => $O(n^2)$, S.C => $O(1)$

0 1 2 3 4

70, 10, 25, 39, 16

0 - 10, 25, 39, 16, 70

1 - 10, 25, 16, 39, 70

2 - 10, 16, 25, 39, 70

3 - 10, 16, 25, 39, 70

```
if(ars[i] > ars[i+1]) {
    swap(ars, i, i+1);
}
```

```
for(i=arr.length-1; i>=1; i--) {
    for(j=0; j<i; j++) {
        if(ars[j] > ar[j+1]) {
            swap();
        }
    }
}
```

0 1 2 3 4 5 6

45, 35, 25, 15, 55, 35, 75

35, 25, 15, 45, 35, 55, 75

0 - 25, 15, 35, 35, 45, 55, 75

1 - 15, 25, 35, 35, 45, 55, 75

2 - 3, 4, 5, 7, 10

i =

j =

```
public static void bubbleSort(int[] arr) {
    for (int i = arr.length - 1; i >= 1; i--) {
        for (int j = 0; j < i; j++) {
            if (arr[j] > arr[j + 1]) swap(arr, j, minIdx: j + 1);
        }
    }
}
```

Insertion Sort

0 1 2 3 4

10, 12, 3, 2, 5

0 - 10, 12, 3, 2, 5

1 - 10, 12, 3, 2, 5

2 - 3, 10, 12, 2, 5

3 - 2, 3, 10, 12, 5

4 - 2, 3, 5, 10, 12

```
for(i=1 to ars.length-1) {
    for(j=i to 1) {
        if(ars[i] < ars[i-1]) {
            swap();
        } else break;
    }
}
```

0 1 2 3 4

7, 8, 9, 10, 67

i = ~~1~~ ~~2~~ ~~3~~ 4

j = 4 ~~3~~ ~~2~~ ~~1~~ 0

T.C => $O(n^2)$, $O(n)$ for Best case

S.C => $O(1)$

```
public static void insertionSort(int[] arr) {
    for (int i = 1; i < arr.length; i++) {
        for (int j = i; j >= 1; j--) {
            if (arr[j] < arr[j - 1]) swap(arr, j, minIdx: j - 1);
            else break;
        }
    }
}
```

0 1 2 3 4

a = 10, 20, 30, 40, 50

i =

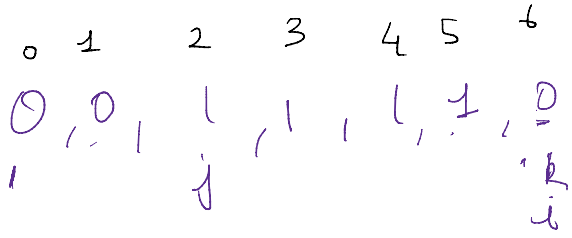
j =

10, 20, 30, 40

40, 30, 20, 10

20, 10, 30, 40

Sort 0 1 2 's - https://leetcode.com/problems/sort-colors/



all elements before j is '0'

j = 0

all elements after k is '2'

k = arr.length - 1

i = 0

T.C => O(n)

```
if (arr[i] == 0)
    swap(i, j)
    j++;
    i++;
else if (arr[i] == 1)
    i++;
else
    swap(i, k);
    k--;
```

H.W => https://leetcode.com/problems/container-with-most-water/

Merge Two Sorted arrays -

nums1[2, 4, 6, 8]

nums2[1, 3, 5, 7]

i = 0
j = 0
k = 0

ans = [

```
while (i < n1.length && j < n2.length) {
    if (nums1[i] < nums2[j]) {
        ans[k] = nums1[i]
        i++;
        k++;
    } else {
        ans[k] = nums2[j]
        j++;
        k++;
    }
}
while (i < n1.length) {
    ans[k] = nums1[i]
    i++;
    k++;
}
while (j < n2.length) {
    ans[k] = nums2[j]
```

```
        ans[k] = nums2[j];
        j++;
    }
    k++;
```