Insertion Sort

array name = a

0	1	2	3	4	5	6
40,	10,	50,	35,	55,	70,	60

i = 1

40

J = i   10   40

~~40~~   ~~10~~

|

(2))   Previous element

while (J > 0 and $a(J) < a(J-1)$)

(1) $10 < 40$

swap(a(J), a(J-1))

J = J-1;

i = 2

0	1	2
10	40	50

J = 2

i = 3

0	1	2	3
		40	
10	~~40~~	~~50~~	~~35~~
	35	~~35~~	50

J = ~~3~~ ~~2~~ 1 $a(J) < a(J-1)$

$35 < 50$

$35 < 40$

$35 < 10$ —— X

0	1	2	3
10	35	40	50

i = 4

0	1	2	3	4
10	35	40	50	55

J = 4

55 < 50 X

i = 5

0	1	2	3	4	5
10	35	40	50	55	70

70 < 55 X

i = 6

0	1	2	3	4	5	6
10	35	40	50	55	~~70~~	~~60~~
					60	70

J = ~~6~~ 5

60 < 70 ✓

60 < 55 X

10, 35, 40, 50, 55, 60, 70

↳ Sorted Array

↳ Insertion sort

Ascending order

0	1	2	3	4
10,	20,	30,	40,	50

10, 20, 30, 40, 50

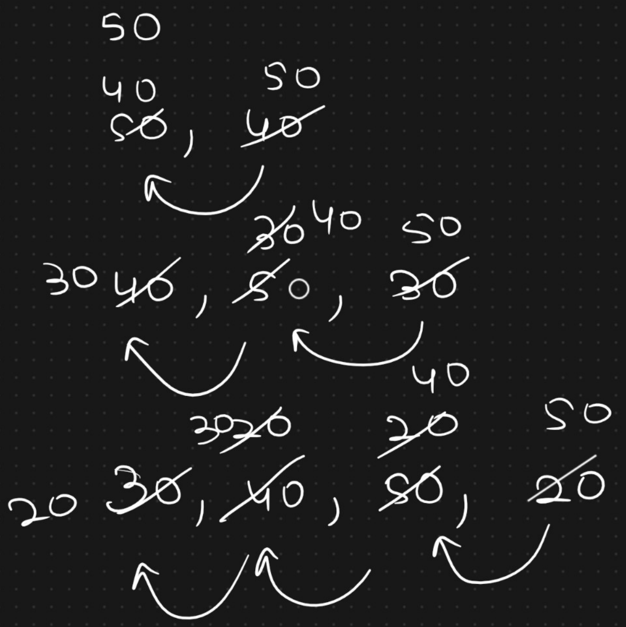
Best case

comparison	swaps
0	0
1	0
1	0
⋮	⋮
1	0
n	0

Total Time complexity = O(n)

Decending order

0	1	2	3	4
50	40	30	20	10



# of elements	comparison	swaps
1	0	0
2	1	1
3	2	2
4	3	3
⋮	⋮	⋮
n	$n-1$	$n-1$

$$\frac{(n-1)n}{2} \qquad \frac{(n-1)n}{2} = \frac{n^2-n}{2}$$

$$\boxed{\text{Total Time complexity} = O(n^2)}$$

Note

already or

→ Array is almost sorted - 10, 15, 25, 40, 5

↳ Insertion Sort

⇓

$O(n)$

↳ Linear Time Complexity

↳ Array is not at all sorted/genuinely sorting

↳ QuickSort/MergeSort

Space Complexity - $O(1)$

Stable Algorithm - Yes

Inplace Algorithm