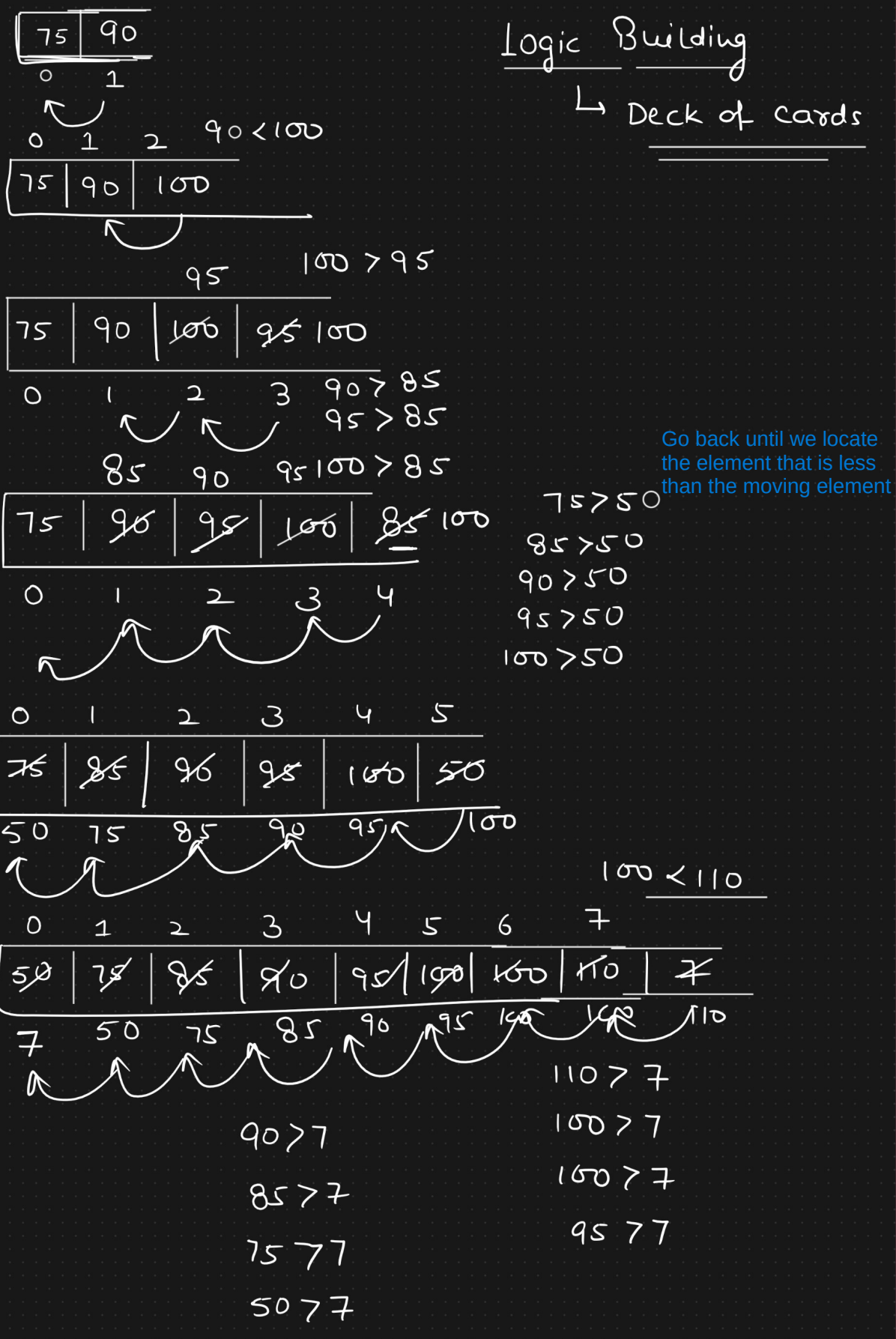
Insertion Sort

75, 90, 100, 95, 85, 50, 100, 110, 7

0 1 2 3 4 5 6 7 8

Logic Building

↳ Deck of cards



Go back until we locate the element that is less than the moving element

7, 50, 75, 85, 90, 95, 100, 100, 110

0 1 2 3 4 5 6 7 8

↳ Sorted array

Best case

Ascending order → n-1 comparisons

→ 0 swaps

10, 20, 30, 40, 50

0 1 2 3 4

n = 5

[10] — No comparison

[10 | 20] 10<20 ①

[10 | 20 | 30] 20<30 ②

[10 | 20 | 30 | 40] 30<40 ③

[10 | 20 | 30 | 40 | 50] 40<50 ④

n → (n-1) comparisons = O(n)

→ 0 swaps

$T(n) = O(n)$

This is the Time complexity of Insertion sort in the best case scenario that is when the array is already sorted.

Please note that in case of sorting of an array Time complexity is measure of time taken for comparisons and time taken for swapping.

Worst case

↳ Decending order

$n = 5$

50, 40, 30, 20, 10

0 1 2 3 4

n elements

	C (Comparison)	S (Swaps)
50 (0)	1	1
50>40	2	2
40 50 (0 1)	3	3
50>30	4	4
40>30	1	1
30 40 50	n-1	n-1

50>20
40>20
30>20

20 30 40 50

50>10
40>10
30>10
20>10

10 20 30 40 50

comparisons → $1+2+3+---+n-1$

$$\Rightarrow \frac{(n-1)\,n}{2} = O(n^2)$$

Swaps

↳ $1+2+3+\ldots+n-1$

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

$$T(n) = O(n^2)$$

Worst case Time Complexity

Best case → Ascending order

(Almost sorted → ascending order)

INSERTION SORT ⟹

$\Downarrow$

$O(n)$

Worst case → Descending order

$\Downarrow$

$O(n^2)$