* Longest Increasing Subsequence

(5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

$O(n^2)$ / $O(n)$ from the beginning to end, and loop from beginning to (index - 1)

fill dp from (max+1) s.t. arr[i] < arr[index]

take the max of dp array.

- (5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9) arr

dp (max, max, -, -, -, -, -, -, -, -, -)

↑↑

binary search for index & fill that index with arr value

(5, 6, 7, max, max, ---)

(5, 6, 7, 1, 2, 8 ---)

do binary search for index, 0-3

then

(1, 6, 7, max, max ---)

(5, 6, 7, 1, 2, 8 ---)

(1, 2, 7, max, max, max, max ---)

(5, 6, 7, 1, 2, 8, 3 ---)

(1, 2, 7, 8, max, max ---)

(- - - -)

from end to begin s.t. arr[i] ≠ max, return.

- use variable to indicate the last position ~~with~~ filled in dp.

do binary search from 0 - lastposition

return last position at the end.

- Inplace change arr without using dp array. Use last index organized to indicate binary searching range.

No: Date: __/__/____

(5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

1 → (5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

2 → (5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

3 → (5, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

4 → (1, 6, 7, 1, 2, 8, 3, 4, 0, 5, 9)

5 → (1, 2, 7, 1, 2, 8, 3, 4, 0, 5, 9)

6 → (1, 2, 7, 8, 2, 8, 3, 4, 0, 5, 9)

7 → (1, 2, 3, 4, 2, 8, 3, 4, 0, 5, 9)

8 → (0, 2, 3, 4, 2, 8, 3, 4, 0, 5, 9)

9 → (0, 2, 3, 4, 5, 8, 3, 4, 0, 5, 9)

10 → (0, 2, 3, 4, 5, 9, 3, 4, 0, 5, 9)

max length