가장 긴 증가하는 부분수열,
LIS (Longest Increasing Subsequence)

A = { 10, 20, 10, 30, 40, 50 }

※ 부분수열 최대 길이 담을 배열준비 DP[A.length+1]

	0	1	2	3	4	5
A	10	20	10	30	40	50

	0	1	2	3	4	5	6
DP	0	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~	~~0~~
		1	2	1	3	~~2~~	~~2~~
						~~3~~ 4	~~3~~ 4 5

※ DP 배열크기 같아도 OK
· 계산시 최소 최대값 = 1
 (나 자신!)

DP[0] = 1
묶음 1로 데이터 1 지금
까지 시작!!

i j
0 0
1 0   DP[1]=1   A[1] > A[0] && DP[1] <= DP[0]
                 true DP[1] = DP[0]+1

2 0   DP[2]=1   A[2] > A[0] && DP[2] <= DP[0]
                 false
2 1   A[2] > A[1] && DP[2] <= DP[1]
                 false → 현재 가능값 이전의 값들 보다 크면 true

3 0   DP[3]=1   A[3] > A[0] && DP[3] <= DP[0]
                 true DP[3] = DP[0]+1 = 2   → 현재값 작다하면 더하면
3 1   A[3] > A[1] && DP[3] <= DP[1]
                 true DP[3] = DP[1]+1 = 3 → 현재 DP[3] 바꾸면
3 2   A[3] > A[2] && DP[3] <= DP[2]
                 false   현재 DP[3] 보다 미만이면 PASS.

4 0   DP[4]=1   A[4] > A[0] && DP[4] <= DP[0]
                 true DP[4] = DP[0]+1 = 1
4 1   A[4] > A[1] && DP[4] <= DP[1]
                 true DP[4] = DP[1]+1 = 2

4 2   A[4] > A[2] && DP[4] <= DP[2]
       false
← 3   A[4] > A[3] && DP[4] <= DP[3]
       true DP[4] = DP[3] +1 = 4

(?) 배열 맨끝에 있는게 과연 최대값인가? NO

보기)   A = {1, 4, 5, 2}

A | 1 | 4 | 5 | 2 |
DP | 1 | 2 | 3 | 2 |

찾는방법)   int max = 0;
for)   for (int num : DP) {

              max = Math.max (max, num);
        }