Question: https://leetcode.com/problems/longest-increasing-subsequence/

So the idea is to maintain a array which stores longest increasing subsequence possible from that position.

For last position it will always be 1.

Now check if ith element is smaller from any element to its right or not, if small then it can add to a increasing subsequence.

So we check the longest subsequence which can be created from that position.

Now we find out the max value from the array which was storing longest increasing subsequence from that position.

Code:  
class Solution {

public int lengthOfLIS(int[] nums) {

int[] len = new int[nums.length];

int highest = 0;

len[nums.length-1] = 1;

for(int i=nums.length-2;i>=0;i--){

len[i]=1;

int j=i+1;

while(j<nums.length){

if(nums[i]<nums[j]){

len[i]=Math.max(len[i],1+len[j]);

}

j++;

}

}

for(int i=0; i<nums.length; i++){

highest=Math.max(highest, len[i]);

}

return highest;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>