[Strictly increasing array](https://csacademy.com/contest/round-61/task/strictly-increasing-array/) [Calculate minimum number of moves to make the whole array strictly increasing, numbers can be negative]

Solution

Suppose the problem was simply named **Non-decreasing Array**. Obviously you want to keep as many number as possible unchanged. It follows that you can select the **longest non-decreasing subsequence** and modify all the other elements so that they'll respect the rule (by making them equal to values from the subsequence we've extracted accordingly). Back to our problem now, we're looking for a **Stricly** Increasing Array. The usual trick to go back to our original and easier problem is to simply solve it for *V*​*i*​ ′​​ = *V*​*i*​​−*i*.

In this way, for some *i* < *j*, if *V*​*i*​′​​ ≤ *V*​*j*​′​​, then *V*​*i*​​+*J*−*I* ≤ *V*​*j*​​, which means we have enough space to fit a stricly increasing array between *i* and *j*.

**Same Problem but now the elements cannot be negative**

Solution

For each element of modified sequence we have bi >= i (1<=i<=n) and we need to find the longest increasing subsequence of the original sequence ai and ai >= i. We keep such ai unchanged and other values can be changed into the value as their index. If we make another ci = ai - i, the answer is the equal to (n - the longest of the non-decreasing subsequence with no negative number) and for longest non-decreasing subsequence , a O(nlogn) solution can be used.

[Sonya and problem with a legend [Div 1C]](http://codeforces.com/problemset/problem/713/C)