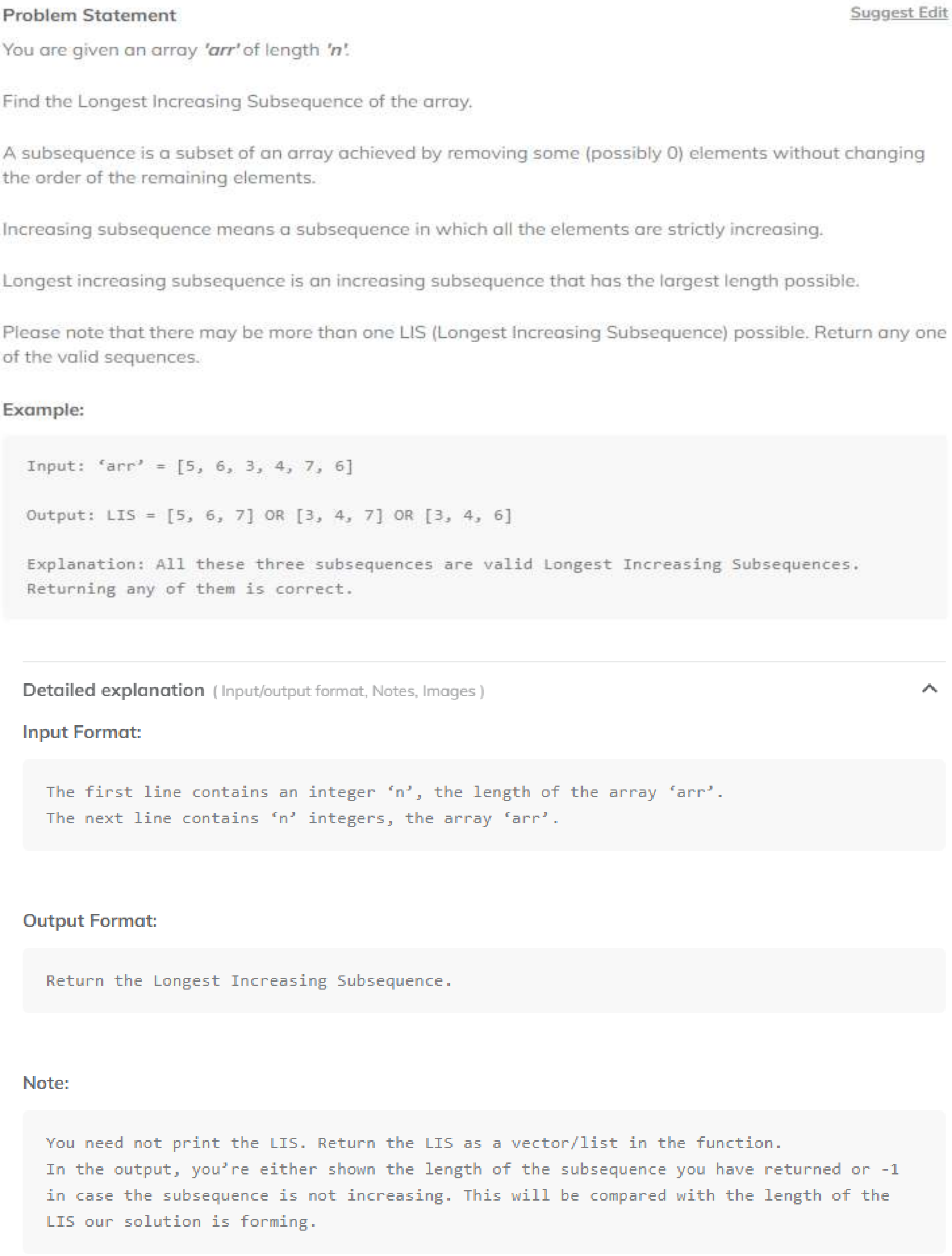
Printing Longest Increasing Subsequence



Problem Statement

Suggest Edit

You are given an array *'arr'* of length *'n'*.

Find the Longest Increasing Subsequence of the array.

A subsequence is a subset of an array achieved by removing some (possibly 0) elements without changing the order of the remaining elements.

Increasing subsequence means a subsequence in which all the elements are strictly increasing.

Longest increasing subsequence is an increasing subsequence that has the largest length possible.

Please note that there may be more than one LIS (Longest Increasing Subsequence) possible. Return any one of the valid sequences.

Example:

```
Input: ‘arr’ = [5, 6, 3, 4, 7, 6]

Output: LIS = [5, 6, 7] OR [3, 4, 7] OR [3, 4, 6]

Explanation: All these three subsequences are valid Longest Increasing Subsequences.
Returning any of them is correct.
```

Detailed explanation (Input/output format, Notes, Images)

Input Format:

```
The first line contains an integer ‘n’, the length of the array ‘arr’.
The next line contains ‘n’ integers, the array ‘arr’.
```

Output Format:

```
Return the Longest Increasing Subsequence.
```

Note:

```
You need not print the LIS. Return the LIS as a vector/list in the function.
In the output, you’re either shown the length of the subsequence you have returned or -1
in case the subsequence is not increasing. This will be compared with the length of the
LIS our solution is forming.
```

Sample Input 1:

```
6
5 6 3 4 7 6
```

Sample Output 1:

```
3
```

Explanation Of Sample Input 1 :

```
There are three valid LIS for the given array: [5, 6, 7], [3, 4, 7] OR [3, 4, 6], and
their length is 3.
```

Sample Input 2 :

```
5
1 2 3 4 5
```

Sample Output 2 :

```
5
```

Explanation Of Sample Input 2 :

```
There is only one valid LIS for the array: [1, 2, 3, 4, 5], and its length is 5.
```

Expected time complexity:

```
The expected time complexity is O(‘n’ ^ 2).
```

Constraints:

```
1 <= ‘n’ <= 500
1 <= ‘arr[i]’ <= 10^5
```