

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1Solution 2

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 // O(n^2) time | O(n) space
4 function longestIncreasingSubsequence(array) {
5   const sequences = new Array(array.length);
6   const lengths = array.map(num => 1);
7   let maxLengthIdx = 0;
8   for (let i = 0; i < array.length; i++) {
9     const currentNum = array[i];
10    for (let j = 0; j < i; j++) {
11      const otherNum = array[j];
12      if (otherNum < currentNum && lengths[j] + 1 >= lengths[i]) {
13        lengths[i] = lengths[j] + 1;
14        sequences[i] = j;
15      }
16    }
17    if (lengths[i] >= lengths[maxLengthIdx]) maxLengthIdx = i;
18  }
19  return buildSequence(array, sequences, maxLengthIdx);
20 }
21
22 function buildSequence(array, sequences, currentIdx) {
23   const sequence = [];
24   while (currentIdx !== undefined) {
25     sequence.unshift(array[currentIdx]);
26     currentIdx = sequences[currentIdx];
27   }
28   return sequence;
29 }
30
31 exports.longestIncreasingSubsequence = longestIncreasingSubsequence;
32
```

Solution 1Solution 2Solution 3

```
1 function longestIncreasingSubsequence(array) {
2   // Write your code here.
3 }
4
5 // Do not edit the line below.
6 exports.longestIncreasingSubsequence = longestIncreasingSubsequence;
7
```

Our Tests

Custom Output

Submit Code

1

const longestIncreasingSubsequence = longestIncreasingSubsequence;

2

const arr = [10, 9, 2, 5, 7, 10, 6];

3

const testCases = [

4

[arr, longestIncreasingSubsequence(arr)],

5

[arr, longestIncreasingSubsequence(arr)],

6

];

1

const longestIncreasingSubsequence = longestIncreasingSubsequence;

2

const arr = [10, 9, 2, 5, 7, 10, 6];

3

const testCases = [

4

[arr, longestIncreasingSubsequence(arr)],

5

[arr, longestIncreasingSubsequence(arr)],

6

];

