Our Solution(s) Run Code

Solution 1

```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

// O(n^2) time | O(n) space
function maxSumIncreasingSubsequence(array) {
  const sequences = new Array(array.length);
  const sums = array.map(num => num);
  let maxSumIdx = 0;
  for (let i = 0; i < array.length; i++) {
    const currentNum = array[i];
    for (let j = 0; j < i; j++) {
      const otherNum = array[j];
      if (otherNum < currentNum && sums[j] + currentNum >= sums[i]) {
        sums[i] = sums[j] + currentNum;
        sequences[i] = j;
      }
    }
    if (sums[i] >= sums[maxSumIdx]) maxSumIdx = i;
  }
  return [sums[maxSumIdx], buildSequence(array, sequences, maxSumIdx)]
}

function buildSequence(array, sequences, currentIdx) {
  const sequence = [];
  while (currentIdx !== undefined) {
    sequence.unshift(array[currentIdx]);
    currentIdx = sequences[currentIdx];
  }
  return sequence;
}

exports.maxSumIncreasingSubsequence = maxSumIncreasingSubsequence;

```

Your Solutions Run Code

Solution 1 Solution 2 Solution 3

```
function maxSumIncreasingSubsequence(array) {
  // Write your code here.
}

// Do not edit the line below.
exports.maxSumIncreasingSubsequence = maxSumIncreasingSubsequence;

```

Our Tests

Custom Output Submit Code

Run or submit code when you're ready.