Algorithm for Longest Increasing Subsequence.

1. Initialize an array **dp** of the same length as the input array **nums** to store the LIS length for each element. Initialize **maxLength** variable to keep track of the maximum LIS length.
2. Iterate over each element in the input array **nums** from left to right.
3. For each element at index **i**, initialize **dp[i]** to 1, as the minimum LIS length for each element is 1.
4. Iterate over all previous elements at indices **j** from 0 to **i-1**.
5. For each previous element at index **j**, check if the current element **nums[i]** is greater than **nums[j]** and if the LIS length ending at index **i** is less than the LIS length ending at index **j** plus one (**dp[j] + 1**).
6. If the above condition is true, update **dp[i]** to **dp[j] + 1**.
7. Update **maxLength** with the maximum value between **maxLength** and **dp[i]**.
8. After iterating over all elements, **maxLength** will contain the length of the Longest Increasing Subsequence.
9. Return **maxLength** as the result.