**Algorithm for Longest Increasing Subsequence Number**

1. **Initialization:**
   * Create an array dp of the same length as the input array nums to store the length of the LIS ending at each index. Initialize it with 1, as the minimum length of any LIS is 1.
2. **Dynamic Programming:**
   * Iterate through each element of the input array nums.
   * For each element at index i, iterate through elements before it (indices j from 0 to i-1).
   * If nums[i] is greater than nums[j] and the current LIS length ending at i is less than LIS length ending at j + 1, update dp[i] to dp[j] + 1.
3. **Find the Maximum Length:**
   * Iterate through the dp array to find the maximum LIS length (maxLength) and its ending index (endIndex).
4. **Reconstruct the LIS:**
   * Start from the endIndex and trace back the LIS using the dp array.
   * Add each element to the LIS list in reverse order.
5. **Output:**
   * Print the length of the LIS and the LIS itself.