1403. Minimum Subsequence in Non-Increasing Order

Given the array `nums`, obtain a subsequence of the array whose sum of elements is **strictly greater** than the sum of the non included elements in such subsequence.

If there are multiple solutions, return the subsequence with **minimum size** and if there still exist multiple solutions, return the subsequence with the **maximum total sum** of all its elements. A subsequence of an array can be obtained by erasing some (possibly zero) elements from the array.

Note that the solution with the given constraints is guaranteed to be **unique**. Also return the answer sorted in **non-increasing** order.

Example 1:

Input: nums = [4,3,10,9,8]
Output: [10,9]
Explanation: The subsequences [10,9] and [10,8] are minimal such that the sum of their elements is strictly greater than the sum of elements not included,
however, the subsequence [10,9] has the maximum total sum of its elements.

Example 2:

Input: nums = [4,4,7,6,7]
Output: [7,7,6]
Explanation: The subsequence [7,7] has the sum of its elements equal to 14 which is not strictly greater than the sum of elements not included (14 = 4 + 4 + 6).
Therefore, the subsequence [7,6,7] is the minimal satisfying the conditions.
Note the subsequence has to returned in non-decreasing order.

Example 3:

Input: nums = [6]
Output: [6]

```
def minSubsequence(self, nums: List[int]) -> List[int]:
        nums.sort()
        Sum = sum(nums)
        x = 0
        temp = []
        for i in range(len(nums)-1,-1,-1):
```

```
            x += nums[i]
            temp.append(nums[i])
            if x > Sum - x:
                return temp
```