Problem # 1673: Find the most competitive subsequence (Medium)

<https://leetcode.com/problems/find-the-most-competitive-subsequence/>

Solution:  
  
Runtime beats 78.17%

1. Let n be the length of nums.

2. Number of attempts = n – k

3. Initialize a stack.

4. Iterate through nums array.

While stack is not empty, and top of the stack is greater than the current element num, and number of attempts is greater than 0, then pop the stack and decrement number of attempts.

5. Add the current elements in nums to the stack.

6. Return the first k elements of the stack.

class Solution:

def mostCompetitive(self, nums: List[int], k: int) -> List[int]:

n = len(nums)

num\_attempts = n - k

stack = []

for num in nums:

while stack and stack[-1] > num and num\_attempts > 0:

stack.pop()

num\_attempts -= 1

stack.append(num)

return stack[:k]