











Solution S

Given an integer array nums sorted in **non-decreasing order**, remove the duplicates **in-place** such that each unique element appears only **once**. The **relative order** of the elements should be kept the **same**.

□ Discuss (999+)

Submissions

Since it is impossible to change the length of the array in some languages, you must instead have the result be placed in the **first part** of the array nums. More formally, if there are k elements after removing the duplicates, then the first k elements of nums should hold the final result. It does not matter what you leave beyond the first k elements.

Return k after placing the final result in the first k slots of nums.

Do **not** allocate extra space for another array. You must do this by **modifying the input array in-place** with O(1) extra memory.

Custom Judge:

Description

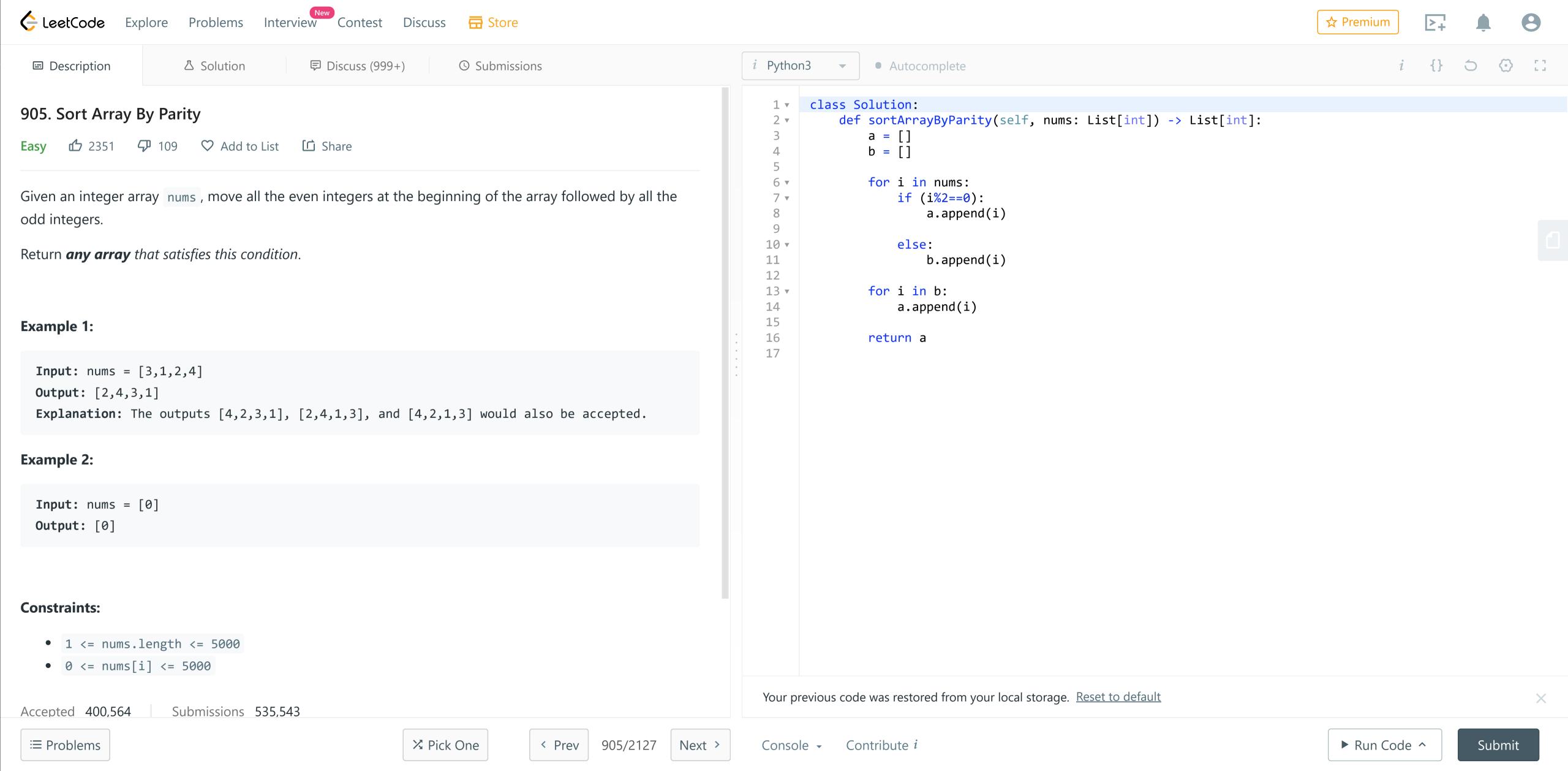
The judge will test your solution with the following code:

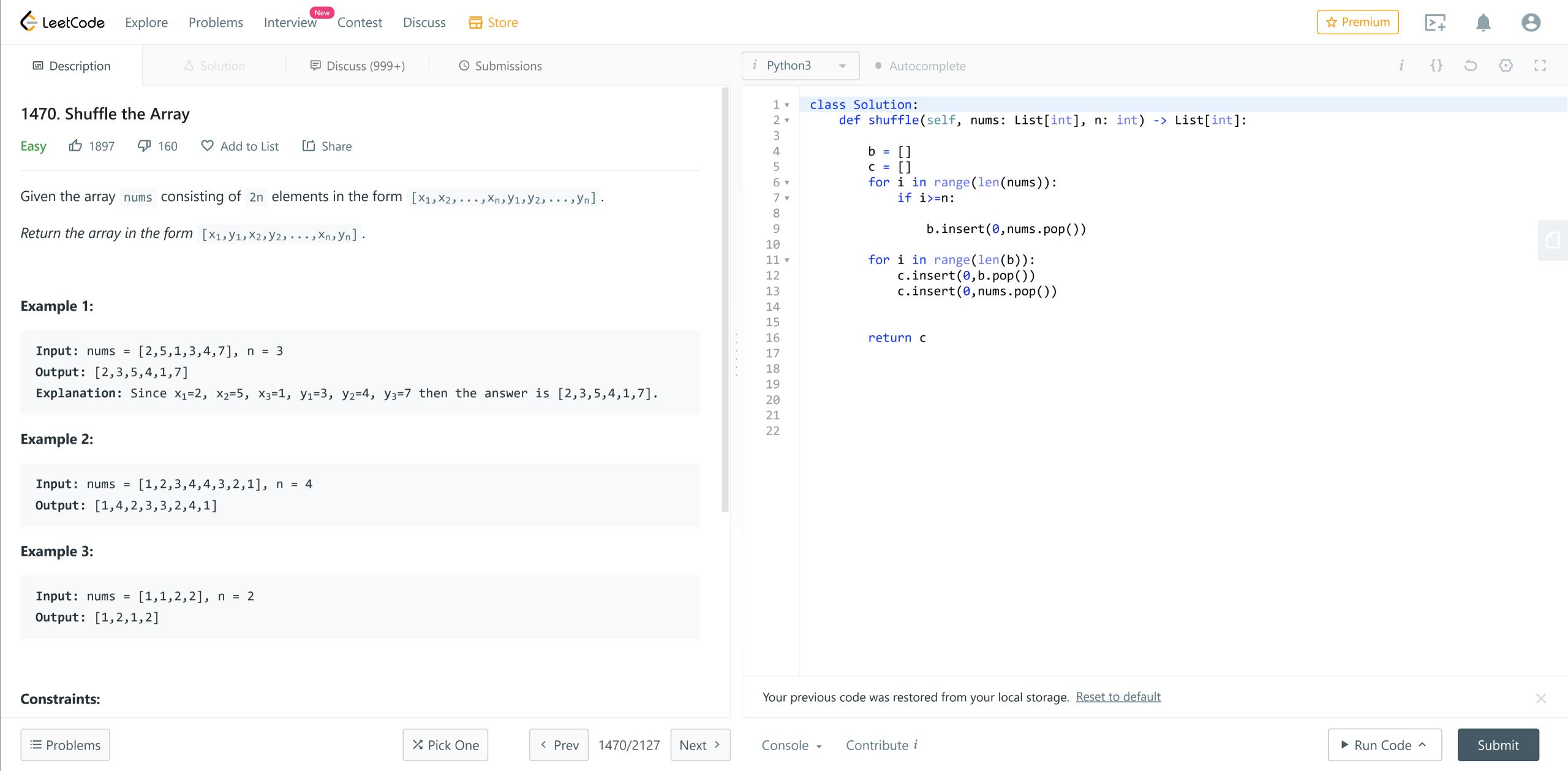
```
int[] nums = [...]; // Input array
int[] expectedNums = [...]; // The expected answer with correct length

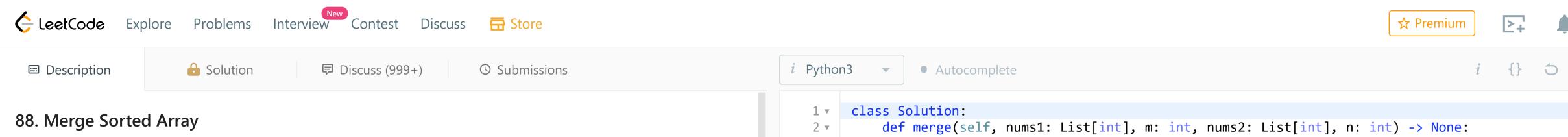
int k = removeDuplicates(nums); // Calls your implementation

assert k == expectedNums.length;
for (int i = 0; i < k; i++) {
    assert nums[i] == expectedNums[i];
}</pre>
```

```
i Python3
                                                                                            i {} 5
                                                                                                          \bigcirc
                 Autocomplete
        class Solution:
            def removeDuplicates(self, nums: List[int]) -> int:
                # for empty list return 0
                if len(nums) == 0:
   4 ▼
                     return 0
                 # point the index to the first element
                # for each element in nums check if it is equal to what index points to
                # if it's not equal, increment index and assign the value to that index
   9
                 index = 0
  10 ▼
                 for element in nums:
                    if nums[index] != element:
  11 ▼
  12
                         index += 1
                         nums[index] = element
  13
  14
                # return unique elements count
  15
                 return index + 1
  16
  17
  18
  19
  20
  21
  22
 Your previous code was restored from your local storage. Reset to default
```







You are given two integer arrays nums1 and nums2, sorted in **non-decreasing order**, and two integers m and n, representing the number of elements in nums1 and nums2 respectively.

Merge nums1 and nums2 into a single array sorted in non-decreasing order.

△ 2632 **¬** 264 **¬** Add to List **□** Share

The final sorted array should not be returned by the function, but instead be *stored inside the array* nums1. To accommodate this, nums1 has a length of m + n, where the first m elements denote the elements that should be merged, and the last n elements are set to 0 and should be ignored. nums2 has a length of n.

Example 1:

```
Input: nums1 = [1,2,3,0,0,0], m = 3, nums2 = [2,5,6], n = 3
Output: [1,2,2,3,5,6]
Explanation: The arrays we are merging are [1,2,3] and [2,5,6].
The result of the merge is [1,2,2,3,5,6] with the underlined elements coming from nums1.
```

Example 2:

```
Input: nums1 = [1], m = 1, nums2 = [], n = 0
Output: [1]
Explanation: The arrays we are merging are [1] and [].
The result of the merge is [1].
```

```
Do not return anything, modify nums1 in-place instead.
              for i in range(m,len(nums1)):
 7 ▼
                  nums1.pop()
 9
              for i in range(n,len(nums2)):
10 ▼
                  nums2.pop()
11
12
              for i in nums2:
13 ▼
                  nums1.append(i)
14
15
              nums1.sort()
16
17
18
19
20
21
```

Your previous code was restored from your local storage. Reset to default

≡ Problems ×

➢ Pick One

< Prev

88/2127 Next >

Console - Contribute i

► Run Code ^

Submit

① []

