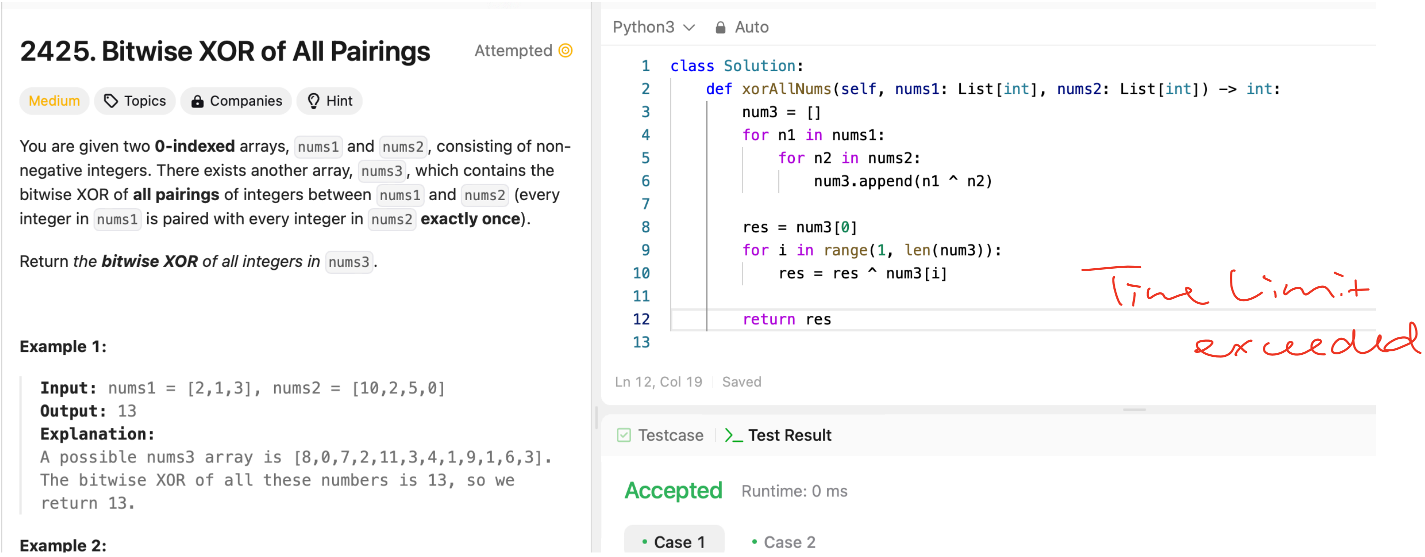


2425. Bitwise XOR of All Pairings

You are given two **0-indexed** arrays, `nums1` and `nums2`, consisting of non-negative integers. There exists another array, `nums3`, which contains the bitwise XOR of **all pairings** of integers between `nums1` and `nums2` (every integer in `nums1` is paired with every integer in `nums2` **exactly once**).

Return *the **bitwise XOR** of all integers in* `nums3`.

Example 1:

```
Input: nums1 = [2,1,3], nums2 = [10,2,5,0]
Output: 13
Explanation:
A possible nums3 array is [8,0,7,2,11,3,4,1,9,1,6,3].
The bitwise XOR of all these numbers is 13, so we
return 13.
```

Example 2:

```
class Solution:
    def xorAllNums(self, nums1: List[int], nums2: List[int]) -> int:
        num3 = []
        for n1 in nums1:
            for n2 in nums2:
                num3.append(n1 ^ n2)

        res = num3[0]
        for i in range(1, len(num3)):
            res = res ^ num3[i]

        return res

```

Time Limit exceeded

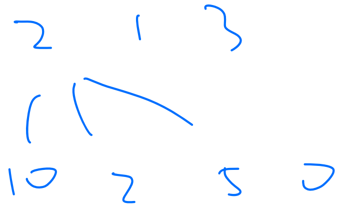
Accepted Runtime: 0 ms

A	B	A^B
1	1	0
0	0	0
1	0	1
0	1	1

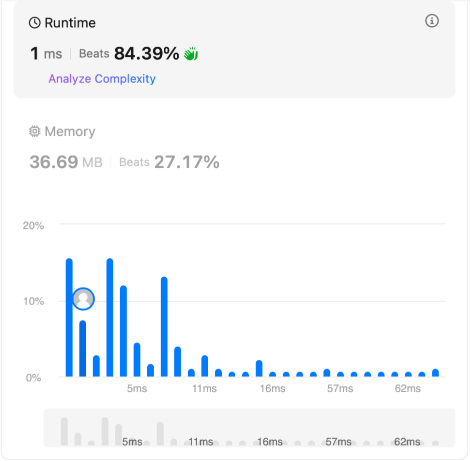
$$(A\hat{}B)\hat{}C = (A\hat{}C)\hat{}B$$

if len(nums1) % 2 == 0 :

nums2 element will be canceled out



vise versa



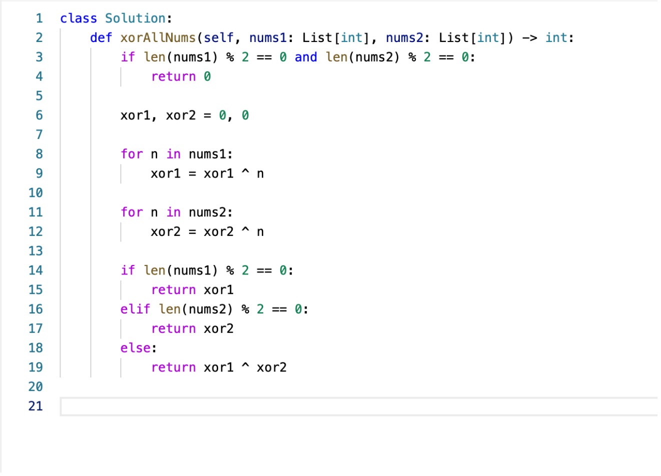
Runtime

1 ms | Beats 84.39%

Analyze Complexity

Memory

36.69 MB | Beats 27.17%



```
class Solution:
    def xorAllNums(self, nums1: List[int], nums2: List[int]) -> int:
        if len(nums1) % 2 == 0 and len(nums2) % 2 == 0:
            return 0

        xor1, xor2 = 0, 0

        for n in nums1:
            xor1 = xor1 ^ n

        for n in nums2:
            xor2 = xor2 ^ n

        if len(nums1) % 2 == 0:
            return xor1
        elif len(nums2) % 2 == 0:
            return xor2
        else:
            return xor1 ^ xor2

```