Problem 1879: Minimum XOR Sum of Two Arrays

Problem Information

Difficulty: Hard
Acceptance Rate: 49.96%
Paid Only: No
Tags: Array, Dynamic Programming, Bit Manipulation, Bitmask

Problem Description

You are given two integer arrays `nums1` and `nums2` of length `n`.

The **XOR sum** of the two integer arrays is `(nums1[0] XOR nums2[0]) + (nums1[1] XOR nums2[1]) + ... + (nums1[n - 1] XOR nums2[n - 1])` (**0-indexed**).

* For example, the **XOR sum** of `[1,2,3]` and `[3,2,1]` is equal to `(1 XOR 3) + (2 XOR 2) + (3 XOR 1) = 2 + 0 + 2 = 4`.

Rearrange the elements of `nums2` such that the resulting **XOR sum** is **minimized**.

Return _the**XOR sum** after the rearrangement_.

Example 1:

Input: nums1 = [1,2], nums2 = [2,3] **Output:** 2 **Explanation:** Rearrange nums2 so that it becomes [3,2]. The XOR sum is (1 XOR 3) + (2 XOR 2) = 2 + 0 = 2.

Example 2:

Input: nums1 = [1,0,3], nums2 = [5,3,4] **Output:** 8 **Explanation:** Rearrange nums2 so that it becomes [5,4,3]. The XOR sum is (1 XOR 5) + (0 XOR 4) + (3 XOR 3) = 4 + 4 + 0 = 8.

Constraints:

* `n == nums1.length` * `n == nums2.length` * `1 <= n <= 14` * `0 <= nums1[i], nums2[i] <= 107`

Code Snippets

C++:

```
class Solution {
public:
int minimumXORSum(vector<int>& nums1, vector<int>& nums2) {

}
};
```

Java:

```
class Solution {
public int minimumXORSum(int[] nums1, int[] nums2) {

}
}
```

Python3:

```
class Solution:
def minimumXORSum(self, nums1: List[int], nums2: List[int]) -> int:
```