Problem 3215: Count Triplets with Even XOR Set Bits II

Problem Information

Difficulty: Medium
Acceptance Rate: 60.56%
Paid Only: Yes
Tags: Array, Bit Manipulation

Problem Description

Given three integer arrays `a`, `b`, and `c`, return the number of triplets `(a[i], b[j], c[k])`, such that the bitwise `XOR` between the elements of each triplet has an **even** number of set bits.

Example 1:

Input: a = [1], b = [2], c = [3]

Output: 1

Explanation:

The only triplet is `(a[0], b[0], c[0])` and their `XOR` is: `1 XOR 2 XOR 3 = 002`.

Example 2:

Input: a = [1,1], b = [2,3], c = [1,5]

Output: 4

Explanation:

Consider these four triplets:

* `(a[0], b[1], c[0])`: `1 XOR 3 XOR 1 = 0112` * `(a[1], b[1], c[0])`: `1 XOR 3 XOR 1 = 0112` * `(a[0], b[0], c[1])`: `1 XOR 2 XOR 5 = 1102` * `(a[1], b[0], c[1])`: `1 XOR 2 XOR 5 = 1102`

Constraints:

* `1 <= a.length, b.length, c.length <= 105` * `0 <= a[i], b[i], c[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
long long tripletCount(vector<int>& a, vector<int>& b, vector<int>& c) {

}
};
```

Java:

```
class Solution {
public long tripletCount(int[] a, int[] b, int[] c) {

}
}
```

Python3:

```
class Solution:
def tripletCount(self, a: List[int], b: List[int], c: List[int]) -> int:
```