Problem 3513: Number of Unique XOR Triplets I

Problem Information

Difficulty: Medium
Acceptance Rate: 25.91%
Paid Only: No
Tags: Array, Math, Bit Manipulation

Problem Description

You are given an integer array `nums` of length `n`, where `nums` is a **permutation** of the numbers in the range `[1, n]`.

A **XOR triplet** is defined as the XOR of three elements `nums[i] XOR nums[j] XOR nums[k]` where `i <= j <= k`.

Return the number of **unique** XOR triplet values from all possible triplets `(i, j, k)`.

Example 1:

Input: nums = [1,2]

Output: 2

Explanation:

The possible XOR triplet values are:

* `(0, 0, 0) -> 1 XOR 1 XOR 1 = 1` * `(0, 0, 1) -> 1 XOR 1 XOR 2 = 2` * `(0, 1, 1) -> 1 XOR 2 XOR 2 = 1` * `(1, 1, 1) -> 2 XOR 2 XOR 2 = 2`

The unique XOR values are `{1, 2}`, so the output is 2.

Example 2:

Input: nums = [3,1,2]

Output: 4

Explanation:

The possible XOR triplet values include:

* `(0, 0, 0) -> 3 XOR 3 XOR 3 = 3` * `(0, 0, 1) -> 3 XOR 3 XOR 1 = 1` * `(0, 0, 2) -> 3 XOR 3 XOR 2 = 2` * `(0, 1, 2) -> 3 XOR 1 XOR 2 = 0`

The unique XOR values are `{0, 1, 2, 3}`, so the output is 4.

Constraints:

* `1 <= n == nums.length <= 105` * `1 <= nums[i] <= n` * `nums` is a permutation of integers from `1` to `n`.

Code Snippets

C++:

```
class Solution {
public:
int uniqueXorTriplets(vector<int>& nums) {

}
};
```

Java:

```
class Solution {
public int uniqueXorTriplets(int[] nums) {

}
}
```

Python3:

```
class Solution:
def uniqueXorTriplets(self, nums: List[int]) -> int:
```