Problem 3719: Longest Balanced Subarray I

Problem Information

Difficulty: Medium
Acceptance Rate: 52.14%
Paid Only: No
Tags: Array, Hash Table, Divide and Conquer, Segment Tree, Prefix Sum

Problem Description

You are given an integer array `nums`.

A **subarray** is called **balanced** if the number of **distinct even** numbers in the subarray is equal to the number of **distinct odd** numbers.

Return the length of the **longest** balanced subarray.

Example 1:

Input: nums = [2,5,4,3]

Output: 4

Explanation:

* The longest balanced subarray is `[2, 5, 4, 3]`. * It has 2 distinct even numbers `[2, 4]` and 2 distinct odd numbers `[5, 3]`. Thus, the answer is 4.

Example 2:

Input: nums = [3,2,2,5,4]

Output: 5

Explanation:

* The longest balanced subarray is `[3, 2, 2, 5, 4]`. * It has 2 distinct even numbers `[2, 4]` and 2 distinct odd numbers `[3, 5]`. Thus, the answer is 5.

Example 3:

Input: nums = [1,2,3,2]

Output: 3

Explanation:

* The longest balanced subarray is `[2, 3, 2]`. * It has 1 distinct even number `[2]` and 1 distinct odd number `[3]`. Thus, the answer is 3.

Constraints:

* `1 <= nums.length <= 1500` * `1 <= nums[i] <= 105`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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