Problem 2170: Minimum Operations to Make the Array Alternating

Problem Information

Difficulty: Medium
Acceptance Rate: 35.03%
Paid Only: No
Tags: Array, Hash Table, Greedy, Counting

Problem Description

You are given a **0-indexed** array `nums` consisting of `n` positive integers.

The array `nums` is called **alternating** if:

* `nums[i - 2] == nums[i]`, where `2 <= i <= n - 1`. * `nums[i - 1] != nums[i]`, where `1 <= i <= n - 1`.

In one **operation** , you can choose an index `i` and **change** `nums[i]` into **any** positive integer.

Return _the**minimum number of operations** required to make the array alternating_.

Example 1:

Input: nums = [3,1,3,2,4,3] **Output:** 3 **Explanation:** One way to make the array alternating is by converting it to [3,1,3,_**1**_ ,_**3**_ ,_**1**_]. The number of operations required in this case is 3. It can be proven that it is not possible to make the array alternating in less than 3 operations.

Example 2:

Input: nums = [1,2,2,2,2] **Output:** 2 **Explanation:** One way to make the array alternating is by converting it to [1,2,_**1**_ ,2,_**1**_]. The number of operations required in this case is 2. Note that the array cannot be converted to [_**2**_ ,2,2,2,2] because in this case nums[0] == nums[1] which violates the conditions of an alternating array.

Constraints:

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

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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