Problem 3201: Find the Maximum Length of Valid Subsequence I

Problem Information

Difficulty: Medium
Acceptance Rate: 54.86%
Paid Only: No
Tags: Array, Dynamic Programming

Problem Description

You are given an integer array `nums`.

A subsequence `sub` of `nums` with length `x` is called **valid** if it satisfies:

* `(sub[0] + sub[1]) % 2 == (sub[1] + sub[2]) % 2 == ... == (sub[x - 2] + sub[x - 1]) % 2.`

Return the length of the **longest** **valid** subsequence of `nums`.

A **subsequence** is an array that can be derived from another array by deleting some or no elements without changing the order of the remaining elements.

Example 1:

Input: nums = [1,2,3,4]

Output: 4

Explanation:

The longest valid subsequence is `[1, 2, 3, 4]`.

Example 2:

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

Output: 6

Explanation:

The longest valid subsequence is `[1, 2, 1, 2, 1, 2]`.

Example 3:

Input: nums = [1,3]

Output: 2

Explanation:

The longest valid subsequence is `[1, 3]`.

Constraints:

* `2 <= nums.length <= 2 * 105` * `1 <= nums[i] <= 107`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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