Problem 3708: Longest Fibonacci Subarray

Problem Information

Difficulty: Medium
Acceptance Rate: 69.30%
Paid Only: No
Tags: Array

Problem Description

You are given an array of **positive** integers `nums`.

A **Fibonacci** array is a contiguous sequence whose third and subsequent terms each equal the sum of the two preceding terms.

Return the length of the longest **Fibonacci** **subarray** in `nums`.

Note: Subarrays of length 1 or 2 are always **Fibonacci**.

Example 1:

Input: nums = [1,1,1,1,2,3,5,1]

Output: 5

Explanation:

The longest Fibonacci subarray is `nums[2..6] = [1, 1, 2, 3, 5]`.

`[1, 1, 2, 3, 5]` is Fibonacci because `1 + 1 = 2`, `1 + 2 = 3`, and `2 + 3 = 5`.

Example 2:

Input: nums = [5,2,7,9,16]

Output: 5

Explanation:

The longest Fibonacci subarray is `nums[0..4] = [5, 2, 7, 9, 16]`.

`[5, 2, 7, 9, 16]` is Fibonacci because `5 + 2 = 7`, `2 + 7 = 9`, and `7 + 9 = 16`.

Example 3:

Input: nums = [1000000000,1000000000,1000000000]

Output: 2

Explanation:

The longest Fibonacci subarray is `nums[1..2] = [1000000000, 1000000000]`.

`[1000000000, 1000000000]` is Fibonacci because its length is 2.

Constraints:

* `3 <= nums.length <= 105` * `1 <= nums[i] <= 109`

Code Snippets

C++:

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

}
};
```

Java:

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

```
}
}
```

Python3:

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