Problem 3400: Maximum Number of Matching Indices After Right Shifts

Problem Information

Difficulty: Medium
Acceptance Rate: 84.22%
Paid Only: Yes
Tags: Array, Two Pointers, Simulation

Problem Description

You are given two integer arrays, `nums1` and `nums2`, of the same length.

An index `i` is considered **matching** if `nums1[i] == nums2[i]`.

Return the **maximum** number of **matching** indices after performing any number of **right shifts** on `nums1`.

A **right shift** is defined as shifting the element at index `i` to index `(i + 1) % n`, for all indices.

Example 1:

Input: nums1 = [3,1,2,3,1,2], nums2 = [1,2,3,1,2,3]

Output: 6

Explanation:

If we right shift `nums1` 2 times, it becomes `[1, 2, 3, 1, 2, 3]`. Every index matches, so the output is 6.

Example 2:

Input: nums1 = [1,4,2,5,3,1], nums2 = [2,3,1,2,4,6]

Output: 3

Explanation:

If we right shift `nums1` 3 times, it becomes `[5, 3, 1, 1, 4, 2]`. Indices 1, 2, and 4 match, so the output is 3.

Constraints:

* `nums1.length == nums2.length` * `1 <= nums1.length, nums2.length <= 3000` * `1 <= nums1[i], nums2[i] <= 109`

Code Snippets

C++:

```
class Solution {
public:
int maximumMatchingIndices(vector<int>& nums1, vector<int>& nums2) {

}
};
```

Java:

```
class Solution {
public int maximumMatchingIndices(int[] nums1, int[] nums2) {

}
}
```

Python3:

```
class Solution:
def maximumMatchingIndices(self, nums1: List[int], nums2: List[int]) -> int:
```