# Problem 3388: Count Beautiful Splits in an Array

## Problem Information

**Difficulty:** Medium
**Acceptance Rate:** 16.92%
**Paid Only:** No
**Tags:** Array, Dynamic Programming

## Problem Description

You are given an array `nums`.

A split of an array `nums` is **beautiful** if:

1. The array `nums` is split into three subarrays: `nums1`, `nums2`, and `nums3`, such that `nums` can be formed by concatenating `nums1`, `nums2`, and `nums3` in that order. 2. The subarray `nums1` is a prefix of `nums2` **OR** `nums2` is a prefix of `nums3`.

Return the **number of ways** you can make this split.

**Example 1:**

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

**Output:** 2

**Explanation:**

The beautiful splits are:

1. A split with `nums1 = [1]`, `nums2 = [1,2]`, `nums3 = [1]`. 2. A split with `nums1 = [1]`, `nums2 = [1]`, `nums3 = [2,1]`.

**Example 2:**

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

**Output:** 0

**Explanation:**

There are 0 beautiful splits.

**Constraints:**

* `1 <= nums.length <= 5000` * `0 <= nums[i] <= 50`

## Code Snippets

### C++:

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

}
};
```

### Java:

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

}
}
```

### Python3:

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