Problem 1712: Ways to Split Array Into Three Subarrays

Problem Information

Difficulty: Medium
Acceptance Rate: 33.86%
Paid Only: No
Tags: Array, Two Pointers, Binary Search, Prefix Sum

Problem Description

A split of an integer array is **good** if:

* The array is split into three **non-empty** contiguous subarrays - named `left`, `mid`, `right` respectively from left to right. * The sum of the elements in `left` is less than or equal to the sum of the elements in `mid`, and the sum of the elements in `mid` is less than or equal to the sum of the elements in `right`.

Given `nums`, an array of **non-negative** integers, return _the number of**good** ways to split_ `nums`. As the number may be too large, return it **modulo** `109 + 7`.

Example 1:

Input: nums = [1,1,1] **Output:** 1 **Explanation:** The only good way to split nums is [1] [1] [1].

Example 2:

Input: nums = [1,2,2,2,5,0] **Output:** 3 **Explanation:** There are three good ways of splitting nums: [1] [2] [2,2,5,0] [1] [2,2] [2,5,0] [1,2] [2,2] [5,0]

Example 3:

Input: nums = [3,2,1] **Output:** 0 **Explanation:** There is no good way to split nums.

Constraints:

* `3 <= nums.length <= 105` * `0 <= nums[i] <= 104`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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