

Problem 3284: Sum of Consecutive Subarrays

Problem Information

Difficulty: Medium

Acceptance Rate: 42.13%

Paid Only: Yes

Tags: Array, Two Pointers, Dynamic Programming

Problem Description

We call an array `arr` of length `n` **consecutive** if one of the following holds:

* `arr[i] - arr[i - 1] == 1` for $_all_ \ 1 \leq i < n$. * `arr[i] - arr[i - 1] == -1` for $_all_ \ 1 \leq i < n$.

The **value** of an array is the sum of its elements.

For example, `[3, 4, 5]` is a consecutive array of value 12 and `[9, 8]` is another of value 17. While `[3, 4, 3]` and `[8, 6]` are not consecutive.

Given an array of integers `nums`, return the **_sum_** of the **values** of all **consecutive** subarrays.

Since the answer may be very large, return it **modulo** `10⁹ + 7`.

Note that an array of length 1 is also considered consecutive.

Example 1:

Input: nums = [1,2,3]

Output: 20

Explanation:

The consecutive subarrays are: `[1]`, `[2]`, `[3]`, `[1, 2]`, `[2, 3]`, `[1, 2, 3]`. Sum of their values would be: `1 + 2 + 3 + 3 + 5 + 6 = 20`.

Example 2:

Input: nums = [1,3,5,7]

Output: 16

Explanation:

The consecutive subarrays are: `[1]`, `[3]`, `[5]`, `[7]`. Sum of their values would be: `1 + 3 + 5 + 7 = 16`.

Example 3:

Input: nums = [7,6,1,2]

Output: 32

Explanation:

The consecutive subarrays are: `[7]`, `[6]`, `[1]`, `[2]`, `[7, 6]`, `[1, 2]`. Sum of their values would be: `7 + 6 + 1 + 2 + 13 + 3 = 32`.

Constraints:

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

Code Snippets

C++:

```
class Solution {
public:
    int getSum(vector<int>& nums) {
        }
};
```

Java:

```
class Solution {  
    public int getSum(int[] nums) {  
        }  
        }
```

Python3:

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