Problem 2031: Count Subarrays With More Ones Than Zeros

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

You are given a binary array

nums

containing only the integers

0

and

1

. Return

the number of

subarrays

in nums that have

more

1

'

s than

0

's. Since the answer may be very large, return it

modulo

10

9

+ 7

.

A

subarray

is a contiguous sequence of elements within an array.

Example 1:

Input:

nums = [0,1,1,0,1]

Output:

9

Explanation:

The subarrays of size 1 that have more ones than zeros are: [1], [1], [1] The subarrays of size 2 that have more ones than zeros are: [1,1] The subarrays of size 3 that have more ones than zeros are: [0,1,1], [1,1,0], [1,0,1] The subarrays of size 4 that have more ones than zeros are: [1,1,0,1] The subarrays of size 5 that have more ones than zeros are: [0,1,1,0,1]

Example 2:

Input:

nums = [0]

Output:

0

Explanation:

No subarrays have more ones than zeros.

Example 3:

Input:

nums = [1]

Output:

1

Explanation:

The subarrays of size 1 that have more ones than zeros are: [1]

Constraints:

1 <= nums.length <= 10

5

0 <= nums[i] <= 1

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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

Python:

```
class Solution(object):
def subarraysWithMoreOnesThanZeroes(self, nums):
"""
:type nums: List[int]
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number[]} nums
* @return {number}
*/
var subarraysWithMoreOnesThanZeroes = function(nums) {

};
```

TypeScript:

```
function subarraysWithMoreOnesThanZeroes(nums: number[]): number {

};
```

C#:

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

}
}
```

C:

```
int subarraysWithMoreOnesThanZeroes(int* nums, int numsSize) {

}
```

Go:

```
func subarraysWithMoreOnesThanZeroes(nums []int) int {

}
```

Kotlin:

```
class Solution {
fun subarraysWithMoreOnesThanZeroes(nums: IntArray): Int {

}
}
```

Swift:

```
class Solution {
func subarraysWithMoreOnesThanZeroes(_ nums: [Int]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn subarrays_with_more_ones_than_zeroes(nums: Vec<i32>) -> i32 {

}
}
```

Ruby:

```
# @param {Integer[]} nums
# @return {Integer}
def subarrays_with_more_ones_than_zeroes(nums)

end
```

PHP:

```
class Solution {

/**
* @param Integer[] $nums
* @return Integer
*/
function subarraysWithMoreOnesThanZeroes($nums) {

}
}
```

Dart:

```
class Solution {
int subarraysWithMoreOnesThanZeroes(List<int> nums) {

}
}
```

Scala:

```
object Solution {
def subarraysWithMoreOnesThanZeroes(nums: Array[Int]): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec subarrays_with_more_ones_than_zeroes(nums :: [integer]) :: integer
def subarrays_with_more_ones_than_zeroes(nums) do

end
end
```

Erlang:

```
-spec subarrays_with_more_ones_than_zeroes(Nums :: [integer()]) -> integer().
subarrays_with_more_ones_than_zeroes(Nums) ->
.
```

Racket:

```
(define/contract (subarrays-with-more-ones-than-zeroes nums)
(-> (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

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

}
};
```

Java Solution:

```
/**
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

class Solution {
public int subarraysWithMoreOnesThanZeroes(int[] nums) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Subarrays With More Ones Than Zeros
Difficulty: Medium
Tags: array, tree, hash, sort, search

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(h) for recursion stack where h is height
"""

class Solution:
def subarraysWithMoreOnesThanZeroes(self, nums: List[int]) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def subarraysWithMoreOnesThanZeroes(self, nums):
"""
:type nums: List[int]
:rtype: int
```

```
"""
```

JavaScript Solution:

```
/**
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

/**
* @param {number[]} nums
* @return {number}
*/
var subarraysWithMoreOnesThanZeroes = function(nums) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

function subarraysWithMoreOnesThanZeroes(nums: number[]): number {

};
```

C# Solution:

```
/*
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

public class Solution {
public int SubarraysWithMoreOnesThanZeroes(int[] nums) {

}
}
```

C Solution:

```
/*
* Problem: Count Subarrays With More Ones Than Zeros
* Difficulty: Medium
* Tags: array, tree, hash, sort, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(h) for recursion stack where h is height
*/

int subarraysWithMoreOnesThanZeroes(int* nums, int numsSize) {

}
```

Go Solution:

```
// Problem: Count Subarrays With More Ones Than Zeros
// Difficulty: Medium
// Tags: array, tree, hash, sort, search
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height
```

```
func subarraysWithMoreOnesThanZeroes(nums []int) int {

}
```

Kotlin Solution:

```
class Solution {
fun subarraysWithMoreOnesThanZeroes(nums: IntArray): Int {

}
}
```

Swift Solution:

```
class Solution {
func subarraysWithMoreOnesThanZeroes(_ nums: [Int]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Subarrays With More Ones Than Zeros
// Difficulty: Medium
// Tags: array, tree, hash, sort, search
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(h) for recursion stack where h is height

impl Solution {
pub fn subarrays_with_more_ones_than_zeroes(nums: Vec<i32>) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[]} nums
# @return {Integer}
def subarrays_with_more_ones_than_zeroes(nums)
```

```
end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[] $nums
* @return Integer
*/
function subarraysWithMoreOnesThanZeroes($nums) {

}
}
```

Dart Solution:

```
class Solution {
int subarraysWithMoreOnesThanZeroes(List<int> nums) {

}
}
```

Scala Solution:

```
object Solution {
def subarraysWithMoreOnesThanZeroes(nums: Array[Int]): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec subarrays_with_more_ones_than_zeroes(nums :: [integer]) :: integer
def subarrays_with_more_ones_than_zeroes(nums) do

end
end
```

Erlang Solution:

```
-spec subarrays_with_more_ones_than_zeroes(Nums :: [integer()]) -> integer().
subarrays_with_more_ones_than_zeroes(Nums) ->
.
```

Racket Solution:

```
(define/contract (subarrays-with-more-ones-than-zeroes nums)
(-> (listof exact-integer?) exact-integer?)
)
```