Problem 2155: All Divisions With the Highest Score of a Binary Array

Problem Information

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

Problem Description

You are given a

0-indexed

binary array

nums

of length

n

.

nums

can be divided at index

i

(where

0 <= i <= n)

into two arrays (possibly empty)

nums

left

and

nums

right

:

nums

left

has all the elements of

nums

between index

0

and

i - 1

(inclusive)

, while

nums

right

has all the elements of nums between index

i

and

n - 1

(inclusive)

.

If

i == 0

,

nums

left

is

empty

, while

nums

right

has all the elements of

nums

.

If

i == n

,

nums

left

has all the elements of nums, while

nums

right

is

empty

.

The

division score

of an index

i

is the

sum

of the number of

0

's in

nums

left

and the number of

1

's in

nums

right

.

Return

all distinct indices

that have the

highest

possible

division score

. You may return the answer in

any order

.

Example 1:

Input:

nums = [0,0,1,0]

Output:

[2,4]

Explanation:

Division at index - 0: nums

left

is []. nums

right

is [0,0,

1

,0]. The score is 0 + 1 = 1. - 1: nums

left

is [

0

]. nums

right

is [0,

1

,0]. The score is 1 + 1 = 2. - 2: nums

left

is [

0

,

0

]. nums

right

is [

1

,0]. The score is 2 + 1 = 3. - 3: nums

left

is [

0

,

0

,1]. nums

right

is [0]. The score is 2 + 0 = 2. - 4: nums

left

is [

0

,

0

,1,

0

]. nums

right

is []. The score is 3 + 0 = 3. Indices 2 and 4 both have the highest possible division score 3. Note the answer [4,2] would also be accepted.

Example 2:

Input:

nums = [0,0,0]

Output:

[3]

Explanation:

Division at index - 0: nums

left

is []. nums

right

is [0,0,0]. The score is 0 + 0 = 0. - 1: nums

left

is [

0

]. nums

right

is [0,0]. The score is 1 + 0 = 1. - 2: nums

left

is [

0

,

0

]. nums

right

is [0]. The score is 2 + 0 = 2. - 3: nums

left

is [

0

,

0

,

0

]. nums

right

is []. The score is 3 + 0 = 3. Only index 3 has the highest possible division score 3.

Example 3:

Input:

nums = [1,1]

Output:

[0]

Explanation:

Division at index - 0: nums

left

is []. nums

right

is [

1

,

1

]. The score is 0 + 2 = 2. - 1: nums

left

is [1]. nums

right

is [

1

]. The score is 0 + 1 = 1. - 2: nums

left

is [1,1]. nums

right

is []. The score is 0 + 0 = 0. Only index 0 has the highest possible division score 2.

Constraints:

n == nums.length

1 <= n <= 10

5

nums[i]

is either

0

or

1

.

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public List<Integer> maxScoreIndices(int[] nums) {

}
}
```

Python3:

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

Python:

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

JavaScript:

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

};
```

TypeScript:

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

};
```

C#:

```
public class Solution {
public IList<int> MaxScoreIndices(int[] nums) {

}
}
```

C:

```
/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* maxScoreIndices(int* nums, int numsSize, int* returnSize) {

}
```

Go:

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

}
```

Kotlin:

```
class Solution {
fun maxScoreIndices(nums: IntArray): List<Int> {

}
}
```

Swift:

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

}
}
```

Rust:

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

}
}
```

Ruby:

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

end
```

PHP:

```
class Solution {

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

}
}
```

Dart:

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

}
}
```

Scala:

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

}
```

```
}
```

Elixir:

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

end
end
```

Erlang:

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

Racket:

```
(define/contract (max-score-indices nums)
(-> (listof exact-integer?) (listof exact-integer?))
)
```

Solutions

C++ Solution:

```
/*
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

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

```
}
};
```

Java Solution:

```
/**
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public List<Integer> maxScoreIndices(int[] nums) {

}
}
```

Python3 Solution:

```
"""
Problem: All Divisions With the Highest Score of a Binary Array
Difficulty: Medium
Tags: array

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""

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

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

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

};
```

TypeScript Solution:

```
/**
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

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

};
```

C# Solution:

```
/*
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public IList<int> MaxScoreIndices(int[] nums) {

}
}
```

C Solution:

```
/*
* Problem: All Divisions With the Highest Score of a Binary Array
* Difficulty: Medium
* Tags: array
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* Note: The returned array must be malloced, assume caller calls free().
*/
int* maxScoreIndices(int* nums, int numsSize, int* returnSize) {

}
```

Go Solution:

```
// Problem: All Divisions With the Highest Score of a Binary Array
// Difficulty: Medium
// Tags: array
```

```
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func maxScoreIndices(nums []int) []int {

}
```

Kotlin Solution:

```
class Solution {
fun maxScoreIndices(nums: IntArray): List<Int> {

}
}
```

Swift Solution:

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

}
}
```

Rust Solution:

```
// Problem: All Divisions With the Highest Score of a Binary Array
// Difficulty: Medium
// Tags: array
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

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

}
}
```

Ruby Solution:

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

end
```

PHP Solution:

```
class Solution {

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

}
}
```

Dart Solution:

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

}
}
```

Scala Solution:

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

}
}
```

Elixir Solution:

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

```
end
end
```

Erlang Solution:

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

Racket Solution:

```
(define/contract (max-score-indices nums)
(-> (listof exact-integer?) (listof exact-integer?))
)
```