Problem 2535: Difference Between Element Sum and Digit Sum of an Array

Problem Information

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

Problem Description

You are given a positive integer array

nums

.

The

element sum

is the sum of all the elements in

nums

.

The

digit sum

is the sum of all the digits (not necessarily distinct) that appear in

nums

.

Return

the

absolute

difference between the

element sum

and

digit sum

of

nums

.

Note

that the absolute difference between two integers

x

and

y

is defined as

|x - y|

.

Example 1:

Input:

nums = [1,15,6,3]

Output:

9

Explanation:

The element sum of nums is 1 + 15 + 6 + 3 = 25. The digit sum of nums is 1 + 1 + 5 + 6 + 3 = 16. The absolute difference between the element sum and digit sum is |25 - 16| = 9.

Example 2:

Input:

nums = [1,2,3,4]

Output:

0

Explanation:

The element sum of nums is 1 + 2 + 3 + 4 = 10. The digit sum of nums is 1 + 2 + 3 + 4 = 10. The absolute difference between the element sum and digit sum is |10 - 10| = 0.

Constraints:

1 <= nums.length <= 2000

1 <= nums[i] <= 2000

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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

Python:

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

JavaScript:

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

};
```

TypeScript:

```
function differenceOfSum(nums: number[]): number {
};
```

C#:

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

}
}
```

C:

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

}
```

Go:

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

}
```

Kotlin:

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

}
}
```

Swift:

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

}
}
```

Rust:

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



}
}
```

Ruby:

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

end
```

PHP:

```
class Solution {

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

}
}
```

Dart:

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

}
}
```

Scala:

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

}
}
```

Elixir:

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

end
end
```

Erlang:

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

Racket:

```
(define/contract (difference-of-sum nums)
(-> (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
* 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:
int differenceOfSum(vector<int>& nums) {

}
};
```

Java Solution:

```
/**
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
* 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 int differenceOfSum(int[] nums) {

}
}
```

Python3 Solution:

```
"""
Problem: Difference Between Element Sum and Digit Sum of an Array
Difficulty: Easy
Tags: array, math

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 differenceOfSum(self, nums: List[int]) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
* 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 differenceOfSum = function(nums) {

};
```

TypeScript Solution:

```
/**
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
* 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 differenceOfSum(nums: number[]): number {

};
```

C# Solution:

```
/*
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
```

```
* 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 int DifferenceOfSum(int[] nums) {

}
}
```

C Solution:

```
/*
* Problem: Difference Between Element Sum and Digit Sum of an Array
* Difficulty: Easy
* Tags: array, math
*
* 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
*/

int differenceOfSum(int* nums, int numsSize) {

}
```

Go Solution:

```
// Problem: Difference Between Element Sum and Digit Sum of an Array
// Difficulty: Easy
// Tags: array, math
//
// 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 differenceOfSum(nums []int) int {

}
```

Kotlin Solution:

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

}
}
```

Swift Solution:

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

}
}
```

Rust Solution:

```
// Problem: Difference Between Element Sum and Digit Sum of an Array
// Difficulty: Easy
// Tags: array, math
//
// 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 difference_of_sum(nums: Vec<i32>) -> i32 {

}
}
```

Ruby Solution:

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

end
```

PHP Solution:

```
class Solution {

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

}
}
```

Dart Solution:

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

}
}
```

Scala Solution:

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

}
}
```

Elixir Solution:

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

end
end
```

Erlang Solution:

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

Racket Solution:

```
(define/contract (difference-of-sum nums)
(-> (listof exact-integer?) exact-integer?)
)
```