Problem 2006: Count Number of Pairs With Absolute Difference K

Problem Information

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

Problem Description

Given an integer array

nums

and an integer

k

, return

the number of pairs

(i, j)

where

i < j

such that

|nums[i] - nums[j]| == k

.

The value of

|x|

is defined as:

x

if

x >= 0

.

-x

if

x < 0

.

Example 1:

Input:

nums = [1,2,2,1], k = 1

Output:

4

Explanation:

The pairs with an absolute difference of 1 are: - [

1

,

2

,2,1] - [

1

,2,

2

,1] - [1,

2

,2,

1

] - [1,2,

2

,

1

]

Example 2:

Input:

nums = [1,3], k = 3

Output:

0

Explanation:

,

There are no pairs with an absolute difference of 3.

Example 3:

Input:

nums = [3,2,1,5,4], k = 2

Output:

3

Explanation:

The pairs with an absolute difference of 2 are: - [

3

,2,

1

,5,4] - [

3

,2,1,

5

,4] - [3,

2

,1,5,

4

]

Constraints:

1 <= nums.length <= 200

1 <= nums[i] <= 100

1 <= k <= 99

Code Snippets

C++:

```
class Solution {
public:
int countKDifference(vector<int>& nums, int k) {

}
};
```

Java:

```
class Solution {
public int countKDifference(int[] nums, int k) {

}
}
```

Python3:

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

Python:

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

```
:type k: int
:rtype: int
"""
```

JavaScript:

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

};
```

TypeScript:

```
function countKDifference(nums: number[], k: number): number {

};
```

C#:

```
public class Solution {
public int CountKDifference(int[] nums, int k) {

}
}
```

C:

```
int countKDifference(int* nums, int numsSize, int k) {

}
```

Go:

```
func countKDifference(nums []int, k int) int {

}
```

Kotlin:

```
class Solution {
fun countKDifference(nums: IntArray, k: Int): Int {

}
}
```

Swift:

```
class Solution {
func countKDifference(_ nums: [Int], _ k: Int) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn count_k_difference(nums: Vec<i32>, k: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer[]} nums
# @param {Integer} k
# @return {Integer}
def count_k_difference(nums, k)

end
```

PHP:

```
class Solution {

/**
* @param Integer[] $nums
* @param Integer $k
* @return Integer
*/
function countKDifference($nums, $k) {

}


}
```

Dart:

```
class Solution {
int countKDifference(List<int> nums, int k) {

}
}
```

Scala:

```
object Solution {
def countKDifference(nums: Array[Int], k: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec count_k_difference(nums :: [integer], k :: integer) :: integer
def count_k_difference(nums, k) do

end
end
```

Erlang:

```
-spec count_k_difference(Nums :: [integer()], K :: integer()) -> integer().
count_k_difference(Nums, K) ->
.
```

Racket:

```
(define/contract (count-k-difference nums k)
(-> (listof exact-integer?) exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

class Solution {
public:
int countKDifference(vector<int>& nums, int k) {

}
};
```

Java Solution:

```
/**
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

class Solution {
public int countKDifference(int[] nums, int k) {

}
}
```

Python3 Solution:

```
"""
Problem: Count Number of Pairs With Absolute Difference K
Difficulty: Easy
Tags: array, hash
```

```
Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) for hash map
"""

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

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

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

};
```

TypeScript Solution:

```
/**
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

function countKDifference(nums: number[], k: number): number {

};
```

C# Solution:

```
/*
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

public class Solution {
public int CountKDifference(int[] nums, int k) {

}
}
```

C Solution:

```
/*
* Problem: Count Number of Pairs With Absolute Difference K
* Difficulty: Easy
* Tags: array, hash
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) for hash map
*/

int countKDifference(int* nums, int numsSize, int k) {

}
```

Go Solution:

```
// Problem: Count Number of Pairs With Absolute Difference K
// Difficulty: Easy
// Tags: array, hash
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

func countKDifference(nums []int, k int) int {

}
```

Kotlin Solution:

```
class Solution {
fun countKDifference(nums: IntArray, k: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func countKDifference(_ nums: [Int], _ k: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Count Number of Pairs With Absolute Difference K
// Difficulty: Easy
// Tags: array, hash
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) for hash map

impl Solution {
pub fn count_k_difference(nums: Vec<i32>, k: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[]} nums
# @param {Integer} k
# @return {Integer}
def count_k_difference(nums, k)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[] $nums
* @param Integer $k
* @return Integer
*/
function countKDifference($nums, $k) {

}
}
```

Dart Solution:

```
class Solution {
int countKDifference(List<int> nums, int k) {

}
}
```

Scala Solution:

```
object Solution {
def countKDifference(nums: Array[Int], k: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec count_k_difference(nums :: [integer], k :: integer) :: integer
def count_k_difference(nums, k) do

end
end
```

Erlang Solution:

```
-spec count_k_difference(Nums :: [integer()], K :: integer()) -> integer().
count_k_difference(Nums, K) ->
.
```

Racket Solution:

```
(define/contract (count-k-difference nums k)
(-> (listof exact-integer?) exact-integer? exact-integer?)
)
```