Problem 2001: Number of Pairs of Interchangeable Rectangles

Problem Information

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

Problem Description

You are given

n

rectangles represented by a

0-indexed

2D integer array

rectangles

, where

rectangles[i] = [width

i

, height

i

]

denotes the width and height of the

i

th

rectangle.

Two rectangles

i

and

j

(

i < j

) are considered

interchangeable

if they have the

same

width-to-height ratio. More formally, two rectangles are

interchangeable

if

width

i

/height

i

== width

j

/height

j

(using decimal division, not integer division).

Return

the

number

of pairs of

interchangeable

rectangles in

rectangles

.

Example 1:

Input:

rectangles = [[4,8],[3,6],[10,20],[15,30]]

Output:

6

Explanation:

The following are the interchangeable pairs of rectangles by index (0-indexed): - Rectangle 0 with rectangle 1: 4/8 == 3/6. - Rectangle 0 with rectangle 2: 4/8 == 10/20. - Rectangle 0 with rectangle 3: 4/8 == 15/30. - Rectangle 1 with rectangle 2: 3/6 == 10/20. - Rectangle 1 with rectangle 3: 3/6 == 15/30. - Rectangle 2 with rectangle 3: 10/20 == 15/30.

Example 2:

Input:

rectangles = [[4,5],[7,8]]

Output:

0

Explanation:

There are no interchangeable pairs of rectangles.

Constraints:

n == rectangles.length

1 <= n <= 10

5

rectangles[i].length == 2

1 <= width

i

, height

i

<= 10

Code Snippets

C++:

```
class Solution {
public:
long long interchangeableRectangles(vector<vector<int>>& rectangles) {

}
};
```

Java:

```
class Solution {
public long interchangeableRectangles(int[][] rectangles) {

}
}
```

Python3:

```
class Solution:
def interchangeableRectangles(self, rectangles: List[List[int]]) -> int:
```

Python:

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

JavaScript:

```
/**
* @param {number[][]} rectangles
* @return {number}
*/
```

```
var interchangeableRectangles = function(rectangles) {

};
```

TypeScript:

```
function interchangeableRectangles(rectangles: number[][]): number {

};
```

C#:

```
public class Solution {
public long InterchangeableRectangles(int[][] rectangles) {

}
}
```

C:

```
long long interchangeableRectangles(int** rectangles, int rectanglesSize,
int* rectanglesColSize) {

}
```

Go:

```
func interchangeableRectangles(rectangles [][]int) int64 {

}
```

Kotlin:

```
class Solution {
fun interchangeableRectangles(rectangles: Array<IntArray>): Long {

}
}
```

Swift:

```
class Solution {
func interchangeableRectangles(_ rectangles: [[Int]]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn interchangeable_rectangles(rectangles: Vec<Vec<i32>>) -> i64 {

}
}
```

Ruby:

```
# @param {Integer[][]} rectangles
# @return {Integer}
def interchangeable_rectangles(rectangles)

end
```

PHP:

```
class Solution {

/**
* @param Integer[][] $rectangles
* @return Integer
*/
function interchangeableRectangles($rectangles) {

}
}
```

Dart:

```
class Solution {
int interchangeableRectangles(List<List<int>> rectangles) {

}
}
```

Scala:

```
object Solution {
def interchangeableRectangles(rectangles: Array[Array[Int]]): Long = {

}
}
```

Elixir:

```
defmodule Solution do
@spec interchangeable_rectangles(rectangles :: [[integer]]) :: integer
def interchangeable_rectangles(rectangles) do

end
end
```

Erlang:

```
-spec interchangeable_rectangles(Rectangles :: [[integer()]]) -> integer().
interchangeable_rectangles(Rectangles) ->
.
```

Racket:

```
(define/contract (interchangeable-rectangles rectangles)
(-> (listof (listof exact-integer?)) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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:
long long interchangeableRectangles(vector<vector<int>>& rectangles) {

}
};
```

Java Solution:

```
/**
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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 long interchangeableRectangles(int[][] rectangles) {

}
}
```

Python3 Solution:

```
"""
Problem: Number of Pairs of Interchangeable Rectangles
Difficulty: Medium
Tags: array, math, 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 interchangeableRectangles(self, rectangles: List[List[int]]) -> int:
# TODO: Implement optimized solution
```

```
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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[][]} rectangles
* @return {number}
*/
var interchangeableRectangles = function(rectangles) {

};
```

TypeScript Solution:

```
/**
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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 interchangeableRectangles(rectangles: number[][]): number {

};
```

C# Solution:

```
/*
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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 long InterchangeableRectangles(int[][] rectangles) {

}
}
```

C Solution:

```
/*
* Problem: Number of Pairs of Interchangeable Rectangles
* Difficulty: Medium
* Tags: array, math, 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
*/

long long interchangeableRectangles(int** rectangles, int rectanglesSize,
int* rectanglesColSize) {

}
```

Go Solution:

```
// Problem: Number of Pairs of Interchangeable Rectangles
// Difficulty: Medium
// Tags: array, math, 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 interchangeableRectangles(rectangles [][]int) int64 {

}
```

Kotlin Solution:

```
class Solution {
fun interchangeableRectangles(rectangles: Array<IntArray>): Long {

}
}
```

Swift Solution:

```
class Solution {
func interchangeableRectangles(_ rectangles: [[Int]]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Number of Pairs of Interchangeable Rectangles
// Difficulty: Medium
// Tags: array, math, 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 interchangeable_rectangles(rectangles: Vec<Vec<i32>>) -> i64 {
```

```
}
}
```

Ruby Solution:

```
# @param {Integer[][]} rectangles
# @return {Integer}
def interchangeable_rectangles(rectangles)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[][] $rectangles
* @return Integer
*/
function interchangeableRectangles($rectangles) {

}
}
```

Dart Solution:

```
class Solution {
int interchangeableRectangles(List<List<int>> rectangles) {

}
}
```

Scala Solution:

```
object Solution {
def interchangeableRectangles(rectangles: Array[Array[Int]]): Long = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec interchangeable_rectangles(rectangles :: [[integer]]) :: integer
def interchangeable_rectangles(rectangles) do

end
end
```

Erlang Solution:

```
-spec interchangeable_rectangles(Rectangles :: [[integer()]]) -> integer().
interchangeable_rectangles(Rectangles) ->
.
```

Racket Solution:

```
(define/contract (interchangeable-rectangles rectangles)
(-> (listof (listof exact-integer?)) exact-integer?)
)
```