Problem 3619: Count Islands With Total Value Divisible by K

Problem Information

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

Problem Description

You are given an

m x n

matrix

grid

and a positive integer

k

. An

island

is a group of

positive

integers (representing land) that are

4-directionally

connected (horizontally or vertically).

The

total value

of an island is the sum of the values of all cells in the island.

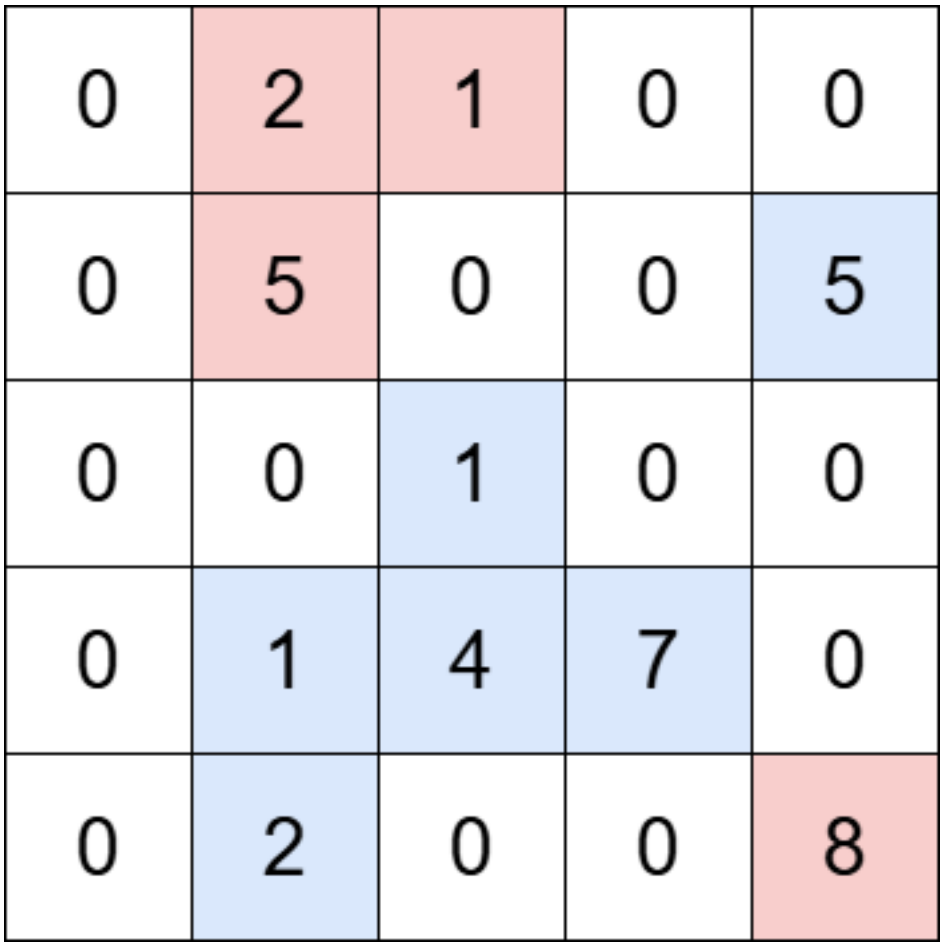
Return the number of islands with a total value

divisible by

k

.

Example 1:



Input:

grid = [[0,2,1,0,0],[0,5,0,0,5],[0,0,1,0,0],[0,1,4,7,0],[0,2,0,0,8]], k = 5

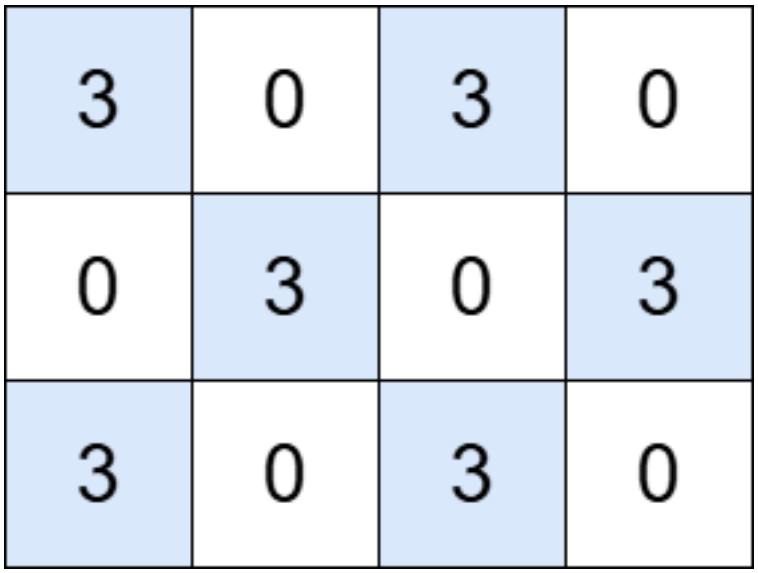
Output:

2

Explanation:

The grid contains four islands. The islands highlighted in blue have a total value that is divisible by 5, while the islands highlighted in red do not.

Example 2:



Input:

grid = [[3,0,3,0], [0,3,0,3], [3,0,3,0]], k = 3

Output:

6

Explanation:

The grid contains six islands, each with a total value that is divisible by 3.

Constraints:

m == grid.length

n == grid[i].length

1 <= m, n <= 1000

1 <= m * n <= 10

5

0 <= grid[i][j] <= 10

6

1 <= k <= 10

6

Code Snippets

C++:

```
class Solution {
public:
int countIslands(vector<vector<int>>& grid, int k) {

}
};
```

Java:

```
class Solution {
public int countIslands(int[][] grid, int k) {

}
}
```

Python3:

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

Python:

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

JavaScript:

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

};
```

TypeScript:

```
function countIslands(grid: number[][], k: number): number {

};
```

C#:

```
public class Solution {
public int CountIslands(int[][] grid, int k) {

}
}
```

C:

```
int countIslands(int** grid, int gridSize, int* gridColSize, int k) {

}
```

Go:

```
func countIslands(grid [][]int, k int) int {

}
```

Kotlin:

```
class Solution {
fun countIslands(grid: Array<IntArray>, k: Int): Int {

}
}
```

Swift:

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

}
}
```

Rust:

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

}
}
```

Ruby:

```
# @param {Integer[][]} grid
# @param {Integer} k
# @return {Integer}
def count_islands(grid, k)

end
```

PHP:

```
class Solution {
```

```
/**
* @param Integer[][] $grid
* @param Integer $k
* @return Integer
*/
function countIslands($grid, $k) {

}
}
```

Dart:

```
class Solution {
int countIslands(List<List<int>> grid, int k) {

}
}
```

Scala:

```
object Solution {
def countIslands(grid: Array[Array[Int]], k: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec count_islands(grid :: [[integer]], k :: integer) :: integer
def count_islands(grid, k) do

end
end
```

Erlang:

```
-spec count_islands(Grid :: [[integer()]], K :: integer()) -> integer().
count_islands(Grid, K) ->
.
```

Racket:

```
(define/contract (count-islands grid k)
(-> (listof (listof exact-integer?)) exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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 countIslands(vector<vector<int>>& grid, int k) {

}
};
```

Java Solution:

```
/**
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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 countIslands(int[][] grid, int k) {

}
```

```
}
```

Python3 Solution:

```
"""
Problem: Count Islands With Total Value Divisible by K
Difficulty: Medium
Tags: array, graph, search

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 countIslands(self, grid: List[List[int]], k: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

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

JavaScript Solution:

```
/**
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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[][]} grid
* @param {number} k
* @return {number}
*/
var countIslands = function(grid, k) {

};
```

TypeScript Solution:

```
/**
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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 countIslands(grid: number[][], k: number): number {

};
```

C# Solution:

```
/*
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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 CountIslands(int[][] grid, int k) {

}
```

```
}
```

C Solution:

```
/*
* Problem: Count Islands With Total Value Divisible by K
* Difficulty: Medium
* Tags: array, graph, search
*
* 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 countIslands(int** grid, int gridSize, int* gridColSize, int k) {

}
```

Go Solution:

```
// Problem: Count Islands With Total Value Divisible by K
// Difficulty: Medium
// Tags: array, graph, search
//
// 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 countIslands(grid [][]int, k int) int {

}
```

Kotlin Solution:

```
class Solution {
fun countIslands(grid: Array<IntArray>, k: Int): Int {

}
}
```

Swift Solution:

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

}
}
```

Rust Solution:

```
// Problem: Count Islands With Total Value Divisible by K
// Difficulty: Medium
// Tags: array, graph, search
//
// 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 count_islands(grid: Vec<Vec<i32>>, k: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[][]} grid
# @param {Integer} k
# @return {Integer}
def count_islands(grid, k)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[][] $grid
* @param Integer $k
* @return Integer
*/
function countIslands($grid, $k) {
```

```
}
}
```

Dart Solution:

```
class Solution {
int countIslands(List<List<int>> grid, int k) {

}
}
```

Scala Solution:

```
object Solution {
def countIslands(grid: Array[Array[Int]], k: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec count_islands(grid :: [[integer]], k :: integer) :: integer
def count_islands(grid, k) do

end
end
```

Erlang Solution:

```
-spec count_islands(Grid :: [[integer()]], K :: integer()) -> integer().
count_islands(Grid, K) ->
.
```

Racket Solution:

```
(define/contract (count-islands grid k)
(-> (listof (listof exact-integer?)) exact-integer? exact-integer?)
)
```