Problem 764: Largest Plus Sign

Problem Information

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

Problem Description

You are given an integer

n

. You have an

n x n

binary grid

grid

with all values initially

1

's except for some indices given in the array

mines

. The

i

th

element of the array

mines

is defined as

mines[i] = [x

i

, y

i

]

where

grid[x

i

][y

i

] == 0

.

Return

the order of the largest

axis-aligned

plus sign of

1

's contained in

grid

. If there is none, return

0

.

An

axis-aligned plus sign

of

1

's of order

k

has some center

grid[r][c] == 1

along with four arms of length

k - 1

going up, down, left, and right, and made of

1

's. Note that there could be

0

's or

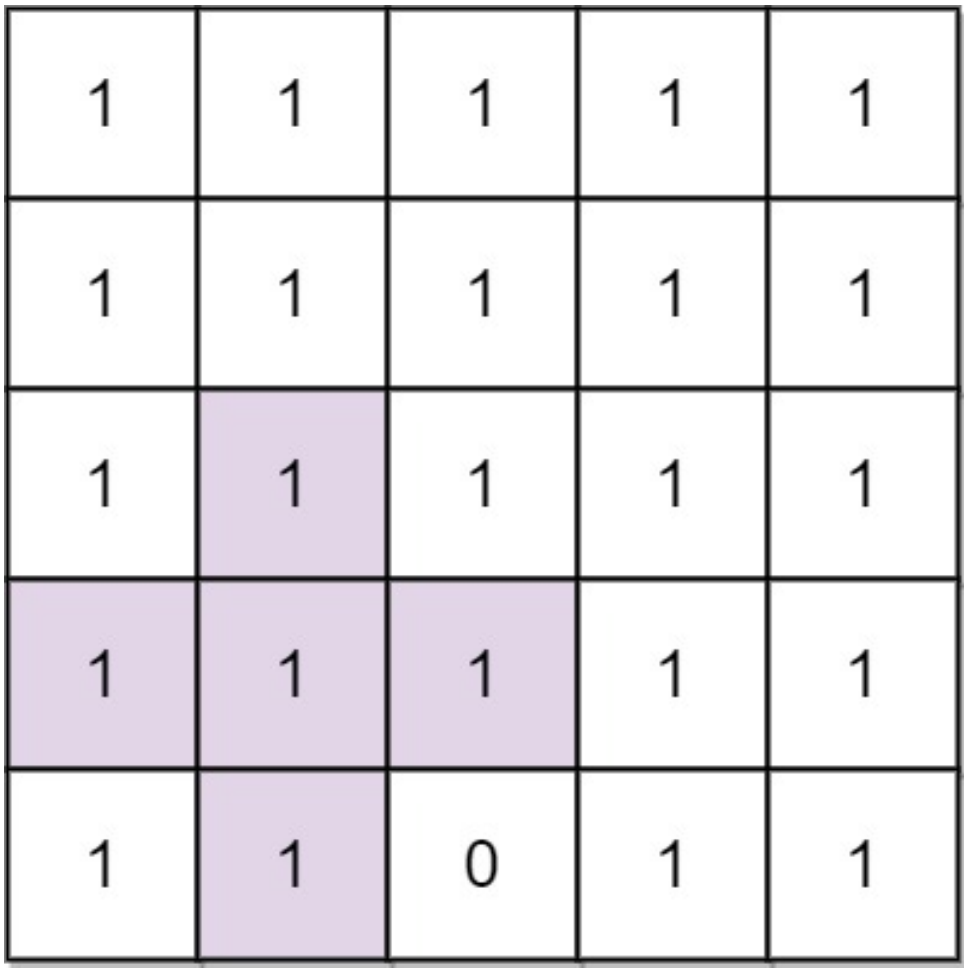
1

's beyond the arms of the plus sign, only the relevant area of the plus sign is checked for

1

's.

Example 1:



Input:

n = 5, mines = [[4,2]]

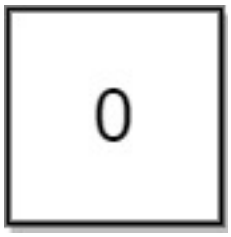
Output:

2

Explanation:

In the above grid, the largest plus sign can only be of order 2. One of them is shown.

Example 2:



Input:

n = 1, mines = [[0,0]]

Output:

0

Explanation:

There is no plus sign, so return 0.

Constraints:

1 <= n <= 500

1 <= mines.length <= 5000

0 <= x

i

, y

i

< n

All the pairs

(x

i

, y

i

)

are

unique

.

Code Snippets

C++:

```
class Solution {
public:
int orderOfLargestPlusSign(int n, vector<vector<int>>& mines) {

}
};
```

Java:

```
class Solution {
public int orderOfLargestPlusSign(int n, int[][] mines) {

}
}
```

Python3:

```
class Solution:
def orderOfLargestPlusSign(self, n: int, mines: List[List[int]]) -> int:
```

Python:

```
class Solution(object):
def orderOfLargestPlusSign(self, n, mines):
"""
:type n: int
:type mines: List[List[int]]
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number} n
* @param {number[][]} mines
* @return {number}
*/
var orderOfLargestPlusSign = function(n, mines) {

};
```

TypeScript:

```
function orderOfLargestPlusSign(n: number, mines: number[][]): number {

};
```

C#:

```
public class Solution {
public int OrderOfLargestPlusSign(int n, int[][] mines) {

}
}
```

C:

```
int orderOfLargestPlusSign(int n, int** mines, int minesSize, int*
minesColSize) {

}
```

Go:

```
func orderOfLargestPlusSign(n int, mines [][]int) int {

}
```

Kotlin:

```
class Solution {
fun orderOfLargestPlusSign(n: Int, mines: Array<IntArray>): Int {

}
}
```

Swift:

```
class Solution {
func orderOfLargestPlusSign(_ n: Int, _ mines: [[Int]]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn order_of_largest_plus_sign(n: i32, mines: Vec<Vec<i32>>) -> i32 {

}
}
```

Ruby:

```
# @param {Integer} n
# @param {Integer[][]} mines
# @return {Integer}
def order_of_largest_plus_sign(n, mines)

end
```

PHP:

```
class Solution {

/**
* @param Integer $n
* @param Integer[][] $mines
* @return Integer
*/
function orderOfLargestPlusSign($n, $mines) {

}
}
```

Dart:

```
class Solution {
int orderOfLargestPlusSign(int n, List<List<int>> mines) {

}
}
```

Scala:

```
object Solution {
def orderOfLargestPlusSign(n: Int, mines: Array[Array[Int]]): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec order_of_largest_plus_sign(n :: integer, mines :: [[integer]]) ::
integer
def order_of_largest_plus_sign(n, mines) do

end
end
```

Erlang:

```
-spec order_of_largest_plus_sign(N :: integer(), Mines :: [[integer()]]) ->
integer().
order_of_largest_plus_sign(N, Mines) ->
.
```

Racket:

```
(define/contract (order-of-largest-plus-sign n mines)
(-> exact-integer? (listof (listof exact-integer?)) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Largest Plus Sign
* Difficulty: Medium
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

class Solution {
public:
int orderOfLargestPlusSign(int n, vector<vector<int>>& mines) {

}
};
```

Java Solution:

```
/**
* Problem: Largest Plus Sign
* Difficulty: Medium
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
```

```
* Space Complexity: O(n) or O(n * m) for DP table
*/

class Solution {
public int orderOfLargestPlusSign(int n, int[][] mines) {

}
}
```

Python3 Solution:

```
"""
Problem: Largest Plus Sign
Difficulty: Medium
Tags: array, dp

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(n) or O(n * m) for DP table
"""

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

Python Solution:

```
class Solution(object):
def orderOfLargestPlusSign(self, n, mines):
"""
:type n: int
:type mines: List[List[int]]
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Largest Plus Sign
* Difficulty: Medium
```

```
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

/**
* @param {number} n
* @param {number[][]} mines
* @return {number}
*/
var orderOfLargestPlusSign = function(n, mines) {

};
```

TypeScript Solution:

```
/**
* Problem: Largest Plus Sign
* Difficulty: Medium
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

function orderOfLargestPlusSign(n: number, mines: number[][]): number {

};
```

C# Solution:

```
/*
* Problem: Largest Plus Sign
* Difficulty: Medium
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
```

```
* Space Complexity: O(n) or O(n * m) for DP table
*/

public class Solution {
public int OrderOfLargestPlusSign(int n, int[][] mines) {

}
}
```

C Solution:

```
/*
* Problem: Largest Plus Sign
* Difficulty: Medium
* Tags: array, dp
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(n) or O(n * m) for DP table
*/

int orderOfLargestPlusSign(int n, int** mines, int minesSize, int*
minesColSize) {

}
```

Go Solution:

```
// Problem: Largest Plus Sign
// Difficulty: Medium
// Tags: array, dp
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

func orderOfLargestPlusSign(n int, mines [][]int) int {

}
```

Kotlin Solution:

```
class Solution {
fun orderOfLargestPlusSign(n: Int, mines: Array<IntArray>): Int {

}
}
```

Swift Solution:

```
class Solution {
func orderOfLargestPlusSign(_ n: Int, _ mines: [[Int]]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Largest Plus Sign
// Difficulty: Medium
// Tags: array, dp
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(n) or O(n * m) for DP table

impl Solution {
pub fn order_of_largest_plus_sign(n: i32, mines: Vec<Vec<i32>>) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer} n
# @param {Integer[][]} mines
# @return {Integer}
def order_of_largest_plus_sign(n, mines)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer $n
* @param Integer[][] $mines
* @return Integer
*/
function orderOfLargestPlusSign($n, $mines) {

}
}
```

Dart Solution:

```
class Solution {
int orderOfLargestPlusSign(int n, List<List<int>> mines) {

}
}
```

Scala Solution:

```
object Solution {
def orderOfLargestPlusSign(n: Int, mines: Array[Array[Int]]): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec order_of_largest_plus_sign(n :: integer, mines :: [[integer]]) ::
integer
def order_of_largest_plus_sign(n, mines) do

end
end
```

Erlang Solution:

```
-spec order_of_largest_plus_sign(N :: integer(), Mines :: [[integer()]]) ->
integer().
```

```
order_of_largest_plus_sign(N, Mines) ->
.
```

Racket Solution:

```
(define/contract (order-of-largest-plus-sign n mines)
(-> exact-integer? (listof (listof exact-integer?)) exact-integer?)
)
```