Problem 1292: Maximum Side Length of a Square with Sum Less than or Equal to Threshold

Problem Information

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

Problem Description

Given a

m x n

matrix

mat

and an integer

threshold

, return

the maximum side-length of a square with a sum less than or equal to

threshold

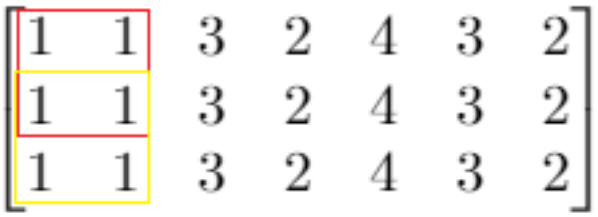
or return

0

if there is no such square

.

Example 1:



Input:

mat = [[1,1,3,2,4,3,2],[1,1,3,2,4,3,2],[1,1,3,2,4,3,2]], threshold = 4

Output:

2

Explanation:

The maximum side length of square with sum less than 4 is 2 as shown.

Example 2:

Input:

mat = [[2,2,2,2,2],[2,2,2,2,2],[2,2,2,2,2],[2,2,2,2,2],[2,2,2,2,2]], threshold = 1

Output:

0

Constraints:

m == mat.length

n == mat[i].length

1 <= m, n <= 300

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

4

0 <= threshold <= 10

5

Code Snippets

C++:

```
class Solution {
public:
int maxSideLength(vector<vector<int>>& mat, int threshold) {

}
};
```

Java:

```
class Solution {
public int maxSideLength(int[][] mat, int threshold) {

}
}
```

Python3:

```
class Solution:
def maxSideLength(self, mat: List[List[int]], threshold: int) -> int:
```

Python:

```
class Solution(object):
def maxSideLength(self, mat, threshold):
"""
:type mat: List[List[int]]
:type threshold: int
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number[][]} mat
* @param {number} threshold
* @return {number}
*/
var maxSideLength = function(mat, threshold) {

};
```

TypeScript:

```
function maxSideLength(mat: number[][], threshold: number): number {

};
```

C#:

```
public class Solution {
public int MaxSideLength(int[][] mat, int threshold) {

}
}
```

C:

```
int maxSideLength(int** mat, int matSize, int* matColSize, int threshold) {

}
```

Go:

```
func maxSideLength(mat [][]int, threshold int) int {
```

```
}
```

Kotlin:

```
class Solution {
fun maxSideLength(mat: Array<IntArray>, threshold: Int): Int {

}
}
```

Swift:

```
class Solution {
func maxSideLength(_ mat: [[Int]], _ threshold: Int) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn max_side_length(mat: Vec<Vec<i32>>, threshold: i32) -> i32 {

}
}
```

Ruby:

```
# @param {Integer[][]} mat
# @param {Integer} threshold
# @return {Integer}
def max_side_length(mat, threshold)

end
```

PHP:

```
class Solution {

/**
* @param Integer[][] $mat
* @param Integer $threshold
```

```
* @return Integer
*/
function maxSideLength($mat, $threshold) {

}
}
```

Dart:

```
class Solution {
int maxSideLength(List<List<int>> mat, int threshold) {

}
}
```

Scala:

```
object Solution {
def maxSideLength(mat: Array[Array[Int]], threshold: Int): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec max_side_length(mat :: [[integer]], threshold :: integer) :: integer
def max_side_length(mat, threshold) do

end
end
```

Erlang:

```
-spec max_side_length(Mat :: [[integer()]], Threshold :: integer()) ->
integer().
max_side_length(Mat, Threshold) ->
.
```

Racket:

```
(define/contract (max-side-length mat threshold)
(-> (listof (listof exact-integer?)) exact-integer? exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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 maxSideLength(vector<vector<int>>& mat, int threshold) {

}
};
```

Java Solution:

```
/**
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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 maxSideLength(int[][] mat, int threshold) {
```

```
}
}
```

Python3 Solution:

```
"""
Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
Difficulty: Medium
Tags: array, 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 maxSideLength(self, mat: List[List[int]], threshold: int) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def maxSideLength(self, mat, threshold):
"""
:type mat: List[List[int]]
:type threshold: int
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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[][]} mat
* @param {number} threshold
* @return {number}
*/
var maxSideLength = function(mat, threshold) {

};
```

TypeScript Solution:

```
/**
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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 maxSideLength(mat: number[][], threshold: number): number {

};
```

C# Solution:

```
/*
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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 MaxSideLength(int[][] mat, int threshold) {

}
}
```

C Solution:

```
/*
* Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
* Difficulty: Medium
* Tags: array, 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 maxSideLength(int** mat, int matSize, int* matColSize, int threshold) {

}
```

Go Solution:

```
// Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
// Difficulty: Medium
// Tags: array, 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 maxSideLength(mat [][]int, threshold int) int {

}
```

Kotlin Solution:

```
class Solution {
fun maxSideLength(mat: Array<IntArray>, threshold: Int): Int {

}
}
```

Swift Solution:

```
class Solution {
func maxSideLength(_ mat: [[Int]], _ threshold: Int) -> Int {

}
}
```

Rust Solution:

```
// Problem: Maximum Side Length of a Square with Sum Less than or Equal to
Threshold
// Difficulty: Medium
// Tags: array, 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 max_side_length(mat: Vec<Vec<i32>>, threshold: i32) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[][]} mat
# @param {Integer} threshold
# @return {Integer}
def max_side_length(mat, threshold)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[][] $mat
* @param Integer $threshold
* @return Integer
*/
function maxSideLength($mat, $threshold) {

}
}
```

Dart Solution:

```
class Solution {
int maxSideLength(List<List<int>> mat, int threshold) {

}
}
```

Scala Solution:

```
object Solution {
def maxSideLength(mat: Array[Array[Int]], threshold: Int): Int = {

}
}
```

Elixir Solution:

```
defmodule Solution do
@spec max_side_length(mat :: [[integer]], threshold :: integer) :: integer
def max_side_length(mat, threshold) do

end
end
```

Erlang Solution:

```
-spec max_side_length(Mat :: [[integer()]], Threshold :: integer()) ->
integer().
max_side_length(Mat, Threshold) ->
```

```
.
```

Racket Solution:

```
(define/contract (max-side-length mat threshold)
(-> (listof (listof exact-integer?)) exact-integer? exact-integer?)
)
```