

Problem 73: Set Matrix Zeroes

Problem Information

Difficulty: Medium

Acceptance Rate: 0.00%

Paid Only: No

Problem Description

Given an

$m \times n$

integer matrix

matrix

, if an element is

0

, set its entire row and column to

0

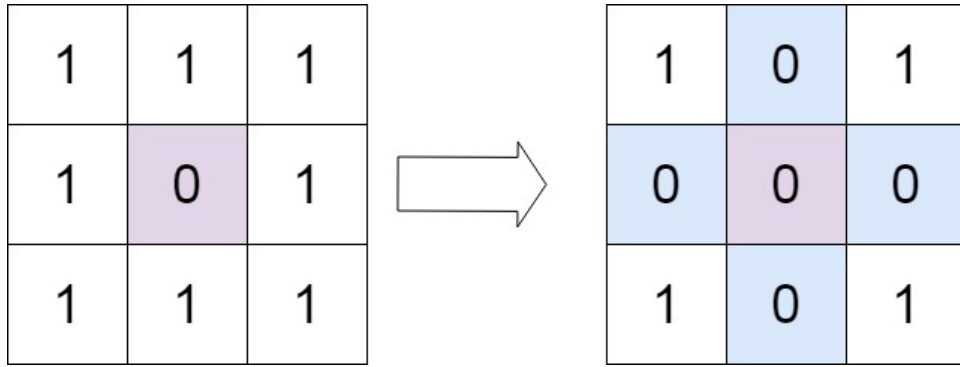
's.

You must do it

in place

.

Example 1:



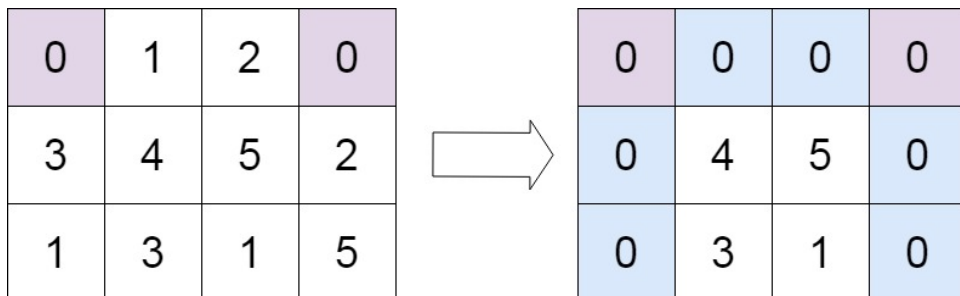
Input:

```
matrix = [[1,1,1],[1,0,1],[1,1,1]]
```

Output:

```
[[1,0,1],[0,0,0],[1,0,1]]
```

Example 2:



Input:

```
matrix = [[0,1,2,0],[3,4,5,2],[1,3,1,5]]
```

Output:

```
[[0,0,0,0],[0,4,5,0],[0,3,1,0]]
```

Constraints:

```
m == matrix.length
```

```
n == matrix[0].length
```

$1 \leq m, n \leq 200$

-2

31

$\leq \text{matrix}[i][j] \leq 2$

31

- 1

Follow up:

A straightforward solution using

$O(mn)$

space is probably a bad idea.

A simple improvement uses

$O(m + n)$

space, but still not the best solution.

Could you devise a constant space solution?

Code Snippets

C++:

```
class Solution {
public:
    void setZeroes(vector<vector<int>>& matrix) {

    }
};
```

Java:

```
class Solution {  
    public void setZeroes(int[][] matrix) {  
  
    }  
}
```

Python3:

```
class Solution:  
    def setZeroes(self, matrix: List[List[int]]) -> None:  
        """  
        Do not return anything, modify matrix in-place instead.  
        """
```

Python:

```
class Solution(object):  
    def setZeroes(self, matrix):  
        """  
        :type matrix: List[List[int]]  
        :rtype: None Do not return anything, modify matrix in-place instead.  
        """
```

JavaScript:

```
/**  
 * @param {number[][]} matrix  
 * @return {void} Do not return anything, modify matrix in-place instead.  
 */  
var setZeroes = function(matrix) {  
  
};
```

TypeScript:

```
/**  
Do not return anything, modify matrix in-place instead.  
 */  
function setZeroes(matrix: number[][]): void {  
  
};
```

C#:

```
public class Solution {  
    public void SetZeroes(int[][] matrix) {  
  
    }  
}
```

C:

```
void setZeroes(int** matrix, int matrixSize, int* matrixColSize) {  
  
}
```

Go:

```
func setZeroes(matrix [][]int) {  
  
}
```

Kotlin:

```
class Solution {  
    fun setZeroes(matrix: Array<IntArray>): Unit {  
  
    }  
}
```

Swift:

```
class Solution {  
    func setZeroes(_ matrix: inout [[Int]]) {  
  
    }  
}
```

Rust:

```
impl Solution {  
    pub fn set_zeroes(matrix: &mut Vec<Vec<i32>> ) {  
  
    }  
}
```

Ruby:

```
# @param {Integer[][]} matrix
# @return {Void} Do not return anything, modify matrix in-place instead.
def set_zeroes(matrix)

end
```

PHP:

```
class Solution {

    /**
     * @param Integer[][] $matrix
     * @return NULL
     */
    function setZeroes(&$matrix) {

    }

}
```

Dart:

```
class Solution {
  void setZeroes(List<List<int>> matrix) {

  }
}
```

Scala:

```
object Solution {
  def setZeroes(matrix: Array[Array[Int]]): Unit = {

  }
}
```

Solutions

C++ Solution:

```

/*
 * Problem: Set Matrix Zeroes
 * Difficulty: Medium
 * 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:
    void setZeroes(vector<vector<int>>& matrix) {

    }
};

```

Java Solution:

```

/**
 * Problem: Set Matrix Zeroes
 * Difficulty: Medium
 * 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 void setZeroes(int[][] matrix) {

    }
}

```

Python3 Solution:

```

"""
Problem: Set Matrix Zeroes
Difficulty: Medium
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 setZeroes(self, matrix: List[List[int]]) -> None:
```

```
# TODO: Implement optimized solution
```

```
pass
```

Python Solution:

```
class Solution(object):
```

```
def setZeroes(self, matrix):
```

```
"""
```

```
:type matrix: List[List[int]]
```

```
:rtype: None Do not return anything, modify matrix in-place instead.
```

```
"""
```

JavaScript Solution:

```
/**
```

```
 * Problem: Set Matrix Zeroes
```

```
 * Difficulty: Medium
```

```
 * 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[][]} matrix
```

```
 * @return {void} Do not return anything, modify matrix in-place instead.
```

```
 */
```

```
var setZeroes = function(matrix) {
```

```
};
```

TypeScript Solution:


```

/**
 * Problem: Set Matrix Zeroes
 * Difficulty: Medium
 * 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
 */

/**
Do not return anything, modify matrix in-place instead.
 */
function setZeroes(matrix: number[][]): void {

};

```

C# Solution:

```

/*
 * Problem: Set Matrix Zeroes
 * Difficulty: Medium
 * 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 void SetZeroes(int[][] matrix) {

    }
}

```

C Solution:

```

/*
 * Problem: Set Matrix Zeroes
 * Difficulty: Medium
 * 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
*/

void setZeroes(int** matrix, int matrixSize, int* matrixColSize) {

}

```

Go Solution:

```

// Problem: Set Matrix Zeroes
// Difficulty: Medium
// 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 setZeroes(matrix [][]int) {

}

```

Kotlin Solution:

```

class Solution {
    fun setZeroes(matrix: Array<IntArray>): Unit {

    }
}

```

Swift Solution:

```

class Solution {
    func setZeroes(_ matrix: inout [[Int]]) {

    }
}

```

Rust Solution:

```

// Problem: Set Matrix Zeroes
// Difficulty: Medium
// 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 set_zeroes(matrix: &mut Vec<Vec<i32>> >) {

}
}

```

Ruby Solution:

```

# @param {Integer[][]} matrix
# @return {Void} Do not return anything, modify matrix in-place instead.
def set_zeroes(matrix)

end

```

PHP Solution:

```

class Solution {

/**
 * @param Integer[][] $matrix
 * @return NULL
 */
function setZeroes(&$matrix) {

}

}

```

Dart Solution:

```

class Solution {
void setZeroes(List<List<int>> matrix) {

}

}

```

Scala Solution:

```
object Solution {  
  def setZeroes(matrix: Array[Array[Int]]): Unit = {  
  
  }  
}
```