Description | Discussion (45) | Solutions (9.8K) | Submissions

C++ | Auto

74. Search a 2D Matrix

Medium | 11.4K | 337 | Companies

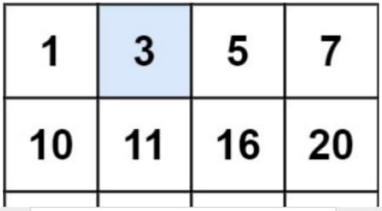
You are given an `m x n` integer matrix `matrix` with the following two properties:

- Each row is sorted in non-decreasing order.
- The first integer of each row is greater than the last integer of the previous row.

Given an integer `target`, return `true` *if* `target` *is in* `matrix` *or* `false` *otherwise.*

You must write a solution in `O(log(m * n))` time complexity.

Example 1:




```cpp
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& mat, int target) {
        int n = mat.size();
        int m = mat[0].size();
        int i = 0 , j = m-1;

        while(i<n && j >= 0){
            if((mat[i][j])==target){
                return 1;
            }else if(mat[i][j] > target){
                j--;
            }else{
                i++;
            }

        }
        return 0;
    }
};
```

Console | Run | Submit

Spirally traversing a matrix

Medium Accuracy: 35.2% Submissions: 199K+ Points: 4

Given a matrix of size r*c. Traverse the matrix in spiral form.

Example 1:

Input:

r = 4, c = 4

matrix[][] = {{1, 2, 3, 4},
{5, 6, 7, 8},
{9, 10, 11, 12},
{13, 14, 15,16}}

Output:

1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10

Explanation:

Input:

C++ (g++ 5.4) Start Timer

```cpp
// } Driver Code Ends
class Solution
{
    public:
    //Function to return a list of integers denoting spiral traversal of matrix.
    vector<int> spirallyTraverse(vector<vector<int> > matrix, int r, int c)
    {
        // code here
        vector<int>ans;
        int top = 0,bottom = r-1;
        int right = c-1 , left = 0;

        while(top <= bottom && left<= right){
            for(int i = left ; i <= right ; i++)
                ans.push_back(matrix[top][i]);
            top++;

            for(int i = top ; i<= bottom ; i++)
                ans.push_back(matrix[i][right]);
            right--;

            if(top <= bottom){
                for(int i =right; i>= left;i--)
                    ans.push_back(matrix[bottom][i]);
                bottom--;
            }
            if(left <= right){
                for(int i = bottom;i>= top ; i--)
                    ans.push_back(matrix[i][left]);
                left++;
            }
        }
        return ans;
    }
};
// } Driver Code Ends
```

Custom Input | Compile & Run | Submit

Problem | Editorial | Submissions | Comments

Median in a row-wise sorted Matrix

Medium Accuracy: 55.05% Submissions: 68K+ Points: 4

Given a row wise sorted matrix of size **R*C** where R and C are always **odd**, find the median of the matrix.

Example 1:

Input:
R = 3, C = 3
M = [[1, 3, 5],
[2, 6, 9],
[3, 6, 9]]
Output: 5
Explanation: Sorting matrix elements gives us {1,2,3,3,5,6,6,9,9}. Hence, 5 is median.

C++ (g++ 5.4) | Start Timer

```
// } Driver Code Ends
//User function template for C++

class Solution{
public:
    int median(vector<vector<int>> &matrix, int R, int C){
        // code here
        vector<int>v;
        for(int i = 0 ; i<R;i++){
            for(int j = 0 ; j<C;j++){
                v.push_back(matrix[i][j]);
            }
        }
        sort(v.begin(),v.end());
        int median = v.size();

        return v[median/2];

    }
};

// } Driver Code Ends
```

Custom Input | Compile & Run | Submit