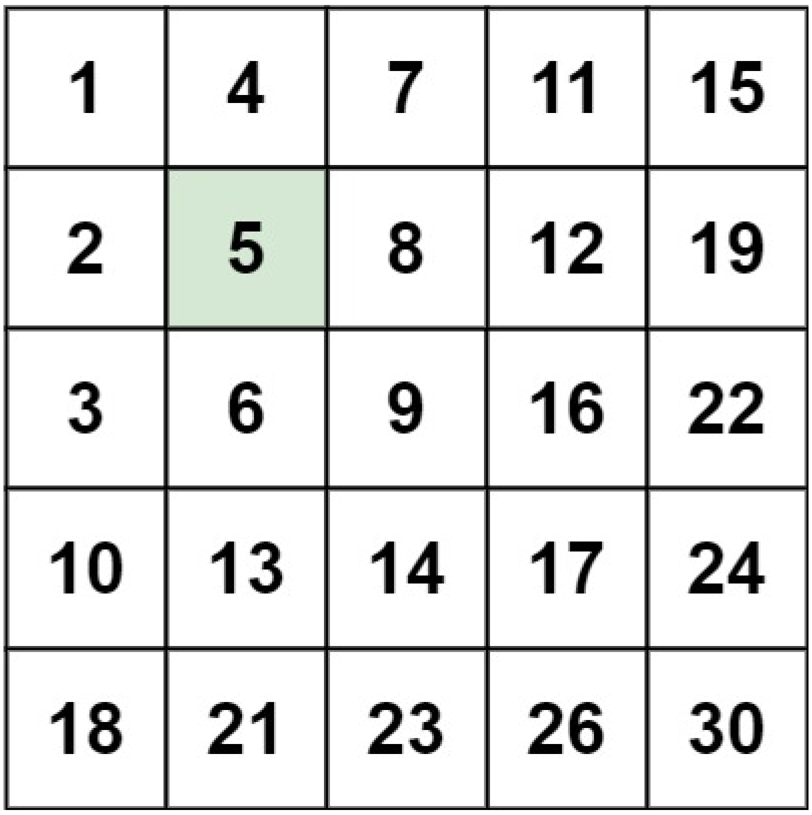
240. Search a 2D Matrix II

Write an efficient algorithm that searches for a `target` value in an `m x n` integer `matrix`. The `matrix` has the following properties:

- Integers in each row are sorted in ascending from left to right.
- Integers in each column are sorted in ascending from top to bottom.

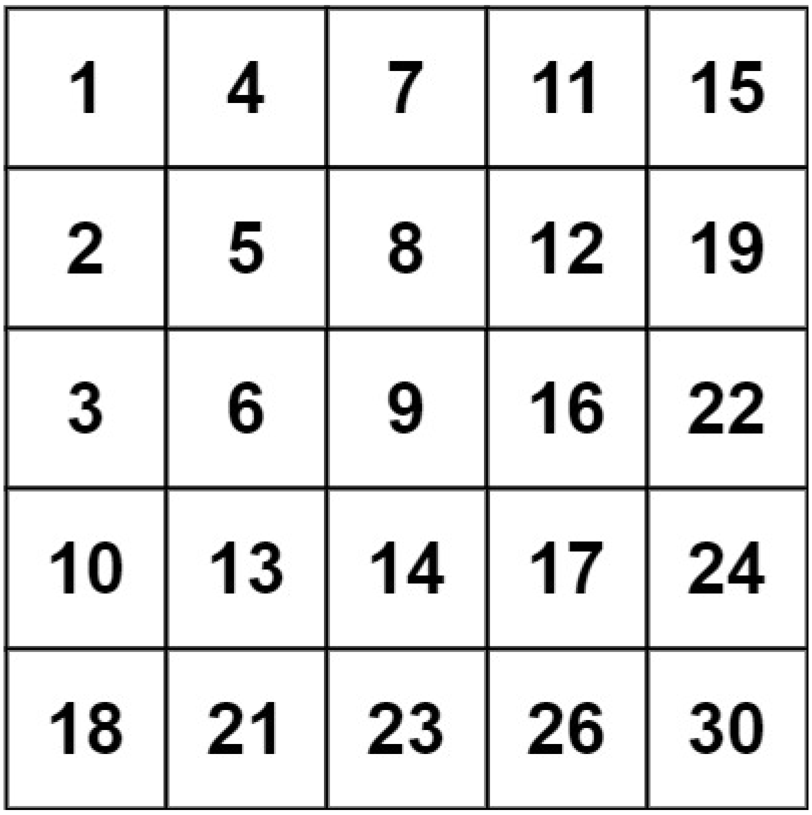
Example 1:



1	4	7	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

```
Input: matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],
[18,21,23,26,30]], target = 5
Output: true
```

Example 2:



1	4	7	11	15
2	5	8	12	19
3	6	9	16	22
10	13	14	17	24
18	21	23	26	30

```
Input: matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],
[18,21,23,26,30]], target = 20
Output: false
```

Constraints:

- `m == matrix.length`
- `n == matrix[i].length`
- `1 <= n, m <= 300`
- `-10<sup>9</sup> <= matrix[i][j] <= 10<sup>9</sup>`
- All the integers in each row are **sorted** in ascending order.
- All the integers in each column are **sorted** in ascending order.
- `-10<sup>9</sup> <= target <= 10<sup>9</sup>`

```python
class Solution:
    def searchMatrix(self, matrix: List[List[int]], target: int) -> bool:
        i = 0
        j = len(matrix[0])-1

        while i<len(matrix) and j>=0:
            temp = matrix[i][j]
            if temp==target:
                return True
            elif temp>target:
                j-=1
            else:
                i+=1
        return False
```