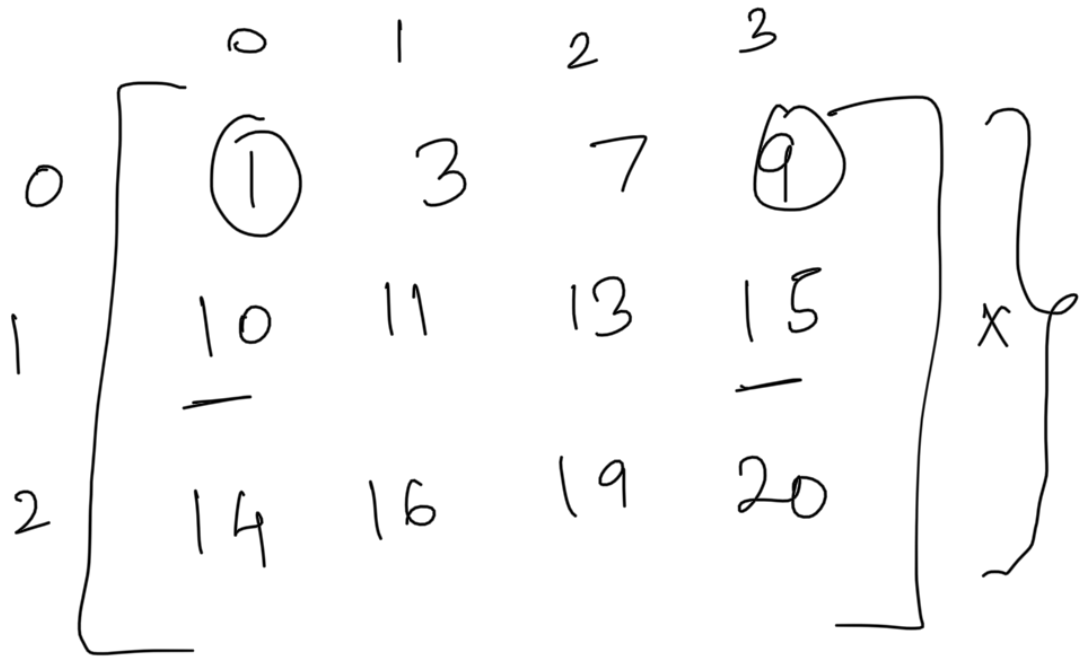
74 Search a 2D Matrix



Brute force : O (m x n)

Optimal solution :

- Binary search on rows (using L+R) to find the row that includes the target. → logm
- After getting the row, ran binary search to search the target. → logn

→ logm + logn

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

        # binary search to find the row which includes the target element
        top_row, bottom_row = 0, ROWS - 1
        while top_row <= bottom_row:
            middle_row = (top_row + bottom_row) // 2
            if target > matrix[middle_row][-1]:
                top_row = middle_row + 1
            elif target < matrix[middle_row][0]:
                bottom_row = middle_row - 1
            else:
                break

        if not top_row <= bottom_row:
            return False

        # binary search (on the computed row) to find the target element
        left_element, right_element = 0, COLS - 1
        while left_element <= right_element:
            middle_element = (left_element + right_element) // 2
            if target > matrix[middle_row][middle_element]:
                left_element = middle_element + 1
            elif target < matrix[middle_row][middle_element]:
                right_element = middle_element - 1
            else:
                return True

        return False
```