```cpp
class Solution {
public:
    bool searchMatrix(vector<vector<int>>& matrix, int target) {

        // 确保矩阵不为空
        if (matrix.empty() || matrix[0].empty())
        {
            return false;
        }
        int M = matrix.size();
        int N = matrix[0].size();
        int left1 = 0;
        // 修改为 M - 1
        int right1 = M - 1;
        // 默认值为 M，表示未找到行
        int hang_index = M;
        //二分查找寻找目标行
        while (left1 <= right1) {
            int mid1 = (left1 + right1) / 2;
            if (target < matrix[mid1][0]) {
                // 更新右边界
                right1 = mid1 - 1;
            } else {
                // 记录当前行
                hang_index = mid1;
                // 移动到下一个可能的行
                left1 = mid1 + 1;
            }
        }
        // 如果没有找到有效的行，返回 false
        if (hang_index == M || matrix[hang_index][0] > target)
            return false;

        // 二分查找目标列
        int left2 = 0;
        // 修改为 N - 1
        int right2 = N - 1;
        while (left2 <= right2) {
            int mid2 = (left2 + right2) / 2;
            // 找到目标
            if (matrix[hang_index][mid2] == target) {
```

```
                return true;
            }
            if (target < matrix[hang_index][mid2]) {
                // 更新右边界
                right2 = mid2 - 1;
            } else {
                // 更新左边界
                left2 = mid2 + 1;
            }
        }

        return false;
    }
};```
```