# 题目

编写一个高效的算法来搜索 m x n 矩阵 matrix 中的一个目标值 target。该矩阵具有以下特性：

每行的元素从左到右升序排列。

每列的元素从上到下升序排列。

**示例:**

现有矩阵 matrix 如下：

[

[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]

]

给定 target = 5，返回 true。

给定 target = 20，返回 false。

**类似题目：**剑指offer 04。

# 分析

## 方法一：分治算法

**思路：**

从右上角开始（左下也可以），往左下角走，右上角那个元素往左元素递减，往下元素递增，一增一减大大降低了搜索的时间复杂度。

targe大了，往下搜

target小了，往左搜

**分析：**

class Solution {

public:

bool searchMatrix(vector<vector<int>>& matrix, int target) {

if(matrix.size()==0)

return false;

int m=matrix.size(); //总行号

int n=matrix.at(0).size(); //总列号

int row = 0;

int col = n-1;

//可以使用右上角或左下角作为参考值

while(row<m && col>=0)

{

if(target < matrix[row][col])

{

col--;

}

else if(target >matrix[row][col])

{

row++;

}

else

{

return true;

}

}

return false;

}

};

**复杂度分析：**

时间复杂度：O(n+m)，因为最差情况是从右上搜到左下角。

空间复杂度：O(1)，因为这种方法只处理几个指针，所以它的内存占用是恒定的。