



11、二维数组的查找

24计算机考研成员一战成硕！



题目描述：

在一个二维数组array中（每个一维数组的长度相同），每一行都按照从左到右**递增**的顺序排序，每一列都按照从上到下递增的顺序排序。请完成一个函数，输入这样的一个二维数组和一个整数，判断数组中是否含有该整数。

```
[  
[1,2,8,9],  
[2,4,9,12],  
[4,7,10,13],  
[6,8,11,15]  
]
```

给定 target = 7，返回 true。

给定 target = 3，返回 false。

数据范围：矩阵的长宽满足 $0 \leq n, m \leq 500$ ，矩阵中的值满足 $0 \leq \text{val} \leq 10^9$

进阶：空间复杂度 $O(1)$ ，时间复杂度 $O(n+m)$

0	1	4	7	11	15
0	2	5	8	12	19
	3	6	9	16	22
	10	13	14	17	24

j	1	4	7	11	15
	2	5	8	12	19
	3	6	9	16	22
i	10	13	14	17	24

j	1	4	7	11	15
	2	5	8	12	19
i	3	6	9	16	22
	10	13	14	17	24

j	1	4	7	11	15
	2	5	8	12	19
i	3	6	9	16	22
	10	13	14	17	24

j	1	4	7	11	15
	2	5	8	12	19
i	3	6	9	16	22
	10	13	14	17	24

j	1	4	7	11	15
	2	5	8	12	19
i	3	6	9	16	22
	10	13	14	17	24

Target = 8
 当前遍历的值：10
 10 > 8，往上走
 i--，相当于去除一行。j=0

Target = 8
 当前遍历的值：3
 3 < 8，没必要再往上选值比较，
 j++，相当于去除一列

Target = 8
 当前遍历的值：6
 6 < 8，没必要再往上选值比较，
 j++，相当于去除一列，4和5不用比较

Target = 8
 当前遍历的值：9
 9 > 8，没必要再往右选值比较，
 i--，相当于去除一列，16和22不用比较

Target = 8
 当前遍历的值：8
 相等，结束



⌚ 倒计时

1、知识点及难度



解人：多动症男孩

难度：中等偏下

知识点：

1. 二维数组按照从左到右递增和从上到下递增的规则排序，利用这一规律可以采用特定的搜索策略，缩小搜索范围。
2. `#include <stdbool.h>` 是一个头文件，它定义了一个新的数据类型 `bool` 和两个常量 `true` 和 `false`，用于表示布尔值。
3. `int** array` 是一个指针变量，它指向一个指针数组，每个指针指向一个 `int` 类型的值。可以将其理解为一个二维数组的首地址。

时间复杂度： $O(n+m)$ ，其中 n 和 m 分别为二维数组的行数和列数。由于每一步操作都会将搜索范围减少一行或一列，最多需要进行 $n+m$ 次操作即可完成搜索。

空间复杂度： $O(1)$ ，算法只使用了常数个额外变量来保存辅助信息，没有使用额外的辅助空间。

2、算法题

思路

1. 初始搜索起始位置为右上角元素，即行为0，列为数组的列数减1。
2. 循环执行以下的步骤，直到搜索范围越界：
 - a. 如果当前元素等于目标值，返回true,表示找到了目标值。
 - b. 如果当前元素大于目标值，说明目标值可能在该元素的左侧，因此列数减1，缩小搜索范围。
 - c. 如果当前元素小于目标值，说明目标值可能在当前元素的下方，因此行数加1，缩小搜索范围。
3. 如果搜索范围越界仍未找到目标值，则返回false,表示未找到目标值。

基本实现-C

```
1 #include <stdio.h>
2 #include <stdbool.h>
3
4 bool searchInArray(int** array, int rows, int cols, int target) {
5     // 搜索的起始位置
6     int row = 0;
```

```

7     int col = cols - 1;
8
9     // 不能超过搜索的边界 (rows为行数, 这里作为位序)
10    while (row < rows && col >= 0) {
11        if (array[row][col] == target) {
12            return true;
13        }else if (array[row][col] > target) {
14            col--; // 目标值可能在当前元素的左侧
15        }else {
16            row++; // 目标值可能在当前元素的下方
17        }
18    }
19    return false; // 超出搜索范围, 未找到目标值。
20 }

```

基本实现C++

```

1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  bool searchInArray(vector<vector<int>>& array, int target) {
6      int rows = array.size(); // 矩阵的行数
7      int cols = array[0].size(); // 矩阵的列数
8
9      // 搜索的起始位置
10     int row = 0;
11     int col = cols - 1;
12
13     while (row < rows && col >= 0) {
14         if(array[row][col] == target) {
15             return true; // 找到目标值
16         } else if (array[row][col] > target) {
17             col--; // 目标值可能在当前元素的左侧
18         }else {
19             row++; // 目标值可能在当前元素的下方
20         }
21     }
22     return false; // 搜索范围越界, 未找到目标值
23 }
24

```

球友解答

Title: _____

Date: _____

13



在一个二维数组array中（每个一维数组的长度相同），每一行都按照从左到右递增的顺序排序，每一列都按照从上到下递增的顺序排序。请完成一个函数，输入这样的一个二维数组和一个整数，判断数组中是否含有该整数。

由题知该二维数组有序，从左上角元素开始

遍历

```
bool find(int a[][], int m, int n, int target) {
```

```
    int i = 0; // 从A[0][0]开始查找
```

```
    int j = n - 1;
```

```
    while (i < m && j >= 0) {
```

```
        if (A[i][j] == target) // 查找成功
```

```
            return true;
```

```
        else if (target < A[i][j])
```

```
            j--;
```

// 目标小，左移

```
        else
```

```
            i++;
```

// 目标大，下移

```
    }
```

```
    return false; }
```

3、总结



总结栏

蓝蓝B站首页: [蓝蓝希望你上岸呀B站首页](#)

蓝蓝公众号: [算法训练营9分计划](#)

蓝蓝知识星球介绍: [关于知识星球的权益](#)

如何在星球打卡记录：

- 计算机考研数据结构算法专项day[1/60]:
- 学习内容：最好能发出自己写的图片
- 遇到的问题：如果无就不用写了
- 小结：这部分一周写一次即可。