杨氏矩阵

有一个数字矩阵，矩阵的每行从左到右是递增的，矩阵从上到下是递增的请编写程序在这样的矩阵中查找某个数字是否存在。要求:时间复杂度小于0(N);

对于这种题目来说，

struct point

{

int x;

int y;

};

struct point find\_num(int (\*arr)[3], int row, int col, int n)

{

int i = row-1;

int j = 0;

struct point p = {-1,-1};

while (( i >= 0) && (j < col ))

{

int flag = 0;

if (n < arr[i][j])

{

if (flag == 0)

i -= 1;

else if (flag == 1)

return p;

}

else if (n > arr[i][j])//j++进来以后 如果再去i-- 就说明找不到了

{

j += 1;

flag = 1;

}

else

{

p.x = i +1;

p.y = j + 1;

return p;

}

}

return p;

}

void test1()

{

int arr[3][3] = { 1,2,3,4,5,7,8,9,10};

int input = 0;

scanf("%d", &input);

struct point p=find\_num(arr,3,3,input);

printf("%d %d\n", p.x, p.y);

}

我们想要实现一个函数 帮我们将元素找出来以后，返回他的坐标。但是，return只能返回一个值，但是坐标起码两个值或以上，因此，我们可以定义一个结构体类型，

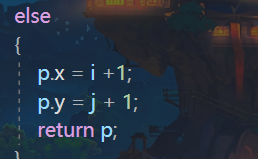
方法一：自定义结构体以返回坐标



并且将函数的返回类型也定义为此自定义结构体类型



当程序找到在数组内的元素后，



将坐标信息传给自定义的结构体后，return出来



在主函数内，用同样的自定义结构体类型将其接受后，就能够打印出相印的下标类型了。

方法二：传参指针 返回坐标

int find\_num2(int(\*arr)[3], int\* row, int\* col, int n)//传参指针 返回坐标

{

int i = \*row - 1;

int j = 0;

while ((i >= 0) && (j < \*col))

{

int flag = 0;

if (n < arr[i][j])

{

if (flag == 0)

i -= 1;

else if (flag == 1)

{

\*row = -1;

\*col = -1;

return 0;

}

}

else if (n > arr[i][j])//j++进来以后 如果再去i-- 就说明找不到了

{

j += 1;

flag = 1;

}

else

{

\*row = i + 1;

\*col = j + 1;

return 1;

}

}

\*row = -1;

\*col = - 1;

return 0;

}

void test2()

{

int arr[3][3] = { 1,2,3,4,5,7,8,9,10 };

int input = 0;

scanf("%d", &input);

int x = 3;

int y = 3;

int ret = find\_num2(arr, &x, &y, input);

printf("%d %d\n",x,y);

}

此种方法，将行列信息先以指针的方式传给函数后，再通过解引用操作将主函数中的两个行列指针改为坐标信息。