在一个二维数组中，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。

两个思路。

1. 先列二分，再从上到下每行遍历二分法。
2. 一种从左下角开始，小的上移，大的右移。

//1.列二分，遍历列，行二分：

 public class Solution {

   public boolean Find(int target, int [][] array) {

                           int col = array [0].length;//列

                           int cur = array.length;//行

                           boolean Final = false;

                           int cur\_min = 0;

                           int cur\_max = cur - 1;

                           int col\_min = 0;

                           int col\_max = col - 1;

                           while(cur\_min<cur\_max){

                               int mid = (cur\_min+cur\_max)/2;

                              if((array[cur\_max][col\_min]>target)&&(array[mid][col\_min]>target)){

                                   cur\_max = mid - 1;

                                   continue;

                               }elseif(array[cur\_max][col\_min]==target){

                                   return true;

                               }else{

                                   //cur\_min = cur\_max;

                                   break;

                               }

                           }

                            int temp1 = col\_min;

                           int temp2 = col\_max;

                           for(int i = 0;i<=cur\_max;i++){

                                    while(col\_min<=col\_max){

                                        intmid = (col\_min+col\_max)/2;

                                        if((array[i][col\_max]>target)&&(array[i][mid]<target)){

                                            col\_min = mid + 1;

                                            continue;

                                        }elseif((array[i][col\_max]>target)&&(array[i][mid]>target)){

                                                 col\_max= mid - 1;

                                            continue;

                                        }else if(((array[i][col\_max]>target)&&(array[i][mid]==target))||(array[i][col\_max]==target)){

                                            return true;

                                        }else{

                                                 break;

                                        }

                                    }

                                    col\_min= temp1;

                                    col\_max= temp2;

                           }

                           return false;

                       }

｝

//2.一种从左下角开始，小的上移，大的右移。

Public class Solution {

    public boolean Find(int [][] array,int target) {

        for(int i=0;i<array.length;i++){

            int low=0;

            int high=array[i].length-1;

            while(low<=high){

                int mid=(low+high)/2;

                if(target>array[i][mid])

                    low=mid+1;

                else if(target<array[i][mid])

                    high=mid-1;

                else

                    return true;

            }

        }

        return false;

    }

}

public class Solution {

public static boolean Find(int target, int [][] array) {

int len = array.length-1;

int col = array[0].length-1;

int i = 0;

while(len>=0&&i<=col){

if(array[len][i]>target){

len--;

continue;

}else if(array[len][i]<target){

i++;

continue;

}else{

return true;

}

}

return false;

}

}