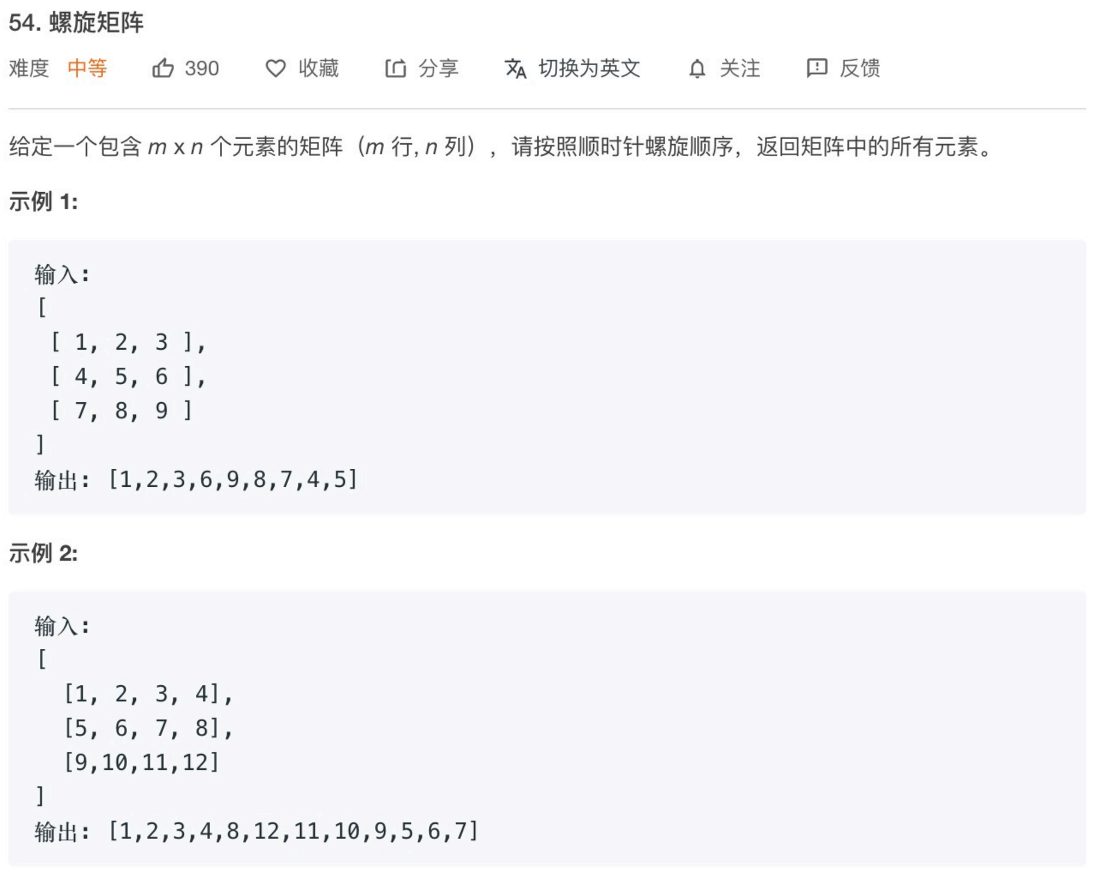
Algorithms

54. 螺旋矩阵

问题描述



54. 螺旋矩阵

难度 中等　390　收藏　分享　切换为英文　关注　反馈

给定一个包含 *m* x *n* 个元素的矩阵（*m* 行, *n* 列），请按照顺时针螺旋顺序，返回矩阵中的所有元素。

示例 1:

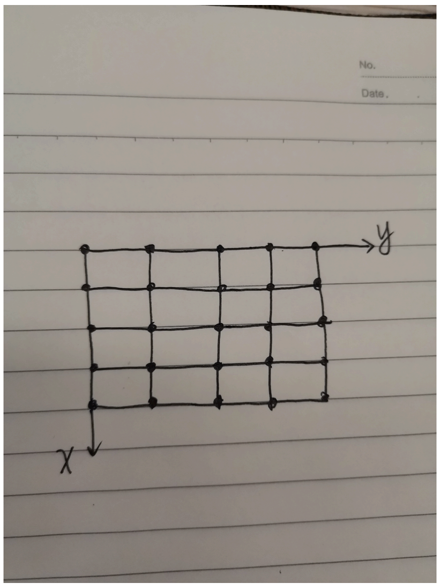
```
输入:
[
 [ 1, 2, 3 ],
 [ 4, 5, 6 ],
 [ 7, 8, 9 ]
]
输出: [1,2,3,6,9,8,7,4,5]
```

示例 2:

```
输入:
[
  [1, 2, 3, 4],
  [5, 6, 7, 8],
  [9,10,11,12]
]
输出: [1,2,3,4,8,12,11,10,9,5,6,7]
```

解法

自原点起，顺时针走完所有节点



```java
import java.awt.Point;
import java.util.ArrayList;
import java.util.List;

class Solution {
  public List<Integer> spiralOrder(int[][] matrix) {
    List<Integer> result = new ArrayList<>();
    if (null == matrix || matrix.length == 0) {
      return result;
    }
    int maxRSize = matrix.length;
    int maxCSize = matrix[0].length;
    Point maxPoint = new Point(maxRSize - 1, maxCSize - 1);
    boolean[][] visitMatrix = new boolean[maxRSize][maxCSize];
    int direc = 0;  // 0-右 1-下 2-左 3-上
    Point point = new Point(0, 0);
    for (int i = 0; i < maxRSize * maxCSize; i++) {
      // 输出当前节点
      result.add(matrix[point.x][point.y]);
      visitMatrix[point.x][point.y] = true;
      // 确定下一个节点
      Point wishPoint = getWishPoint(point, direc);
      if (check(maxPoint, visitMatrix, wishPoint)) {
        point = wishPoint;
        continue;
      }
      direc++;
      if (direc == 4) {
        direc = 0;
      }
      point = getWishPoint(point, direc);
    }
    return result;
  }

  private Point getWishPoint(Point point, int direc) {
    int wishX = point.x;
    int wishY = point.y;
    if (direc == 0) {
      wishY++;
    } else if (direc == 1) {
      wishX++;
    } else if (direc == 2) {
      wishY--;
    } else {
      wishX--;
    }
    return new Point(wishX, wishY);
  }

  private boolean check(Point maxPoint, boolean[][] visitMatrix, Point wishPoint) {
    return wishPoint.x >= 0 &&
           wishPoint.y >= 0 &&
           wishPoint.x <= maxPoint.x &&
           wishPoint.y <= maxPoint.y &&
           !visitMatrix[wishPoint.x][wishPoint.y];
  }
}
```