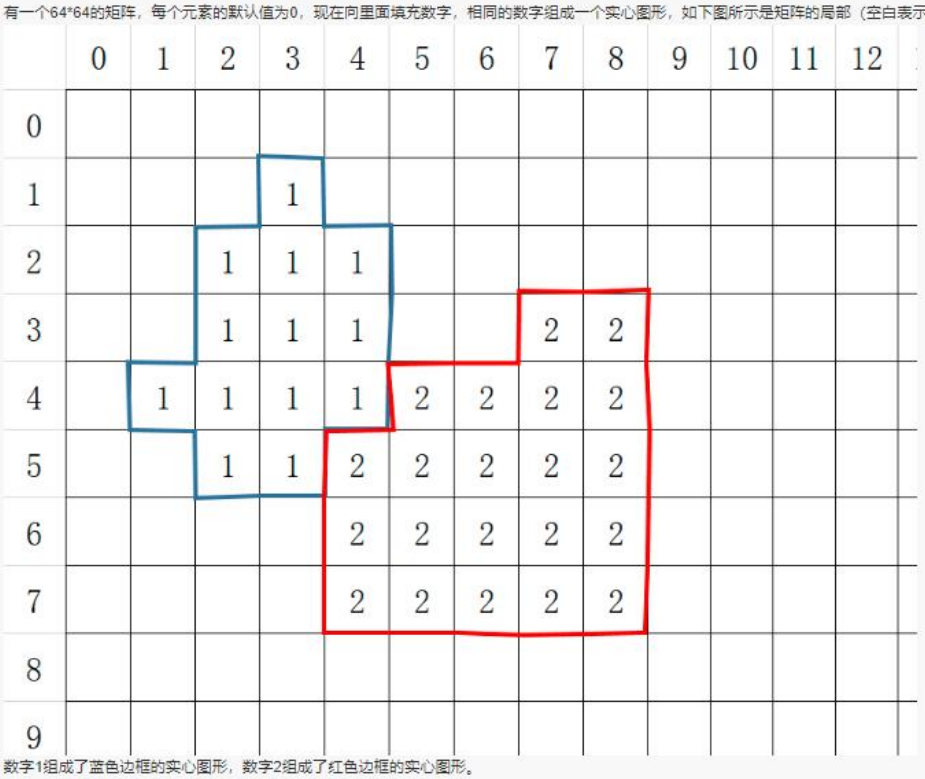
题目描述：有一个64*64的矩阵，每个元素的默认值为0，现在向里面填充数字，相同的数字组成一个实心图形，如下图所示是矩阵的局部（空白表示填充0）：



数字1组成了蓝色边框的实心图形，数字2组成了红色边框的实心图形。

单元格的边长规定为1个单位，请根据输入，计算每个非0值填充出来的实心图形的周长。

输入描述：

2

1 1 3 2 2 2 3 2 4 3 2 3 3 3 4 4 1 4 2 4 3 4 4 5 2 5 3

2 3 7 3 8 4 5 4 6 4 7 4 8 5 4 5 5 5 6 5 7 5 8 6 4 6 5 6 6 6 7 6 8 7 4 7

5 7 6 7 7 7 8

输入数据说明如下：

1、第一行输入 N，表示一共有 N 个图形，N>0 且 N<64*64；

2、矩阵左上角单元格坐标记做（0,0），第一个数字表示行号，第二个数字表示列号；

3、接下来是 N 行，每行第一个数字是矩阵单元格填充的数字，后续每两个一组，表示填充该数字的单元格的坐标；

4、答题者无需考虑数据格式非法的场景，题目用例不考察数据格式；

5、题目用例保证同一个填充值只会有一行输入数据。

输出描述：

18 20

1、一共输出 N 个数值，每个数值表示某一输入行表示图形的周长；

2、输出顺序需和输入的各行顺序保持一致，即第 1 个数是输入的第 1 个图形的周长，第 2 个数是输入的第 2 个图形的周长，以此类推。

补充说明：

示例 1

输入：

```
2
1 1 3 2 2 2 3 2 4 3 2 3 3 3 4 4 1 4 2 4 3 4 4 5 2 5 3
2 3 7 3 8 4 5 4 6 4 7 4 8 5 4 5 5 5 6 5 7 5 8 6 4 6 5 6 6 6 7 6 8 7 4 7 5 7 6 7 7 7 8
```

输出：

```
18 20
```

说明：

本样例中，经过观察和计算，1 组成的图形的周长为 18 个单位，2 组成的图形的周长为 20 个单位。

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        // 2
        //1 (1 3) (2 2) (2 3) (2 4) (3 2) (3 3) (3 4) (4 1) (4 2) (4 3) (4 4) (5 2) (5 3)
        //2 (3 7 (3 8 (4 5 (4 6 (4 7 (4 8 (5 4 (5 5 (5 6 (5 7 (5 8 (6 4 (6 5 (6 6 (6 7 (6 8 (7 4 (7 5 (7 6 (7 7 (7 8
        int count = Integer.parseInt(in.nextLine());
```

```
            for (int i = 0; i < count; i++) {
                String[] t = in.nextLine().split(" ");
                int num = Integer.parseInt(t[0]);
                int size = (t.length - 1) / 2;
                int[][] point = new int[64][64];
                List<Zuo> list = new ArrayList<>();
                for (int j = 1; j < t.length; j += 2) {
                    int x = Integer.parseInt(t[j]);
                    int y = Integer.parseInt(t[j + 1]);
                    point[x][y] = num;
                    list.add(new Zuo(x, y));
                }

                // 开始处理
                int ans = size * 4;
                for (int j = 0; j < list.size(); j++) {
                    for (int k = 0; k < list.size(); k++) {
                        if (k == j) {
                            continue;
                        }
                        Zuo a = list.get(k);
                        Zuo b = list.get(j);
                        if ((a.x==b.x||a.y==b.y)
&&(Math.abs(a.x-b.x)==1||Math.abs(a.y-b.y)==1)) {
                            ans--;
                        }
                    }
                }
                System.out.print(ans + " ");


            }

        }
}

class Zuo {
    int x;
    int y;
    public Zuo(int x, int y) {
        this.x = x;
        this.y = y;
    }
```

}