

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 ）：

	0	1	2	3	4	5	6	7	8	9	10
0											
1				1							
2			1	1	1						
3			1	1	1			2	2		
4		1	1	1	1	2	2	2	2		
5			1	1	2	2	2	2	2		
6					2	2	2	2	2		
7					2	2	2	2	2		
8											
9											

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。

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。

```
def calc(str1):
    ground=[['0' for i in range(64)] for j in range(64)]
    fill=str1[0]
    line=str1.split()[1:]
    n=len(line)//2
    res=4*n
    for i in range(n):
        x=int(line[2*i])
        y=int(line[2*i+1])
        ground[x][y]='1'
    for i in range(n):
        x=int(line[2*i])
        y=int(line[2*i+1])
        cnt=0
        if 0<=x-1<64 and 0<=y<64:
            if ground[x-1][y]=='1':
                cnt+=1
        if 0<=x<64 and 0<=y-1<64:
            if ground[x][y-1]=='1':
                cnt+=1
        if 0<=x<64 and 0<=y+1<64:
            if ground[x][y+1]=='1':
                cnt+=1
        if 0<=x+1<64 and 0<=y<64:
            if ground[x+1][y]=='1':
                cnt+=1
        res=res-cnt
    return res
```

```
N=int(input())
```

```
for i in range(N):
```

```
    print(calc(input()),end=' ')
```