

数值同化

题目描述：

存在一个 $m \times n$ 的二维数组，其成员取值范围为 $0,1,2$ 。其中值为 1 的元素具备同化特性，每经过 $1S$ ，将上下左右值为 0 的元素同化为 1 。而值为 2 的元素，免疫同化。将数组所有成员随机初始化为 0 或 2 ，再将矩阵的 $[0,0]$ 元素修改成 1 ，在经过足够长的时间后，求矩阵中有多少个元素是 0 或 2 （即 0 和 2 数量之和）。

输入描述：

输入的前两个数字是矩阵大小。后面的数字是矩阵内容。

输出描述：

返回矩阵中非 1 的元素个数

补充说明：

m 和 n 不会超过 30 (含 30)。

示例 1

输入：

```
4 4
0 0 0 0
0 2 2 2
0 2 0 0
0 2 0 0
```

输出：

```
9
```

说明：

输入数字前两个数字是矩阵大小。后面的是数字是矩阵内容。
这个矩阵的内容如下：

```
{
    0,0,0,0
    0,2,2,2
    0,2,0,0
}
```

```
0,2,0,0  
}
```

起始位置 $(0,0)$ 被修改为 1 后，最终只能同化矩阵为：

```
{  
  
1,1,1,1  
  
1,2,2,2  
  
1,2,0,0  
  
1,2,0,0  
}
```

所以矩阵中非 1 的元素个数为 9 。

```
import sys
```

```
line = sys.stdin.readline().strip('\n')
```

```
m, n = map(int, line.split(' '))
```

```
mat = []
```

```
for line in sys.stdin:
```

```
    l = line.strip('\n')
```

```
    a = l.split(' ')
```

```
    a = list(map(int, a))
```

```
    mat += [a]
```

```
dic_path = dict()
```

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

```
    for j in range(n):
```

```
        towards = []
```

```
            for di, dj in [[0, 1], [0, -1], [1, 0], [-1, 0]]:
```

```
                if 0 <= i + di < m and 0 <= j + dj < n and mat[i + di][j + dj] == 0:
```

```
                    towards += [';', join(map(str, [i + di, j + dj]))]
```

```
            k = ';'.join(map(str, [i, j]))
```

```
            dic_path[k] = towards
```

```
visited = set()
```

```
que = ['0,0']
```

```
while True:
```

```
    q = que.pop(0)
```

```
    if q not in visited:
```

```
        visited.add(q)
```

```
        que.extend(dic_path[q])
```

```
    if que == []:
```

```
        break
```

```
print(m * n - len(visited))
```