Peak 2D

ข้อมูล 2 มิติขนาด R x C สามารถเขียนแทนด้วยลิสต์ขนาด R ช่อง โดยที่แต่ละช่องเก็บลิสต์ของจำนวนเต็ม C ตัว จงเขียนฟังก์ชัน count_peak(data) ที่คืนค่าเป็นจำนวนเต็มแทนจำนวนจุดสูงสุดจากข้อมูล 2 มิติที่กำหนดให้

```
def read_data():
    dat = []
    R = int(input())
    for r in range(R):
        dat.append([int(e) for e in input().strip().split()])
    return dat
def count_peak(data):
    # เขียนโค้ดตรงนี้


exec(input().strip()) # ต้องมีคำสั่งนี้ตอนส่งให้ grader ตรวจ
```

โดย "จุดสูงสุด" คือ <u>ข้อมูลที่มีค่า**มากกว่า**ข้อมูล ด้านซ้าย ด้านขวา ด้านบน และด้านล่าง</u>
(แปลว่าถ้าข้อมูลอยู่ริมขอบด้านใดด้านหนึ่ง หรือมุมด้านใดด้านหนึ่ง จะไม่นับเป็นจุดสูงสุด)

หาก data มีค่าเป็น [[0, 1, 2, 3], [1, 9, 2, 3], [2, 1, 8, 3], [3, 3, 3, 3]] จะสามารถแสดงเป็น 2 มิติได้เป็น

```
0 1 2 3
1 9 2 3
2 1 8 3
3 3 3 3
```

จุดริมขอบ จะไม่นับเป็นจุดสูงสุด

8 นับเป็นจุดสูงสุดเพราะว่ามีค่ามากกว่า ข้อมูลด้านบน (2) ด้านซ้าย (1) ด้านล่าง (3) ด้านขวา (3)

จุดสูงสุดมี 2 จุดได้แก่ 9 8

ข้อมูลนำเข้า

คำสั่งภาษา Python ที่ใช้ทดสอบการทำงานของฟังก์ชัน

ข้อมูลส่งออก

ผลที่ได้รับจากการสั่งทำงานคำสั่งที่ได้รับ

ตัวอย่าง

input (จากแป้นพิมพ์)	output (ทางจอภาพ)

`dat=read_data();print(count_peak(dat))`
`4`
`0 1 2 3`
`1 9 2 3`
`2 1 8 3`
`3 3 3 3`
	`2`

`dat=read_data();print(count_peak(dat))`
`3`
`0 0 0`
`0 1 0`
`0 0 0`
	`1`

`dat=read_data();print(count_peak(dat))`
`3`
`0 0 0`
`0 0 0`
`0 0 0`
	`0`

input (จากแป้นพิมพ์)	output (ทางจอภาพ)
dat=read_data();print(count_peak(dat)) 3 0 1 2 3 4 5 6 7 8 9 0 1	0
dat=read_data();print(count_peak(dat)) 5 -4 -3 -2 -1 0 3 99 -2 12 7 12 -111 -1 -99 5 55 111 -200 -1 -99 -4 -3 -2 -1 0	4
print(count_peak([[0,1,2,3],[1,9,2,3],[2,1,8,3],[3,3,3,3]]))	2