

## 2559\_2\_Function\_Tiling\_Puzzle

คงเคยเล่นเกมเลื่อนแผ่นพลาสติกเล็กๆ จำนวน 15 อัน ให้เรียง 1 ถึง 15 จากซ้ายไปขวาลงล่าง ดังตัวอย่างในรูปข้างล่างนี้

1	7	2	3
6		8	4
5	9	10	11
13	14	15	12



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

เราสามารถแทนตารางข้างบนนี้ด้วย **list of lists of ints** (แทนช่องว่างด้วยเลข 0) เช่น ตารางรูปข้างจะแทนด้วย

```
[ [1,7,2,3], [6,0,8,4], [5,9,10,11], [13,14,15,12] ]
```

ให้สังเกตว่า ไม่ใช่ทุกตารางจะสามารถเลื่อนไปสู่เป้าหมายที่ต้องการได้ เช่น แคสลับ 14 กับ 15 ในตารางทางขวบนนี้ ซึ่งแทนด้วย

```
[ [1,2,3,4], [5,6,7,8], [9,10,11,12], [13,15,14,0] ]
```

 ก็ไม่สามารถเลื่อนไปสู่เป้าหมายได้ (ขอไม่พิสูจน์)

โจทย์ข้อนี้เกี่ยวกับการตรวจตาราง (ในรูปของ **list of lists**) ว่าเป็นตารางที่สามารถเลื่อนไปหาเป้าหมายได้หรือไม่

เราตรวจได้โดยไม่ต้องลงมือเลื่อนหมายเลขในตาราง ดังนี้ (การตรวจสอบนี้ใช้ได้กับตารางที่มีขนาด  $n \times n$  ใดๆ)

1. **flatten**: เปลี่ยน **list of lists of ints** เป็น **list of ints** โดยตัด 0 ทิ้ง เช่น (ขอแสดงกรณีตารางขนาด  $3 \times 3$ )

จาก [ [1,2,0], [3,5,6], [4,7,8] ] ก็เปลี่ยนเป็น [1,2,3,5,6,4,7,8]

2. **inversions**: หาจำนวน **inversion** ซึ่งคือจำนวนคู่ของข้อมูลใน **list of ints** ว่า **มีกี่คู่ที่ตัวซ้ายมากกว่าตัวขวา**

เช่น จากตัวอย่างข้างบนนี้ มีข้อมูล 8 ตัว ก็มีทั้งหมด  $8 \times 7 / 2 = 28$  คู่

ดังนี้ (1,2), (1,3), (1,5), (1,6), (1,4), (1,7), (1,8), (2,3), (2,5), (2,6), (2,4), (2,7), (2,8), (3,5), (3,6), (3,4), (3,7), (3,8), (5,6), (5,4), (5,7), (5,8), (6,4), (6,7), (6,8), (4,7), (4,8), (7,8)

ซึ่งมี 2 คู่ที่ตัวซ้ายมากกว่าตัวขวา ดังนั้น จำนวน **inversion** จึงมีค่าเป็น 2

3. ตารางที่ได้รับ จะเลื่อนได้ไปสู่เป้าหมายได้ ก็เมื่อมีลักษณะ ตรงตามเงื่อนไขข้างล่างนี้

จำนวนแถวของตาราง	จำนวน <b>inversions</b>	หมายเลขแถวของตารางที่เลข 0 อยู่ (แถวบนสุดคือแถวที่ 0)
เลขคี่	เลขคู่	อยู่แถวใดก็ได้
เลขคู่	เลขคี่	เลขคู่
	เลขคู่	เลขคี่

จากตัวอย่างในขั้นตอนที่ 1 ตารางมีจำนวน 3 แถวเป็นเลขคี่ จำนวน **inversions** เป็นเลขคู่ จึงสามารถเลื่อนไปยังเป้าหมาย

ได้(รายละเอียดอ่านเพิ่มเติมได้ที่ <https://www.cs.bham.ac.uk/~mdr/teaching/modules04/java2/TilesSolvability.html>)

จึงเขียนฟังก์ชันต่าง ๆ ที่ทำงานตาม **comment** ที่เขียนไว้ในโครงของโปรแกรมข้างล่างนี้

```
def row_number(t, e): # return row number of t containing e (top row is row #0)

def flatten(t):      # return a list of ints converted from list of lists of ints t

def inversions(x):    # return the number of inversions of list x

def solvable(t):      # return True if tiling t (list of lists of ints) is solvable
                      # otherwise return False

exec(input().strip()) # do not remove this line
```

### ข้อมูลนำเข้า

คำสั่งในการทดสอบฟังก์ชันที่เขียน

### ข้อมูลส่งออก

ผลที่ได้จากคำสั่งที่ป้อนเป็นข้อมูลนำเข้า

## ตัวอย่าง

input	output (ทางจอภาพ)
<code>print(row_number([[0,8,7],[6,5,4],[3,2,1]], 0))</code>	0
<code>print(flatten([[0,8,7],[6,5,4],[3,2,1]]))</code>	[8, 7, 6, 5, 4, 3, 2, 1]
<code>print(inversions([8,7,6,5,4,3,2,1]))</code>	28
<code>print(solvable([[0,8,7],[6,5,4],[3,2,1]]))</code>	True