

## Convolution

จงเขียนฟังก์ชัน `convolute(M, K)` รับ `M` และ `K` เป็นลิสต์ซ้อนลิสต์ ที่แทนเมทริกซ์ของจำนวนเต็ม โดยที่ `M` และ `K` มีขนาดเท่ากัน ฟังก์ชันนี้คืนจำนวนเต็มที่เป็นผลรวมของผลคูณของจำนวนในเมทริกซ์ `M` และ `K` ที่ตำแหน่งเดียวกัน เช่น

$$\begin{matrix} & M & & K \\ \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} & * & \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \end{matrix}$$

$$\begin{aligned} &= (1 * -1) + (2 * -2) + (3 * -1) + (4 * 0) + (5 * 0) + (6 * 0) + (7 * 1) + (8 * 2) + (9 * 1) \\ &= \quad -1 \quad \quad -4 \quad \quad -3 \quad \quad +0 \quad \quad +0 \quad \quad +0 \quad \quad +7 \quad \quad +16 \quad \quad +9 \\ &= 24 \end{aligned}$$

```
def convolute(M, K):
```

```
    exec(input().strip()) #ต้องมีบรรทัดนี้เมื่อส่งไป grader
```

### ข้อมูลนำเข้า

คำสั่งในการทดสอบฟังก์ชันที่เขียน

### ข้อมูลส่งออก

ผลที่ได้จากคำสั่งที่ป้อนเป็นข้อมูลนำเข้า

### ตัวอย่าง

Input	Output
<code>print(convolute([[1,2],[3,4]],[[1,1],[1,1]]))</code>	10
<code>print(convolute([[1,2,3],[4,5,6],[7,8,9]],[[-1,-2,-1],[0,0,0],[1,2,1]]))</code>	24
<code>print(convolute([[1,2],[3,4]],[[0,0],[0,0]]))</code>	0
<code>print(convolute([[1,2],[1,2]],[[-1,-1],[1,1]]))</code>	0