



Lab	
HW	
Until	

การบ้านปฏิบัติการ 13

n-dimensional Lists and Nested Collections (20 คะแนน)ข้อกำหนด

- i. การเรียกใช้ฟังก์ชันเพื่อการทดสอบ ต้องอยู่ภายใต้เงื่อนไข `if __name__ == '__main__':` เพื่อให้สามารถ `import` ไปเรียกใช้งานจาก Script อื่น ๆ ได้อย่างเป็นมาตรฐาน
 - ii. ทุกข้อต้องมีการสร้างฟังก์ชัน `my_id()` โดยให้คืนค่าสายอักขระแทนเลขประจำตัวนักศึกษา 9 หลัก
- 1) 4 คะแนน (Lab13_1_5XXXXXXX.py) [Attachment] ให้เขียนฟังก์ชัน `matrix_mult(m1, m2)` เพื่อทำการหาผลคูณของเมทริกซ์ `m1` และ เมทริกซ์ `m2` (wikipedia: <https://goo.gl/S0DDZv>) โดยฟังก์ชันจะทำงานแบบ Non-destructive กล่าวคือจะคืนค่าผลคูณที่ได้โดยไม่เปลี่ยนแปลงเมทริกซ์ `m1` และ `m2` ที่อยู่ในในรูปแบบ List สองมิติ ทั้งนี้หากไม่สามารถหาผลคูณได้ให้คืนค่า None

"Dot Product"

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 \\ \end{bmatrix}$$

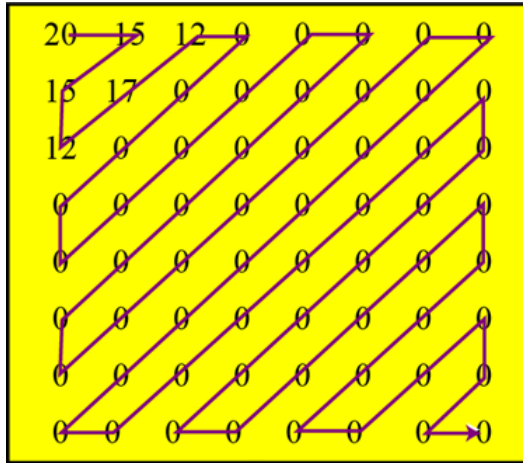
InputOutput

[[1, 2, 3], [4, 5, 6]] [[7, 8], [9, 10], [11, 12]]	[[58, 64], [139, 154]]
[[1, 2, 3], [4, 5, 6]] [[7, 8, 5, 9, 3], [9, 10, -3, 7, 13], [11, 12, 6, 2, 9]]	[[58, 64, 17, 29, 56], [139, 154, 41, 83, 131]]

- การวิเคราะห์ปัญหา

- Input: จำนวนข้อมูล _____ ชนิดข้อมูล _____
- Output: (แสดงค่า) จำนวนข้อมูล _____ ชนิดข้อมูล _____
(คืนค่า) จำนวนข้อมูล _____ ชนิดข้อมูล _____

- 2) 4 คะแนน (Lab13_2_6XXXXXXX.py) [Attachment] ในการบีบอัดภาพใน Format JPEG หลังจากทำขั้นตอน Discrete Cosine Transform แล้ว เราจะต้องอ่านข้อมูลที่ได้ใน Pixel Block ของจำนวนเต็ม ขนาด 8×8 ในรูปแบบ สลับฟันปลา ดังแสดงด้านล่าง เพื่อนำ List ที่ได้ไปทำการบีบอัดแบบ Run-Length Compression อีกที



20 15 15 12 17 12 0 0 0 0 ... 0

ให้เขียนฟังก์ชัน `read_pixel(pixel_block)` เพื่อคืนค่า List แทนตัวเลขในลำดับการอ่าน Matrix ในรูปแบบ สลับฟันปลา ดังอธิบายเมื่อ `pixel_block` เป็น Matrix จัตุรัสของจำนวนเต็ม

Input	Output
[[1, 2, 6], [3, 5, 7], [4, 8, 9]]	[1, 2, 3, 4, 5, 6, 7, 8, 9]

• การวิเคราะห์ปัญหา

- Input: จำนวนข้อมูล _____ ชนิดข้อมูล _____
- Output: (แสดงค่า) จำนวนข้อมูล _____ ชนิดข้อมูล _____
(คืนค่า) จำนวนข้อมูล _____ ชนิดข้อมูล _____

- 3) 4 คะแนน (HW13_1_6XXXXXXX.py) [Attachment] ให้เขียนฟังก์ชัน `square_matrix(list_x)` เพื่อให้ List 2 มิติ `list_x` ที่มีสมาชิกเป็นจำนวนเต็มกลายเป็น matrix จัตุรัสโดยเติม 0 เพื่อให้มีขนาด row และ column เท่ากัน โดยจะต้องคงทุก element ใน List เดิมไว้ และจำนวน 0 ที่เติมต้องเป็นจำนวนที่น้อยที่สุดที่เป็นไปได้ ทั้งนี้กำหนดให้ ฟังก์ชันทำงานแบบ Destructive และแต่ละ element จะต้องไม่เป็น alias ซึ่งกันและกัน

Input	Output
[[2, 3, 4], [1, 2, 3]]	[[2, 3, 4], [1, 2, 3], [0, 0, 0]]

- 5) 4 คะแนน (HW13_3_6XXXXXXX.py) [Attachment] ให้เขียนฟังก์ชัน `sum_d_product(m)` เพื่อคืนค่าผลบวกของผลคูณทแยงใน matrix m ที่มีขนาด $n \times n$ เมื่อ n สามารถเขียนในรูปของ 2^x (x เป็นจำนวนเต็มบวก)

a	b
c	d

โดยกรณี matrix m ขนาด 2×2 เช่น

1	2
3	4

ดังนั้น matrix จะมี `sum_d_product()` = $(1 \times 4) + (3 \times 2) = 10$

กรณีต้องการหา `sum_d_product()` ของ matrix ขนาดใหญ่กว่า 2×2 ทำได้โดยการหา `sum_d_product()`

ของ matrix ย่อย ขนาด 2×2 ก่อน แล้วหา `sum_d_product()` ของ matrix ผลลัพธ์อีกที

เช่นกรณี matrix ขนาด 4×4 จะมีขั้นตอนดังนี้

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

→

56	18
96	33

→

3576

หรือกรณี matrix 8×8

3	4	3	3	0	4	4	3
1	4	1	1	1	2	3	2
5	5	2	3	3	5	5	5
1	5	1	3	1	5	3	5
3	0	5	1	4	5	3	1
5	3	2	3	3	1	5	5
2	2	4	1	4	1	4	5
5	5	0	1	0	4	3	1

→

16	6	4	17
30	9	20	40
9	17	19	20
20	4	16	19

→

324	500
376	681

→

408644

Input

Output

[[3, 3, 3, 2], [2, 0, 3, 1], [2, 1, 2, 3], [1, 0, 2, -1]]	33
[[1, 1, 5, -1], [12, 2, -2, 0], [4, 8, 8, 12], [4, 12, 12, 15]]	3856
[[0, -1, -1, 3, 2, 3, -1, 3], [3, -1, -1, 2, 0, -1, 2, 1], [3, 0, 1, 2, 3, 1, 3, 1], [2, 2, 1, -1, -1, 2, 0, 3], [1, 3, 2, 1, 3, 2, 2, 1], [1, 2, 2, 1, 3, 3, 1, 3], [2, 2, 2, 2, 2, 2, 3, 3], [1, 3, 2, 3, 1, 1, 2, 2]]	-6290