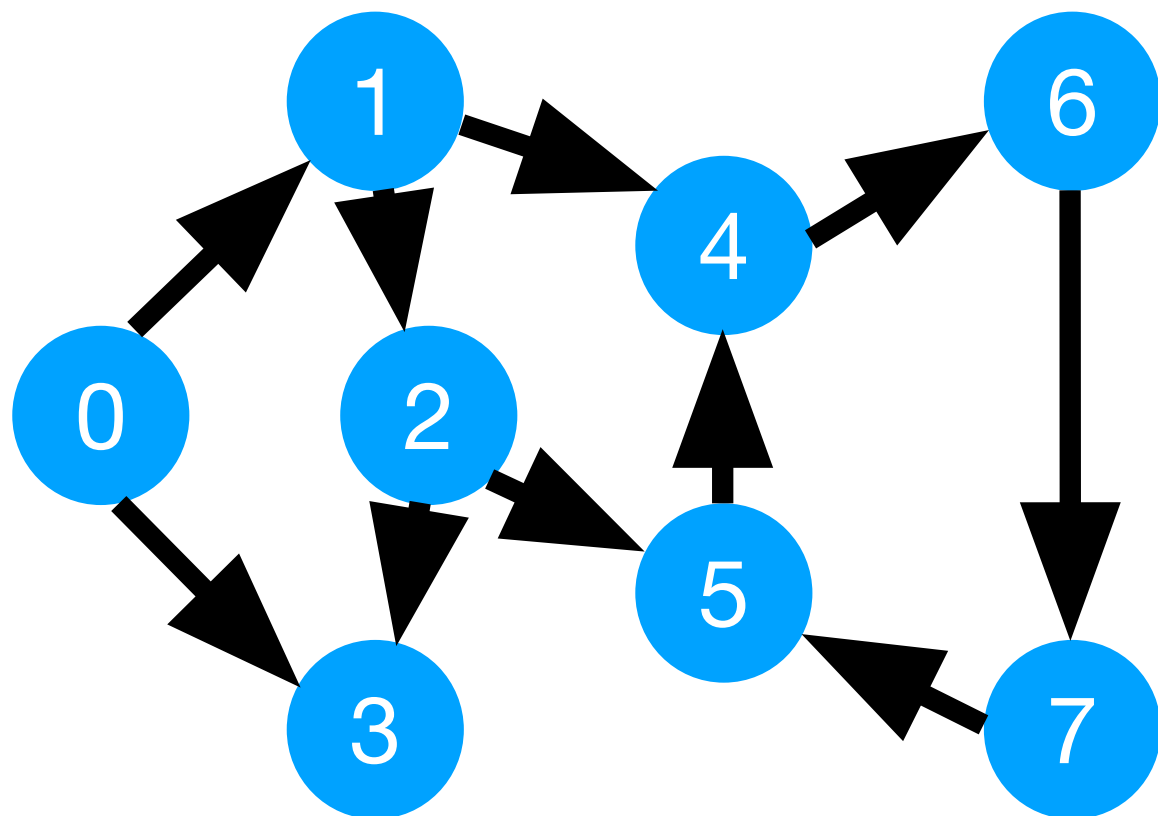# Linear Algebra
HW1 Cycle Detection

袁培傑

Outline

1. Task Introduction

2. Problem 1
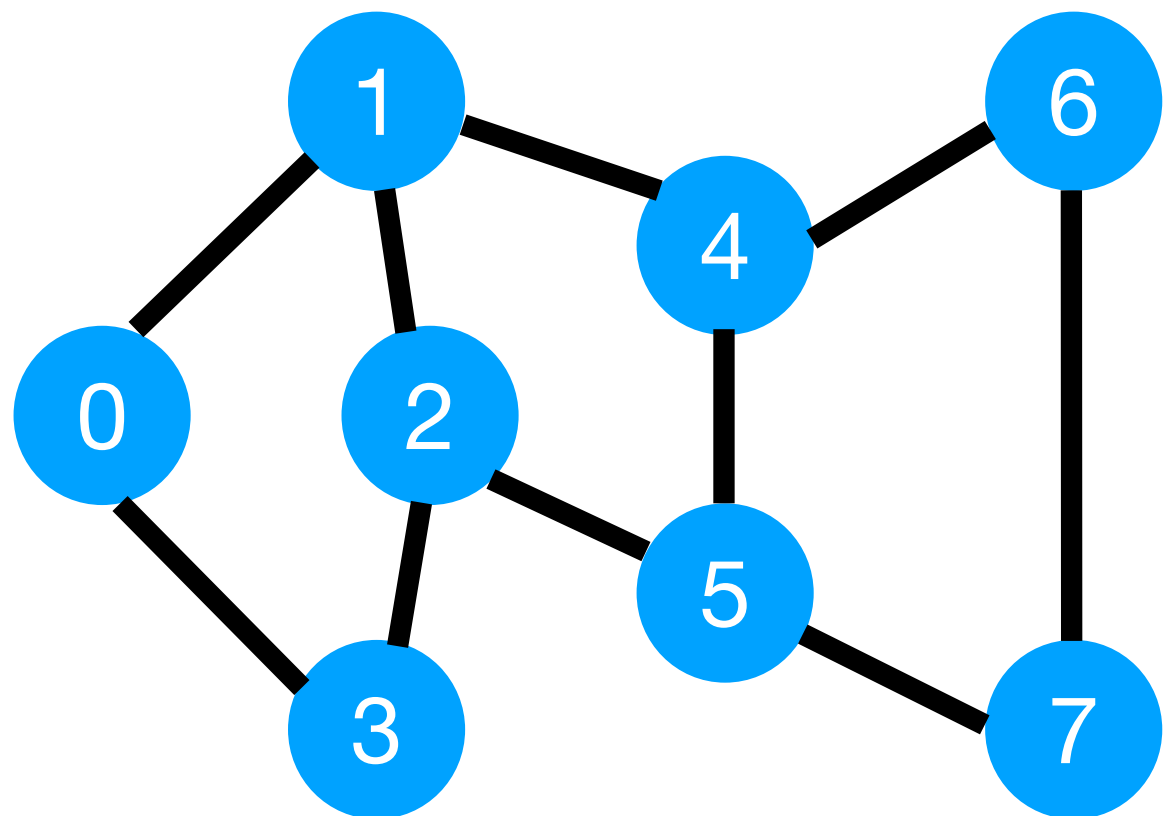
3. Problem 2

4. Python Tips

5. Rules

Task Introduction

Graph

- A graph contains some nodes and edges

- The edges can be directed or undirected

- This task is given the **directed graph**, we need to find out whether **there is a cycle** in the graph
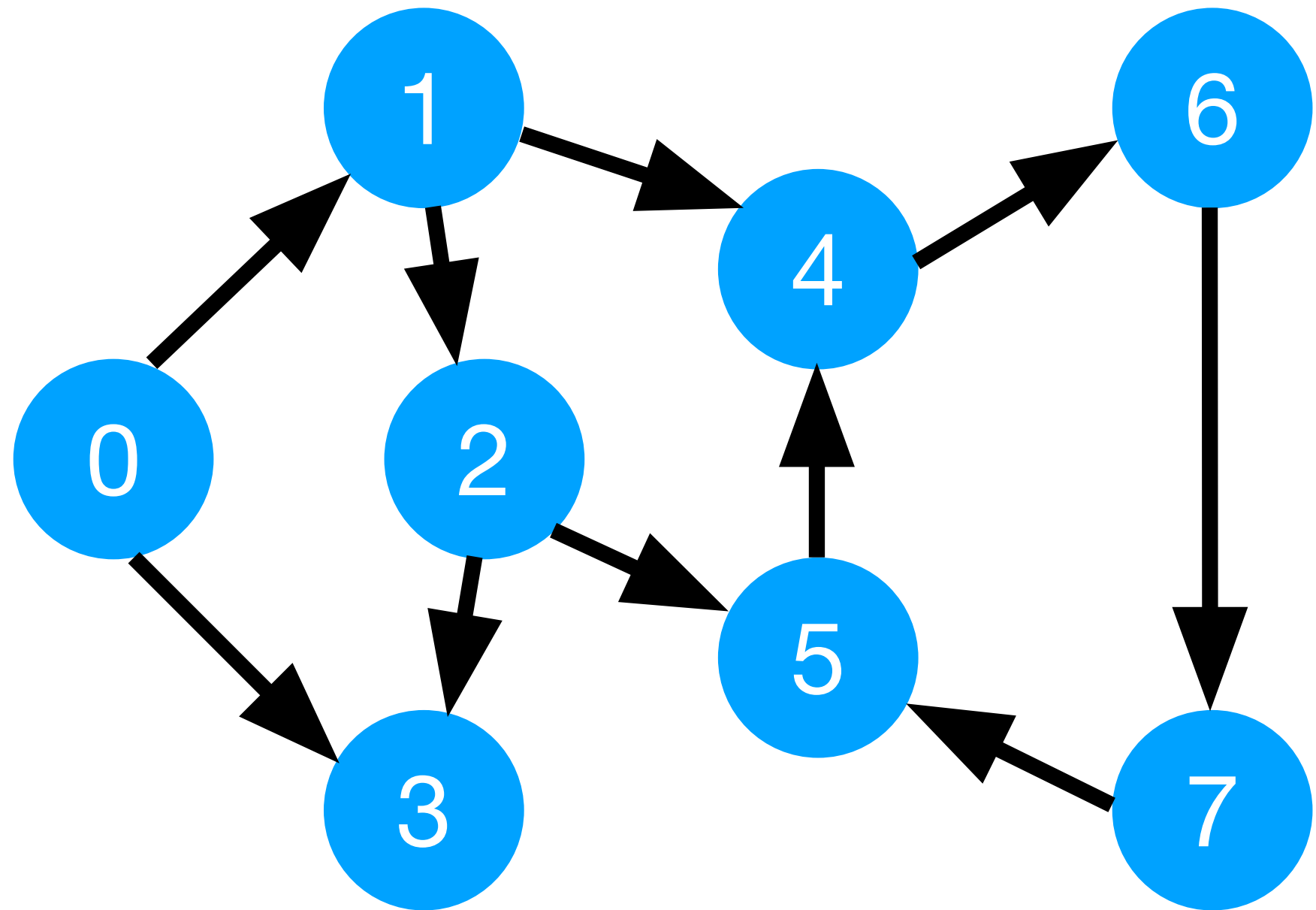

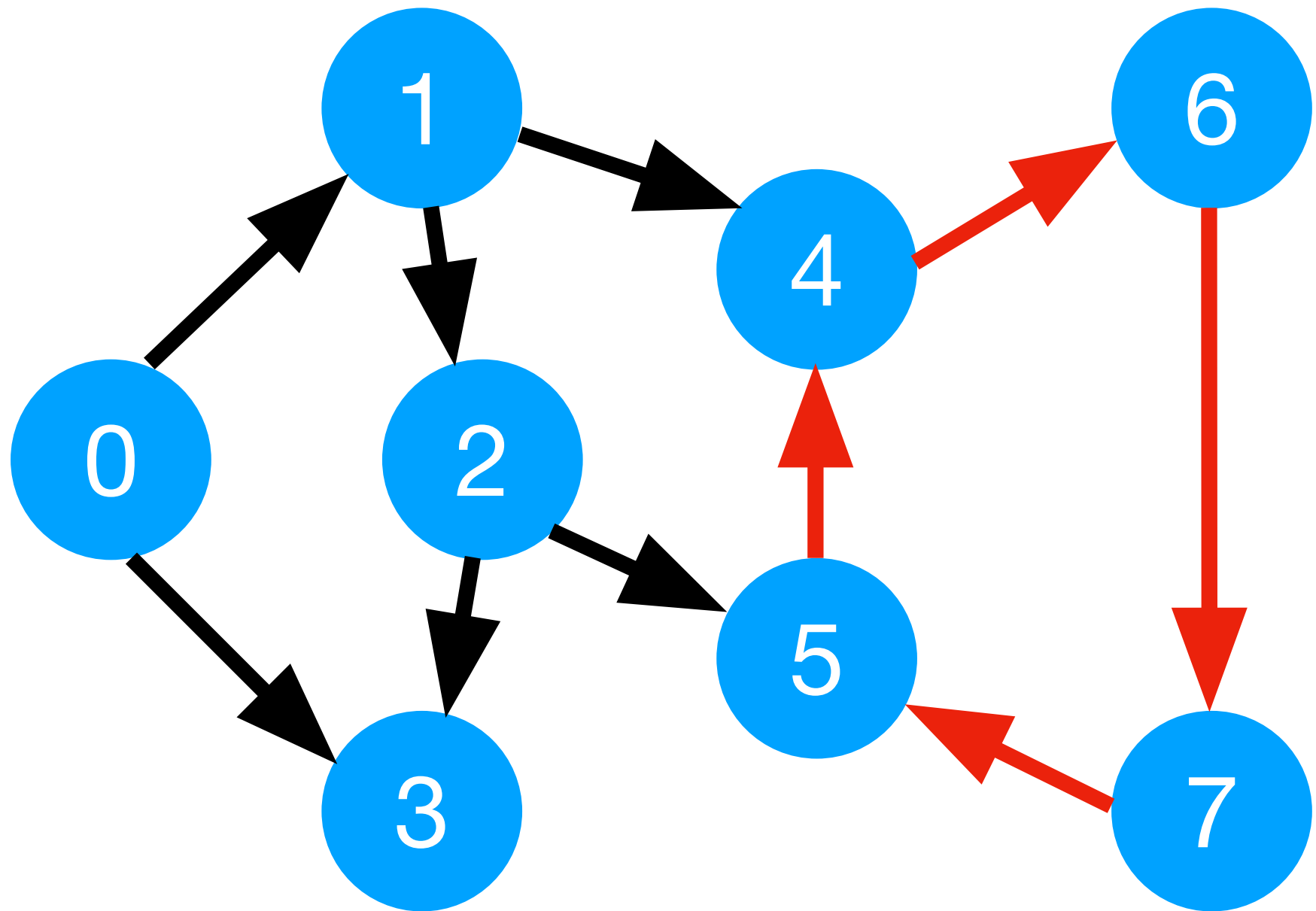**Directed Graph**   **Undirected Graph**

Directed Graph

0 → 1
0 → 3
1 → 2
1 → 4
2 → 3
2 → 5
4 → 6
5 → 4
6 → 7
7 → 5

Cycle Detection

0 → 1
0 → 3
1 → 2
1 → 4
2 → 3
2 → 5
4 → 6
5 → 4
6 → 7
7 → 5

Problem 1

Graph Representation

	0	1	2	3	4	5	6	7
0 → 1 **0**	-1	1						
0 → 3 **1**	-1			1				
1 → 2 **2**		-1	1					
1 → 4 **3**		-1			1			
2 → 3 **4**			-1	1				
2 → 5 **5**			-1			1		
4 → 6 **6**					-1		1	
5 → 4 **7**					1	-1		
6 → 7 **8**							-1	1
7 → 5 **9**						1		-1

- A row is a connection

- If a connection is from 0 to 1, the value of column 0 will be -1 and the value of column 1 will be 1
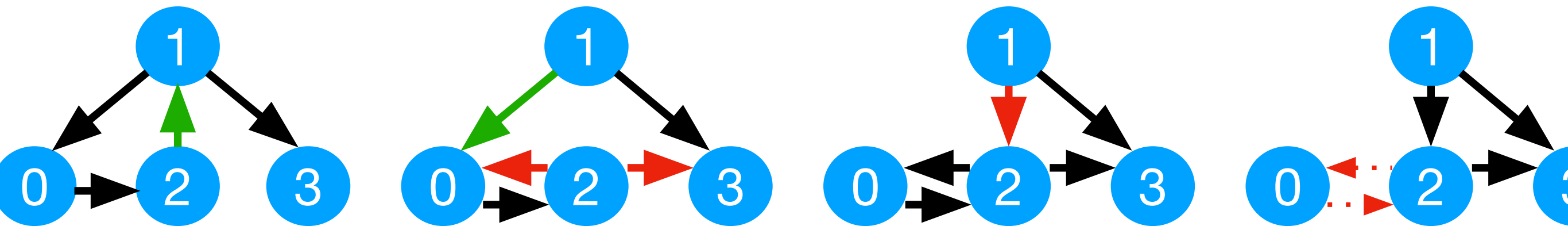
- 0 otherwise

Linear Dependent

- Linear independence in row

- In cycle detection, we can only do **ADDITION**

Linear Dependent to Detect Cycle
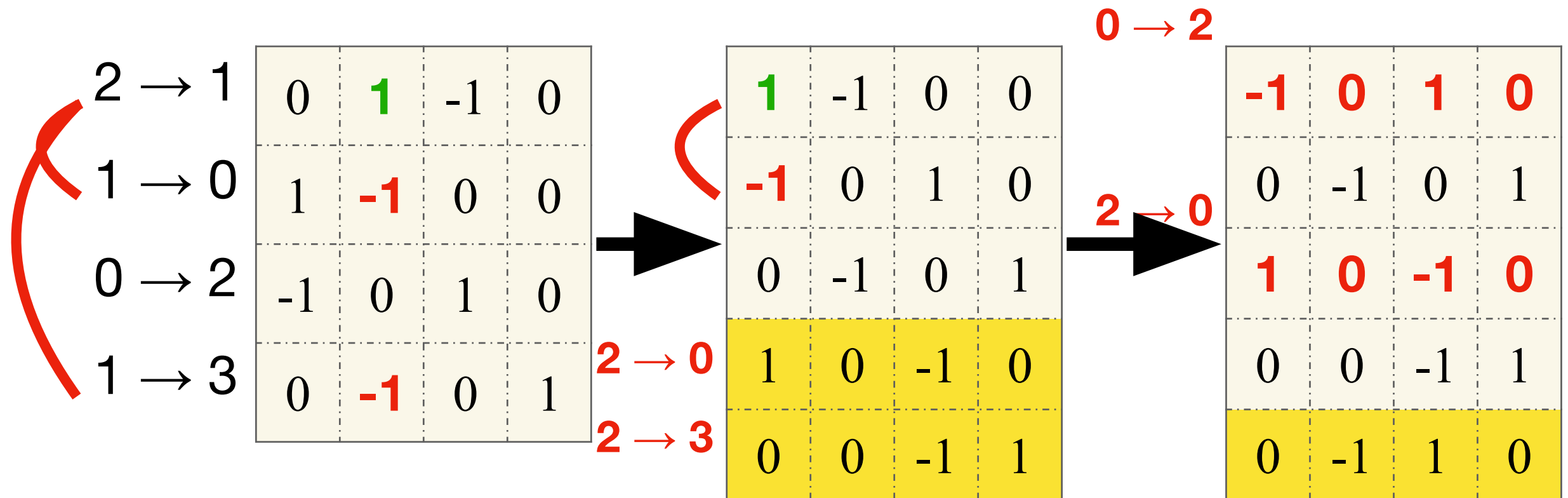
Termination

- If we get a **ALL 0** row after addition, then the graph has a cycle.

- If we do addition on all the edges and we don't get a ALL 0, then the graph does not have a cycle.

p1.py

```python
def has_cycle(sets):
    # TODO
    # return True if the graph has cycle; return False if not


    return False
'''

  HINT: You can `print(sets)` to show what the matrix looks like
    If we have a directed graph with 2->3 4->1 3->5 5->2 0->1
          0  1  2  3  4  5
      0   0  0 -1  1  0  0
      1   0  1  0  0 -1  0
      2   0  0  0 -1  0  1
      3   0  0  1  0  0 -1
      4  -1  1  0  0  0  0
    The size of the matrix is (5,6)
'''
```
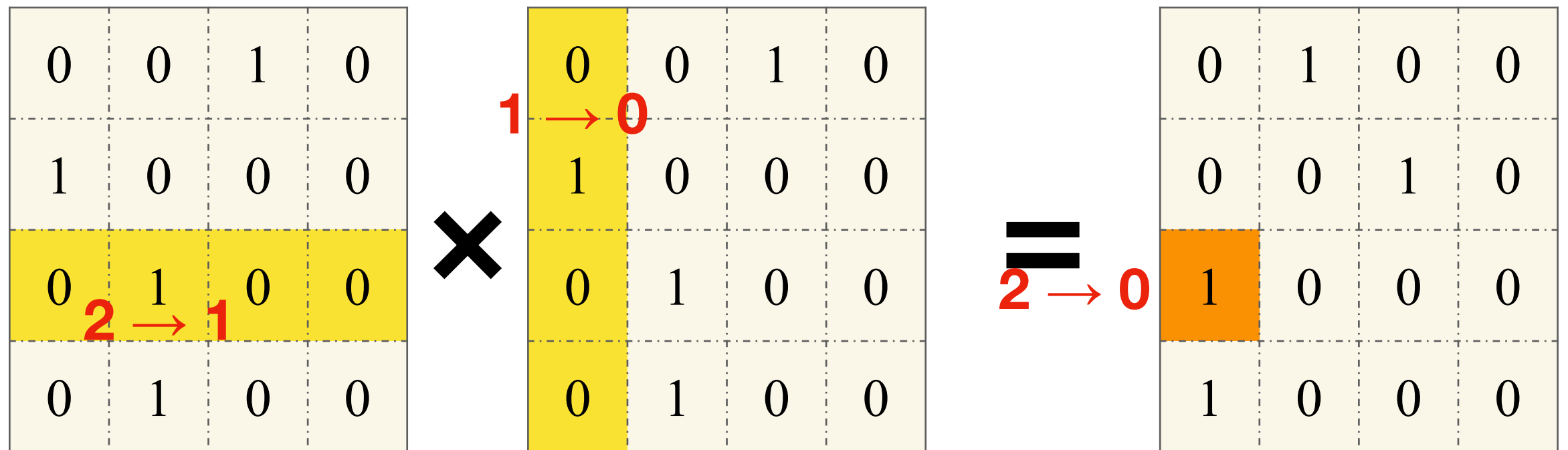
Problem 2

Graph Representation

$0 \rightarrow 1$

$0 \rightarrow 3$

$1 \rightarrow 2$

$1 \rightarrow 4$

$2 \rightarrow 3$

$2 \rightarrow 5$

$4 \rightarrow 6$

$5 \rightarrow 4$

$6 \rightarrow 7$

$7 \rightarrow 5$

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

- A cell(x,y) = 1 if there is a connection from x to y

- 0 otherwise

Matrix Multiplication

$$
\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}
\times
\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}
=
\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}
$$

$2 \to 1$ $1 \to 0$ $2 \to 0$

- Do multiplication at most N times. (N: # of nodes)

Matrix Multiplication
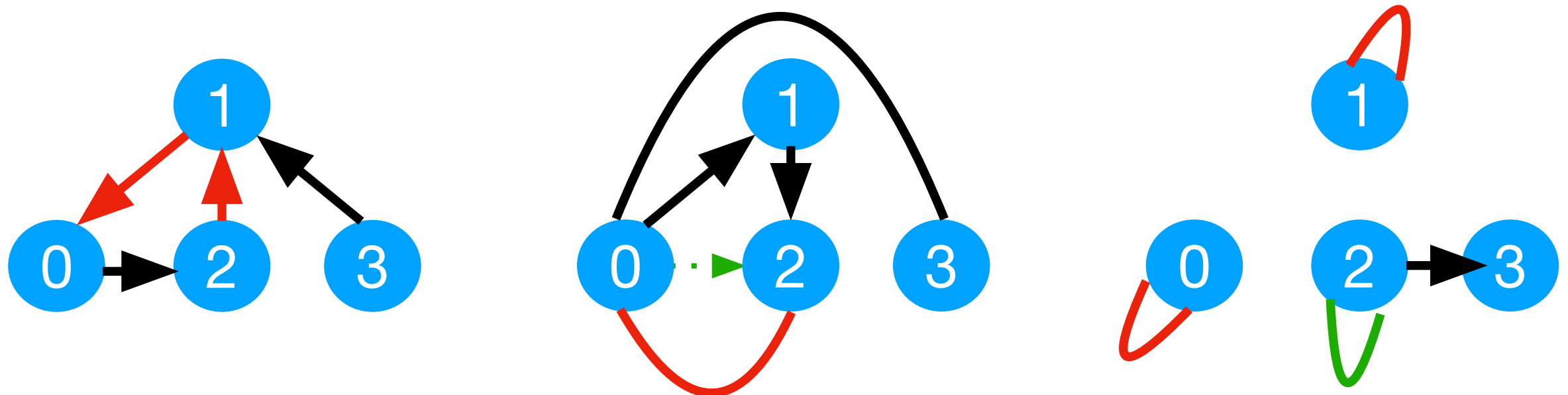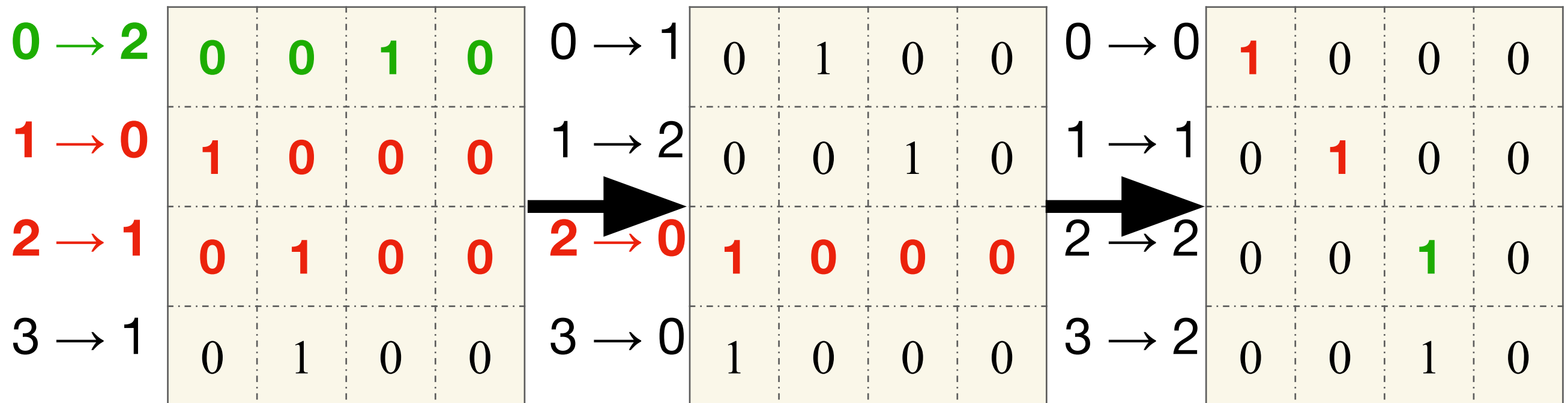
Termination

- If we find a value bigger than 1 in the diagonal of matrix, then the graph has a cycle.

- If we do N times of multiplication and all value in the diagonal are 0, then the graph does not have a cycle.

p2.py

```python
def has_cycle(sets):
    # TODO
    # return True if the graph has cycle; return False if not

    return False
'''

  HINT: You can `print(sets)` to show what the matrix looks like
    If we have a directed graph with 2->3 4->1 3->5 5->2 0->1
         0  1  2  3  4  5
      0  0  1  0  0  0  0
      1  0  0  0  0  0  0
      2  0  0  0  1  0  0
      3  0  0  0  0  0  1
      4  0  1  0  0  0  0
      5  0  0  1  0  0  0
    The size of the matrix is (6,6)
'''
```

Code Implementation

- You should only complete the function `has_graph(sets)` in p1.py and p2.py. **DO NOT** modify the other parts of code

- This function will return True or False

  - If the graph(sets) has cycles, this function should return True

  - Otherwise, return False

Python Tips

List

- Generate an empty list

  - L = list() or L = []

- Get a sublist of a list L

  - From A to B: L[A:B]

  - From begin to B: L[:B]

  - From A to end: L[A:]

0	1	2	3	4	5	6	7	8	9

0   1   2   3   4   5   6   7   8   9   10
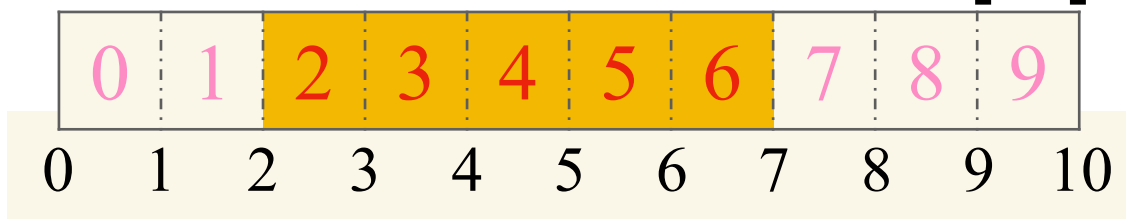
- Push an object in list

  - L.append(object)

    L = [2, 4, [1, 2], [3, 6]]
    object = 8
    L.append(object)
    >>> L = [2, 4, [1, 2], [3, 6], 8]

- Extend a list L2 and push it in list L

  - L.extend(L2)

    L2 = [4, [2, 3]]
    L.append(L2)
    >>> L = [2, 4, [1, 2], [3, 6], 8, 4, [2, 3]]

NumPy

- Create a numpy object

  - L = numpy.array([3, 2, 5, 1])

- Two numpy matrix A, B multiplication

  - numpy.matmul(A, B)

    A = [[1, 2], [3, 4]]

    B = [[2, 3], [4, 5]]

    numpy.matmul(A, B)

    >>> [[10, 13], [22, 29]]

Rules

Run Code

- python p1.py r07922072
  Yes
  No
  No
  … 12 outputs

- You should run your code with your 學號

  - You can generate answer text file by running command
    python p1.py 自己的學號 > p1_ans.txt
    python p2.py 自己的學號 > p2_ans.txt

  - Or just type them by yourself

```
Yes
No
No
Yes
No
Yes
Yes
Yes
No
No
Yes
No
```

Code Structure

- Code you **download**
  |—p1.py (TODO)
  |—p2.py (TODO)
  |—graph_gen.py

- Code you **submit** should be put in a
  folder and compressed in a zip file

  r07922072_hw1.zip
  r07922072_hw1
    |—p1.py
    |—p2.py
    |—graph_gen.py
    |—p1_ans.txt
    |—p2_ans.txt

Rules

- 不要抄作業, 不要交別人的答案, 作弊一律0分計算

- 上傳 zip 檔案到 CEIBA

- 注意繳交的資料夾學號開頭英文用**小寫**

- **DEADLINE: 2018/10/18(四) 23:59 (GMT+8:00)**

- **遲交每過一天：分數×0.8 (per day)**

- **格式、檔案、各種奇怪的錯誤讓我無法改作業：分數×0.8**

Q & A

ask on Facebook