BÀI TẬP TUẦN 2

1. Bài tập tại lớp

Viết các thuật toán dưới đây bằng Python.

Bài 1: Tạo một mảng numpy có kích thước 2×4. Toàn bộ giá trị bằng 0.

Bài 2: Cho numpy array một chiều với giá trị ngẫu nhiên. Lọc và trả về mảng numpy gồm các giá trị nguyên dương.

Bài 3: Dùng hàm argmax trong numpy, tìm chỉ mục của số lớn nhất trong mảng 1 chiều.

Bài 4: An được giao nhiệm vụ tổ chức thi tiếng Anh trên máy tính cho hai phòng với 7 bạn mỗi bên. Tuy nhiên, vì hôm ấy chỉ còn 1 phòng máy trống, An phải lựa chọn thí sinh để đảm bảo có nhiều nhất 7 bạn tham gia.

An cũng biết được phòng 2 có thể thay bằng hình thức thi giấy. An đặt hai danh sách hai bên, chọn lựa giữa hai bạn vị trí đầu tiên, thứ hai, thứ ba,... đến hết. Tiêu chí chọn như sau:

- Giữa phòng 1 và 2, ưu tiên phòng 1.
- Nếu thí sinh phòng 1 xin thôi thi, chọn thí sinh còn lại.
- Nếu cả hai thí sinh đều thôi thi, gán "None" để bổ sung sau.

Viết hàm nhận vào hai mảng numpy một chiều chứa số thứ tự thí sinh. Trả về mảng chứa danh sách thí sinh cuối cùng. Lưu ý rằng số thứ tự giữa hai phòng không trùng nhau và thí sinh thôi thi có số thứ tự âm. Ví dụ:

$$room\_1 = [1, 2, -3, 4, 5, 6, -7]$$

$$room\_2 = [8, 9, 10, 11, 12, -13, -14]$$

Đáp án là: $[1, 2, 10, 4, 5, 6, None]$.

Tên hàm: combine_rooms(room_1, room_2)

✨Bài 5: Cho hai ma trận $A, B \in R^{m \times n}$. Phép cộng ma trận được thực hiện như sau:

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} + \begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mn} \end{bmatrix} = \begin{bmatrix} a_{11} + b_{11} & \cdots & a_{1n} + b_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} + b_{m1} & \cdots & a_{mn} + b_{mn} \end{bmatrix}$$

Khi thực hiện trên máy tính, nếu $B \in R^m$ là một vector, phần mềm sẽ thực hiện broadcast vector bằng cách tạo ma trận mới với mỗi cột chính là B. Ví dụ:

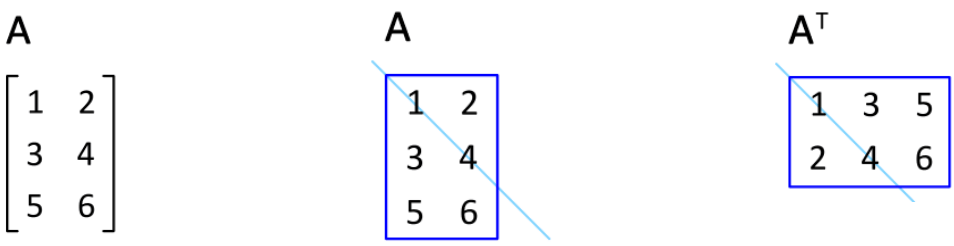
$$\begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix} + \begin{bmatrix} 6 \\ 7 \end{bmatrix} = \begin{bmatrix} 2 & 3 \\ 4 & 5 \end{bmatrix} + \begin{bmatrix} 6 & 6 \\ 7 & 7 \end{bmatrix} = \begin{bmatrix} 8 & 9 \\ 11 & 12 \end{bmatrix}$$

Mô phỏng lại broadcast bằng một hàm nhận một vector numpy array và số cột. Hàm trả về kết quả broadcast. Chẳng hạn, với $\begin{bmatrix} 6 \\ 7 \end{bmatrix}$ và số cột là 3, kết quả là $\begin{bmatrix} 6 & 6 & 6 \\ 7 & 7 & 7 \end{bmatrix}$

Tên hàm: broadcast(vec, n)

2. Bài tập về nhà

Bài 6: Ma trận chuyển vị (transpose) được tạo ra bằng cách lật ma trận ban đầu theo trục chéo:



https://upload.wikimedia.org/wikipedia/commons/e/e4/Matrix_transpose.gif

Cho ma trận mảng numpy, trả về chuyển vị của ma trận đó.

Tên hàm: transpose(mat)

Bài 7: Tích ma trận và tích hadamard là hai phép tính phổ biến. Hãy viết chương trình nhận hai ma trận đầu vào, in trên màn hình tích của chúng. Nếu không thể tính, hãy in "Khong co tich ma tran" hoặc "Khong co tich Hadamard".

Tên hàm: product(mat_a, mat_b)

Bài 8: Viết hàm nhận tham chiếu là ma trận và số chỉ mục của cột. Thay toàn bộ giá trị cột đó là 1 rồi trả về kết quả.

Tên hàm: replace_col(mat, col_ind)

Lưu ý:

- *Nộp 1 file .py duy nhất. Chỉ chạy hàm khi module đó là main.*
- *Hàm trả về kết quả (return) và không gọi hàm nhập/in như input() hay print().*
- *Hàm tuân thủ tên như ghi chú dưới mỗi bài.*
- *Yêu cầu người dùng nhập lại nếu dữ liệu không hợp lệ.*
- *Thời gian chạy mỗi bài không quá 1 giây.*
- *Nếu đề không ghi rõ, mặc định mảng numpy chứa số thực.*