TP 4

Notre but est de travailler avec les matrices de nombres entiers. Une matrice peut donc être vue comme un objet qui a comme paramètres un nombre de lignes, un nombre de colonnes et une liste contenant les coéfficients.

Exercice 1
Définir une classe Matrice et écrire son constructeur de manière à ce que trois appels à ce construsteur soient possible :

- `m = Matrice ()` : crée une matrice vide.
- `m = Matrice (3, 3)` : initialise une matrice à 3 lignes, 3 colonnes avec tous les coéfficients à 0.
- `m = Matrice (3, 3, l)` : initialise une matrice 3×3 avec les valeurs contenues das l comme coéfficients.

Rappel : pour offrir plusieurs constructeurs, donner des valeurs par défaut aux attributs d'instances.

Exercice 2
Ajouter la méthode `__str__`qui permetra d'afficher dans un `print` une matrice ligne par ligne.

Exercice 3
Écrire la méthode `read_matrice` qui invite l'utilisateur à saisir le nombre de lignes, puis de colonnes et la liste des coefficients.

Exercice 4
Ajouter les méthode de manipulation des matrices suivantes :

- `__add__` : retourne la matrice résultat de l'addition de la matrice courante avec la matrice passée en argument. Ainsi nous pourrons appeler l'opérateur $+$
- `__sub__` : retourne la matrice résultat de la soustraction de la matrice courante avec la matrice passée en argument. Ainsi nous pourrons appeler l'opérateur $-$
- `__mul__` : retourne la matrice résultat du produit de la matrice courante avec la matrice passée en argument. Ainsi nous pourrons appeler l'opérateur $*$
- `transpose` : retourne la transposée de la matrice courante.