

Examen Semestre 2
Matière : Informatique

Classes : MP2, PC2 et PT2
Date : Mai 2019

Nombre de pages : 4
Durée : 2 h

Exercice 1

La base de données **cinema.sqlite** porte sur tous les films de cinéma, leur casting, acteurs et réalisateurs, jusqu'à l'année 2016. Elle contient les trois tables suivantes :

La table **acteur**, portant sur les acteurs de cinéma, et ayant pour attributs :

- **id** de type INT servant d'identifiant, c'est la clé primaire,
- **nom** de type VARCHAR() (chaîne de caractères), contenant son nom ;

La table **film**, portant sur les films de cinéma, et ayant pour attributs :

- **id** de type INT servant d'identifiant, c'est la clé primaire,
- **titre** de type VARCHAR() (chaîne de caractères), contenant son titre,
- **annee** de type INT, contenant son année de sortie,
- **score** de type FLOAT, contenant sa note attribuée par un jury d'internautes,
- **realisateur** de type VARCHAR() contenant le nom de son réalisateur ;

La table **casting**, portant sur les castings (c'est-à-dire les acteurs) des différents films :

- **filmid** de type INT servant d'identifiant du film,
- **acteurid** de type INT, l'identifiant d'un acteur du film.

On demande les requêtes SQL permettant d'obtenir les informations suivantes :

- 1) Les titres des 10 films les mieux notés de l'année 1999.
- 2) Le titre des films ayant obtenu une note supérieure à la note moyenne.
- 3) Les noms des réalisateurs ayant tourné au moins un film dont la note est supérieure à la moyenne.
- 4) Les noms des réalisateurs dont aucun film n'a une note inférieure à la moyenne.
- 5) Les noms des acteurs par ordre alphabétique, et pour chacun le nombre de films auxquels il a participé.
- 6) La liste des 10 réalisateurs ayant les meilleures notes moyennes de leurs films, classés par moyenne décroissante.
- 7) Les noms de tous les acteurs ayant tourné dans le film 'Star Wars'.

Exercice 2

Un nombre complexe est dit « entier de Gauss » s'il est de la forme : $a + ib$ avec $(a, b) \in \mathbb{Z}^2$. Définir une classe nommée **GaussInt** permettant d'instancier des objets représentant les nombres complexes de type « entier de Gauss » et possédant les méthodes suivantes :

1) Un constructeur permettant d'initialiser les attributs **re** et **im** d'instance. Ces attributs représentent respectivement la partie réelle et imaginaire d'un objet de type entier de Gauss. Les arguments du constructeur doivent être des entiers relatifs, autrement une exception est levée.

2) Une méthode **norme** renvoyant la norme d'un entier de Gauss.

Nota : A l'entier de Gauss $n = a + ib$, on associe sa norme $N(n) = a^2 + b^2$.

3) La surcharge d'une méthode spéciale d'affichage dont l'invocation sur une instance d'attributs $re = 4$ et $im = -7$, permet d'afficher : (4,-7).

4) La surcharge de la méthode spéciale **__add__** qui permet d'additionner deux entiers de Gauss ou un entier de Gauss et un entier. Une exception doit être levée en cas où l'argument passé n'est pas un entier ou un entier de Gauss.

La méthode **__add__** ne supporte pas l'utilisation d'objet d'instance sur le côté droit de l'opérateur +, quand l'objet situé à gauche ne l'est pas.

Par exemple :

```
>>> n = GaussInt(4, -7)
>>> n + 2
(6, -7)
>>> 2 + n
TypeError: unsupported operand type(s) for +: 'int' and 'GaussInt'
```

Python appelle **__radd__** uniquement lorsque l'objet situé à droite de l'opérateur + est une instance de la classe **GaussInt** et que l'objet situé à gauche ne l'est pas.

5) Ecrire un code permettant la surcharge de la méthode spéciale **__radd__** qui permet d'additionner un entier et un entier de Gauss.

6) La surcharge de la méthode spéciale **__mul__** qui permet de multiplier deux entiers de Gauss ou un entier de Gauss et un entier. Une exception doit être levée en cas où l'argument passé n'est pas un entier ou un entier de Gauss.

Exercice 3

La factorisation de **Cholesky** consiste, pour une matrice carrée symétrique définie positive $A(n, n)$, à déterminer une matrice triangulaire inférieure L telle que : $A = LL^T$.

Une matrice symétrique est une matrice carrée qui est égale à sa propre transposée.

Une matrice $A(n, n)$ est dite définie positive si, pour tout vecteur colonne non nul x à n éléments réels, on a : $x^T A x > 0$.

Principe de la méthode

Soit A une matrice carrée d'ordre 3. L'équation $A = LL^T$ s'écrit comme suit :

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{bmatrix} \begin{bmatrix} L_{11} & L_{21} & L_{31} \\ 0 & L_{22} & L_{32} \\ 0 & 0 & L_{33} \end{bmatrix}$$

Après avoir effectué le produit matriciel LL^T , on obtient :

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} = \begin{bmatrix} L_{11}^2 & L_{11}L_{21} & L_{11}L_{31} \\ L_{11}L_{21} & L_{21}^2 + L_{22}^2 & L_{21}L_{31} + L_{22}L_{32} \\ L_{11}L_{31} & L_{21}L_{31} + L_{22}L_{32} & L_{31}^2 + L_{32}^2 + L_{33}^2 \end{bmatrix}$$

Comme la matrice A est symétrique, on a :

$$A_{12} = A_{21}, A_{13} = A_{31} \text{ et } A_{23} = A_{32}.$$

En identifiant les matrices A et LL^T élément par élément, on obtient six équations (en raison de la symétrie, seuls les éléments triangulaires inférieurs ou supérieurs doivent être pris en compte) avec six composantes inconnues de L .

En résolvant ces équations dans **un certain ordre**, il est possible d'avoir une seule inconnue dans chaque équation.

En comparant les éléments de la **première colonne**, en commençant par la première ligne et en descendant, on peut calculer L_{11} , L_{21} et L_{31} dans cet ordre :

$$\begin{aligned} A_{11} &= L_{11}^2 & L_{11} &= \sqrt{A_{11}} \\ A_{21} &= L_{11}L_{21} & L_{21} &= A_{21}/L_{11} \\ A_{31} &= L_{11}L_{31} & L_{31} &= A_{31}/L_{11} \end{aligned}$$

Ensuite la **deuxième colonne**, en commençant par la deuxième ligne, on aura L_{22} et L_{32} :

$$\begin{aligned} A_{22} &= L_{21}^2 + L_{22}^2 & L_{22} &= \sqrt{A_{22} - L_{21}^2} \\ A_{32} &= L_{21}L_{31} + L_{22}L_{32} & L_{32} &= (A_{32} - L_{21}L_{31})/L_{22} \end{aligned}$$

Enfin la **troisième colonne** donne L_{33} :

$$A_{33} = L_{31}^2 + L_{32}^2 + L_{33}^2 \quad L_{33} = \sqrt{A_{33} - L_{31}^2 - L_{32}^2}$$

On peut extrapoler les résultats pour une matrice $n \times n$.

Algorithme de factorisation :

Les éléments L_{ij} ($i, j = 1, 2, \dots, n$) de la matrice triangulaire inférieure L sont donnés par :

$$L_{11} = \sqrt{A_{11}} \quad L_{i1} = A_{i1}/L_{11}, \quad i = 2, 3, \dots, n$$

$$L_{jj} = \sqrt{A_{jj} - \sum_{k=1}^{j-1} L_{jk}^2}, \quad j = 2, 3, \dots, n$$

$$L_{ij} = (A_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk}) / L_{jj}, \quad j=2, 3, \dots, n-1, \quad i=j+1, j+2, \dots, n.$$

1) Ecrire une fonction **Cholesky(A)** qui prend en argument une matrice carrée A et retourne une matrice triangulaire inférieure L telle que : $A = LL^T$.

On se propose de résoudre le système d'équations linéaires $Ax = b$, où A est une matrice carrée d'ordre n , x vecteur colonne des inconnues de taille n et b second membre du système de taille n .

Comme A est factorisée par la méthode de Cholesky, la résolution du système $Ax = b$ est ramenée à la résolution de deux systèmes triangulaires, et ce pour réduire la complexité globale de calcul.

On a :

$$Ax = b \Leftrightarrow LL^T x = b \Leftrightarrow \begin{cases} Ly = b, \\ L^T x = y. \end{cases}$$

Etape 1 : Résolution du système $Ly = b$

On peut trouver les composantes de y par des substitutions élémentaires, y_1 puis y_2 , etc. Cette étape est appelée *descente*.

2) Donner l'algorithme de résolution qui permet de calculer le vecteur y , à partir de la matrice triangulaire inférieure L et du vecteur b .

3) En déduire une fonction **descente(L, b)** qui prend en argument la matrice triangulaire L et le vecteur b et retourne le vecteur y .

Etape 2 : Résolution du système $L^T x = y$

Il reste à calculer les composantes du vecteur x en résolvant le système triangulaire supérieur : $Ux = y$ (avec $U = L^T$), ce qui se fait de manière similaire, mais en calculant d'abord x_n puis, de proche en proche, les autres inconnues x_i , de $i = n - 1$ à $i = 1$ (étape dite de *remontée*).

4) Donner l'algorithme de résolution qui permet de calculer le vecteur x , à partir de la matrice triangulaire supérieure U et du vecteur y .

5) En déduire une fonction **remontee(U, y)** qui prend en argument la matrice triangulaire supérieure U (qui est la transposée de L) et le vecteur y résultat de l'étape 1 et retourne le vecteur x qui représente la solution du système initial.

6) En appelant les fonctions précédentes, écrire un code pour résoudre le système d'équations linéaires définit par :

$$A = \begin{pmatrix} 4 & 6 & 8 \\ 6 & 34 & 52 \\ 8 & 52 & 129 \end{pmatrix} \text{ et } b = \begin{pmatrix} 0 \\ -160 \\ -452 \end{pmatrix}$$