Programmation pour le calcul scientifique
Année : 2019-2020
Formation : L3 Ingénierie Mathématique

TP7 : Classes, surcharge d'opérateur

1 Matrices creuses, encore.

Le but est de transformer le code que nous avions écrit pour décrire les matrices creuses carrées, de façon à être dans l'esprit de la POO.

1. Copiez vos fichiers du TP correspondant, ou récupérez une correction, et changez votre type structuré en classe.
2. Créez des constructeurs pour que les initialisations suivantes soient valides :

```
int M,N;
SparseMatrix A(M,N); // M is the size of the matrix, N is the number of non-zero coefficients
                     // This constructor only pre-allocates memory

std::vector<int>    rowIndices,colIndices;
std::vector<double> values;
SparseMatrix A(rowIndices,colIndices,values);

std::string fileName;
SparseMatrix A(fileName); // Reads the content of the matrix from a text file
```

3. Qu'en est-il du destructeur ?
4. Incorporez les fonctions précédemment implémentées dans la classe (trace, norme infinie, multiplication avec un vecteur).
5. Modifiez votre fonction `main` en accord avec ces changements.
6. Écrivez une méthode privée `posIndex(const int i, const int j)` qui retourne l'indice dans le tableau de valeurs non nulles du coefficient d'indices naturels (i, j). La fonction retournera -1 si le coefficient ne fait pas partie des coefficients non-nuls.
7. Écrire une méthode `getValue(const int i, const int j)` qui retourne le coefficient de la matrice.
8. Surchargez l'opérateur `()` pour réaliser l'opération `A(i,j) = 2.;`. Il faudra pour cela retourner une référence vers le coefficient, et non simplement sa valeur. Si le coefficient n'est pas non-nul, ajoutez le à la matrice.
9. Nous souhaitons maintenant ajouter des matrices entre elles, ou les multiplier avec un vecteur de manière « naturelle ». Surchargez l'opérateur `+` de façon à pouvoir écrire

```
SparseMatrix A1,A2;
SparseMatrix A3(A1+A2);
```

10. Surchargez l'opérateur d'assignation pour pouvoir écrire

```
SparseMatrix A1,A2,A3;
A3 = A1+A2;
```

11. Surchargez l'opérateur `*` de façon à pouvoir écrire

```
SparseMatrix A;
std::vector<double> b,x;
b = A*x;
```

12. Surchargez l'opérateur « de façon à pouvoir afficher la matrice à l'écran ou la sauvegarder dans un fichier.