Projet C++: Les matrices

Ecole Polytechnique de l'Université de Tours

Avril 2019

Préambule

Lors de ce projet tuteuré nous allons travailler les *matrices*.

L'objectif est de réaliser une librairie de classes et fonctions permettant de manipuler des matrices d'éléments. La réalisation de ce projet va nécessiter l'utilisation de la méthodologie UML pour modéliser le logiciel à développer, ainsi que le langage C++ pour la mise en œuvre. Vous devrez à l'issue de ce projet rendre un rapport présentant le travail que vous avez réalisé. Ce projet est à réaliser dans un volume de 12h de TPs, volume réparti comme suit (ce n'est qu'une suggestion) :

1. Modélisation UML : 3h pour la réalisation du diagramme de classes (seul diagramme qui vous est obligatoirement demandé dans ce projet). Ce temps inclus de la rédaction de votre rapport.
2. Développement C++ : 9h. Ce temps inclus de la rédaction de votre rapport.

Vous travaillez en binômes.

A l'issue de ces 12h vous devrez rendre votre rapport, les sources de votre projet ainsi qu'une version exécutable sous Windows et qui a été compilée dans le profil Release sous Visual Studio.

Spécifications fonctionnelles

Passons maintenant à la spécification du travail à réaliser. La librairie que vous devez fournir doit permettre :

1. De créer des objets Matrice pour lequel le type des éléments est quelconque : Je peux donc avoir une matrice d'entiers, de caractères, de valeurs réelles, d'objets d'une autre classe, ...
2. De faire des opérations « élémentaires » sur une matrice,
    a. multiplier une matrice par un nombre : si M est une matrice et c une constante alors on veut pouvoir faire cM et Mc. Cette opération ne doit pas modifier le contenu de M.
    b. Diviser une matrice par un nombre : si M est une matrice et c une constante alors on veut pouvoir faire M/c. Cette opération ne doit pas modifier le contenu de M.
    c. Afficher à l'écran une matrice.
3. De faire des opérations un peu plus complexes mettant en jeu plusieurs matrices,

a. A partir d'une matrice M donnée, calculer sa matrice transposée M^T. Cette opération ne doit pas modifier le contenu de M.
b. Faire l'addition de deux matrices A et B : on veut pouvoir calculer $A+B$ et $B+A$. Cette opération ne doit pas modifier le contenu de A ou B.
c. Faire la soustraction de deux matrices A et B : on veut pouvoir calculer $A-B$ et $B-A$. Cette opération ne doit pas modifier le contenu de A ou B.
d. Faire le produit de deux matrices A et B : on veut pouvoir calculer AB et BA. Cette opération ne doit pas modifier le contenu de A ou B.

Par ailleurs, vous êtes libre de proposer de nouvelles fonctionnalités tant qu'elles relèvent du confort d'utilisation de votre code.

Contraintes techniques

Pour la réalisation de votre projet vous devrez respecter les quelques contraintes énoncées ci-dessous :

1. Les variables, types, classes, fonction et méthodes devront respecter les conventions de nommage,
2. Vous devez mettre en œuvre les bonnes pratiques du Génie Logiciel telles qu'elles ont pu être présentées lors des cours d'Algorithmes Orientés Objets et de Langages Orientés Objets,
3. Vous devrez intégrer la gestion des exceptions dans votre programme : utilisez pour cela la classe Cexception vue en TPs,
4. Vous travaillerez sous Visual Studio et votre projet devra être ***application console win32***,

En plus des classes nécessaires pour réaliser ce qui a été demandé ci-dessus à propos des matrices, vous devrez développer une fonction qui permet de lire le contenu d'une matrice dans un fichier texte et créer un objet matrice en mémoire pour stocker les données lues dans le fichier. Le format du fichier texte (imposé) est donné ci-dessous et illustré dans le cadre d'un exemple : il repose sur l'utilisation de quelques balises.

TypeMatrice=<type_base_C>
NBLignes=<Nombre_de_lignes_de_la_matrice>
NBColonnes=<Nombre_de_colonnes_de_la_matrice>
Matrice=[
<Ligne_1 : autant d'éléments que de colonnes>
<Ligne_n : autant d'éléments que de colonnes>
...
<Ligne_NBLignes : autant d'éléments que de colonnes>
]

Un exemple :

TypeMatrice=int
NBLignes=2

NBColonnes=3
Matrice=[
5 3 6
1 -2 8
]

Note : votre procédure devra **uniquement** gérer le cas où TypeMatrice vaut double. Pour les autres types, vous lèverez une exception.

La fonction principale de votre projet devra alors faire l'algorithme suivant :

- Pour chaque nom de fichier passé en paramètre, lire le fichier et créer la matrice associée,
- Demander à l'utilisateur de saisir une valeur *c*,
- Afficher le résultat de la multiplication de chacune des matrices par la valeur *c*,
- Afficher le résultat de la division de chacune des matrices par la valeur *c*,
- Afficher le résultat de l'addition de toutes les matrices entre elles : $M_1+M_2+M_3+....$,
- Afficher le résultat de l'opération suivante : $M_1-M_2+M_3-M_4+M_5-M_6+....$
- Afficher le résultat du produit des matrices.

Eléments à fournir à l'issu du projet

A l'issu du projet vous devrez fournir :

- Un rapport présentant les modèles UML que vous avez pu construire, la façon dont votre code a été développé et une présentation des choix que vous avez pu être amené à réaliser, ... Plus tout ce qui est nécessaire pour la compréhension de votre projet.
- Les sources complètes (avec le projet sous Visual Studio),
- Un exécutable fonctionnant sous Windows et compilé en mode Release.