TD MOO : C++ / UML
Séance 5/6

TD 5– Héritage

Le but de ce TD :

- maitriser l'écriture de nouvelles classes simples (exemple pédagogique),
- maitriser les mécanismes d'héritage et de surcharge
- mettre en œuvre le polymorphisme

Préparation : exercice 1 (et les notions de cours associées)

Exercice I : modélisation d'une hiérarchie de classes

Faire le diagramme UML des trois classes suivantes (seulement sur papier) :

- **`Un`** qui possède
  - un champ X de type double privé,
  - une fonction `GetX` qui retourne X,
  - une fonction `SetX` qui modifie X,
  - un constructeur permettant d'initialiser X
  - une fonction `Affiche` qui affiche X avec le texte « Un.X = »
- **`Deux`** qui dérive (ou hérite) de `Un` et qui possède en plus :
  - un champ Y de type double privé,
  - une fonction `GetY` qui retourne Y,
  - une fonction `SetY` qui modifie Y,
  - un constructeur permettant d'initialiser X et Y,
  - une fonction `Affiche` qui affiche X et Y avec le texte « Deux. (X, Y) = ».
- **`Trois`** qui dérive de `Deux` et qui possède en plus :
  - un champ Z de type double privé,
  - une fonction `GetZ` qui retourne Z,
  - une fonction `SetZ` qui modifie Z,
  - un constructeur permettant d'initialiser X, Y et Z,
  - la fonction `Affiche` qui affiche X, Y et Z avec le texte «Trois.(X, Y, Z)=».

Exercice II : réalisation d'une hiérarchie de classes

1 Implémenter votre solution en respectant les diagrammes de l'exercice **I**.

2 Vérifier le fonctionnement de vos trois classes dans une fonction main.

3 Modifier/débugger votre code pour répondre aux questions.

3-a Un objet de type `Deux` peut-il accéder directement au champ `X` ? Pourquoi ?

3-b Quand on crée un objet de type `Trois`, combien de constructeurs sont exécutés ? Dans quel ordre sont effectués les constructeurs ? Comment avez-vous fait pour trouver les constructeurs exécutés et l'ordre ?

3-c Quand un objet de type `Trois` est détruit, dans quel ordre sont exécutés les destructeurs ? Comment avez-vous procédé pour trouver l'ordre ?

3-d A partir d'un objet `Trois`, comment exécuter la fonction `Affiche` de la classe `Deux` ? Et celle de la classe `Un` ?

4 Les pointeurs d'objet :

Créer un pointeur de type `Trois` et l'affecter à un espace alloué à un objet `Trois`. C'est-à-dire :

```
Trois *_pt = new Trois( 1, 2, 3) ;
_pt->Affiche() ;  // pour les questions qui suivent
delete _pt ;      // ne pas oublier !!!!
```

4-a Le nombre et l'ordre dans lequel s'effectue les constructeurs ont il changé ?

4-b Qu'en est-il pour les destructeurs ?

4-c La fonction `Affiche` appelée est elle la plus adaptée (nombre de valeurs affichées par rapport au nombre de valeurs en mémoire) ?

5 Avantages et problèmes avec les pointeurs d'objet :

Créer un pointeur de type et `Un` et l'affecter à un espace alloué à un objet `Trois`. C'est-à-dire :

```
Un *_pt = new Trois( 1, 2, 3) ; // si si
_pt->Affiche() ;  // pour les questions qui suivent
delete _pt ;      // ne pas oublier !!!!
```

5-a Le nombre et l'ordre dans lequel s'effectue les constructeurs ont il changé ?

5-b Qu'en est-il pour les destructeurs ?

5-c La fonction `Affiche` appelée est elle la plus adaptée ?

5-d Mettre le mot clé `virtual` devant la déclaration des fonctions `Affiche`. Quelle fonction `Affiche` est maintenant appelée par l'instruction : `_pt->Affiche() ;`

5-e a Mettre le mot clé `virtual` devant la déclaration des destructeurs des classes `Un`, `Deux` et `Trois`. Avec les destructeurs `virtual`, quel est le nombre et l'ordre des destructeurs utilisés par l'instruction suivante ?

```
delete _pt ;
```

5-f Trouver une application des pointeurs sur le type le plus haut dans la hiérarchie.

6 Même questions que la **5-d** avec des références :

Créer une référence de type et `Un` et l'affecter à un espace alloué d'un objet `Trois`. C'est-à-dire :

```
Trois tr( 5, 4, 3) ;
Un &_ref = tr ;
_ref.Affiche() ;  // pour les questions qui suivent
```

7 Conclure sur les conditions de mise en œuvre du polymorphisme et les intérêts du polymorphisme.