

Exercice 6 révisions chapitres 2 à 6

OBJECTIF :

A l'issue de la réalisation de cet exercice, les étudiants doivent être capables de réaliser un ensemble de classes avec une association, de l'héritage, du polymorphisme et une composition. Ils doivent aussi maîtriser les aspects du C++ tels que la surcharge des fonctions, les paramètres optionnels et le passage par référence.

Ceci dans le cadre d'une application console avec le Visual Studio 2015.

PREPARATION DU PROJET

Vous trouverez un modèle UML Ex6ModeleUML.zargo et le fichier de canevas sous K:\ES\Maitres-Eleves\SLO\Modules\SL228_POBJ\CoursCpp\Exercices\Ex6, ainsi que le fichier HexUtil.h

Il s'agit de créer un projet VisualStudio en C++, une application console. Nommez le projet Ex6. Lorsque le projet est créé, copiez Ex6ModeleUML.zargo, Ex6.cpp et les autres fichiers dans le répertoire Ex6 du projet, puis générez les fichiers au même endroit. Ensuite ajouter tous les fichiers .cpp dans la section "fichiers sources" et tous les fichiers .h dans la section "fichiers d'en-tête".

Le projet se compose de TestA et TestB. Le modèle UML est en relation avec le TestB.

ACTIONS DU TESTA

Le TestA utilise la classe HexUtil. Le fichier HexUtil.h est fourni, par contre le fichier HexUtil.cpp est à réaliser entièrement.

La classe HexUtil contient des méthodes de saisie pour des int, short, float et double.

La classe HexUtil contient la méthode ShowHex surchargée pour les types int, short, float et double. La méthode ShowHex affiche en hexadécimal la valeur des octets composant les variables de différent type.

ACTIONS AU NIVEAU DU PROGRAMME PRINCIPAL

Dans la section TestA du canevas, il faut :

- Créer un objet MyHexUtil du type HexUtil
- Utiliser SaisirInt et ShowHex avec la valeur int saisie.
- Utiliser SaisirShort et ShowHex avec la valeur short saisie.
- Utiliser SaisirFloat et ShowHex avec la valeur float saisie.
- Utiliser SaisirDouble et ShowHex avec la valeur double saisie.

REALISATION DES METHODES DE SAISIE DE HEXUTIL.CPP

Il faut réaliser les 4 méthodes de saisie. Il s'agit d'afficher un message indiquant la saisie réalisée, d'effectuer la saisie et de retourner la valeur saisie.

Veuillez-vous référer à l'exemple pour le détail des messages.

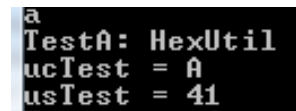
REALISATION DES METHODES SHOWHEX DE HEXUTIL.CPP

Il faut réaliser les 4 méthodes ShowHex. En fonction du type du paramètre, il faut afficher un message, puis en utilisant un pointeur **unsigned char *pData**, extraire et afficher en hexadécimal la valeur de chacun des octets de la variable reçue. Remarque : l'affichage doit être réalisé en commençant par l'octet de poids fort.

Veuillez-vous référer à l'exemple pour le détail des affichages.

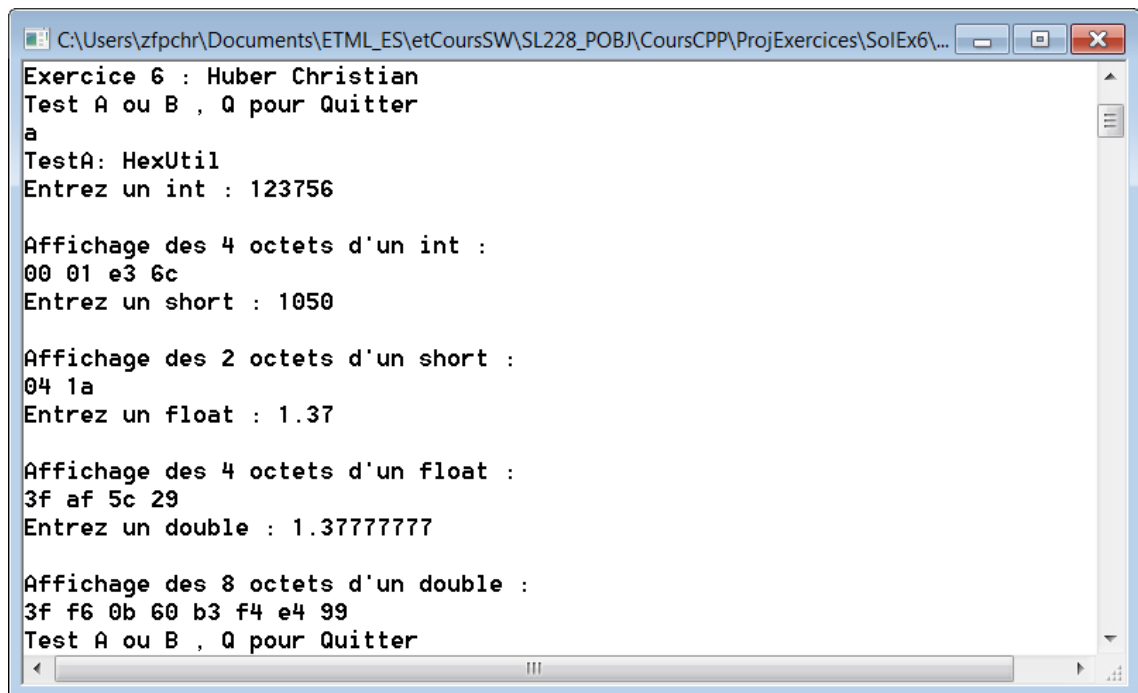
Aide : cout affiche les unsigned char comme des caractères, il est donc nécessaire de passer par une variable temporaire de type unsigned short.

```
unsigned char ucTest = 0x41  
unsigned short usTest = 0x41  
  
cout << "ucTest = " << hex << ucTest << endl;  
cout << "usTest = " << hex << usTest << endl;
```



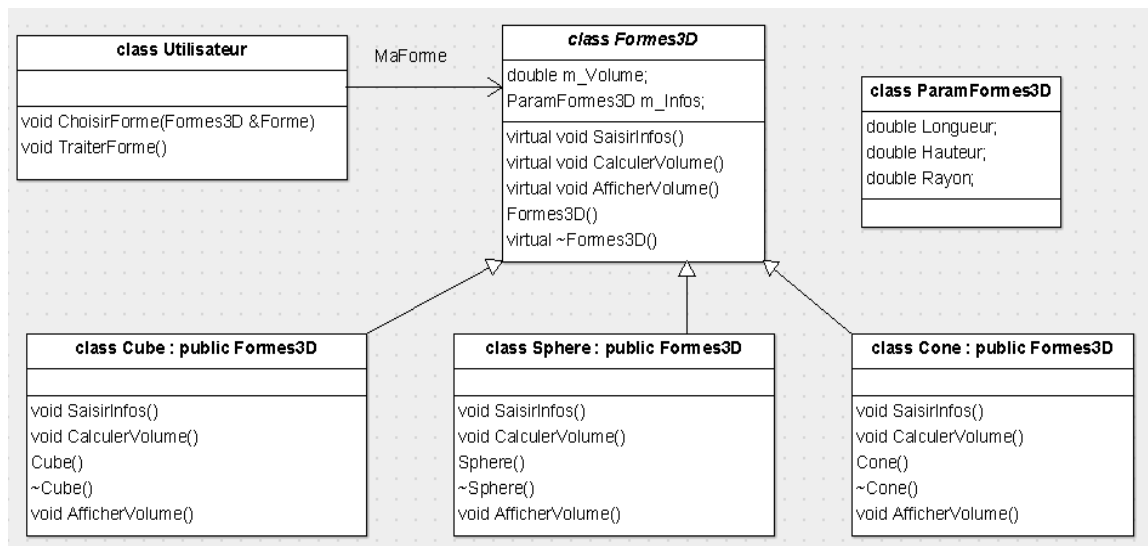
```
a  
TestA: HexUtil  
ucTest = A  
usTest = 41
```

EXEMPLE DE RÉSULTAT DU TESTA



ACTIONS DU TESTB

MODELE UML FOURNIS



ACTIONS AU NIVEAU DU PROGRAMME PRINCIPAL

Dans la zone de déclaration des variables, créez 3 Utilisateurs (Albert, Bernard Christian). Il faut aussi créer un Cube, une Sphere et un Cone.

Dans la section TestB appelez pour chaque utilisateur la méthode **ChoisirForme** en passant en paramètre l'objet du type Cube, Sphere ou Cone. Remarque :Il s'agit d'un passage par référence C++.

Ensuite pour les 3 utilisateurs appelez la méthode **TraiterForme**.

REALISATION DES METHODES DE LA CLASSE UTILISATEUR

La méthode **ChoisirForme (Formes3D &Forme)** doit simplement associer l'objet reçu en paramètre (en utilisant Formes3D *MaForme)

La méthode **TraiterForme()** doit effectuer la saisie des paramètres, le calcul du volume et l'affichage, ceci en utilisant successivement les méthodes de la classe Formes3D.

- Appeler la méthode **SaisirInfos()** de la classe Formes3D.
- Appeler la méthode **CalculerVolume()** de la classe Formes3D.
- Appeler la méthode **AfficherVolume()** de la classe Formes3D.

REALISATION DES METHODES DE LA CLASSE FORMES3D

Les 3méthodes virtuelle **SaisirInfos()**, **CalculerVolume()** et **AfficherVolume()** restent vides.

Dans le constructeur, mettre à 0 les champs de m_infos et afficher "Constructeur de Forme3D". Dans le destructeur afficher "Destructeur de Formes3D".

REALISATION DES METHODES DES CLASSES DERIVEES

Il faut compléter le constructeur de chacune des trois classes en appelant le constructeur de la classe de base.

Il faut aussi afficher "Constructeur de Cube" ou " Constructeur de Sphere" ou "Constructeur de Cone", en correspondance avec la classe.

REALISATION DE SAISIRINFOS

Il faut implémenter la méthode **SaisirInfos()** en tenant compte des spécificités de chaque classe.

Pour le détail des messages veuillez-vous référer à l'exemple de résultat.

Remarque : La classe ParamFormes3D est proche d'une structure avec 3 champs. Les attributs sont public ce qui permet de les utiliser directement, en plus il n'y a pas de composition, ce qui permet d'établir un attribut de ce type pour la classe Formes3D

Pour un **Cube** on demande la longueur de l'arrête et on l'enregistre dans le champ Longueur de m_Infos.

Pour une **Sphere** on demande le rayon et on l'enregistre dans le champ Rayon de m_Infos.

Pour un **Cone** on demande le rayon et la hauteur on enregistre dans les champs Rayon et Hauteur de m_Infos.

Pour le détail des messages veuillez-vous référer à l'exemple de résultat.

REALISATION DE CALCULERVOLUME

Il faut implémenter la méthode **CalculerVolume()** en tenant compte des spécificités de chaque classe.

Pour un **Cube** on utilise la longueur de l'arrête que l'on obtient du champ Longueur de m_Infos.

Pour une **Sphere** on utilise le rayon que l'on obtient du champ Rayon de m_Infos.

Pour un **Cone** on utilise le rayon et la hauteur que l'on obtient des champs Rayon et Hauteur de m_Infos.

REALISATION DE AFFICHERVOLUME

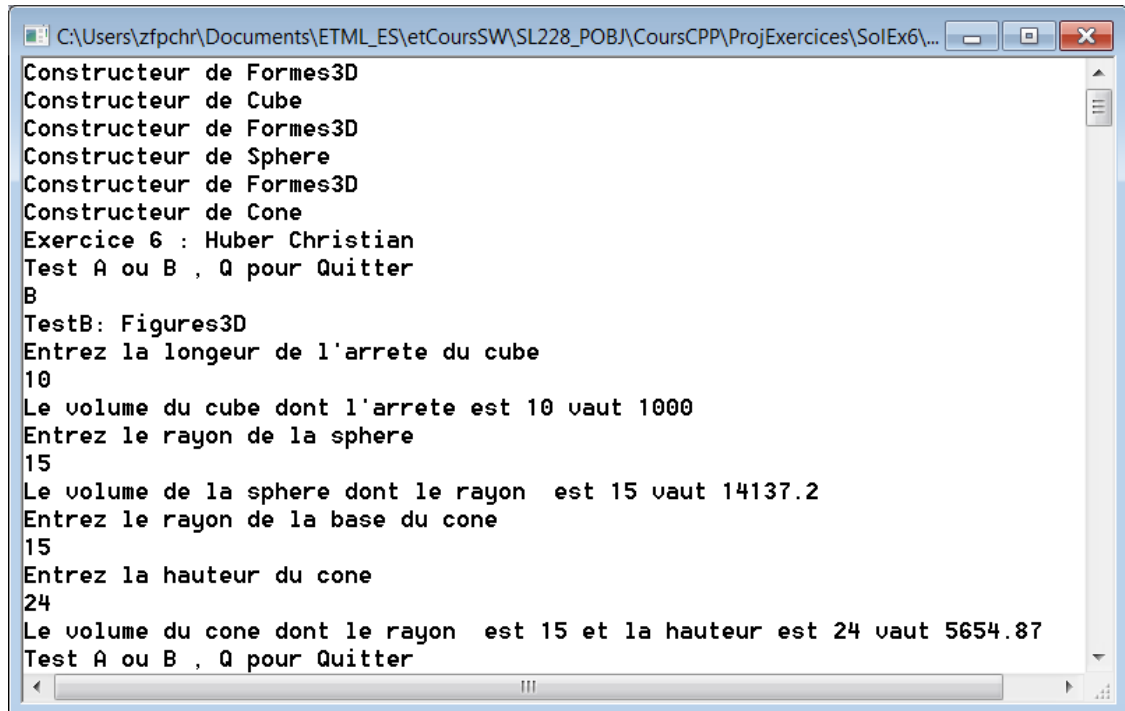
Il faut implémenter la méthode **AfficherVolume()** en tenant compte des spécificités de chaque classe.

Pour un **Cube** on indique qu'il s'agit d'un Cube dont l'arrête vaut xxx et que le volume est de vvvv.

Pour une **Sphere** on indique qu'il s'agit d'une sphère dont le rayon vaut xxx et que le volume est de vvvv.

Pour un **Cone** on indique la valeur du rayon et de la hauteur ainsi que le volume obtenu.

Pour le détail des messages veuillez-vous référer à l'exemple de résultat.

EXEMPLE DE RESULTAT

```
C:\Users\zfpchr\Documents\ETML_ES\etCoursSW\SL228_POBJ\CoursCPP\ProjExercices\SolEx6\...
Constructeur de Formes3D
Constructeur de Cube
Constructeur de Formes3D
Constructeur de Sphere
Constructeur de Formes3D
Constructeur de Cone
Exercice 6 : Huber Christian
Test A ou B , Q pour Quitter
B
TestB: Figures3D
Entrez la longueur de l'arrete du cube
10
Le volume du cube dont l'arrete est 10 vaut 1000
Entrez le rayon de la sphere
15
Le volume de la sphere dont le rayon est 15 vaut 14137.2
Entrez le rayon de la base du cone
15
Entrez la hauteur du cone
24
Le volume du cone dont le rayon est 15 et la hauteur est 24 vaut 5654.87
Test A ou B , Q pour Quitter
```