TP MOPO – Garage

Préambule

Le but de ce TP est de mettre en place une application programmée en langage C++, et basée sur un modèle objet permettant de stocker les informations relatives au fonctionnement d'un garage.

Le résultat de votre travail, qui devra être déposé en fin de séance sur MOODLE, sera noté en fonction d'un certain nombre de critères parmi lesquels figurent, entre autres, l'aspect fonctionnel, la qualité de l'implémentation et des commentaires, ainsi que le respect du standard de programmation proposé en cours.

Il prendra la forme d'une archive contenant :

- les codes sources
- le fichier CMakeLists.txt
- un compte-rendu des éventuelles difficultés que vous avez rencontré et des solutions que vous avez trouvé pour les contourner.

Vous avez accès via Moodle au corrigé du TP précédent (application 'Agence Immobilière'). Il contient notamment quelques méthodes utilitaires pour simplifier la lecture des données entrées par l'utilisateur (StringUtils.cpp / .h), ainsi qu'un bon aperçu du travail attendu. Courage, c'est votre dernier TP de C++ !

Définition du modèle objet

Les données à traiter sont les suivantes :

- Les clients
- Les véhicules

Données à stocker :

- Client : Nom, prénom, adresse, téléphone, adresse e-mail
- Véhicule : Nom, Type (moto, voiture, utilitaire)
- Les motos ont une cylindrée, les voitures un nombre de places, et les utilitaires un volume (en m3)

Les véhicules ont tous une immatriculation, ainsi qu'un statut (en vente, en location).

Mettre en place les classes nécessaires au stockage de ces données à l'intérieur d'une application C++ compilable avec un fichier 'cmake'.

Bien que non obligatoire, il est fortement recommandé d'utiliser la STL, et plus particulièrement std::vector pour le stockage des tableaux dynamiques. Cela vous permettra de facilement créer ou détruire des entrées à l'aide des méthodes 'push_back' et 'erase'.

Implémentation de l'interface utilisateur

Voici les fonctionnalités à implémenter pour permettre à un utilisateur d'interagir avec l'application :

- Ajout d'un nouveau client (demande à l'utilisateur des données associées).
- Affichage de la liste des clients

- Ajout d'un véhicule.
- Suppression d'un véhicule
- Affichage du nombre total / la liste des véhicules en fonction de leur type et de leur statut

Note :

Vous pouvez utiliser la classe template « std::vector » pour le stockage d'un tableau :

```
#include <vector>

class CClient
{
        // ...
}

std::vector<CClient> aClient;

aClient.push_back(CClient("jean", "dupont", "16 rue Einstein, 25000 Besancon", "0381500000"));
```

Suppression du 3ème élément d'un std::vector<CClient> :

```
#include <vector>

class CClient
{
        // ...
}

std::vector<CClient> aClient;

aClient.erase(aClient.begin() + 3);
```