Objectif

Le but de ce TP est d'assimiler les notions suivantes :

- La notion de classe
- Constructeur et destructeur
- Membre statique
- Utilisation des opérateurs new et delete

Exercice 1

L'objectif de cet exercice est de gérer les notes des étudiants d'une institution à l'aide d'une classe C++ **Etudiant** définie par :

Les attributs suivants :

- **Code** : l'identifiant de l'étudiant **(auto incrémenté)**
- **nom**: nom d'un étudiant
- **nbrNotes**: le nombre de notes de l'étudiant
- ***tabNotes**: tableau contenant les notes d'un étudiant (**allocation dynamique**).

Les méthodes suivantes :

- Un constructeur d'initialisation
- Un constructeur avec arguments
- Un destructeur ~**Etudiant** ()
- Les **getters** et **setters**
- **void saisie** () : permettant la saisie des note d'un étudiant
- **void affichage** () : permettant l'affichage des informations d'un étudiant
- **float moyenne** () : retourne comme résultat la moyenne des notes de l'étudiant.
- **bool admis** () : retourne comme résultat la valeur ***true***, si un étudiant est admis et la valeur ***false***, sinon. Un étudiant est considéré comme étant admis lorsque la moyenne de ses notes est supérieure ou égale à 10.
- **bool comparer():** qui compare la moyenne des deux étudiants, retourne comme résultat la valeur true, si deux étudiants ont la même moyenne et la valeur false, sinon.

1- Déclarer et définir la classe Etudiant en créant les deux fichiers : etudiant.h et etudiant.cpp
2- Ecrire un programme principal main.cpp dans lequel :
   a. Créer cinq étudiants
   b. Saisir leurs notes
   c. Afficher les notes des cinq étudiants
   d. Calculer la moyenne des cinq étudiants
   e. Afficher les étudiants admis
   f. Afficher les étudiants ayant la même moyenne

Exercice 2 surcharge d'opérateur +=

Nous reprenons l'exemple de la classe point du TP2 et nous souhaitons réaliser ce type d'opération logique :

```
int main() {
point my_point1(2.0, 3.0);
point my_point2(4.0, 5.0);
my_point1 += my_point2; }
```

1 - Définir et implémenter la méthode suivante pour surcharger l'opérateur += :

point & operator+=(point & point_);

Modifier alors les fichiers *point.h*, *point.cpp* et *main.cpp*

2 – Récrire le programme principale en simulant un exemple d'utilisation de l'opérateur +=. Réaffichez les nouvelles valeurs de l'abscisse et de l'ordonnée du point en question.

Exercice 3 les membres statiques

Un *Compte* bancaire possède à tout moment une donnée : son *solde*. Ce solde peut être positif (compte créditeur) ou négatif (compte débiteur).

Chaque compte est caractérisé par un *code* incrémenté automatiquement.

Le *code* et le *solde* d'un compte sont accessibles en lecture seulement.

A sa création, un compte bancaire a un *solde* nul et un *code* incrémenté.

Il est aussi possible de créer un compte en précisant son *solde* initial.

Utiliser son compte consiste à pouvoir y faire des dépôts et des retraits. Pour ces deux opérations, il faut connaître le montant de l'opération.

L'utilisateur peut aussi consulter le solde de son compte par la méthode *affiche_solde()*.

Questions

1) Définir la classe *Compte* à travers les fichiers *compte.h* et *compte.cpp*
2) Identifier la donnée statique
3) Identifier les membres méthodes
4) Proposer un programme principale qui a pour but de :
    a) Créer trois comptes bancaires en affectant un solde
    b) Effectuer des opérations de dépôts et de retraits
    c) Afficher le solde final