TP4 – Héritage

Le but de ce TP est de créer une nouvelle classe à partir d'une classe existante. On utilisera la classe CompteBancaire écrite au premier TP.

1 Créez la classe dérivée Livret dans le fichier Livret.h

La classe Livret hérite de la classe Compte (un livret est un compte). Un Livret a un taux d'intérêt et un plafond. Il est associé à une banque.

Ajouter également l'entête des fonctions suivantes :

- constructeur avec arguments et valeurs par défaut,
- constructeur par copie,
- destructeur,
- afficher les attributs,
- calcul des intérêts sur une période donnée,
- affichez les résultats d'une simulation sur une période donnée (10 ans par défaut).

2 Codez le fichier Livret.cpp

Codez toutes les méthodes de la classe Livret dans le fichier **Livret**.cpp

3 Testez l'héritage dans un fichier de test main4.cpp

Déclarez un livret livretA qui a un taux d'intérêt de 2.5%.
Affichez LivretA.
Calculez les intérêts sur un an.
Faites une simulation sur 2 ans.

4 Testez le polymorphisme dans un autre fichier de test mainPoly.cpp

Un objet polymorphe est un objet qui peut prendre différentes formes. En C++, il dispose de méthodes virtuelles et conserve des informations sur son type réel. A l'exécution, la fonction appelée sur un objet polymorphe dépendra du type réel de l'objet.

Pour montrer le fonctionnement du polymorphisme, nous allons utiliser :

- **la notion d'héritage** déjà vue entre la classe Compte et sa classe dérivée Livret.
- **la notion de virtualité** implantée par les fonctions virtuelles,
- **les pointeurs (**pour la notion de liaison dynamique**).**

Dans le fichier mainPoly.cpp, déclarez un pointeur ptr1 sur un compte qui pointe sur un compte.
Déclarez un pointeur ptr2 sur un compte qui pointe sur un livret.
Affichez ptr1.
Affichez ptr2.
Que se passe-t-il lors de l'affichage ? Les deux affichages sont-ils identiques ?
Que faut-il faire pour y remédier ?

Exercice supplémentaire : Expression arithmétique et héritage

On désire représenter une expression arithmétique en vue de son évaluation. Toute expression doit pouvoir être évaluée et affichée.

Une variable est une expression qui a un nom et une valeur.

Un entier est une expression qui a seulement une valeur.

Le produit de deux expressions est une expression.

La somme de deux expressions est une expression.

Décrivez les classes nécessaires (classe de base Expression et ses classes dérivées Variable, Entier, Produit, Somme) pour permettre l'exécution du programme suivant:

```
main()
{
  Var v("v"); v=2;
 Entier n = 3;
 Expression & e = v + (v * n);
 e.afficher() ;
  cout << e.eval();
}
```