HLIN603
Feuille de TD/TP N°2
Héritage Multiple et Généricité Paramétrique en C++

Exercice 1 : Hiérarchie d'héritage de comptes bancaires

Cet exercice reprend une partie de la hiérarchie d'héritage définie dans l'exercice 2 de la feuille de TD/TP N°1.

1) Proposez trois classes C++ représentant respectivement :

   - Une classe ***CompteBancaire*** disposant d'un attribut ***solde*** et d'un destructeur qui affiche la valeur du solde que la banque est supposée rendre au client lors de la fermeture du compte.

   - Une classe ***CompteRemunere*** représentant les comptes auxquels on sert un intérêt. En particulier, lors de la fermeture, le solde est augmenté de 10%.

   - Une classe ***CompteDepot*** représentant les comptes de dépôt classiques. Lors de la fermeture, on prélève sur ces comptes des frais de gestion de 100 euros.

   - Une classe ***CompteDepotRemunere***.

2) Représentez une instance de chaque classe dans le cas de l'héritage non virtuel, puis dans le cas de l'héritage virtuel.

3) - Ajoutez une méthode ***deposer*** dans ***CompteBancaire***, spécialisez-la dans le cas des ***CompteRemunere*** en ajoutant un intérêt de 1% à la somme déposée, et dans le cas des ***CompteDepot*** en retirant 1euro de frais de gestion et en ajoutant 10euros si le dépôt est supérieur à 1000 euros. Effectuez un dépôt sur une instance de ***CompteDepotRemunere***.
   - Que se passe-t-il dans chacun des cas d'héritage (virtuel ou non) ?
   - Comment pouvez-vous résoudre le problème ?

4) Donnez l'ordre des opérations effectuées lorsqu'un compte est fermé (l'instance est "détruite" au sens de C++) dans chacun des cas d'héritage (virtuel ou non).

5) On suppose que ***CompteDepotRemunere*** possède une méthode ***deposer***, et hérite "d'abord" de ***CompteRemunere*** puis de ***CompteDepo***t.

   - Ajoutez une classe ***CompteDepotAvecCarteCredit*** et une classe ***CompteDepotRemunereEtAvecCarteCredit*** qui hérite "d'abord" de ***CompteDepotAvecCarteCredit***.
   - Dans le cas de l'héritage virtuel, que se passe-t-il lorsqu'on ferme un compte de la classe ***CompteDepotRemunereEtAvecCarteCredit*** ?

Exercice 2 : Une classe paramétrée Dictionnaire

Un dictionnaire est un ensemble d'associations, c'est-à-dire de couples (clé, valeur), où la clé est d'un type donné ***TypeCle*** et la valeur d'un type donné ***TypeValeur***. Une association peut être représentée par la classe ***Assoc*** vue en cours. Un dictionnaire possède habituellement les méthodes suivantes :

- put: place une clé et une valeur associée dans le dictionnaire;
- get: prend une clé et renvoie la valeur associée;
- estVide: dit si le dictionnaire est vide;
- taille: retourne le nombre d'associations clé-valeur effectivement présentes dans le dictionnaire;
- contient: prend une clé et dit si elle est présente dans le dictionnaire;
- affiche: affiche le contenu du dictionnaire sur un flot de sortie.

1) **Spécification.** Écrivez la partie "signatures des méthodes" du fichier .h correspondant au dictionnaire générique.
   - Ajoutez la surcharge de ***l'opérateur <<*** pour l'affichage et la surcharge de ***l'opérateur =*** pour l'affectation.
2) **Implémentation par un tableau d'associations.** Le tableau d'associations est alloué dynamiquement, par défaut avec une taille de 10. L'indice d'une association est calculé grâce à une fonction de hachage appliquée à la clé (voir en annexe).
   - En utilisant les exemples de résultats de la fonction de hachage, faites quelques essais d'insertions "à la main" pour voir comment les choses se passent : par exemple put("abricot",235); put("amande",1023); put("ananas",242); put ("pomme",83); ...
   - Complétez la partie "attributs" du fichier .h, et écrivez le fichier .cc correspondant.

Pour faciliter l'écriture de certaines méthodes, on peut écrire la méthode privée :

```
void CherchCl(const TypeCle& cl, int& i, int& res);
/* cherche la cle cl dans le dictionnaire :
      si cl est presente: renvoie res=1, i indice de la case de cl dans T;
      si cl est absente et le dictionnaire non plein: renvoie res=0, i indice de case possible pour cl dans T;
      si cl est absente et le dictionnaire plein: renvoie res=2, i non significatif. */
```

3) **Utilisation.** Instanciez vos classes pour avoir un dictionnaire dont les clés sont des chaînes et les valeurs des entiers.
   - Écrivez un programme simple qui teste le fonctionnement de ce dictionnaire (ajoutez des couples, affichez le dictionnaire, rajoutez des couples pour tester l'agrandissement, affichez, etc.).
   - Utiliser la classe dictionnaire pour stocker les mots d'un texte lu sur l'entrée standard et comptabiliser le nombre d'occurrences de chacun.