Gestion de comptes bancaires

TP noté - Programmation par objets 2 (UE ULIN606)

1 Présentation du projet

L'objectif de ce projet est de développer une application de gestion de comptes bancaires. Un compte bancaire peut être classé suivant deux critères. Le premier permet de classer un compte suivant ce qu'il rapporte (rémunéré ou dépôt) ; le second critère classe les comptes selon leur type de propriétaire (personne ou entreprise). Chacun peut avoir plusieurs comptes (rémunérés et/ou dépôt). Certains comptes sont à la fois des comptes rémunérés et dépôt.

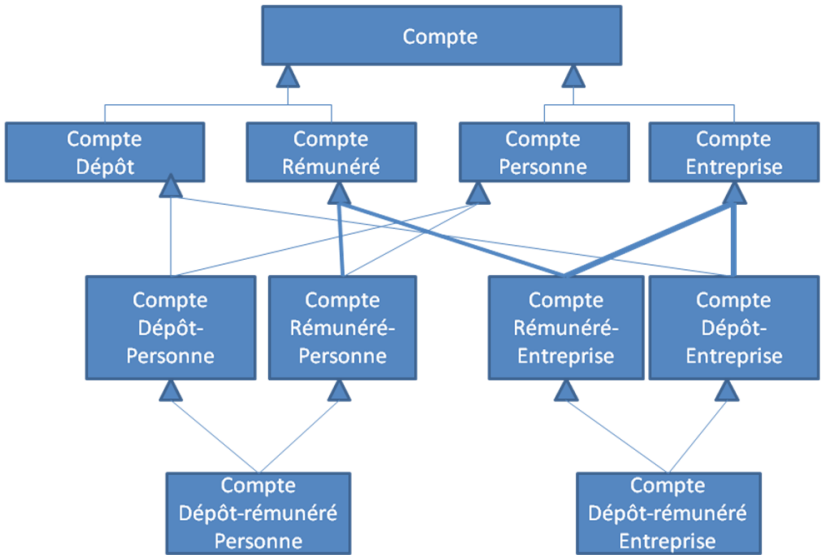

FIGURE 1 – Diagramme minimal des classes de l'application de gestion de comptes bancaires.

2 Précisions sur les classes

Compte

Nous définissons les attributs suivants pour la classe `Compte` :
- Solde : de type réel ;
- Propriétaires : de type `Set` (librairie `STL`) de `Client` ;
- Gestionnaire : de type `Conseiller` ;
- Date d'ouverture du compte : de type `Date` ;
- Liste des opérations du mois en cours : une opération est une association d'une clé et d'une valeur. La clé correspond à la date d'opération. La valeur correspond au montant de l'opération ; Plusieurs

opérations peuvent avoir lieu pendant la même date. Le type de cet attribut est un template de la librairie `STL`.

Nous définissons les méthodes et fonctions suivantes :
- Afficher les informations concernant un compte en utilisant l'opérateur `<<` ;
- Afficher les informations concernant les propriétaires du compte ;
- Saisir les informations concernant un compte en utilisant l'opérateur `>>` ;
- Créditer un compte : méthode déposer ;
- Débiter un compte : méthode retirer.

En plus, cette classe doit définir un seul attribut représentant la liste de tous les comptes ouverts. Elle doit définir des méthodes permettant d'ouvrir et de fermer un compte. La méthode permettant de fermer un compte doit afficher la valeur du solde que la banque est supposée rendre au client lors de la fermeture du compte.

Compte rémunéré

Nous définissons, au minimum, les attributs spécifiques suivants pour la classe `CompteRemunere` :
- Taux de rémunération : la même valeur du taux est valable pour tous les comptes rémunérés et est égale à 0.02 ;
- Liste de 12 associations (mois, intérêt pour le mois) : le type de cet attribut est la classe template `Dictionnaire` définie en TD.

Nous définissons, au minimum, les méthodes suivantes :
- Calcul de la valeur d'intérêt mensuel : cette valeur est égale à la moyenne du solde journalier pendant un mois * taux appliqué ;
- Fermeture de compte : cette méthode affiche la valeur du solde après ajout de la valeur des intérêts mensuels ;
- Déposer une somme : cette méthode ajoute 1 euro par tranche de 1000 euros à la somme déposée ;
- Afficher les informations concernant un compte en utilisant l'opérateur `<<` ;
- Saisir les informations concernant un compte en utilisant l'opérateur `>>`.

Compte dépôt

Nous définissons, au minimum, les attributs spécifiques suivants pour la classe `CompteDepot` :
- La valeur minimale nécessaire pour ouvrir un compte dépôt ;
- La valeur maximale autorisée pour un compte dépôt ;
- La valeur des frais de gestion à prélever lors de la fermeture du compte.

Nous définissons, au minimum, les méthodes spécifiques suivantes :
- Fermeture de compte : cette méthode affiche la valeur du solde après prélèvement des frais de gestion ;
- Déposer une somme : cette méthode retire 1 euro par tranche de 1000 euros de frais de gestion de la somme déposée, et ajoute 20 euros si le dépôt est supérieur à 10000 euros ;
- Afficher les informations concernant un compte en utilisant l'opérateur `<<` ;
- Saisir les informations concernant un compte en utilisant l'opérateur `>>`.

Compte Personne

Nous définissons, au minimum, l'attribut suivant pour la classe `ComptePersonne` :
- Valeur du découvert autorisé.

Nous définissons, au minimum, les méthodes spécifiques suivantes :
- Afficher les informations concernant un compte en utilisant l'opérateur `<<` ;

– Saisir les informations concernant un compte en utilisant l'opérateur `>>`.

Il est à signaler que lors de la création d'un compte de ce type, les propriétaires du compte doivent être de type `Personne`.

Compte Entreprise

Nous définissons, au minimum, l'attribut suivant pour la classe `CompteEntreprise` :
– Liste des personnes autorisées à effectuer des retraits sur le compte : de type `Set` de `Personne`.

Nous définissons, au minimum, les méthodes spécifiques suivantes :
– Afficher les informations concernant un compte en utilisant l'opérateur `<<` ;
– Saisir les informations concernant un compte en utilisant l'opérateur `>>`.

Il est à signaler que lors de la création d'un compte de ce type, les propriétaires du compte doivent être de type `Entreprise`.

3 Travail demandé

1. Implémenter l'ensemble des classes décrites précédemment ainsi que l'ensemble des autres classes dont elles dépendent. Proposer différents constructeurs pour chaque classe (sans paramètres et avec paramètres). Pour l'héritage multiple, proposer différents schémas (ordre de déclaration) d'héritage.
2. Implémenter une classe GestionComptes permettant d'offrir des services liés à l'ensemble des méthodes permettant la gestion des compotes. Principalement :
   (a) Ouvrir des comptes (pour tous les types de compte) ; utiliser les différents constructeurs proposés.
   (b) Déposer une somme.
   (c) Afficher la liste des opérations du mois en cours d'un compte.
   (d) Afficher la liste des comptes ouverts avec leurs informations.
   (e) Afficher les informations concernant les propriétaires d'un compte.
   (f) Chercher un compte suivant le nom et prénom et date de naissance de l'un des propriétaires.
   (g) Fermer un compte en donnant le nom de ses propriétaires.
   (h) Fermer des comptes (avec affichage du solde à remettre au client).
3. Créer différentes classes implémentant des types d'exceptions à gérer. Pensez à créer des hiérarchies de classes. Parmi les exemples d'exceptions à gérer : chercher un compte qui n'existe pas, ajouter un compte rémunéré ou dépôt d'une personne qui existe déjà, fermeture d'un compte qui n'existe pas, retrait d'une somme qui dépasse le montant du solde, dépôt d'une somme qui dépasse le plafond autorisé, création d'un compte `Entreprise` avec comme propriétaire des personnes, etc. Pensez aux autres exceptions qui doivent être gérées.
4. Tester dans une méthode main l'ensemble des méthodes de la classe `GestionComptes`. Tester des cas nécessitant de lever des exceptions. Pour chaque schéma d'héritage multiple, donner le résultat d'exécution de :
   (a) Ouvrir un compte `CompteDepotRemunerePersonne` avec une somme de 1600 euros ;
   (b) Déposer 3000 euros ;
   (c) Fermer le compte créé et afficher le montant du solde à remettre au client.
5. La réalisation d'une interface graphique en `QT` pour lancement des méthodes de la classe `GestionComptes` sera considérée comme un plus considérable.