Feuille de TP N°05

Héritage, polymorphisme et classes abstraites

Gestion d'une banque

1 Préambule

L'objectif de ce TP est d'implémenter des classes modélisant le fonctionnement simplifié des comptes bancaires gérés par une banque.
Une méthode `main` fournie vous aidera à tester votre implémentation. La modélisation est déjà faites et vous pouvez avoir un aperçu de la hiérarchie des classes ainsi que de leurs interfaces dans le schéma ci-dessous. Votre travail va consister essentiellement à implémenter les méthodes des classes `Client`, `CompteNonBloque`, `CompteCourant`, `ComptePEL` et `CompteLDD` en tenant compte du schéma d'héritage.

2 La classe Client

Une banque possède un ensemble de clients, chacun pouvant être titulaire d'un ou plusieurs comptes. Tout client d'une banque y possède au moins un compte. Dans chaque banque, un client peut posséder plusieurs comptes d'épargne mais ne peut être titulaire que d'un seul compte courant. Un compte ne peut appartenir qu'à un seul client. Un client est identifié de façon unique par son nom.

Écrire une classe `Client` pour la représentation des clients de la banque. Cette classe possèdera les champs privés suivants :
– `nom` : Nom de la personne.
– `prenom` : Prénom de la personne.
– `adresse` : Adresse de la personne.
– `dateNaissance` : date de naissance de la personne.
Un client est en plus décrit par la liste de ses comptes.

Le format des nom et prénom sera le suivant (cf. méthodes `toLowerCase()` et `toUpperCase()` de la classe `java.lang.String()`) :
– Le nom sera stocké en majuscule .
– Le prénom commencera par une majuscule et le reste sera en minuscule.

Redéfinissez les méthodes `toString()` et `equals(...)` de la classe `Object`. La méthode `toString()` renverra une chaîne de caractères décrivant la personne sous la forme `Samir AITAHMED (12 avril 1970)`.

Remarque 1 *Dans la littérature, il est conseillé de redéfinir la méthode* `int hashCode()` *de la classe* `Object` *chaque fois que l'on redéfinit la méthode* `equals(...)` *(le code de hashage est un entier permettant d'identifier de façon unique un objet de la classe, deux codes proches étant censés représenter des objets proches).*

Pour l'affichage de la date, on pourra se référer à :
– la documentation de la classe `Calendar` (méthode `getDisplayName(...)`)
– la documentation de la classe `SimpleDateFormat` qui hérite de la classe DateFormat
– ou utiliser la méthode de classe format de la classe String utilisant les mêmes arguments que la méthode (tant décriée par certains mais néanmoins utile) printf(...)

3 Des classes pour les comptes bancaires

Un compte bancaire est identifié par un numéro de compte, qui est un entier positif unique. Ce numéro est attribué à l'ouverture du compte et ne peut être modifié par la suite. Un compte est associé à un client. Une fois le compte créé, le titulaire du compte ne peut plus être modifié.

Remarque 2 *Le numéro de compte est affecté automatiquement et vous devez garantir que deux comptes différents auront toujours des numéros de compte distincts. Vous utiliserez un compteur d'instances avec un attribut* `static`.

Il est possible de déposer et de retirer de l'argent sur un compte (si le retrait sur ce compte est autorisé). Le solde est la somme d'argent disponible sur un compte. Si le solde d'un compte est négatif, le compte est à découvert. Dans ce cas, des agios (frais de découvert) sont prélevés.

Il existe plusieurs types de comptes. Les *comptes courants* permettent de réaliser des opérations de versement et de retrait auprès de la banque. Des frais de découvert (5% du solde actuel) sont ajoutés à toute opération dont le résultat laisse le solde du compte négatif. Les *comptes épargne* sont rémunérés, chaque produit d'épargne a un taux fixé par le règlement de la banque. Certains comptes épargne sont *bloqués*, c'est-à-dire que le retrait sur ces comptes est interdit, seul le dépôt est autorisé. Le découvert est interdit sur les comptes épargne. Parmi les plans d'épargne, on distingue :
- Le Plan Épargne Logement (PEL) qui fait travailler l'argent à raison de 3% et peut faciliter l'accès aux prêts. Par contre l'argent sera bloqué jusqu'à la fin du plan.
- Le Livret de Développement Durable (LDD) est moins rémunérateur que le PEL, seulement 2%, mais il permet d'effectuer un retrait à tout instant.

Par ailleurs, on sait qu'il y aura d'autres types de comptes à l'avenir.

Le solde du compte est actualisé après chaque opération en tenant compte des éventuels frais de découvert. Il est également possible d'effectuer un virement entre deux comptes différents. Le solde des deux comptes est mis à jour selon les règles énoncées précédemment.

a) Implémentez les méthodes des classes `CompteNonBloque`, `CompteCourant`, `ComptePEL` et `CompteLDD` de telle sorte que :
   - En cas de tentative d'opération illégale (un retrait d'un montant négatif, un dépôt d'un montant négatif, un retrait qui aboutirait à un solde négatif sur un compte épargne, ouverture d'un compte avec un solde négatif, ...) le programme affiche un message puis s'arrête (appel à `System.exit(0)`).
   - Si après un retrait le compte courant a un solde négatif, le programme affiche un message avertissant le client qu'il va payer des agios.

b) Pour chacune de ces classes, prévoir un constructeur permettant d'initialiser chaque attribut avec des paramètres. Prévoir aussi dans chacune de ces classes une méthode `toString()`.

c) Testez l'implémentation de chaque classe en exécutant la méthode `main` de `TestComptes.java`.

4 Une classe Banque

Écrivez la classe `Banque` qui contient une liste de comptes (un attribut privé de type `Vector <Compte> listeComptes`) et une liste de clients (un attribut privé de type `Vector <Client> listeClients`).
Écrivez les attributs, le constructeur et la méthode `toString()` de la classe `Banque`.
Les opérations publiques sur une `Banque` sont :
- `ajouterClient(...)`
- Ouvrir un nouveau compte (qui est bien sûr ajouté dans `Vector`) :
  `public int ajouterCompte(Client titulaire, double montant)` ;
- Créditer un compte dont on connait le numéro :
  `public void depot(int numero, double montant)` ;
- Débiter un compte dont on connaît le numéro :
  `public void retrait(int numero, double montant)` ;
- Afficher le solde et le nom du titulaire d'un compte dont on connait le numéro :
  `public void afficher(int numero)` ;
- Cherche dans `listeCompte` s'il existe un compte dont le numéro est indiqué en paramètre à la méthode :

`public Compte chercherCompte(int numero)` ;
Cette méthode retourne ce compte s'il en existe un, retourne `null` sinon.
- Afficher la liste des comptes qui sont à découvert (on se contentera d'afficher les numéros des comptes qui sont à découvert) :
`public void afficheCompteDecouvert()`.

Vous écrirez ensuite une méthode `main()` dans laquelle vous exécuterez les instructions suivantes :
- Créer deux clients, Dupont et Durand, et les ajouter à la banque.
- Pour monsieur Dupont, créer un compte courant avec un seuil de découvert autorisé de -500. Ajouter 1000 sur son compte. Retirer 100. Retirer 2000.
- Pour monsieur Dupont, créer un compte épargne logement avec un taux d'intérêt de 3%. Ajouter 200 sur ce compte.
- Pour monsieur Durand, créer un compte courant. Ajouter 800.
- Afficher le contenu de la banque : clients et comptes.

5 Gestion des exceptions

Modifiez votre programme de façon à lancer une exception lors d'une tentative d'opération illégale.

Créez une exception `CompteException` qui hérite d'`Exception` et qui ne définit qu'un seul constructeur (et aucune autre méthode) `CompteException(String)` qui appelle le constructeur de la classe mère.

6 Quelques notions bancaires

Un compte bancaire est identifié par un numéro de compte. Ce numéro de compte est un entier positif permettant de désigner et distinguer sans ambiguïté possible chaque compte géré par l'établissement bancaire. Chaque compte possède donc un numéro unique.

Solde du compte est la somme d'argent disponible sur un compte est exprimée en Dinars. Cette somme est désignée sous le terme de solde du compte. Ce solde est un nombre décimal qui peut être positif (solde créditeur), nul ou négatif (solde débiteur).

Créditer un compte consiste à ajouter un montant positif au solde du compte.

Débiter un compte consiste à retirer un montant positif au solde du compte ne dépassant pas le débit maximal autorisé. Le solde résultant ne doit en aucun cas être inférieur au découvert maximal autorisé pour ce compte.

Agios Lorsque le compte en banque présente un solde négatif, il est dit " à découvert ". En contrepartie du fonctionnement du découvert, la banque perçoit des agios, c'est à dire des intérêts débiteurs.

Bon travail !