TDD Test Driven Developpment

Dans cet exercice il s'agit d'adopter une méthode dirigée par les tests (Test Driven Development) pour réaliser une classe qui répresente des comptes bancaires.

4.1. "Cahier des charges"

Il s'agit de définir une (des) classe(s) JAVA permettant de modéliser des comptes 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. Ce numéro est attribué par la banque à l'ouverture du compte et ne peut être modifié par la suite. Dans le code de la classe Compte et des tests que vous allez écrire, vous ne vous préoccuperez pas de la manière dont ce numéro est obtenu ni si il est unique : vous créerez des objets compte en fournissant un numéro arbitraire, le code de la classe compte ne se souciant pas de l'unicité de ce numéro (cela sera le problème du programme utilisant les comptes).
- Un compte est associé à une personne titulaire du compte, cette personne étant décrite par son nom, son prénom et son adresse. Une fois le compte créé, le titulaire du compte ne peut plus être modifié. Une même personne peut être titulaire de plusieurs comptes.
- La somme d'argent disponible sur un compte est exprimée en Euros. Cette somme est désignée sous le terme de solde du compte. Ce solde est un nombre décimal qui peut être positif, nul ou négatif.
- Le solde d'un compte peut être éventuellement (et temporairement) être négatif. Dans ce cas, on dit que le compte est à découvert. Le decouvert d'un compte est nul si le solde du compte est positif ou nul, il est égal à la valeur absolue du solde si ce dernier est négatif.
- En aucun cas le solde d'un compte ne peut être inférieur à une valeur fixée pour ce compte. Cette valeur est définie comme étant - (moins) le découvert maximal autorisé pour ce compte. Par exemple pour un compte dont le découvert maximal autorisé est 2000 €, le solde ne pourra pas être inférieur à -2000 €. Le découvert maximal autorisé peut varier d'un compte à un autre, il est fixé arbitrairement par la banque à la création du compte et peut être ensuite révisé selon les modifications des revenus du titulaire du compte.
- 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. Le solde résultant ne doit en aucun cas être inférieur au découvert maximal autorisé pour ce compte.
- Lors d'une opération de retrait, un compte ne peut être débité d'un montant supérieur à une valeur désignée sous le terme de débit maximal autorisé. Comme le découvert maximal autorisé, le débit maximal autorisé peut varier d'un compte à un autre et est fixé arbitrairement par la banque à la création du compte. Il peut être ensuite révisé selon les modifications des revenus du titulaire du compte.
- Effectuer un virement consiste à débiter un compte au profit d'un autre compte qui sera crédité du montant du débit.
- Lors de la création d'un compte seul l'identité du titulaire du compte est indispensable. En l'absence de dépot initial le solde est fixé à 0. Les valeurs par défaut pour le découvert maximal autorisé et le débit maximal autorisé sont respectivement de 800 € et 1000 €. Il est éventuellement possible d'attribuer d'autres valeurs à ces caractéristiques du compte lors de sa création.
- Toutes les informations concernant un compte peuvent être consultées : numéro du compte, identité du titulaire, montant du découvert maximal autorisé, montant du débit maximal autorisé, situation du compte (est-il à découvert ?), montant du débit autorisé (fonction du solde courant et du débit maximal autorisé).

4.2. Travail demandé

Il vous est démandé de définir et tester une classe (**Compte**) qui doit permettre à une application de créer et utiliser autant de comptes bancaires que nécessaires, chaque compte étant un objet, instance (ou exemplaire) de la classe **Compte**.

Question 1 : A partir du "cahier des charges" précédent élaborer une spécification d'une classe Java modélisant un compte bancaire.

Il s'agira en analysant le texte ci-dessus de :

- définir les attributs (variables d'instance, variables de classe) de la classe **Compte**,
- d'identifier les méthodes publiques proposées par la classe **Compte**. Pour chaque méthode on prendra soin, outre la définition de sa signature, de spécifier son comportement sous la forme d'un commentaire documentant.
- de proposer un ou plusieurs constructeurs pour la classe **Compte**. Là aussi on complétera la donnée de la signature de chaque constructeur avec un commentaire documentant détaillant son utilisation.

Ecrivez le squelette de la classe **Compte** en mettant dans le corps de chaque constructeur ou méthode une instruction

```
throw new UnsupportedOperationException("Opération pas encore implémentée)
;
```

indications pour la classe Compte:

- le numéro de compte est attribué à la création d'un compte, il doit donc apparaître comme paramètre du(des) constructeur(s), une fois l'objet Compte créé il ne peut ensuite être modifié.
- lorsqu'une opération illégale est effectuée (par exemple débiter un compte avec un montant négatif), levez une exception de type **IllegalArgumentException**

Question 2 : Une fois les spécifications de votre classe **Compte** effectuées écrivez un programme JUnit pour ses tests unitaires

1. générez le code du programme JUnit pour les tests untitaires,
2. ecrivez les différents cas de test.

Question 3 : Réalisez l'implémentation de la classe **Compte** et au fur et à mesure que vous complétez le code vérifiez que le code que vous avez écrit passe les tests

Question 4 : Une fois l'implémentation de **Compte** terminée, vérifiez la couverture de vos tests. Si le code de votre classe n'est pas couvert en totalité complétez les tests.