**TRAVAUX PRATIQUES N °**1 à 5

Gestion de compte bancaire

**Objectifs:**

* Prise en main d'un environnement de développement NetBeans
* Manipulation de la syntaxe de base de JAVA (variable, traduction des notions algorithmiques en JAVA, utilisation des fonctions, initiation à l’objet)
* Réalisation d’une application de gestion de compte bancaire par une succession de TP liés les uns aux autres.

**Outils/Matériel:**

Environnement NetBeans sous Windows

TP 1 : Gestion d’un compte bancaire

On commence ici une suite de TP pour gérer un compte bancaire.

**Question 1**

Objectif : gérer l’affichage d’un menu, la saisie d’informations entrées au clavier et leur affichage à l’écran

1. L'objectif de ce premier programme est d'écrire toutes les instructions qui permettent l'affichage du menu ainsi que la saisie des données demandées et le traitement associé. Le programme affiche tous les messages de toutes les options, sans contrôle sur le choix de l'utilisateur.

Le menu principal s'affiche de la façon suivante :

* + 1. Créer un compte
    2. Afficher un compte
    3. Créer une ligne comptable
    4. Sortir

1. Lors de la création du compte, le programme doit poser les questions suivantes

Type du compte [Types possibles : courant, joint, épargne]:

* + - * Numéro du compte :
      * Première valeur créditée :
      * Taux de placement , à afficher uniquement pour le compte épargne

1. A la fin de ces saisies, toutes les valeurs entrées sont affichées.
2. Lors de l’affichage d’un compte, demandez au programme de vérifier que le numéro du compte saisi par l'utilisateur existe, de façon à :
   * + - Afficher le numéro du compte, le type, la valeur initiale et son taux dans le cas d’un compte d'épargne, si le compte existe.
       - Afficher un message indiquant que le numéro du compte n'est pas valide, si le compte n'existe pas.
3. Lors de la ligne comptable, le programme doit poser les questions suivantes

- Le numéro du compte concerné (avec vérification de son existence) .

- La somme à créditer (valeur positive) ou à débiter (valeur néga­tive) .

- La date de l'opération , format chaîne de caractères.

- Le motif de l'achat ou de la vente [thèmes possibles : Salaire, Loyer, Alimentation, Divers]

- Le mode de paiement [Types possibles : CB, n° du Chèque, Virement].

1. Créer une boucle pour faire réapparaître le menu une fois l'option choisie réalisée.

**Question 2**

Objectif : maîtriser l’appel et l’écriture de fonctions

1. Créer deux fonctions, MenuPrincipal et AfficherCompte.
2. Menuprincipal : elle permet d’afficher le menu et de récupérer en sortie l’option choisie par l’utilisateur.
3. AfficherCompte : elle affiche toutes les caractéristiques du compte . Proposer deux versions à cette fonction ; la première prendra comme paramètre de fonction toutes les caractéristiques du compte ; la seconde s’appuiera sur des variables de classe.

TP 2 : Gestion d’un compte bancaire avec des classes

**Objectif**: Conception d’une classe, création d’objet.

**Question 1**

1. Créer la classe Compte
2. Dans un fichier nommé Compte. java, définissez la classe Compte en procédant de la façon suivante :
3. Déterminez les données qui définissent le compte bancaire
4. Écrivez les méthodes associées :
   * créerCompte
   * afficherCompte
5. Modifier le programme principal (main) pour la création du compte et l’affichage du compte

**Question 2**

1. Modifier le menu pour la gestion d’une ligne comptable
   * Le menu principal s'affiche de la façon suivante :
     1. Créer un compte
     2. Afficher un compte
     3. Créer une ligne comptable
     4. Sortir
2. Créer la classe LigneComptable
3. Déterminez les données qui définissent la ligne comptable
4. Écrivez les méthodes associées:
   * créerLigneComptable
   * afficherLigneComptable

**Question 3**

1. Lier un Compte à une ligne comptable
2. Faire un diagramme de classe UML.
3. Dans la classe Compte
4. Définissez une nouvelle donnée décrivant une ligne comptable.
5. Ecrivez la méthode créerLigne ( ) qui permette les actions suivantes :
   * créer en mémoire l'objet ligne.
   * faire appel à la méthode créerLigneComptable ( ) par l'intermédiaire de l'objet ligne de façon à enregistrer les valeurs numériques associées à la ligne créée ;
   * modifier la valeur courante du compte à partir de la valeur (débit ou crédit) saisie dans la méthode créerLigneComptable ( ).
6. c. Modifiez la méthode afficherCpte () de façon à afficher les informations stockées dans ligne, en utilisant la méthode afficherLigne ( ) .
7. Modifier le programme principal (main) pour la création de la ligne comptable et l’affichage de cette ligne comptable.

TP 3: Heritage

**Objectif**: **Comprendre l'héritage**

**Protection des données héritées**

Sachant qu'un compte d'épargne est un compte bancaire ayant un taux de rémunération:

a. Écrivez la classe CpteEpargne en prenant soin de déclarer la nouvelle donnée en mode

private.

b. Modifiez le type Compte de façon à supprimer tout ce qui fait appel au compte d'épargne (donnée et méthodes).

Un compte d'épargne modifie la valeur courante par le calcul des intérêts, en fonction du taux d'épargne. Il ne peut ni modifier son numéro, ni son type.

c. Faire un diagramme de classe UML.

**Le contrôle des données d'un compte d'épargne**

Sachant que le taux d'un compte d'épargne ne peut être négatif, écrivez la méthode contrôleTaux ( ) pour que celle-ci demande à l’utilisateur de rentrer un taux non négatif .

**Le constructeur d'une classe dérivée**

Ecrire le constructeur de la classe CpteEpargne pour qu’il appelle le constructeur de la classe mère et appelle la méthode contrôleTaux ( ) définie à la question suivante.

Modifiez ce constructeur de façon à ce que la donnée typeCpte prenne la valeur Epargne.

**Le polymorphisme**

De la méthode affiche afficherCpte( ) :

a. Dans la classe CpteEpargne, écrivez la méthode afficherCpte ( ) , sachant qu'afficher les données d'un compte d'épargne revient à afficher les données d'un compte, suivies du taux d'épargne.

De l'objet C, déclaré de type Compte :

b. Dans l'application, modifiez l'option 1, de façon à demander à l'utilisateur s’il souhaite créer un compte simple ou un compte d'épargne. Selon la réponse, construisez l'objet C en appelant le constructeur approprié.

TP 4: Tableaux , fichiers et listes dynamiques

**Question 1**

**Objectif**: Maîtriser les tableaux .

créer un tableau de 10 lignes comptables dans la classe Compte.

Faire un diagramme de classe UML.

a. Si le nombre de lignes créées est inférieur à 10 lignes , créer en mémoire une ligne comp­table.

c. Si le nombre de lignes est supérieur à 10 lignes, supprimer la plus ancienne ligne comptable , et stocker la nouvelle ligne comptable en dernière position du tableau ligne.

d. Modifier la valeur courante du compte en fonction du crédit ou débit réalisé par la nouvelle ligne comptable.

e. Modifier la méthode afficherCompte() de façon à afficher l’ensemble des lignes saisies en cours d’exécution du programme.

**Question 2**

**Objectif :** implémenter un tri à bulle.

a. Le tableau de comptes devra être trié sur la base d’un tri à bulle, le critère de tri étant le numéro de compte. Ce tri à bulle sera implémenté dans une méthode.

b. Faire appel à cette méthode pour trier le tableau de comptes.

c. pensez à gérer les exceptions.

**Question 3**

**Objectif**: Maîtriser les ArrayList et la généricité

Reprendre la question 1 en remplaçant le tableau de lignes comptables par un ArrayList. Ne pas gérer la limite des 10 lignes. Mettre en place le tri de la liste de compte avec la démarche suivante :

1. La classe Compte Héritera dans un premier temps de l’interface Comparable avec l’écriture suivante :
   * 1. Class Compte implements Comparable
2. Surcharger la méthode CompareTo de la classe Compte de manière à comparer deux comptes
3. Appeler Collections.sort( tableau de comptes) : cet appel fera appel à la méthode statique sort de la classe Collections avec comme paramètre le tableau de comptes.
4. Changer ensuite la déclaration de la classe Compte de la manière suivante :
   * 1. Class Compte implements Comparable <Compte>
     2. Faire les changements nécessaires dans votre code

TP 5: JDBC

**Question 1**

**Objectif**: **écriture en base de données**

**Ecriture en base de données simple**

1. Proposer la sauvegarde des comptes et des lignes comptables en base de données.