

TD 2

Programmation Java

Exercice 1 :

On veut modéliser un compte bancaire. Chaque compte est défini par :

- un numéro de compte (chaîne de caractères),
- un nom de titulaire (chaîne),
- un solde (double).

1) Créer une classe **CompteBancaire** contenant ces trois attributs privés.

2) Ajouter trois constructeurs :

- un constructeur par défaut qui crée un compte avec :
 - numéro = "0000",
 - nom = "Inconnu",
 - solde = 0.0.
- un constructeur à deux paramètres (numéro, nom) → solde par défaut à 0.0.
- un constructeur à trois paramètres (numéro, nom, solde).

3) Dans le constructeur à deux paramètres, réutiliser le constructeur à trois paramètres avec `this(...)`.

4) Ajouter les méthodes suivantes :

- `void deposer(double montant)` → ajoute au solde.
- `void retirer(double montant)` → retire du solde (si le solde est suffisant).
- `void afficher()` → affiche toutes les informations du compte.

5) Dans la classe **MainCompte**, crée plusieurs objets pour tester :

- un compte par défaut,
- un compte avec deux paramètres,
- un compte avec trois paramètres.

Effectue quelques dépôts et retraits, puis affiche les résultats.

Exercice 2 :

On souhaite créer une classe **TableCarres** qui contient un tableau statique carres représentant les carrés des entiers de 0 à 9.

Déclarez une classe **TableCarres** contenant :

un attribut statique `int[] carres` ;

un bloc d'initialisation statique permettant d'allouer le tableau et de le remplir avec les carrés des nombres de 0 à 9 ;

une méthode statique `afficher()` qui affiche le contenu du tableau sur une même ligne.

Créez une classe `TestTableCarres` contenant la méthode `main`.

Depuis cette méthode : Appelez `TableCarres.afficher()` pour afficher les valeurs du tableau.

Résultat :

Bloc static exécuté : tableau initialisé !

0 1 4 9 16 25 36 49 64 81

Exercice 3 :

Vous allez concevoir une petite application de suivi de dépenses pour un utilisateur. Cette application permet de :

1. Créer un utilisateur avec un nom et un solde initial.
2. Enregistrer une série de dépenses (montant + description).
3. Afficher les informations du profil.
4. Consulter le solde mis à jour après chaque dépense.
5. Afficher un récapitulatif des dépenses.
6. Calculer récursivement le total des dépenses

Détails techniques

Classe `Depense`

- `double montant` : le montant dépensé.
- `String description` : courte description (ex. : "Transport", "Nourriture").
- Méthode `toString()` pour afficher une dépense.

Classe `Utilisateur`

- `String nom`
- `double solde`
- `Depense[] depenses` : un tableau fixe de dépenses (ex. taille max 10)
- `int nbDepenses` : compteur du nombre réel de dépenses enregistrées

Méthodes :

- `void ajouterDepense(double montant, String description)` : ajoute une dépense si solde suffisant.
- `void afficherDepenses()` : affiche la liste des dépenses.
- `double calculerDepenses(int n)` : méthode récursive pour calculer le total des n premières dépenses.
- `void afficherProfil()` : affiche nom, solde, nombre de dépenses.

Exemple de test dans `main()`

```
public class Test {  
    public static void main(String[] args) {  
        Utilisateur u = new Utilisateur("Salma", 1000.0);  
        u.afficherProfil();  
        u.ajouterDepense(150.0, "Transport");  
    }  
}
```

```
u.ajouterDepense(200.0, "Courses");
u.ajouterDepense(100.0, "Livres");
u.afficherDepenses();

        System.out.println("Total    des    dépenses    :    "    +
u.calculerDepenses(u.getNbDepenses()) + " MAD");
    u.afficherProfil();
}
}
```

Exercice 4 :

Vous devez créer une application Java simple qui modélise un étudiant et ses évaluations continues. Chaque étudiant peut avoir un ensemble limité de notes (maximum 5), et le système doit permettre d'enregistrer les notes, afficher les informations de l'étudiant, et calculer sa moyenne générale de manière récursive.

Classes à créer :

1. Classe Evaluation

- o Attributs :
 - int note (entre 0 et 20)
 - String matiere (facultatif si les chaînes ne sont pas encore vues)
- o Méthodes :
 - Constructeur
 - int getNote()
 - String toString() (affiche la note et la matière si disponible)

2. Classe Etudiant

- o Attributs :
 - String nom
 - int codeApogee
 - Evaluation[] evaluations (taille fixe = 5)
 - int nbEvaluations
- o Méthodes :
 - Constructeur
 - void ajouterNote(int note)

- void afficherNotes()
- double calculerMoyenne(int n) // méthode récursive
- String toString() (nom, codeApogée, nombre d'évaluations)

Exemple de test dans main()

```
public class Test {  
    public static void main(String[] args) {  
        Etudiant e1 = new Etudiant("Omar El Idrissi", 123456);  
        e1.ajouterNote(15);  
        e1.ajouterNote(13);  
        e1.ajouterNote(17);  
  
        e1.afficherNotes();  
        System.out.println("Moyenne : " + e1.calculerMoyenne(e1.getNbEvaluations()));  
  
        System.out.println(e1);  
    }  
}
```