

# TDs Entrainement Java Guillaume Sanchez

---

## Exercice 1 : Création d'une classe Personne

---

### 1. Crée une classe Personne avec les attributs suivants :

- nom (String)
- âge (int)

```
public class Personne {  
  
    private String nom;  
    private int age;  
  
    Personne(){  
        this.nom = "Inconnu";  
        this.age = 0;  
    }  
  
    Personne(String nomInput, int ageInput){  
        this.nom = nomInput;  
        this.age = ageInput;  
    }  
}
```

### 2. Crée les getters et setters pour ces attributs.

```
public class Personne {  
  
    private String nom;  
    private int age;  
  
    Personne(){  
        this.nom = "Inconnu";  
        this.age = 0;  
    }  
  
    Personne(String nomInput, int ageInput){  
        this.nom = nomInput;  
        this.age = ageInput;  
    }  
}
```

```

    }

    public String getNom(){
        return this.nom;
    }

    public void setNom(String nomInput){
        this.nom = nomInput;
    }

    public int getAge(){
        return this.age;
    }

    public void setAge(int ageInput){
        this.age = ageInput;
    }
}

```

**3. Dans la classe Main, crée un objet Personne, affecte-lui un nom et un âge, puis affiche ses valeurs.**

```

public class App {
    public static void main(String[] args) throws Exception {

        // Création d'une personne :
        Personne personne1 = new Personne("Jean", 25);
        // Affichage des données de la personne :
        System.out.println(personne1.getNom() +
            " a " + personne1.getAge() + " ans.");
    }
}

```

Ce code donne en résultat :

```
Jean a 25 ans.
```

## Exercice 2 : Gestion d'un Compte Bancaire

---

**1. Crée une classe CompteBancaire avec les attributs privés :**

- titulaire (String)
- solde (double)

```

public class CompteBancaire {

    private double solde;
    private String titulaire;

    CompteBancaire(){
        this.solde = 0;
        this.titulaire = "Inconnu";
    }

    CompteBancaire(double soldeInput, String titulaireInput){
        this.solde = soldeInput;
        this.titulaire = titulaireInput;
    }
}

```

## 2. Implémente les getters et setters, mais :

- Empêche la modification du solde en dehors des méthodes de la classe.
- Ajoute une méthode déposer(double montant) qui augmente le solde.
- Ajoute une méthode retirer (double montant) qui diminue le solde uniquement si l'utilisateur a assez d'argent.

```

public class CompteBancaire {

    private double solde;
    private String titulaire;

    CompteBancaire(){
        this.solde = 0;
        this.titulaire = "Inconnu";
    }

    CompteBancaire(double soldeInput, String titulaireInput){
        this.solde = soldeInput;
        this.titulaire = titulaireInput;
    }

    public void déposer(double montant){
        this.setSolde(this.getSolde() + montant);
    }

    public void retirer(double montant){
        this.setSolde(this.getSolde() - montant);
    }
}

```

```

    public double getSolde(){
        return this.solde;
    }

    private void setSolde(double soldeInput){
        this.solde = soldeInput;
    }

    public String getTitulaire(){
        return this.titulaire;
    }

    public void setTitulaire(String titulaireInput){
        this.titulaire = titulaireInput;
    }
}

```

### 3. Dans Main, crée un compte, effectue des dépôts et retraits, et affiche le solde.

```

public class App {
    public static void main(String[] args) throws Exception {

        // Création d'un compte bancaire :
        CompteBancaire compte1 = new CompteBancaire(1000, "Jean");

        // affichage des données initialisées :
        System.out.println(compte1.getTitulaire()
        + " a un solde de " + compte1.getSolde() + " euros.");

        // dépôt de 500 euros :
        compte1.deposer(500);
        // affichage du nouveau solde :
        System.out.println(compte1.getTitulaire()
        + " a un solde de " + compte1.getSolde() + " euros.");

        // retrait de 200 euros :
        compte1.retirer(200);
        // affichage du nouveau solde :
        System.out.println(compte1.getTitulaire()
        + " a un solde de " + compte1.getSolde() + " euros.");
    }
}

```

Ce code donne en résultat :

```

Jean a un solde de 1000.0 euros.
Jean a un solde de 1500.0 euros.

```

## Exercice 3 : Gestion d'un Produit

---

### 1. Crée une classe **Produit** avec :

- nom (String)
- prix (double)
- quantiteStock (int)

```
public class Produit {  
  
    private String nom;  
    private double prix;  
    private int quantiteStock;  
  
    Produit(){  
        this.nom = "Inconnu";  
        this.prix = 0;  
    }  
  
    Produit(String nomInput, double prixInput){  
        this.nom = nomInput;  
        this.prix = prixInput;  
    }  
}
```

### 2. Implémente les **getters** et **setters**, en ajoutant :

- Une validation dans setPrix(double prix): ne pas autoriser un prix négatif.
- Une validation dans setQuantiteStock(int quantite): ne pas accepter une quantité négative.

```
public class Produit {  
  
    private String nom;  
    private double prix;  
    private int quantiteStock;  
  
    Produit(){  
        this.nom = "Inconnu";  
        this.prix = 0;  
    }  
  
    // Méthodes à implémenter  
}
```

```

Produit(String nomInput, double prixInput){
    this.nom = nomInput;
    this.prix = prixInput;
}

public String getNom(){
    return this.nom;
}

public void setNom(String nomInput){
    this.nom = nomInput;
}

public double getPrix(){
    return this.prix;
}

public void setPrix(double prixInput){
    if(prixInput < 0){
        System.out.println("
        Le prix ne peut pas être négatif.");
    }
    else{
        this.prix = prixInput;
    }
}

public int getQuantiteStock(){
    return this.quantiteStock;
}

public void setQuantiteStock(int quantiteStockInput){
    if(quantiteStockInput < 0){
        System.out.println("
        La quantité ne peut pas être négative.");
    }
    else{
        this.quantiteStock = quantiteStockInput;
    }
}
}

```

### 3. Dans Main, crée un produit et teste les restrictions.

```

public class App {
    public static void main(String[] args) throws Exception {
        // Création d'un produit :
    }
}

```

```

Produit produit1 = new Produit("Ordinateur", 1000);

// Essai de modification du prix avec une valeur négative
// Affiche un message d'erreur :
produit1.setPrix(-5);

// Essai de modification du prix avec une valeur positive :
produit1.setPrix(5);

// Affichage du nouveau prix :
System.out.println("Le prix est " +
produit1.getPrix() + " euros.");

// Essai de modification du stock avec une valeur négative
// Affiche un message d'erreur :
produit1.setQuantiteStock(-5);

// Essai de modification du stock avec une valeur positive :
produit1.setQuantiteStock(5);

// Affichage de la nouvelle quantité de stock :
System.out.println("La Quantité du Stock est " +
produit1.getQuantiteStock());
    }
}

```

Ce code donne en résultat :

```

Le prix ne peut pas être négatif.
Le prix est 5.0 euros.
La quantité ne peut pas être négative.
La Quantité du Stock est 5

```

## Exercice 4 : Gestion des étudiants

---

### 1. Crée une classe Etudiant avec les attributs :

- nom (String)
- moyenne (double)

```

public class Etudiant {

    private String nom;
    private double moyenne;

```

```

Etudiant() {
    this.nom = "Inconnu";
    this.moyenne = 0;
}

Etudiant(String nomInput, double moyenneInput) {
    this.nom = nomInput;
    this.moyenne = moyenneInput;
}
}

```

## 2. Implémente les getters et setters avec une contrainte :

- La moyenne doit être comprise entre 0 et 20. Si une valeur hors de cet intervalle est donnée, elle n'est pas prise en compte.

```

public class Etudiant {

    private String nom;
    private double moyenne = -1;

    /* -1 instancié par défaut pour qu'elle
    ne soit pas prise en compte en cas de mauvaise
    donnée
    */

    Etudiant() {
        this.nom = "Inconnu";
        this.moyenne = -1;
    }

    Etudiant(String nomInput, double moyenneInput) {
        this.nom = nomInput;
        if(moyenneInput < 0 || moyenneInput > 20) {
            System.out.println("La moyenne doit être comprise entre 0 et 20.");
        }
        else{
            this.moyenne = moyenneInput;
        }
    }

    public String getNom() {
        return this.nom;
    }

    public void setNom(String nomInput) {
        this.nom = nomInput;
    }
}

```



```

    }

    public double getMoyenne() {
        return this.moyenne;
    }

    public void setMoyenne(double moyenneInput) {
        if(moyenneInput < 0 || moyenneInput > 20) {
            System.out.println("La moyenne doit être comprise entre 0 et 20.");
        }
        else{
            this.moyenne = moyenneInput;
        }
    }
}

```

### 3. Ajoute une méthode `afficherDetails()` pour afficher les infos de l'étudiant.

```

public void afficherDetails() {
    System.out.println("Nom : " + this.nom);
    if(this.moyenne >= 0 && this.moyenne <= 20) {
        System.out.println("Moyenne : " + this.moyenne);
    }
    else {
        System.out.println("Moyenne : Non définie.");
    }
}

```

### 4. Dans `Main`, crée plusieurs étudiants et teste les restrictions.

```

public class App {
    public static void main(String[] args) throws Exception {

        System.out.println("-----");
        // Création d'un Etudiant :
        Etudiant etudiant1 = new Etudiant("Jean", 15);
        // Utilisation de la méthode afficherDetails() :
        etudiant1.afficherDetails();
        System.out.println("-----");
        // Création d'un second Etudiant avec une moyenne incorrecte supérieur à 20 :
        Etudiant etudiant2 = new Etudiant("Paul", 25);
        // Utilisation de la méthode afficherDetails() :
        etudiant2.afficherDetails();
        System.out.println("-----");
    }
}

```

```

// Création d'un troisième Etudiant avec une moyenne incorrecte inferieur à 0 :
Etudiant etudiant3 = new Etudiant("Marie", -5);
etudiant3.afficherDetails();
System.out.println("-----");
// Création d'un quatrième Etudiant avec une moyenne correcte :
Etudiant etudiant4 = new Etudiant("Luc", 18);
etudiant4.afficherDetails();
System.out.println("-----");
// Tentative de changement de la moyenne avec une valeur positive incorrecte :
etudiant4.setMoyenne(25);
etudiant4.afficherDetails();
System.out.println("-----");
// Tentative de changement de la moyenne avec une valeur negative incorrecte :
etudiant4.setMoyenne(-30);
etudiant4.afficherDetails();
System.out.println("-----");
// Tentative de changement de la moyenne avec une valeur correcte :
etudiant4.setMoyenne(10);
etudiant4.afficherDetails();
System.out.println("-----");
    }
}

```

Ce code donne en résultat :

```

-----
Nom : Jean
Moyenne : 15.0
-----
La moyenne doit être comprise entre 0 et 20.
Nom : Paul
Moyenne : Non définie.
-----
La moyenne doit être comprise entre 0 et 20.
Nom : Marie
Moyenne : Non définie.
-----
Nom : Luc
Moyenne : 18.0
-----
La moyenne doit être comprise entre 0 et 20.
Nom : Luc
Moyenne : 18.0
-----
La moyenne doit être comprise entre 0 et 20.
Nom : Luc
Moyenne : 18.0
-----
Nom : Luc

```

Moyenne : 10.0

-----