Développement Orientée Objets — *IUT Montpellier-Sète*
Année 2022-2023 — *Département Informatique*
M. Bougeret, R. Lebreton, C. Nadal, V. Poupet, G. Trombettoni, P. Valicov

Extrait du sujet d'examen de 2023 : durée 1H30

CONSIGNES :

Document autorisé : une feuille A4 manuscrite recto/verso. Matériel électronique interdit.

Le barème (excédentaire) est donné à titre indicatif.

Les réponses non argumentées ne seront pas considérées !

Votre écriture doit être claire et lisible.

Chaque partie est à rendre sur une copie séparée.

Partie 1 – tours de notions

Durée estimée : 45 minutes

1. (*1 point*) Quel est l'affichage produit lorsque le code suivant est compilé et exécuté ?

```java
import java.util.Objects;

public class Truc {
    private String nom;

    public Truc(String nom) {
        this.nom = nom;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass())
            return false;
        Truc truc = (Truc) o;
        return Objects.equals(nom, truc.nom);
    }
}
```

```java
public class App {
    public static void main(String[] args) {
        Truc s1 = new Truc("Patate");
        Truc s2 = new Truc("Patate");
        Truc s3 = new Truc("Vache");

        if (s1 == s2)
            System.out.println("Pareils 1");
        if (s1.equals(s2))
            System.out.println("Égaux 1");

        s2 = s3;

        if (s1 == s2)
            System.out.println("Pareils 2");
        if (s1.equals(s2))
            System.out.println("Égaux 2");

        s3 = s1;

        if (s1 == s3)
            System.out.println("Pareils 3");
        if (s1.equals(s3))
            System.out.println("Égaux 3");

        if (s2 == s3)
            System.out.println("Pareils 4");
        if (s2.equals(s3))
            System.out.println("Égaux 4");
    }
}
```

2. (*3 points*) Expliquez les problèmes du programme donné ci-dessous (il ne vous est pas demandé de corriger) :

```
public abstract class Vehicule {
    private String marque;
    private int prix;

    public Vehicule(int p) {
        prix = this.p;
    }

    public void setMarque(String m) {
        if (!marque.equals(m))
            marque = m;
    }

    public String getMarque() {
        return marque;
    }
}
```

```
public abstract class QuatreRoues extends Vehicule {
   public QuatreRoues(String marque, String modele) {
      super(marque, modele);
   }
}
```

```
public class Voiture extends QuatreRoues {
    public Voiture(String marque, String modele) {
        super(marque, modele);
    }
}
```

```
public class Camion extends QuatreRoues {
    public Camion(String marque, String modele) {
        super(marque, modele);
    }
}
```

```
public class AppVehicules {
    public static void main(String[] args) {
        Vehicule vehicule = new QuatreRoues("Peugeot", "206");
        QuatreRoues q = new Voiture("Renault", "Zoe");
        q.setMarque("Peugeot");
        Voiture v = new Voiture("Toyota", "Yaris");
        q = v;
        Camion camion = new Camion("Volvo", "FH");
        v = camion;
        Vehicule vehicule2 = new Camion("Iveco", "Trakker");
        vehicule2 = v;
        v = vehicule2;
    }
}
```

Remarque : pour chaque erreur, qui n'en est pas une, des points seront enlevés.

3. (*1 point*) Soient les 3 classes suivantes.

```
public class Docteur {
    public void attendreConsultation() {
        System.out.println("Attente d'une consultation");
        consulter();
    }

    public void consulter() {
        System.out.println("Le médecin consulte");
        payer();
    }

    public void payer() {
        System.out.println("Facture du docteur : 25 €");
    }
}
```

```
public class Chirurgien extends Docteur {
    @Override
    public void consulter() {
        System.out.println("Le chirurgien consulte");
        payer();
    }

    @Override
    public void payer() {
        System.out.println("Facture du chirurgien : 50 €");
    }

    public void opérer() {
        System.out.println("Le chirurgien opère");
    }
}
```

```
public class AppDocteur {
    public static void main(String[] args) {
        Docteur d = new Chirurgien();
        d.attendreConsultation();
        d.consulter();
        d.opérer();
    }
}
```

Pour chaque ligne de `AppDocteur`, dire si elle compile. Si c'est le cas, indiquez l'affichage produit par cette ligne.

4. (*1 point*) On vous rappelle que la collection `HashSet` de l'API Java permet d'éviter les doublons. Soit une classe `Toto` dont le code ne vous est pas donné. Soient les instructions suivantes :

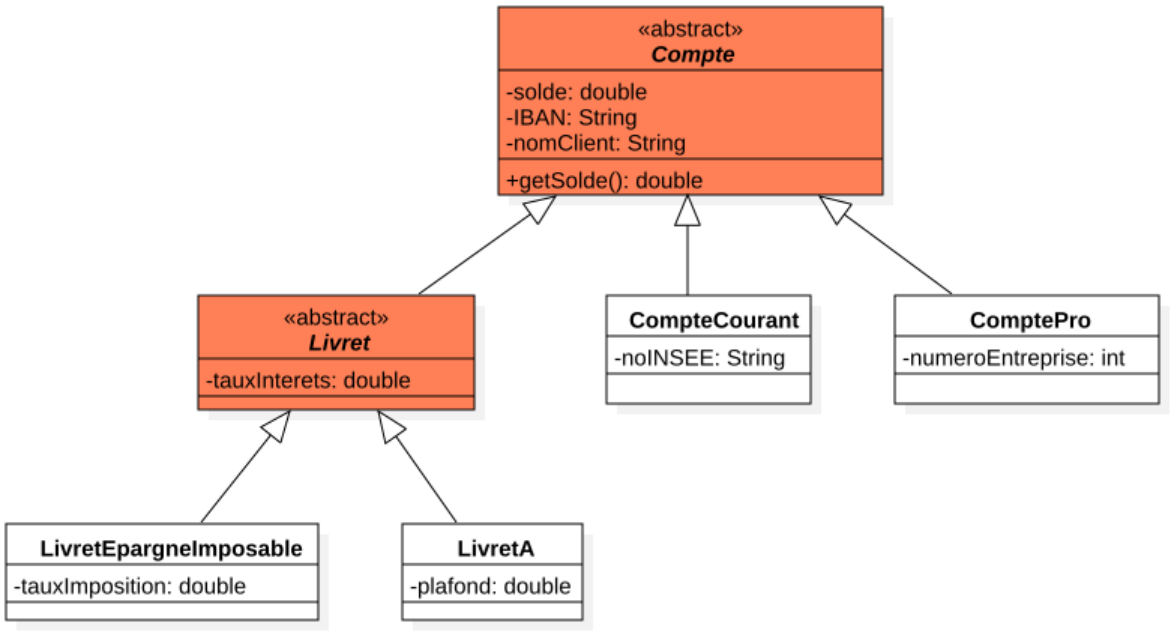
```
HashSet<Toto> ensemble = new HashSet<>();
Toto t1 = // du code initialisant t1
Toto t2 = // du code initialisant t2
ensemble.add(t1); ensemble.add(t2);
```

Expliquez (sans écrire du code) les contraintes que la classe `Toto` doit satisfaire, afin que l'objet `HashSet<Toto> ensemble` fonctionne correctement, à savoir que les doublons soient évités à l'invocation de la fonction `add(...)`.

Partie 2 – Comptes bancaires

Durée estimée : 45 minutes

Vous êtes chargés de compléter un logiciel gérant des comptes bancaires. Le code source principal est déjà écrit. Voici son diagramme de classes :



Pour vous permettre de mieux vous situer, voici le code de la classe `Compte` :

```
public abstract class Compte {
    private double solde;
    private String IBAN; // coordonnées bancaires du compte
    private String nomClient;

    public Compte(double solde, String IBAN, String nomClient) {
        this.solde = solde;
        this.IBAN = IBAN;
        this.nomClient = nomClient;
    }

    public double getSolde() {
        return solde;
    }

    @Override
    public String toString() {
        return "Le solde du compte est" + getSolde();
    }
}
```

Dans les questions qui vont suivre, la classe `Compte` doit rester intacte.

1. (*1 point*) Écrivez le code de la fonction `getSolde()` de la classe `Livret` afin qu'elle retourne le solde du compte, majoré par le taux d'intérêts. La valeur de l'attribut `tauxInterets` est un pourcentage : un réel compris entre 0 et 1 (vous pouvez supposer que c'est toujours le cas et n'avez rien à tester là-dessus).
   Par exemple, si le solde du Livret vaut 100 et le taux d'intérêts du Livret est 0.04, alors la fonction du `getSolde()` devra retourner : 104.

2. (*1 point*) Avec le prélèvement à la source par le gouvernement, il faut que le solde des livrets d'épargnes imposables tienne compte du taux d'imposition (attribut correspondant dans la classe `LivreEpargneImposable`). Redéfinissez la méthode `getSolde()` de la classe `LivretEpargneImposable` afin qu'elle retourne le solde du livret, en enlevant ce qui est dû à l'État. Comme pour la question précédente, vous supposerez que l'attribut `tauxImposition` est un pourcentage correspondant à un réel entre 0 et 1.

   Pour simplifier, on suppose aussi dans cette question que la taxe s'applique sur le solde+intérêts du livret (imposable).

3. (*3 points*) Il est possible de détenir un *compte groupé*, qui est un compte regroupant plusieurs comptes. Le nombre de "sous-comptes" autorisés dans un tel compte groupé est potentiellement illimité et il est toujours possible d'en ajouter d'autres. De plus, il est possible d'avoir des comptes groupés dans un compte groupé. Chaque compte groupé aura son propre *coût*, fixé par l'utilisateur (donnée de type `double`) à la construction du compte groupé. Ce coût correspond aux frais bancaires qui permettent de détenir un compte groupé.

   Pour résumer, voici les caractéristiques d'un compte groupé :

   — on peut y ajouter des comptes à travers une méthode `ajouter(...)`
   — la fonction `getSolde()` retournera toujours la somme des soldes des comptes de ce compte groupé
   — le coût (ou frais de création d'un compte groupé) est fixé à la construction du compte groupé.

   Écrivez le code `Java` permettant l'intégration des comptes groupés dans le logiciel actuel.

   Indication : la classe `Compte` devra rester intacte (aucune modification du fichier .java correspondant).

4. (*1 point*) Écrivez une classe principale `AppComptes` avec la méthode `main(String args[])`, qui effectue les actions suivantes dans l'ordre :

   — Crée un compte groupé pour le client "Toto"
   — Lui ajoute un compte courant de "Riri" et un compte groupé de "Fifi". Le compte groupé de "Fifi" doit posséder un livret A et un compte Pro .
   — affiche le solde du compte groupé de "Toto"

   Pour vous permettre d'instancier les objets correctement, voici la signature des constructeurs des classes `LivretA` et `CompteCourant` :

```
public LivretA(double solde, String IBAN, String nomClient, double tauxInterets, double plafond)

public CompteCourant(double solde, String IBAN, String nomClient, String noINSEE)
```

   Vous initialiserez les différents attributs avec des valeurs au choix.