TP Java : À l'ouest de Java-Town…

Objectifs du TP : Comprendre la notion d'héritage et d'interface. Mot clés **super, extends** et **implements**

Dans ce TP, nous allons réaliser un programme Java permettant d'écrire facilement des histoires de Western.
Dans nos histoires, nous auront des brigands, des cowboys, des shérifs, des barmen et des dames en détresse.

Exercice 1 : Diagramme de classe (UML)

Question 1.1: Réaliser un diagramme

Avant de commencer, **réalisez un diagramme de classe du TP.**
Sachez qu'il est nécessaire de lire le sujet plusieurs fois avant de commencer le diagramme.
Vous pourrez utiliser l'outil de votre choix, comme https://www.draw.io/

Exercice 2 : Western

Question 2.1: Tous humains…

Les intervenants de nos histoires sont tous des humains.
Un humain est caractérisé par son nom et sa boisson favorite.
La boisson favorite d'un humain est, par d´défaut, de l'eau.
Un humain pourra parler, nous aurons donc une méthode `parler(String texte)` qui affiche :

```
(nom de l'humain ) – texte
```

Un humain peut également se présenter et boire.

- Lorsqu'il se présente, il dit bonjour, son nom, et indique sa boisson favorite
- Lorsqu'il boit, il dira : « `Ah ! un bon verre de (sa boisson favorite) ! GLOUPS !` »

Toutes les variables de la classe humain sont privées.

Pour connaître le nom d'un humain, on aura besoin d'une méthode `quelEstTonNom()` qui renvoie la chaîne contenant le nom de cet humain.

Pour connaître sa boisson favorite, on aura également besoin d'une méthode.

Réalisez la classe Humain.
Le constructeur reçoit le nom de l'humain crée en paramètre

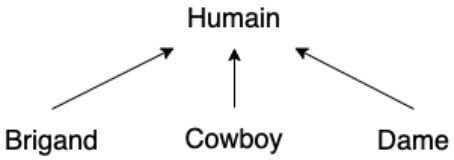
Réalisez la classe Histoire.
Elle contiendra votre fonction principale main.
Dans cette classe Histoire, vous créerez un humain qui se présente et qui boit.

Question 2.2: Brigand, cowboys, dames et détresses

Les dames, les cowboys et les brigands sont tous des humains : ils ont tous un nom et peuvent tous se présenter.

Par contre, il y a certaines différences entre ces 3 classes d'individus :
- Les brigands peuvent kidnapper les dames
- Les dames peuvent se faire enlever et se faire libérer
- Les cowboys peuvent les libérer

On a donc le petit schéma suivant :



Nous pouvons dire que les classes Brigand, Cowboy et Dame sont les classes filles de la classe Humain.

En java, on peut indiquer cela avec le mot clé **extends** :

```
public class Brigand extends Humain {
}
```

Grâce à cette écriture, on dit qu'un brigand est un humain car la classe Brigand hérite de la classe Humain. Par conséquent, il possèdera un nom, une boisson favorite, une méthode présentation, …, sans avoir besoin de les redéfinir.

Il est également possible d'ajouter des variables et des méthodes à la classe Brigand.

Une Dame est un humain caractérisé par :
- La couleur de sa robe (une chaîne de caractère)
- Un état (libre ou captive)

Elle peut :
- Se faire kidnapper (auquel cas elle hurle)
- Se faire libérer par un cowboy (elle remercie alors le héros qui la libère)
- Changer de robe, en s'écriant « `Regardez ma nouvelle robe (couleur) !` »

Un Brigand est un humain qui est caractérisé par :
- Un look (méchant par défaut)
- Un nombre de dames enlevées
- La récompense offerte lorsqu'il est capturé (100 $ par défaut)
- Un état (en prison ou pas)

Il peut :
- Capturer une dame en s'exclamant « `Ah ah ! (nom de la dame), tu es mienne désormais` »
- Se faire emprisonner par un cowboy. Il s'écrit alors « `Damned, je suis fait ! (Nom du cowboy), tu m'as eu !` »

Une méthode pour connaitre la récompense obtenue en cas de capture doit être prévue.

Un Cowboy est un humain qui est caractérisé par :
- Sa popularité (0 pour commencer, augmentera de 1 à chaque dame délivrée)
- Un adjectif pour le caractériser (vaillant par défaut)

Il peut :
- le cowboy s'exclame « `Prend ça, rascal !` »
- Il peut également libérer une dame (en la flattant)
- tirer sur un brigand (un commentaire indique alors « `le (adjectif) (nom) tire sur (nom du méchant). PAN ! »`)

Réalisez les 3 classes Brigand, Cowboy et Dame.
Modifiez votre classe Histoire pour tester ces classes.

Question 2.3: Redéfinition

Au sein d'une classe fille, il est possible d'ajouter des attributs et des méthodes.
Il est également possible de changer les attributs et méthodes de la classe mère.
Pour cela, il suffit simplement de les redéfinir dans la classe fille : On parle alors de redéfinition **(override).**

Lorsque l'on demande le nom à une dame, elle répond `Miss (son nom).`
Lorsque l'on demande le nom à un cowboy, il dira simplement son nom.
Lorsque l'on demande le nom à un brigand, il dira `(son nom) le (son look).`
Exemple : Bob le méchant

Redéfinissez la méthode `quelEstTonNom()` **dans les classes Brigand et Dame.**

Testez dans votre classe Histoire.

Les méthodes que vous avez écrite *remplacent* le contenu de la méthode `quelEstTonNom()` de la classe Humain.

Question 2.4: super

On désire aussi changer la présentation des brigands, dames et cowboys.

Un Brigand parlera aussi de son look, du nombre de dames qu'il a enlevé et de la récompensé offerte pour sa capture.

Une Dame ne pourra pas s'empêcher de parler de la couleur de sa robe.

Un Cowboy dira ce que les autres disent de lui (son adjectif) et parlera de sa popularité.

Si on réécrit la méthode qui permet à une personne de se présenter, la méthode de la classe Humain sera remplacée. Or, nous souhaitons que cette méthode soit appelée.

Par exemple, Le brigand Bob se présentera de la manière suivante :

```
(Bob) – Bonjour, je suis Bob le méchant et j'aime le Tord-Boyaux.
(Ci-dessus, c'est la méthode de la classe Humain)
(Bob) – J'ai l'air méchant et j'ai déjà kidnappé 5 dames !
(Bob) – Ma tête est mise à prix à 100$ !
```

Au sein d'une sous-classe de la classe Humain, nous pouvons redéfinir la méthode `presentation()`, et appeler la méthode de présentation de la classe mère grâce à la syntaxe `super.presentation()`

Aussi, nous allons définir une boisson par défaut à chaque sous-classe :
- lait pour la dame
- tord-boyaux pour le brigand
- whisky pour le cowboy

Pour cela, on crée un constructeur pour chacune de ces sous-classes dans lequel on appelle le constructeur de la classe mère (c'est à dire `super(…)` avant d'attribuer la boisson par défaut.

Question 2.5: Un verre, patron !

Un Barman est un humain dont la boisson favorite est le Vin. Il peut servir n'importe quel humain, en lui donnant un verre de sa boisson favorite.

Il est caractérisé par le nom de son bar. Par défaut, il s'agira du bar `Chez (nom du barman).`

La classe Barman aura 2 constructeurs :
- Soit on crée un barman en indiquant uniquement son nom
- Soit on indique également le nom de son bar

Quand un barman se présente, il n'oublie pas de mentionner le nom de son bar.
Quand un barman parle, il termine toutes ses phrases par `Coco`.

Réalisez la classe Barman.

Testez dans la classe Histoire.

Question 2.6: I'm the law

On ajoute à notre histoire des shérifs.

Un Shérif est un cowboy qui peut coffrer des brigands, en criant « `Au nom de la loi, je vous arrête !` »

Il peut également rechercher un brigand. (Un commentaire indique alors qu'il placarde une affiche dans toute la ville, et il dit, par exemple : « `OYEZ OYEZ BRAVE GENS !! 200 $ a qui arrêtera Bob le brigand, mort ou vif !!` ». Lorsqu'un brigan est recherché, sa mise a prix est doublée.

Tout le monde s'accorde pour dire que les shérifs sont honnêtes. (adjectif peut être **protected**)

Il est caractérisé par le nombre de brigands qu'il a coffré qu'il ne manquera pas de préciser lorsqu'il se présente.

Il refuse de se faire appeler autrement que par `Shérif son nom.`

Réalisez la classe Shérif.
Testez dans la classe Histoire.

Question 2.7: Les shérifs sont des cowboys dans l'âme

Comme un shérif est un cowboy, on peut créer un cowboy en faisant :
```
Cowboy Clint = new Sherif("Clint")
```

Testez.

Quelles sont les fonctions qui sont appelées (quand le cowboy se présente par exemple).
Peut-on demander à ce cowboy de coffrer un brigand ?

Exercice 3 : Interfaces

Pour développer l'histoire, nous allons faire intervenir des Ripoux.

Un Ripoux est un Shérif qui est Brigand.

Comment feriez-vous ?

On aimerait faire en sorte qu'un ripoux hérite de 2 classes (Shérif et Brigand).
L'héritage multiple n'est pas possible en Java. En revanche, Il est possible d'utiliser des interfaces.

Une interface défini un comportement qui doit être implémenté par une classe. Elle liste les méthodes que doit posséder une classe qui implémente cette interface.

Ainsi, nous allons créer une interface **HorsLaLoi.**
Cette interface indiquera qu'une classe qui désigne un hors la loi doit être munie :

- D'une méthode pour kidnapper les dames
- D'une autre pour se faire emprisonner par un cowboy
- D'une autre pour donner son nom
- D'une autre pour donner la hauteur de sa récompense.

Par exemple :

```
public interface HorsLaLoi {

    public void estEmprisonne(Cowboy c);

    public void kidnappe(Dame dame);

    public int getMiseAPrix();

    public String quelEstTonNom();
}
```

Les signatures des méthodes dans l'interface seront éventuellement à remplacer par les noms des méthodes que vous avez utilisées). Une fois que la classe implémentera l'interface, celle-ci devra obligatoirement comporter ces 4 méthodes, avec un contenu.

Pour dire qu'une classe suit une interface, on utilise le mot clé **implements.**

```
public class Brigand implements HorsLaLoi{

    @Override
    public void estEmprisonne(Cowboy c) {
    }

    @Override
    public void kidnappe(Dame dame) {
    }

    @Override
    public int getMiseAPrix() {
    }

    @Override
    public String quelEstTonNom() {
    }
}
```

Dans la classe Brigand, il faut être sûr que toutes les méthodes décrite dans l'interface HorsLaLoi existent.

Question 3.1: Brigand

Écrivez l'interface HorsLaLoi et dites qu'un brigand est un hors-la-loi.

Question 3.2: Les cowboys chassent les hors-la-loi

Les classes qui implémentent cette interface HorsLaLoi peuvent être manipulées comme des objets de type HorsLaLoi. Ainsi, dans nos fonctions, nous pouvons utiliser l'interface HorsLaLoi.

Ainsi, nous pouvons dire qu'un Cowboy ne coffre pas seulement les Brigands, mais tous les HorsLaLoi.
De même, une Dame peut se faire kidnapper par n'importe quel HorsLaLoi.

Modifiez les fonctions concernées pour tenir compte de ceci.
(Vous devriez normalement uniquement remplacer Brigand par Hors la loi).

Testez dans votre classe Histoire.

Question 3.3: Tous pourris

Maintenant que vous avez une interface HorsLaLoi, vous pouvez ajouter des classes qui définissent les comportements de celle-ci.

Créez maintenant la classe Ripoux en utilisant l'interface HorsLaLoi.

Créez un ripou dans votre classe Histoire et testez.

Question 3.4: Calamity Jane

Nous pouvons ajouter une dame brigand à notre histoire.

Elle est une dame qui a décidé de passer du côté des HorsLaLoi.

Créez la classe DameBrigand et testez.

Question 3.5: Ugh !

Pour finaliser notre histoire, nous allons créer un indien.

Un Indien est un humain qui est caractérisé par :
- Son nombre de plumes (qu'il mentionne quand il se présente)
- Son totem (Coyote par d´défaut)

Sa boisson favorite est le jus de racine.

Il termine toutes ses phrases par `Ugh !`
Il peut scalper un visage pale (il peut alors s'ajouter une plume)

Par conséquent, **un VisagePale** est un humain qu'on pourra scalper. Un VisagePale disposera donc d'une méthode `scalp()` (il s'écrira alors, par exemple, `Aïe ma tête !`

Réalisez la classe Indien et l'interface VisagePale. Les brigands, les cowboys et les dames sont des VisagesPales.

Testez votre histoire finale.