Programmation Objet en Java - Feuille de TD 2

Exercice 1. À votre avis, qu'affiche le programme suivant (penser à dessiner les objets en mémoire pour répondre) ? Attention ce n'est pas du Python...

```
public class TestString {
    public static void main(String[] args) {
        int n = 2;
        int m = 2;

        System.out.println(n == m);

        String s1 = "toto";
        String s2 = new String(s1);
        String s3 = s1;

        System.out.println(s1 == s1);
        System.out.println(s1 == s2);
        System.out.println(s3 == s1);

        System.out.println(s2.equals(s1));
        System.out.println(s3.equals(s1));

        String s4 = "toto";
        System.out.println(s1 == s4);
    }
}
```

Exercice 2. Poissons

- Écrire un record `Fish` pour représenter des poissons. Un poisson a une espèce représenté par une chaîne de caractères `species`, un booléen `seaFish` qui indique si c'est un poisson de mer ou pas et une longueur `length` en cm.
- Modifier le constructeur canonique pour qu'il lève une exception si l'espèce est `null` ou si la longueur n'est pas comprise en entre 0 et 300 (vous pouvez utiliser `Objects.checkIndex` pour ça).
- Ajouter un constructeur qui permet de créer un poisson de mer à partir de son espèce et sa longueur.
- Ajouter une méthode `ifIsSeaFish` qui renvoie le poisson auquel on l'applique si c'est un poisson de mer, et `null` sinon.

Exercice 3. Listes de poissons

- Écrire une classe `FishTest` qui contient une méthode `main` qui crée 3 poissons.
- Créer et afficher un liste non modifiable `list1` contenant ces 3 poissons.
- Créer et afficher un liste modifiable `list2` contenant ces 3 poissons.
- Est-il possible d'ajouter un `Fish` dans `list1`. Pourquoi ?
- Est-il possible d'ajouter un `Bird` dans `list2`. Pourquoi ?
- Ajouter une méthode `allSeaFishes` qui prend une `ArrayList` de poissons en paramètre et renvoie `true` si tous les poissons qu'elle contient sont des poissons de mer, et `false` sinon. Quelle est la particularité de cette méthode ?

Remarque : vous pouvez consulter la documentation des `ArrayList` ici :

<https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java.util/ArrayList.html>.

- g) Écrire une méthode `allSeaFishes2` qui fait la même chose que la précédente mais qui utilise la boucle *for-each-in* aperçue dans le cours (c'est la même chose qu'en Python, sans oublier le typage).
- h) Ajouter une méthode `removeSeaFishes` qui prend une liste de poissons en paramètre et supprime tous les poissons de mer qu'elle contient.
- i) Ajouter une méthode `withoutSeaFishes` prend une liste de poissons en paramètre et renvoie une liste qui contient les même poissons, mais sans les poissons de mer. À votre avis, quelle méthode est mieux : celle-ci ou la précédente ?
- j) Ajouter une méthode `sameFishes` prend deux listes de poissons en paramètre et renvoie `true` si les liste contiennent exactement les mêmes poissons, et `false` sinon.
- k) À quel(s) endroit(s) aurait-on pu utiliser des listes non modifiables dans le code que vous venez d'écrire ?