# Programmation Objet en Java - Feuille de TP 6

**Toutes les classes de ce TP seront dans un package nommé `fr.dut.zoo`.**

**Exercice 1.**
On souhaite modéliser un zoo composé d'animaux herbivore ou carnivores.

a) Créer une classe `Animal` qui représente un animal avec un champs booléen `carnivore` qui indique si l'animal est carnivore ou non et un champs `weight` de type `int` qui correspond au poids de l'animal en kg. Écrire le constructeur permettant d'initialiser les deux champs.

b) Écrire la méthode permettant d'afficher un `Animal`.

c) Écrire une méthode `main` dans une classe `AnimalTest`. Créer un `Animal` carnivore et un autre herbivore.

d) Écrire une classe `Zoo` contenant des animaux. On utilisera un champs `zoo` de type `ArrayList<Animal>` pour stocker les animaux.

e) Écrire une méthode `add` qui permet d'ajouter un `Animal` dans le `Zoo`. Attention à ce que vous autorisez à ajouter dans le `Zoo`.

f) Écrire la méthode permettant d'afficher tout le `Zoo`.

g) Sachant qu'un carnivore à besoin du quart de son poids en viande par semaine, écrire une méthode `getMeatForAWeek` dans la classe `Animal` qui renvoie la quantité de viande nécessaire pour le nourrir. Pour un herbivore, cette quantité sera zéro.

h) Écrire une méthode `getMeatForAWeek` dans Zoo qui renvoie la quantité de viande nécessaire pour nourrir les animaux du Zoo pour une semaine.

**Exercice 2.**
Pour attirer des visiteurs, un `Zoo` peut recruter des animaux stars qui, contrairement aux animaux normaux, ont un nom de type `String`.

a) Écrire une classe `StarAnimal` qui dérive de `Animal` avec un constructeur et de quoi afficher un `StarAnimal` avec son nom suivi de ses caractéristiques.

b) Les carnivores qui sont des `StarAnimal` ont un régime alimentaire nécessitant la moitié de leur poids en viande. Que doit-on ajouter ou modifier et dans quelles classes pour que la méthode `getMeatForAWeek` de `Zoo` fonctionne correctement ?

c) Rajouter ce qu'il faut dans le code pour pouvoir tester si deux `StarAnimal` sont égaux.

d) Quelle(s) autres(s) méthode(s) doit-on obligatoirement ajouter ? Compléter le code en conséquence.

e) Créer un petit `Zoo` avec au moins un `StarAnimal` et tester les différentes méthodes que vous avez écrites.

f) Toujours pour attirer les visiteurs, le zoo propose un petit spectacle avec tous les animaux stars mais aussi les herbivores de plus de 20 kg. Écrire une méthode `getAnimalsForTheShow` qui renvoie une liste des animaux participant au spectacle. Quel doit-être son type de retour? Attention, pour cette question, vous ne devez pas utiliser `instanceof`, il y a une solution plus simple et plus propre.

**Exercice 3.**

Les zoos peuvent désormais avoir des oiseaux (classe `Bird`) qui sont des animaux qui peuvent être herbivores ou carnivores, et dont on connaît la taille (entière en cm) mais pas le poids. Un oiseau carnivore consomme un centième de sa taille en viande par semaine.
Note: pour l'instant, les oiseaux ne peuvent pas être des animaux star et ne font pas partie du spectacle.

a) Écrire la classe `Bird` correspondante et rajouter ce qu'il faut pour que l'on puisse ajouter des oiseaux dans un `Zoo` au même titre que les autres animaux.

b) Modifier la classe `Zoo` pour que l'on puisse y ajouter des oiseaux. Vous ne devez avoir qu'une seule méthode d'ajout.

c) Vérifier que toutes les méthodes de `Zoo` fonctionnent, sachant qu'un `Zoo` peut désormais contenir des oiseaux.

d) Faire en sorte de ne pas avoir de code dupliqué entre les classes des différentes espèces.

**Exercice 4.**

Dans ce exercice, on cherche à étendre et à rendre plus efficace le travail d'un `Zoo`.

a) Les méthodes `getMeatForAWeek` et `getAnimalsForTheShow` ont une complexité linéaire en le nombre d'animaux du `Zoo`. Peut-on améliorer cela, en mémorisant un peu plus d'information ? Si oui, faire les modifications nécessaires dans le code. Attention à ce que de votre code ne provoque pas d'erreurs involontaires.

b) On veut désormais pouvoir retirer un animal d'un `Zoo`. Pour cela, ajouter une méthode `remove`. Si l'animal n'existe pas dans le `Zoo`, la méthode renvoie `false`, sinon `true`.

c) Est-ce que les amélioration éventuelles de la première question sont toujours valides ? Sinon, corriger le code.

d) Il arrive qu'un `Zoo` accepte de "prêter" l'un de ses animaux, à condition qu'il en possède au moins deux. Écrire une méthode `lendables` qui renvoie le nombre d'animaux que l'on peut prêter, c'est à dire que si on les retire du zoo, il en reste toujours au moins un exemplaire au zoo. Par exemple, si un zoo possède 5 girafes identiques, il peut en prêter 4.

e) Un `Zoo` peu collaborer avec un autre `Zoo` pour que leur animaux se reproduisent ; pour que cela fonctionne, il faut trouver des animaux identiques dans chaque `Zoo`. Écrire une méthode `couples` qui renvoient la liste des animaux présents dans deux `Zoo`s à la fois.

f) Quelle est la complexité des deux dernières méthodes que vous avez écrites ? Peut-on faire mieux ?