Série 4

Exercice 1

Le but de ce TP est de représenter un zoo. Celui-ci est modélisé par une instance de la classe Zoo dans laquelle les animaux sont stockés dans une **table**. Chaque animal est modélisé par une instance de la classe Animal qui est un objet composite. Un programme TestZoo permettra de construire un zoo dans lequel l'utilisateur ajoutera des animaux.

Question 1 : Définir la classe Animal
La classe Animal est composée de deux chaînes de caractères représentant le nom et la race de l'animal ainsi que de deux entiers correspondant à son âge et son poids.
***Constructeurs* :** Cette classe possède un constructeur qui prend en paramètres le nom, la race, l'âge et le poids.
Méthodes :

- **Modificateurs / Accesseurs** Pour permettre d'accéder et de modifier tous les attributs.
- **__str__** : Renvoie une chaîne de caractères décrivant l'ensemble des informations concernant l'animal.
- **Déplacer** : présente la manière de déplacement d'un animal.

Question 2 :
Créez des classes pour les différentes races : chien, oiseau, poisson
Ajouter les éléments nécessaires pour chaque classe (attributs, constructeurs et méthodes)

Question 3 : Définir la classe Zoo
Un zoo est composé d'un ensemble d'animaux que l'on mettra dans une liste d'animaux.
Méthodes :

- **ajouterAnimal** : Cette méthode permet d'ajouter un animal dans le zoo.
- **moyAge** : Cette méthode renvoie la moyenne d'âge de tous les animaux.
- **moyPoids** : Cette méthode renvoie la moyenne des poids de tous les animaux.
- **afficheTypeAnimal** : Cette méthode prend en paramètre une chaîne représentant une race et affiche tous les animaux de cette race, puis la moyenne de leur âge et de leur poids.
- **__str__** : Cette méthode affiche les caractéristiques de tous les animaux du zoo.

Question 4 : Définir TestZoo
Dans le programme principal, l'utilisateur saisit un zoo qui est une instance de la classe Zoo. Les animaux sont saisis par l'utilisateur et ajoutés au zoo. Le programme affiche ensuite la liste de tous les animaux.

Exercice 2

Ecrivez une interface ***Forme*** avec les méthodes abstraites suivantes:

- **perimetre(): renvoie le périmètre de la forme,**
- **aire(): renvoie l'aire de la forme.**

Ecrivez une classe ***Carre*** implémentant l'interface ***Forme*** avec les attributs suivants:

- **cote: Le côté du carré,**

La classe ***Carre*** doit disposer des constructeurs suivants

- **Constructeur par défaut**
- **Constructeur avec arguments**

La classe ***Carre*** doit contenir des accesseurs et mutateurs pour les différents attributs et les méthodes suivantes:

- **perimetre(): Donne le périmètre du carré,**
- **aire(): Donne l'aire du carré,**
- **toString(): Donne une représentation du Carré.**

Ecrivez aussi une classe de ***testForme*** afin de tester les classes avec les différentes méthodes.