TP Création d'une interface

On souhaite créer une application permettant de gérer un jeu de combat entre héros.

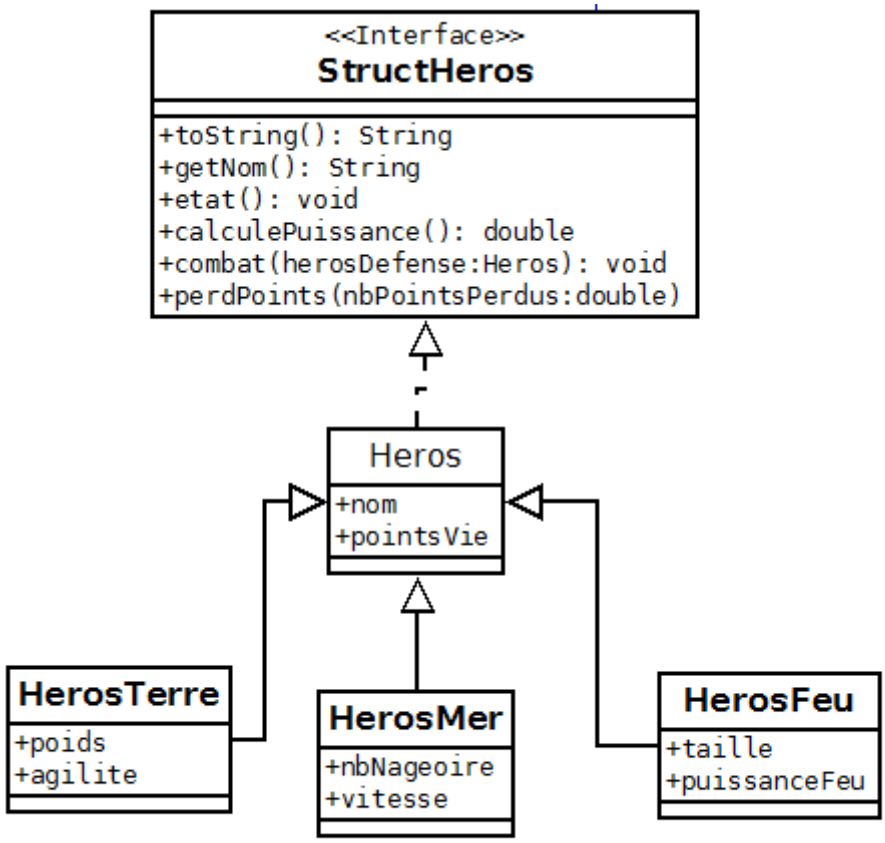
Il existe 3 catégories de héros correspondant à 3 classes :

- **Les héros de terre** : Classe HerosTerre. Ces héros sont caractérisés par un nom, un poids (en kg) et un degré d'agilité. Ces héros ont une puissance que l'on peut calculer grâce à la formule suivant : *puissance =poids/20 * degré d'agilité*

- **Les héros de feu** : Classe HerosFeu. Ces héros sont caractérisés par un nom, une taille (en centimètres) et une puissance de feu. Ces héros ont une puissance que l'on peut calculer grâce à la formule suivante : *puissance = taille/100 * puissance de feu*

- **Les héros des mers** : Classe HerosMer. Ces héros sont caractérisés par un nom, un nombre de nageoires et une vitesse en km/h. Ces héros ont une puissance que l'on peut calculer grâce à la formule suivante *: puissance = nombre de nageoires * vitesse*

Afin d'imposer aux trois types de Heros le même fonctionnement, vous allez créer une architecture d'application avec une Interface StructHeros et une classe abstraite Heros sous le modèle suivant :



Interface StructHeros

Les méthodes de l'interface StructHeros imposent aux catégories de héros de définir :

- Une méthode toString qui retourne les caractéristiques du héros sous forme de chaine de caractère
- Un accesseur getNom qui retourne le nom du héros
- Une méthode etat qui affiche le nom et l'état du héros (vivant avec nombre de points de vie restant ou mort)
- Une méthode perdPoints qui retire au héros le nombre de points de vie passé en paramètres.
- Une méthode calculePuissance qui retourne la puissance du héros
- Une méthode combat qui gère le combat entre deux héros

Classe abstraite Heros

La classe abstraite Heros implémente l'interface StructHeros et contient les attributs communs à tous les héros.
Elle possède un constructeur pour initialiser les attributs communs.
Elle définit le code des méthodes getNom, perdPoints et etat.
Elle redéfinit les autres méthodes de l'interface comme méthodes abstraites afin d'imposer aux sous-classes la rédaction du code de ces méthodes.

Classe HerosTerre

Elle possède les attributs poids et agilité et un constructeur pour initialiser tous les attributs de la classe. Elle doit obligatoirement créer les méthodes calculePuissance, toString et combat.

La méthode toString permettra d'afficher les informations du héros sous le format suivant :

```
Héros de terre Terramon vie : 150.0
pèse 100.0 kg, agilité 25.0
```

La méthode combat fonctionne de la façon suivante :

- Si la puissance du héros qui attaque est supérieure à la puissance du héros en défense, l'attaquant fait perdre 40 points au défenseur.
- Si les deux puissances sont égales, aucun ne perd de points
- Si la puissance du défenseur est supérieure à celle de l'attaquant, celui-ci perd 15 points.

A la fin de la méthode combat, on appelle la méthode état sur les deux héros.

Classe HerosFeu

Elle possède les attributs taille et puissance de feu et un constructeur pour initialiser tous les attributs de la classe. Elle doit obligatoirement créer les méthodes calculePuissance, toString et combat.

La méthode toString permettra d'afficher les informations du héros sous le format suivant :

```
Héros de feu Firomon vie : 140.0
mesure 213.0 cm, puissance de Feu 7.0
```

La méthode combat fonctionne de la façon suivante :

- Il doit rester au héros de feu au moins 20 points de vie pour pouvoir porter une attaque, sinon l'attaque n'aura pas lieu.
- Si la puissance du héros qui attaque est supérieure à la puissance du héros en défense, l'attaquant fait perdre 60 points au défenseur.
- Si les deux puissances sont égales, il fait perdre 20 points au défenseur.
- Si la puissance du défenseur est supérieure à celle de l'attaquant, aucun ne perd de points.

A la fin de la méthode combat, on appelle la méthode état sur les deux héros.

Classe HerosMer

Elle possède les attributs nombre de nageoire et vitesse et un constructeur pour initialiser tous les attributs de la classe. Elle doit obligatoirement créer les méthodes calculePuissance, toString et combat.

La méthode toString permettra d'afficher les informations du héros sous le format suivant :

```
Héros de mer Hydromon vie : 170.0
possède 6.0 nageoires, vitesse 30.0 km/h
```

La méthode combat fonctionne de la façon suivante :

- Si la puissance du héros qui attaque est supérieure à la puissance du héros en défense, l'attaquant fait perdre 50 points au défenseur.
- Si les deux puissances sont égales et qu'il reste au moins 50 points de vie à l'attaquant, il fait perdre 25 points au défenseur.
- Si la puissance du défenseur est supérieure à celle de l'attaquant et qu'il reste au moins 50 points de vie à l'attaquant, le défenseur perd 15 points. Si le nombre de points de vie de l'attaquant n'est pas suffisant, celui-ci perd 10 points.

A la fin de la méthode combat, on appelle la méthode état sur les deux héros.