**Examen de synthèse (60%)**

**Durée : 3 h**

(4 heures sont accordées)

**Instructions**

* Respecter les normes de programmation et documenter votre code.
* Écrire votre **nom et prénom** sur la feuille de code principale (main.py).
* **Remettre sur LÉA et partager votre dépôt sur GitHub avec l’enseignante : Hasna-Hocini**

## **Critères d’évaluation et pondération de l’évaluation certificative finale**

|  |  |
| --- | --- |
| Critères d’évaluation | Pondération |
| * 1. Production d’un diagramme de classes conforme à l’énoncé (00Q6.1, 00Q6.2) | 20 % |
| * 1. Choix et positionnement appropriés des éléments graphiques pour l’affichage et la saisie (00Q6.4) | 15 % |
| * 1. Programmation adéquate de la logique applicative selon les principes de l’orienté-objet, validation des données entrées et gestion appropriée des erreurs (00Q6.3, 00Q6.5) | 35 % |
| * 1. Production de tests adéquats (00Q6.5) | 20 % |
| * 1. Production d’une documentation de qualité respectueuse des normes du cours et respect des règles d’orthographe et de grammaire (00Q6.6) | 10 % |

**Voir grille critériée jointe** pour plus de détails.

# **Exercice 1 : Diagramme de classe UML (20 pts)**

Un zoo compte plusieurs animaux. Les animaux occupent des enclos et sont soignés par des vétérinaires.

Chaque animal est caractérisé par son numéro d’animal, son surnom, son poids et la famille d’animaux à laquelle il appartient. Les numéros des animaux doivent commencer par une lettre suivie d’un tiret puis de quatre chiffres, exemple : A-4152. Le surnom de l’animal ne comporte aucune validation alors que son poids doit être un nombre réel positif. Finalement, la famille de l’animal peut être l’une des trois familles suivantes : Mammifères, oiseaux ou reptiles.

Un animal est donc soit un mammifère, soit un oiseau soit un reptile.

Un mammifère est un animal. Il est donc caractérisé par les informations qui caractérisent l’animal et celle qui caractérise le mammifère : la couleur de son poil. Cette couleur peut être noire, blanche, brune, grise, beige ou multi couleurs.

Un oiseau est un animal. Il est par conséquent caractérisé par les informations qui caractérisent l’animal et aussi l’information qui caractérise l’oiseau : la longueur de son bec en centimètres qui est une valeur réel positive.

Un reptile est un animal. Il est donc caractérisé par les informations qui caractérisent l’animal et aussi par une information booléenne qui indique si le reptile est venimeux ou pas.

Un enclos (maison de l’animal) est caractérisé par son numéro, sa taille (petit, moyen et grand), et sa localisation dans le zoo. Le numéro de l’enclos est un nombre entier sur 5 caractères. La localisation de l’enclos dans le zoo est dans l’une des trois sections du zoo : section A, B ou C.

Un enclos sert de maison pour les animaux. Chaque animal est affecté à un seul enclos et dans chaque enclos peuvent cohabiter un ou plusieurs animaux.

Dans ce zoo travaillent plusieurs vétérinaires. Un vétérinaire est caractérisé par son numéro d’employé (3 premiers caractères du nom suivis par 2 chiffres), son nom (longueur 50 caractères maximum), son prénom (longueur 50 caractères maximum) et sa date de naissance.

Nous pouvons calculer l’âge du vétérinaire avec une méthode privée et avec cette information nous pouvons déterminer s’il doit prendre sa retraite ou pas en utilisant une méthode publique. L’âge de retraite est 60 ans.

Un vétérinaire s’occupe d’un ou plusieurs animaux. Chaque animal est soigné par un ou plusieurs vétérinaires.

## Questions (20 points) :

Élaborer le diagramme de classe UML correspondant à cet énoncé. Pour ce faire il faudra :

1. Déterminer les classes candidates du diagramme de classe.
2. Ajouter les attributs et les méthodes aux classes candidates. Ne pas représenter les getter, setter et propriétés.
3. Représenter correctement l’héritage et les associations décrits dans l’énoncé. Ne pas oublier de représenter les attributs issus des différentes associations.

# **Exercice 2 (20 pts)**

Utiliser le code du projet **Zoo** fourni, le décompresser avant de le compléter selon l’énoncé suivant :

# **Classe Animal**

Dans le dossier **Classes** ajouter à la classe **Animal** les membres suivants :

* **Numero\_animal** : doit commencer par deux lettres suivies d’un tiret puis de cinq chiffres, exemple : LI-41524.
* **surnom** de l’animal : ne comporte aucune validation
* **poids** de l’animal : doit être un nombre entier supérieur à 15 lb.
* **famille** de l’animal à laquelle il appartient : soit mammifères, soit oiseaux, soit reptiles.
* le constructeur avec paramètres et valeurs par défaut.
* l’attribut **nb\_animaux** qui permet de compter le nombre d’animaux instanciés.
* l’attribut **ls\_animaux** qui contient la liste de tous les objets de type **Animal** instanciés.
* les méthodes d’accès (getters et setters) et propriétés pour les attributs qui le nécessitent.

Ajouter deux attributs sachant que :

* Chaque animal habite dans un enclos.
* Chaque animal est soigné par une liste de vétérinaires.
* Ajouter la méthode **ajouterEnclosVeterinaire()** qui permet d’ajouter (assigner) un enclos à la liste des enclos d’un des vétérinaires qui s’occupent de l’animal (voir la classe Veterinaire). Cette méthode prend en paramètre un **numero\_emp** et un objet de type **Enclos.**
* Ajouter la méthode magique \_\_str\_\_ ().

**Classe Enclos**

Dans le dossier **Classes** ajouter à la classe **Enclos** les membres suivants**:**

* **numero\_enclos** qui doit être composé d’un nombre entier sur 5 caractères suivi de trois lettres. Exemple : 12345ABC
* **nom\_enclos** qui est composé d’un maximum de 25 lettres.
* **taille** de l’enclos qui doit avoir l’une des trois valeurs : petit, moyen ou grand.
* **type** de l’enclos qui est soit intérieur soit extérieur.
* **localisation** de l’enclos dans le zoo : section A, B ou C.
* un attribut qui permet d’obtenir la liste des animaux qui partagent un enclos.
* une liste qui permet de sauvegarder tous les enclos instanciés.
* Un attribut qui permet d’obtenir le nombre d’objets **Enclos** instanciés.
* le constructeur avec paramètres et valeurs par défaut.
* les méthodes d’accès et propriétés pour les attributs avec validation.
* La méthode **estAdapte()** qui permet de vérifier si l’enclos a une taille adaptée au nombre d’animaux qui y vivent. Un enclos de petite taille est conçu pour accueillir jusqu’à 2 animaux. Un enclos de taille moyenne jusqu’à 4 animaux et un enclos de grande taille jusqu’à 6 animaux. La méthode estAdapte() retourne **True** si la taille de l’enclos est adaptée et **False** sinon.

**Classe Mammifere**

* Un mammifère est un animal. Ajouter au projet la classe **Mammifere**.
* Ajouter à cette classe l’attribut d’instance **couleur de poil** qui peut avoir les valeurs noire, blanche, brune, grise, beige ou multi couleurs.
* Ajouter les méthodes d’accès et propriétés si nécessaire.
* Ajouter le constructeur avec paramètres et valeurs par défaut.
* Ajouter la méthode magique \_\_str\_\_().

**Classe Oiseau**

* Un oiseau est un animal. Ajouter au projet la classe **Oiseau.**
* Ajouter à cette classe l’attribut d’instance **longueur de bec** qui doit être une valeur réelle positive.
* Ajouter les méthodes d’accès et propriétés si nécessaire.
* Ajouter le constructeur avec paramètres et valeurs par défaut.

**Classe Reptile**

* Un reptile est un animal. Ajouter au projet la classe **Reptile.**
* Ajouter à cette classe l’attribut d’instance **Venimeux** qui doit être une valeur booléenne égale True ou False.
* Ajouter les méthodes d’accès et propriétés si nécessaire.
* Ajouter le constructeur avec paramètres et valeurs par défaut.

**Classe Veterinaire**

Voir le code de la classe Veterinaire dans le projet fourni **Zoo** et le compléter en ajoutant :

* Un attribut qui contient la liste des Enclos assignés au vétérinaire.
* Ajouter la méthode privée nommée **calculerAge**() qui permet de calculer l’âge du vétérinaire.
* Ajouter la méthode publique **prendreRetraite** () qui permet de retourner l’une des deux valeurs **True** ou **False** selon que l’employé a atteint l’âge de 60 ans ou pas.
* Ajouter la méthode **ajouterEnclos ()** qui prend en paramètre un objet de type **Enclos** et l’ajoute à la liste des enclos assignés au vétérinaire.

# **Exercice 3 (30 pts)**

**Créer une boîte de dialogue (15pts) :**

Utiliser Qt-Designer pour créer **une** boîte de dialogue qui permet d’ajouter, de modifier et de supprimer un **enclos**. Programmer le gestionnaire d’événement du bouton « **Enclos** » de la fenêtre principale pour permettre l’ouverture de cette boîte de dialogue.

**Validation des données, sérialisation, gestion d’exception (15 pts)**

1. Dans la classe **Animal**, ajouter la méthode serialiserAnimal (). Cette méthode permet de sérialiser un objet de type Animal vers un fichier de type json qui porte le nom « numero\_animal.json ». Exemple : **LI-41524.json**
2. Quand vous cliquez sur le bouton « **Animal** », la boîte de dialogue animal s’ouvre. Lors de l’ouverture de cette boîte de dialogue, remplir la comboBox « **Enclos de l’animal** » avec les numéros d’enclos tirés de la **liste des enclos (liste de classe)**. Pour remplir cette liste de classe, instancier quelques enclos dans le code (**Important** : Ne pas préparer cette liste en programmant l’ajout d’un enclos dans la boîte de dialogue « **Enclos** ».)
3. Dans la boîte de dialogue animal, associer un code python au bouton « **Ajouter** ». Ce code permet de **valider les entrées de l’utilisateur** selon les principes de l’orientée objet et d’afficher éventuellement les messages d’erreurs dans les labels d’erreur correspondants. L’option choisie par l’utilisateur dans la comboBox **Famille de l’animal** (Mammifère, oiseau ou reptile) permet d’afficher les contrôles associés au choix. Donc ça affiche soit la couleur du poil si la famille de l’animal est **Mammifère,** soit les contrôles qui correspondent à la longueur du bec si la famille de l’animal est **Oiseau,** soit les contrôles qui correspondent à Venimeux si la famille de l’animal est **Reptile**. Si toutes les entrées sont valides, l’objet est créé. L’animal créé est donc un objet de l’une des classes **Mammifere**, **Oiseau** ou **Reptile**. Il est important de s’assurer que l’association entre l’animal instancié et son enclos est créée.L’objet créé est ajouté à la liste des animaux et est également sérialisé dans un fichier json.

**Attention** : Ajouter la gestion d’exception. Le code doit être modulaire.

# **Exercice 4 (20 pts)**

**Production de tests adéquats**

* Créer un test unitaire pour tester la méthode **estAdapte()** de la classe Enclos.
* Créer un test unitaire pour tester la méthode **prendreRetraite()** de la classe **Veterinaire**.

Utiliser des jeux d’essai et des assertions qui permettent une couverture complète du code.

**Production d’une documentation de qualité (10%)**

Ne pas oublier de documenter votre code. Faire attention au respect des règles de grammaire et d’orthographe. Respecter les normes UML.

Bonne réussite 😊!