Travaux pratiques

Exercice 1

Les nombres complexes sont mis en œuvre grâce à une classe *Complexe*. Un nombre complexe est caractérisé par deux nombres réels : sa partie réelle (*pReel*) et sa partie imaginaire (*pImag*). Les **attributs** retenus sont donc deux double pReel et pImag qui sont privés. Un autre choix de mémorisation (module et argument par exemple) n'affectera pas les utilisateurs de la classe puisque ses attributs sont privés. Les méthodes sont toutes publiques, et constituent le jeu de fonctions disponibles pour traiter des nombres complexes :

- double **partieRC** () : fournit la partie réelle du nombre complexe.
- double **partieIC** () : fournit la partie imaginaire du nombre complexe.
- double module () : fournit le module du nombre complexe.
- double argument () : fournit l'argument du nombre complexe.
- Complexe oppose () : fournit l'opposé du nombre complexe.
- Complexe conjugue () : fournit le conjugué du nombre complexe.
- Complexe plus (Complex z) : fournit l'addition du nombre complexe et de z.
- Complexe moins (Complex z) : fournit la soustraction du nombre complexe et de z.
- Complexe multiplier (Complex z) : fournit la multiplication du nombre complexe et de z.

String toString () : fournit la chaîne de caractères d'édition du nombre complexe sous la forme (pReel + pImag i).

Exercice 2

Un parc auto se compose des voitures et des camions qui ont des caractéristiques communes regroupées dans la classe Véhicule.

- Chaque véhicule est caractérisé par son matricule, l'année de son modèle, son prix.
- Lors de la création d'un véhicule, son matricule est incrémenté selon le nombre de véhicules créés.
- Tous les attributs de la classe véhicule sont supposés privés. Ce qui oblige la création des accesseurs (getters) et des mutateurs (setter) ou les propriétés.
- La classe Véhicule possède également deux méthodes abstraites démarrer() et accélérer() qui seront définies dans les classes dérivées et qui afficheront des messages personnalisés.
- La méthode toString() de la classe Véhicule retourne une chaîne de caractères qui contient les valeurs du matricule, de l'année du modèle et du prix.

- Les classes Voiture et Camion étendent la classe Véhicule en définissant concrètement les méthodes accélérer() et démarrer() en affichant des messages personnalisés.

Travail à faire :

- Créer la classe abstraite Véhicule.
- Créer les classes Camion et Voiture.
- Créer une classe Test qui permet de tester la classe Voiture et la classe Camion.

Exercice 3

Créer une classe « Animal », qui possède comme attribut son milieu de vie. Créer ensuite deux autres classes : « Animaux_domestique » et « Animaux_sauvage » qui héritent de la classe « Animal ».

- Un animal domestique possède un nombre de pattes et un maître qui a un nom. Créer une classe maître pouvant représentant le maître des animaux domestiques.
- Définir dans la classe « Animaux_domestique » un constructeur qui appelle le constructeur de sa classe mère « Animal » et une méthode permettant de représenter les animaux domestiques (créer une méthode nommée ***affichage*** permettant d'afficher les informations d'un animal) Si on prend l'exemple d'un animal domestique qui vis en milieu terrestre et dont le maître est jean Paul, on aura l'affichage suivant : « ***Je suis un animal domestique. Je vis en milieu terrestre. Mon maître s'appelle Jean Paul*** »
- Créer deux classes « Chien » et « Chat » caractérisées par la couleur de leur peau et qui héritent de la classe « Animaux_domestique ». Ajouter à ces deux classes un constructeur qui doit appeler le constructeur de leur classe mère « Animaux_domestique ». Redéfinir ensuite la méthode ***affichage*** de la classe Animaux_domestique de manière à afficher par exemple le message suivant : « ***Moi c'est le chat. Je suis un animal domestique. Je vis en milieu terrestre. Mon maître s'appelle Jean Paul. Je suis de couleur grise*** »
- Ajouter à la classe « Animal », deux méthodes : crier() et se_deplacer(). Ces deux méthodes n'ont pas de corps. Penser à rendre la classe animal abstraite.
- Quelle remarque faites-vous ?
- Donner une redéfinition des méthodes crier() et se_deplacer() dans les classes filles « Chien » et « Chat ».
- Créer deux autres classes « Lion » et « Singe » qui héritent cette fois-ci de la classe « Animaux_sauvage » et qui elles aussi redéfinissent les méthodes abstraites de la classe animal.
- Enfin créer une liste d'animaux domestiques et d'animaux sauvages. Afficher leur façon de crier, de se déplacer et de se nourrir.

Exercice 4

Soit la classe abstraite Employé caractérisée par les attributs suivants :

- Matricule
- Nom
- Prénom
- Date de naissance

La classe Employé doit disposer des méthodes suivantes :

- un constructeur d'initialisation
- des propriétés pour les différents attributs
- la méthode Tostring
- une méthode abstraite getSalaire.

Un ouvrier est **un employé** qui se caractérise par sa date d'entrée à la société.

- Tous les ouvriers ont une valeur commune appelée SMIG=2500 DH
- L'ouvrier a un salaire mensuel qui est : Salaire = SMIG + (Ancienneté en année) *100.
- De plus, le salaire ne doit pas dépasser SMIG*2.

Un cadre est **un employé** qui se caractérise par un indice.

- Le cadre a un salaire qui dépend de son indice :
  - 1 : salaire mensuel 13000 DH
  - 2 : salaire mensuel 15000 DH
  - 3 : salaire mensuel 17000 DH
  - 4 : salaire mensuel 20000 DH

Un patron est **un employé** qui se caractérise par un chiffre d'affaire et un pourcentage.

- Le chiffre d'affaire est commun entre les patrons.
- Le patron a un salaire annuel qui est égal à x% du chiffre d'affaire : Salaire = CA*pourcentage/100

Travail à faire :

1. Créer la classe abstraite Employé.

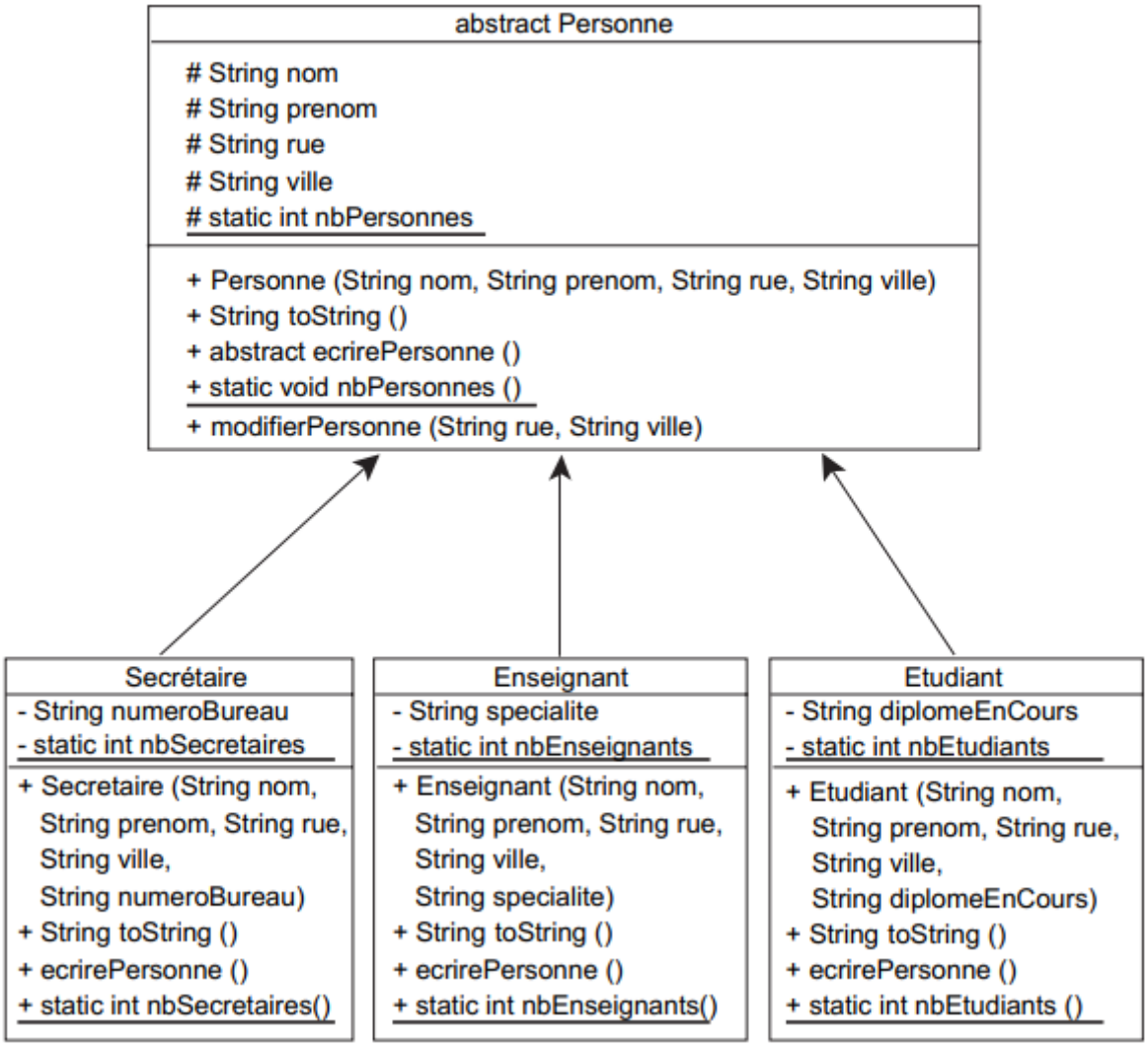
2. Créer la classe Ouvrier, la classe Cadre et la classe Patron qui héritent de la classe Employé, et prévoir les Constructeurs et la méthode toString de chacune des 3 classes.
3. Implémenter la méthode getSalaire() qui permet de calculer le salaire pour chacune des classes.

Exercice 5

Dans un établissement d'enseignement et de manière schématique, on trouve trois sortes de personnes : des administratifs (au sens large, incluant des techniciens) représentés par la catégorie secrétaire, des enseignants et des étudiants. Chaque personne est caractérisée par son nom et prénom, son adresse (rue et ville) qui sont des attributs privés et communs à toutes les personnes. On peut représenter une personne suivant un schéma UML comme indiqué sur la figure ci-après. Les variables d'instances (attributs) sont le *nom, le prénom, la rue et la ville*. *nbPersonnes* est une variable de classe (donc static) qui comptabilise le nombre de Personne dans l'établissement. Cette variable de classe existe en un exemplaire indépendant de tout objet.

On définit les méthodes public suivantes de la classe Personne :

- le constructeur Personne (String nom, String prenom, String rue, String ville) : crée et initialise un objet de type Personne.
- String toString () : fournit une chaîne de caractères correspondant aux caractéristiques (attributs) d'une personne.
- ecrirePersonne () : pourrait écrire les caractéristiques d'une personne. Sur l'exemple ci-dessous, elle ne fait rien. Elle est déclarée abstraite.
- static nbPersonnes () : écrit le nombre total de personnes et le nombre de personnes par catégorie. C'est une méthode de classe.
- modifierPersonne (String rue, String ville) : modifie l'adresse d'une personne et appelle ecrirePersonne () pour vérifier que la modification a bien été faite.



Une Secretaire est une Personne. Elle possède toutes les caractéristiques d'une Personne (nom, prenom, rue, ville) plus les caractéristiques spécifiques d'une secrétaire soit sur l'exemple, un numéro de bureau. Les méthodes suivantes sont définies pour un objet de la classe Secretaire :

- Secretaire (String nom, String prenom, String rue, String ville, String numeroBureau) : le constructeur d'un objet de la classe Secretaire doit fournir les caractéristiques pour construire une Personne, plus les spécificités de la classe Secretaire (numéro de bureau).
- String toString () : fournit une chaîne contenant les caractéristiques d'une Secretaire.
- ecrirePersonne () : écrit "Secrétaire : " suivi des caractéristiques d'une Secretaire.
- De même, un Enseignant est une Personne enseignant une spécialité (mathématiques, informatique, anglais, gestion, etc.)

Un Etudiant est une Personne préparant un diplôme (diplomeEnCours). Les méthodes pour Enseignant et Etudiant sont similaires à celles de Secretaire.

Une variable privée static dans chaque classe compte le nombre de personnes créées dans chaque catégorie. Une méthode static du même nom que la variable fournit la valeur de cette variable static (nbSecretaires, nbEnseignants, nbEtudiants).