

TP OOP N°3

Master 1 MISC

École d'ingénieurs du Littoral-Côte-d'Opale

Université du Littoral Côte d'Opale

Document à rendre :

un rapport de TP qui contient

les codes utilisés,

avec description et références à l'énoncé,

et commentaires éventuels.

**Attention**

| Lisez ATTENTIVEMENT les explications et les consignes de travail.

**Attention**

| Remarque : vous conserverez une trace numérique de toutes les actions réalisées dans le fichier de scripts. Cela vous permettra de relancer des ensembles de commandes en une seule fois. Vous remettrez le fichier à la fin de chaque séance.

Exercice 1 : 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 (get...) et des mutateurs (set...) ou les propriétés.
- La classe Véhicule possède également deux méthodes 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.

Q1. Créer la classe Véhicule.

Q2. Créer les classes Camion et Voiture.

Q3. Créer une classe Test qui permet de tester la classe Voiture et la classe Camion

Exercice 2 :

Q1. Ecrivez une classe Bâtiment avec les attributs suivants :

- adresse

La classe Bâtiment doit disposer des constructeurs suivants :

- Batiment(),
- Batiment (adresse).

La classe Bâtiment doit contenir des accesseurs et mutateurs (ou propriétés) pour les différents attributs. La classe Bâtiment doit contenir une méthode *ToString()* donnant une représentation du Bâtiment.

Q2. Ecrivez une classe **Maison** héritant de **Bâtiment** avec les attributs suivants :

- **NbPieces** : Le nombre de pièces de la maison.

La classe **Maison** doit disposer des constructeurs suivants :

- **Maison()**,
- **Maison(adresse, nbPieces)**.

La classe **Maison** doit contenir des accesseurs et mutateurs (ou des propriétés) pour les différents attributs. La classe **Maison** doit contenir une méthode *ToString()* donnant une représentation de la **Maison**.

Q3. Ecrivez aussi un programme afin de tester ces deux classes.

Exercice 3. Soit les classes suivantes :

une classe **Personne** qui comporte trois champs privés, nom, prénom et date de naissance. Cette classe comporte un constructeur pour permettre d'initialiser les données. Elle comporte également une méthode polymorphe **Afficher** pour afficher les données de chaque personne.

une classe **Employé** qui dérive de la classe **Personne**, avec en plus un champ *Salaire* accompagné de sa propriété, un constructeur et la redéfinition de la méthode **Afficher**.

une classe **Chef** qui dérive de la classe **Employé**, avec en plus un champ *Service* accompagné de sa propriété, un constructeur et la redéfinition de la méthode **Afficher**.

une classe **Directeur** qui dérive de la classe **Chef**, avec en plus un champ *Société* accompagné de sa propriété, un constructeur et la redéfinition de la méthode **Afficher**.

Q1. Ecrire les classes **Personne**, **Employé**, **Chef** et **Directeur**.

Q2. Créez un programme de test qui comporte tableau de huit personnes avec cinq employés, deux chefs et un directeur (8 références de la classe **Personne** dans lesquelles ranger 5 instances de la classe **Employé**, 2 de la classe **Chef** et 1 de la classe **Directeur**).

Q3. Affichez l'ensemble des éléments du tableau à l'aide d'un boucle.

Exercice 4 : Soit la classe Employé caractérisée par 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 \$
- L'ouvrier a un salaire mensuel qui est : $\text{Salaire} = \text{SMIG} + (\text{Ancienneté en année}) * 100$.
- De plus, le salaire ne doit pas dépasser $\text{SMIG} * 2$.

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 \$
 - 2 : salaire mensuel 15000 \$
 - 3 : salaire mensuel 17000 \$
 - 4 : salaire mensuel 20000 \$

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 : $\text{Salaire} = \text{CA} * \text{pourcentage} / 100$

Q1. Créer la classe Employé.

Q2. 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.

Implémenter la méthode GetSalaire() qui permet de calculer le salaire pour chacune des classes.

Q3. Réaliser le programme de test pour lancer les différents appels pour vérifier la fonctionnalité.