Exercices

1. Mettre en place le concept du polymorphisme.

**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.

Travail à faire:

1) Ecrire les classe Personne, Employé, Chef et Directeur.

2) 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).

Affichez l'ensemble des éléments du tableau à l'aide de for

2) L’heritage des classes –Comptes

Objectif

* Créer une classe dérivée.
* Ajouter des méthodes à une classe dérivée.
* Redéfinir des méthodes dans une classe dérivée.

## Énonce

* Un compte bancaire possède à tout moment une donnée : son solde. Ce solde peut être positif (compte créditeur) ou négatif (compte débiteur).
* Chaque compte est caractérisé par un code incrémenté automatiquement.
* le code et le solde d'un compte sont accessibles en lecture seulement.
* A sa création, un compte bancaire a un solde nul et un code incrémenté.
* Il est aussi possible de créer un compte en précisant son solde initial.
* Utiliser son compte consiste à pouvoir y faire des dépôts et des retraits. Pour ces deux opérations, il faut connaître le montant de l'opération.
* L'utilisateur peut aussi consulter le solde de son compte par la méthode toString().
* Un compte Epargne est un compte bancaire qui possède en plus un champ « Taux Intérêt=6 » et une méthode calculIntérêt() qui permet de mettre à jour le solde en tenant compte des intérêts.
* Un ComptePayant est un compte bancaire pour lequel chaque opération de retrait et de versement est payante et vaut 5

**Questions :**

1. Définir la classe Compte.
2. Définir la classe CompteEpargne.
3. Définir la classe ComptePayant.
4. Créer un programme permettant de tester ces classes avec les actions suivantes:

* Créer une instance de la classe Compte , une autre de la classe CompteEpargne et une instance de la classe ComptePayant
* Faire appel à la méthode déposer() de chaque instance pour déposer une somme quelconque dans ces comptes.
* Faire appel à la méthode retirer() de chaque instance pour retirer une somme quelconque de ces comptes.
* Faire appel à la méthode calculInterêt() du compte Epargne.
* Afficher le solde des 3 comptes.

## 3) Objectif:

* Créer une classe abstraite.
* Dériver une classe abstraite.
* Implémenter une méthode abstraite.

## Énonce:

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 , son modèle et 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 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

## 4) Objectif:

* Créer une classe dérivée.
* Ajouter des méthodes à une classe dérivée.
* Redéfinir des méthodes dans une classe dérivée.

## Énonce:

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.

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.

Ecrivez aussi un programme afin de tester ces deux classes.

## 5) Objectif : Manipuler les collections de type Liste.

Soit à développer une application pour la gestion d’un stock.

Un article est caractérisé par son numéro de référence, son nom, son prix de vente et une quantité en stock.

Le stock est représenté par une collection d’articles**.**

### Travail à faire:

Créer la classe article contenant les éléments suivants :

* Les attributs/propriétés.
* Un constructeur d’initialisation.
* La méthode toString().

Dans la classe Program créer :

Le stock sous forme d'une collection d’articles de votre choix.

Un menu présentant les fonctionnalités suivantes :

1. Rechercher un article par référence.
2. Ajouter un article au stock en vérifiant l’unicité de la référence.
3. Supprimer un article par référence.
4. Modifier un article par référence.
5. Rechercher un article par nom.
6. Rechercher un article par intervalle de prix de vente.
7. Afficher tous les articles.
8. Quitter