

TP 3 – 2<sup>ème</sup> Partie – Héritage et classe abstraite

**Exercice 1/**

On désire créer une hiérarchie de personnels pour une entreprise.  
Pour chaque classe, écrivez un constructeur qui reçoit en paramètres toutes les données nécessaires.

**Question 1 /**

Écrivez une classe **abstraite** `Employe`. Un employé est caractérisé par son nom, son âge, le nombre d'année d'expérience (`Nb_A`) et un numéro généré automatiquement lors de chaque création.

**Question 2 /**

Écrivez une classe `Ouvrier` qui hérite de la classe `Employe`. Un `Ouvrier` est un `Employe` caractérisé en plus par un nombre d'heure fixe (`Nb_H_O= 35`).

**Question 3/**

Écrivez une classe `Gerant` qui hérite de la classe `Employe`. Un `Gérant` est un `Employe` caractérisé en plus par un nombre d'heure fixe (`Nb_H_G= 30`).

**Question 4/**

On souhaite calculer le salaire de chaque employé, salaire qui se calcule de la façon suivante :

- pour un ouvrier:  $4 * Nb\_H\_O * (10 + Nb\_A/2)$ ,
- pour un gérant :  $5 * Nb\_H\_G * (20 + Nb\_A/2)$

Pensez à la notion de **méthode abstraite**.

**Question 5/**

On souhaite afficher pour chaque employé sa catégorie (`Gérant` ou `Ouvrier`) et toutes les autres informations (numéro, nom, âge, salaire).

Pensez à la notion de réutilisation.

**Question 6/**

Écrivez une classe `ListeEmployes` ayant en variable d'instance une `ArrayList` d'`Employe` et comportant des méthodes pour y ajouter un employé, la trier selon le salaire (utilisez l'algorithme de votre choix), l'afficher, etc.

**Question 7/**

Écrivez une méthode `selectionner` acceptant 2 arguments (`min` et `max`) et permettant de retourner une instance de `ListeEmployes` composés des employés ayant un âge compris strictement entre `min` et `max`.

**Question 8/**

Dans une classe `Exercice1`, écrivez un programme qui :

- crée 3 ouvriers et 1 gérant,
- affiche le nombre des employés créés,
- crée une instance de `ListeEmploye` et y insère les 3 ouvriers et le gérant,
- affiche les informations pour tous les employés de la liste,
- trie les employés de la liste selon le salaire,
- ré-affiche les informations pour tous les employés de la liste,
- sélectionne et affiche les employés dont l'âge est compris entre 20 et 30 ans

## Exercice 2/

Une société de location de bateaux désire modéliser différentes catégories de bateaux :

- les voiliers,
- les bateaux à moteur,
- les *motor-yachts* (bateaux à moteur avec un équipage professionnel).

Un voilier se caractérise par un nom, un poids, un port d'attache, une longueur et une surface de voilure. Un bateau à moteur se caractérise par un nom, un poids, un port d'attache, une longueur, et la puissance du moteur (en kW). Un *motor-yacht* se caractérise par un nom, un poids, un port d'attache, une longueur, la puissance du moteur et le nombre de membres d'équipage.

On souhaite également calculer la somme due au titre de la taxe de francisation et qui se calcule de la façon suivante :

- pour un voilier : 50 € par mètre de longueur,
  - pour un bateau à moteur : 70 € par mètre de longueur + 5 € par kW,
  - pour un *motor-yacht* : 70 € par mètre de longueur + 5 € par kW  
+ 20 € par membre d'équipage
- c'est-à-dire ce qui est dû au titre du bateau à moteur plus ce qui s'ajoute au titre des membres d'équipage du *motor-yacht*.

L'affichage d'un bateau doit permettre de lister ses caractéristiques (selon sa catégorie) ainsi que la somme due au titre de sa taxe de francisation.

**Question 1/** Réfléchissez à un arbre d'héritage pour décrire cette modélisation.

Pour chaque classe vous indiquerez :

- **si elle est abstraite ou concrète**,
- les variables de classe,
- les variables d'instance,
- les entêtes des méthodes abstraites,
- les entêtes des méthodes d'instance,
- les entêtes des méthodes de classe.

Attention à bien commenter ces informations.

Il est à noter que certaines de ces rubriques peuvent être vides (par exemple, une classe sans variables de classe).

**Question 2/** Proposez une réalisation en langage Java.

Rappel : n'oubliez pas d'appliquer le principe d'encapsulation.

**Question 3/** Définissez une classe `ListeBateaux` comportant en variable d'instance une `ArrayList` de bateaux et des méthodes d'instance permettant :

- d'ajouter un bateau dans cette `ArrayList`,
- de lister les bateaux enregistrés dans cette `ArrayList`.
- de « retourner » la longueur d'un bateau dont le nom est passé en paramètre (on admet qu'il n'y a pas d'homonyme),
- de lister tous les bateaux d'un port d'attache dont le nom est passé en paramètre.

Complétez le programme principal.