

TD Classe Abstraite et Interface (<https://www.cours-gratuit.com/>)

cet exercice vous permettra de concevoir une hiérarchie de classes utilisant la notion d'interface. Il vous servira également de révision pour les notions d'héritage, de classes abstraites et de polymorphisme.

Le directeur d'une entreprise de produits chimiques souhaite gérer les salaires et primes de ses employés au moyen d'un programme Java.

Un employé est caractérisé par son nom, son prénom, son âge et sa date d'entrée en service dans l'entreprise.

Dans un fichier `Salaires.java`, codez une classe abstraite `Employe` dotée des attributs nécessaires, d'une méthode abstraite `calculerSalaire` (ce calcul dépendra en effet du type de l'employé) et d'une méthode `getNom` retournant une chaîne de caractère obtenue en concaténant la chaîne de caractères "L'employé " avec le prénom et le nom.
Dotez également votre classe d'un constructeur prenant en paramètre l'ensemble des attributs nécessaires.

Calcul du salaire

Le calcul du salaire mensuel dépend du type de l'employé. On distingue les types d'employés suivants :

- Ceux affectés à la *Vente*. Leur salaire mensuel est le 20 % du *chiffre d'affaire* qu'ils réalisent mensuellement, plus 400 Francs.
- Ceux affectés à la *Représentation*. Leur salaire mensuel est également le 20 % du *chiffre d'affaire* qu'ils réalisent mensuellement, plus 800 Francs.
- Ceux affectés à la *Production*. Leur salaire vaut le *nombre d'unités* produites mensuellement multipliées par 5.
- Ceux affectés à la *Manutention*. Leur salaire vaut leur *nombre d'heures* de travail mensuel multipliées par 65 francs.

Codez dans votre fichier `Salaires.java` une hiérarchie de classes pour les employés en respectant les conditions suivantes :

- La super-classe de la hiérarchie doit être la classe `Employe`.
- Les nouvelles classes doivent contenir les attributs qui leur sont spécifiques ainsi que le codage approprié des méthodes `calculerSalaire` et `getNom`, en changeant le mot "employé" par la catégorie correspondante.
- Chaque sous classe est dotée de constructeur prenant en argument l'ensemble des attributs nécessaires.
- N'hésitez pas à introduire des classes intermédiaires pour éviter au maximum les redondances d'attributs et de méthodes dans les sous-classes

Employés à risques

Certains employés des secteurs *production* et *manutention* sont appelés à fabriquer et manipuler des produits dangereux.

Après plusieurs négociations syndicales, ces derniers parviennent à obtenir une prime de risque mensuelle.

Complétez votre programme `Salaires.java` en introduisant deux nouvelles sous-classes d'employés. Ces sous-classes désigneront les employés des secteurs *production* et *manutention* travaillant avec des produits dangereux.

Ajouter également à votre programme une interface pour les *employés à risque* permettant de leur associer une *prime mensuelle* fixe de 200.-.

Collection d'employés

Satisfait de la hiérarchie proposée, notre directeur souhaite maintenant l'exploiter pour afficher le salaire de tous ses employés ainsi que le salaire moyen.

Ajoutez une classe `Personnel` contenant une "collection" d'employés. Il s'agira d'une collection polymorphique d'`Employe` - regardez le cours si vous ne voyez pas de quoi il s'agit.

Définissez ensuite les méthodes suivantes à la classe `Personnel` :

- `void ajouterEmploye(Employe)`
qui ajoute un employé à la collection.
- `void calculerSalaires()`
qui affiche le salaire de chacun des employés de la collection.

- `double salaireMoyen()`
qui affiche le salaire moyen des employés de la collection.

Testez votre programme avec le main suivant :

```
1 class Salaires {
2     public static void main(String[] args) {
3         Personnel p = new Personnel();
4         p.ajouterEmploye(new Vendeur("Pierre", "Business", 45, "1995", 30000));
5         p.ajouterEmploye(new Representant("Léon", "Vendtout", 25, "2001", 2000));
6         p.ajouterEmploye(new Technicien("Yves", "Bosseur", 28, "1998", 1000));
7         p.ajouterEmploye(new Manutentionnaire("Jeanne", "Stocketout", 32, "1998", 45));
8         p.ajouterEmploye(new TechnARisque("Jean", "Flippe", 28, "2000", 1000));
9         p.ajouterEmploye(new ManutARisque("Al", "Abordage", 30, "2001", 45));
10
11         p.afficherSalaires();
12         System.out.println("Le salaire moyen dans l'entreprise est de " + p.salaireMoyen() + "
13     }
14 }
```