4éme atelier

Polymorphisme, class abstraite, méthode abstraite, interface

Exercice 1 :

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.

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 chaine de caractère obtenue en concaténant la chaine 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 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.

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'un tableau polymorphique d'Employe.

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.

Que produit le programme suivant :

```
class Salaires {
    public static void main(String[] args) {
       Personnel p = new Personnel();
       p.ajouterEmploye(new Vendeur("Pierre", "Business", 45,
"1995", 30000));
       p.ajouterEmploye(new Representant("Léon", "Vendtout",
25, "2001", 20000));
       p.ajouterEmploye(new Technicien("Yves", "Bosseur", 28,
"1998", 1000));
       p.ajouterEmploye(new Manutentionnaire("Jeanne",
"Stocketout", 32, "1998", 45));
        p.ajouterEmploye(new TechnARisque("Jean", "Flippe",
28, "2000", 1000));
        p.ajouterEmploye(new ManutARisque("Al", "Abordage",
30, "2001", 45));

      p.afficherSalaires();
        System.out.println("Le salaire moyen dans l'entreprise
est de " + p.salaireMoyen() + " francs.");
    }
}
```

Exercice 2 :

1. **Définissez la classe «Personne» qui possède :**
    - deux attributs privés, nommés «nom» et «prenom» de types String ;
    - un attribut privé, nommé «DateNaissance» de type String (qui doit être saisie sous la forme 23/12/1980) ;
    - un constructeur qui permet d'initialiser les différents attributs ;
    - une méthode abstraite «description()» qui servira à afficher le statut de la personne(Etudiant, Professeur, autre).
    - redéfinissez la méthode « toString() » pour retourner les différentes informations concernant une personne.
2. **Définissez la classe «Professeur» qui hérite de la classe «Personne». Ajoutez à la classe«Professeur» :**
    - un attribut privé «coursEnseigne» (cours enseigné par le professeur).
3. **Définissez la classe «Etudiant» qui hérite de la classe «Personne». Ajoutez à la classe«Etudiant» :**
    - un attribut privé «cne».
    - un tableau de type double qui contient 6 notes.
    - un tableau de type boolean prévu pour contenir les inscriptions aux modules.
    - une méthode « moyenne() » qui permet de calculer la moyenne des notes selon l'inscription aux modules.
    - une méthode « mention() » qui permet d'afficher la mention.

Si l'étudiant est inscrit dans 6 modules, la méthode affichera la mention, sinon elle affichera le message : « Etudiant non inscrit dans 6 modules ».

4. **Définissez la classe «EtudiantEtranger» qui hérite de la classe «Etudiant».**

Ajoutez l'attribut privé «nationalité».

5. **Redéfinissez dans les différentes sous classes, la méthode «toString()»**
6. **Définissez la classe «SmiS6» qui contient le tableau statique modules. Le tableau contient les noms des 6 modules.**
7. **Utilisez la classe «GestionSMI» pour les tests :**
    a) Créez 5 objets de type « Personne » : 4 étudiants, dont un étranger ; 1 professeur. Initialisez les différents attributs.
    b) Pour les 4 étudiants : le premier et le deuxième (étranger) sont inscrits dans 6 modules ; le troisième est inscrit dans les modules M1 et M2 ; le quatrième est inscrit dans les modules M2, M3 et M5.
    c) Saisissez les notes des différents étudiants et affichez la moyenne correspondante.
    d) En utilisant la boucle (for - pour chaque élément) affichez les différentes notes des étudiants.
    e) Affichez l'âge des différents étudiants. Pour obtenir l'année en cours, vous pouvez utiliser la classe «GregorianCalendar»