TP noté Java POO

19 mai 2019

Durée : 1h30 - AUCUN DOCUMENT - SUJET À JOINDRE À LA COPIE

Instructions:
-Ce TP noté de 1h30 est un travail en binôme sur NetBeans dans le même groupe de TP.
-Compresser par zip, rar ou 7z tous les dossiers et fichiers de votre projet.
-Exemple, pour le duo Manolo Hina et JP Segado, déposer le dossier Hina_Segado.zip.
-Déposer ce dossier compressé à la fin de la séance de TP sur la page campus.
-Des extraits de la documentation des API Java sont fournis en annexe dans la dernière page.
-Votre code devra être commenté pour plus de clarté, sous peine d'être pénalisé.

1 Gestion des horaires de travail

Le directeur d'une entreprise souhaite gérer les horaires de travail de ses employés avec un programme Java. Un employé est caractérisé par son nom, son prénom, son âge et sa date de recrutement dans l'entreprise. Écrivez une classe abstraite **Employé** dotée:

- des attributs nécessaires.
- d'un constructeur prenant en paramètre l'ensemble des attributs nécessaires.
- d'une méthode abstraite calculerHoraire(int semaine) ce calcul prendra en paramètre le type de l'employé et la semaine de travail.
- d'une méthode getNom() retournant une chaîne de caractère obtenue en concaténant "Employé: " avec le prénom et le nom.

2 Héritage d'employés

Le calcul des horaires hebdomadaires dépend des missions de l'employé. On distingue les missions suivantes :

- **Ceux affectés à la Vente:** Leur horaire est de 32h les trois premières semaines du mois et de 48h la dernière semaine du mois.
- **Ceux affectés à la Production:** Leur horaire est de 30h les semaines paires et de 42h les semaines impaires.
- **Ceux affectés à la Manutention:** Leur horaire est constant et est de 35h par semaine.

Implémentez les classes **Vente**, **Production** et **Manutention** en respectant les conditions suivantes :

- La super-classe de la hiérarchie doit être la classe Employé.
- Les nouvelles classes doivent contenir les attributs qui leur sont spécifiques ainsi que les méthodes calculerHoraire et getNom(), en remplaçant "Employé: " par la catégorie correspondante (Vendeur, Manutentionnaire ou Technicien).
- Chaque sous-classe est dotée de constructeur prenant en argument l'ensemble des attributs nécessaires.

3 Collection d'employés

Satisfait de la hiérarchie proposée, le directeur souhaite maintenant exploiter ce programme pour afficher les horaires de tous ses employés ainsi que l'horaire moyen. Ajoutez une classe **Personnel** contenant une ArrayList d'employés.

Définissez ensuite les méthodes suivantes dans la classe Personnel :

- void ajouterEmploye(Employe) qui ajoute un employé à la collection.
- void calculerHoraires(int i) qui affiche l'horaire de chacun des employés de la collection pour la semaine i.
- int HoraireMensuel() qui affiche l'horaire sur un mois des employés de la collection.

Testez votre programme avec le main suivant :

```
public static void main(String[] args) {
Personnel p = new Personnel();
p.ajouteEmploye(new Vente("Pierre", "Business", 45, 2005));
p.ajouteEmploye(new Vente("Léon", "Vendtout", 25, 2011));
p.ajouteEmploye(new Production("Yves", "Bosseur", 28, 2000));
p.ajouteEmploye(new Manutention("Jeanne", "Stocketout", 32, 2008));

p.calculerHoraires(2);
System.out.println("Horaire moyen dans cette entreprise est de " +,p.HoraireMensuel() + " heures.");
}
```

qui devrait vous afficher :

```
Vendeur Pierre Business 32 heures.
Vendeur Léon Vendtout 32 heures.
Technicien Yves Bosseur 30 heures.
Manutentionnaire Jeanne Stocketout 35 heures.
Horaire moyen dans cette entreprise est de 143 heures.
```

4 Exception d'employés

Le directeur de l'entreprise souhaite rajouter des fonctionnalités à son programme de manière à ce que les employés recrutés avant 2007 et âgés de plus de 40 ans puissent bénéficier d'une prime.

- Ajouter une classe d'exception ExceptionEmployé.
- Ajouter une méthode vérifierPrime() throws ExceptionEmployé qui génère une exception lorsqu'un employé satisfait ces contraintes.
- Ajouter une méthode vérifierPrimes() dans la classe Personnel pour afficher uniquement les employés concernés par la prime.

TP noté Java POO

19 mai 2019

Durée : 1h30 - AUCUN DOCUMENT - SUJET À JOINDRE À LA COPIE

Instructions:
-Ce TP noté de 1h30 est un travail en binôme sur NetBeans dans le même groupe de TP.
-Compresser par zip, rar ou 7z tous les dossiers et fichiers de votre projet.
-Exemple, pour le duo Manolo Hina et JP Segado, déposer le dossier Hina_Segado.zip.
-Déposer ce dossier compressé à la fin de la séance de TP sur la page campus.
-Des extraits de la documentation des API Java sont fournis en annexe dans la dernière page.
-Votre code devra être commenté pour plus de clarté, sous peine d'être pénalisé.

1 Gestion des horaires de travail

Le directeur d'une entreprise souhaite gérer les horaires de travail de ses employés avec un programme Java. Un employé est caractérisé par son nom, son prénom, son âge et sa date de recrutement dans l'entreprise. Écrivez une classe abstraite **Employé** dotée:

- des attributs nécessaires.
- d'un constructeur prenant en paramètre l'ensemble des attributs nécessaires.
- d'une méthode abstraite calculerHoraire(int semaine) ce calcul prendra en paramètre le type de l'employé et la semaine de travail.
- d'une méthode getNom() retournant une chaîne de caractère obtenue en concaténant "Employé: " avec le prénom et le nom.

2 Héritage d'employés

Le calcul des horaires hebdomadaires dépend des missions de l'employé. On distingue les missions suivantes :

- **Ceux affectés à la Vente:** Leur horaire est de 32h les trois premières semaines du mois et de 48h la dernière semaine du mois.
- **Ceux affectés à la Production:** Leur horaire est de 30h les semaines paires et de 42h les semaines impaires.
- **Ceux affectés à la Manutention:** Leur horaire est constant et est de 35h par semaine.

Implémentez les classes **Vente**, **Production** et **Manutention** en respectant les conditions suivantes :

- La super-classe de la hiérarchie doit être la classe Employé.
- Les nouvelles classes doivent contenir les attributs qui leur sont spécifiques ainsi que les méthodes calculerHoraire et getNom(), en remplaçant "Employé: " par la catégorie correspondante (Vendeur, Manutentionnaire ou Technicien).
- Chaque sous-classe est dotée de constructeur prenant en argument l'ensemble des attributs nécessaires.

3 Collection d'employés

Satisfait de la hiérarchie proposée, le directeur souhaite maintenant exploiter ce programme pour afficher les horaires de tous ses employés ainsi que l'horaire moyen. Ajoutez une classe **Personnel** contenant une ArrayList d'employés.

Définissez ensuite les méthodes suivantes dans la classe Personnel :

- void ajouterEmploye(Employe) qui ajoute un employé à la collection.
- void calculerHoraires(int i) qui affiche l'horaire de chacun des employés de la collection pour la semaine i.
- int HoraireMensuel() qui affiche l'horaire sur un mois des employés de la collection.

Testez votre programme avec le main suivant :

```
public static void main(String[] args) {
Personnel p = new Personnel();
p.ajouteEmploye(new Vente("Pierre", "Business", 45, 2005));
p.ajouteEmploye(new Vente("Léon", "Vendtout", 25, 2011));
p.ajouteEmploye(new Production("Yves", "Bosseur", 28, 2000));
p.ajouteEmploye(new Manutention("Jeanne", "Stocketout", 32, 2008));

p.calculerHoraires(2);
System.out.println("Horaire moyen dans cette entreprise est de " +,p.HoraireMensuel() + " heures.");
}
```

qui devrait vous afficher :

```
Vendeur Pierre Business 32 heures.
Vendeur Léon Vendtout 32 heures.
Technicien Yves Bosseur 30 heures.
Manutentionnaire Jeanne Stocketout 35 heures.
Horaire moyen dans cette entreprise est de 143 heures.
```

4 Exception d'employés

Le directeur de l'entreprise souhaite rajouter des fonctionnalités à son programme de manière à ce que les employés recrutés avant 2007 et âgés de plus de 40 ans puissent bénéficier d'une prime.

- Ajouter une classe d'exception ExceptionEmployé.
- Ajouter une méthode vérifierPrime() throws ExceptionEmployé qui génère une exception lorsqu'un employé satisfait ces contraintes.
- Ajouter une méthode vérifierPrimes() dans la classe Personnel pour afficher uniquement les employés concernés par la prime.