Travaux Pratiques Java
Les bases de la POO

Gaëtan MARECAT

2023 - 2024

Pour ce TP et les suivants, téléchargez l'IDE IntelliJ.

Créer un nouveau projet vide Java. Pour chaque exercice, faire un nouveau package dans le dossier src/.

1 Première Class

1.1 Base

Créez une classe Java appelée "Personne" (sans constructeur pour le moment) avec les caractéristiques suivantes :

1. Un attribut nom de type String pour stocker le nom de la personne.
2. Un attribut age de type int pour stocker l'âge de la personne.
3. Une méthode afficherDetails() qui affiche le nom et l'âge de la personne.

Ensuite, créez une classe principale "Main" qui instancie deux objets de la classe "Personne" en utilisant le constructeur Personne(). Définir leurs attributs, puis les affiche en utilisant la méthode afficherDetails().

Rajouter un constructeur qui prends en paramètre le nom et l'âge. Que constatez-vous ?

Rajouter ensuite un constructeur vide en plus du précédent.

2 Héritage

2.1 Reprise de l'exercie précédent

1. Avec les différents constructeurs, créer une troisième personne qui a des attributs avec des valeurs différentes des deux premières (un autre nom et un autre âge) puis une quatrième avec des attributs identiques à la troisième (même nom et même âge).
2. Afficher en console les égalités entre la personne 1 et la personne 3 puis entre la personne 3 et la personne 4 avec "=="
3. Faire la même chose que l'étape précédente mais avec la méthode equals() de la class Object à la place de "==".
4. Afficher également les personnes en console en les printant directement puis faire de même avec la méthode toString().
   Tips : pour afficher l'égalité, utiliser cette syntaxe : System.out.println("personne 1 == personne 3 : " + (personne1 == personne2)) ;

2.2 Surcharge

Redéfinir les méthodes equals() et toString() de la classe Object dans la classe Personne. Puis refaire les manipulations précédentes.

1. toString() : cette méthode doit retourner l'objet Personne sous forme de String
2. equals() : cette méthode doit comparer les objets de type Personne

2.3 Formes Géométriques

Écrire un programme Java qui permet de gérer différentes formes géométriques, telles que des cercles et des rectangles. Créer une classe de base abstraite appelée *FormeGeometrique* avec les méthodes suivantes :

1. *public abstract double calculerAire() ;* Une méthode abstraite pour calculer l'aire de la forme géométrique.
2. *public abstract double calculerPerimetre() ;* Une méthode abstraite pour calculer le périmètre de la forme géométrique.
3. *public abstract void afficherDetails() ;* Une méthode pour afficher les détails de la forme géométrique.

Ensuite, créez deux classes dérivées, *Cercle* et *Rectangle*, qui héritent de la classe FormeGeometrique et implémentent les méthodes abstraites pour calculer l'aire et le périmètre spécifiques à chaque forme. *Cercle* doit avoir en attribut au moins le rayon et *Rectangle* doit au moins avoir en attribut longueur et largeur. Vous être libre de définir les constructeurs de ces classes, si besoin se référer au guide proposé ci-dessous.

Redéfinir également les méthode *equals()* et *toString()* de la classe *Object* dans les classes *Rectange* et *Cercle*.

Créer un programme qui créer deux cercles et deux rectangles dans un tableau puis de calculer l'aire totale et le périmètre total de toutes les formes, et d'afficher les détails de chaque forme.

Guide proposé si besoin pour faire l'exercice

1. Créer une classe abstraite FormeGeometrique avec les méthodes demandées
2. Créer la classe Cercle. Ajouter un attribut rayon à la classe Cercle puis faire un constructeur qui prend en paramètre le rayon du cercle. Hériter la classe de FormeGeometriqueUtiliser et surcharger les méthodes de cette classe dans votre classe Cercle à l'aide de l'attribut rayon.
3. Créer la classe Rectangle. Ajouter un attribut largeur et un attribut longueur à la classe Rectangle puis faire un constructeur qui prend en paramètre la largeur et la longueur du rectangle. Hériter la classe de FormeGeometriqueUtiliser et surcharger les méthodes de cette classe dans votre classe Rectangle à l'aide des attributs largeur et longueur.
4. Faire une classe principale avec une méthode main
5. Dans cette méthode créer les deux cercles et les deux rectangle puis calculée l'air total à l'aide d'un parcours du tableau

3 Interface

Créez une interface Java appelée "Animal" avec les méthodes suivantes :

1. emettreSon() qui renvoie le son que fait l'animal.
2. afficherInfos() qui affiche des informations spécifiques à l'animal.

Créez une classe "Chien" qui implémente l'interface "Animal" et définit un attribut nom de type String. Implémentez les méthodes de l'interface pour que le chien émette un son et affiche des informations spécifiques (par exemple, la race du chien).

Créez une classe "Chat" qui implémente l'interface "Animal" et définit un attribut nom de type String. Implémentez les méthodes de l'interface pour que le chat émette un son et affiche des informations spécifiques (par exemple, le nombre de griffes du chat).

Dans une classe principale "Main", créez un objet "Chien" et un objet "Chat", définissez leurs attributs, puis faites-les émettre un son et afficher leurs informations spécifiques.

4 List

Créez un programme Java qui permet à l'utilisateur de saisir plusieurs noms jusqu'à ce qu'il décide d'arrêter la saisie. Stockez ces noms dans une liste (par exemple, une ArrayList). Une fois la saisie terminée, affichez la liste des noms. Demandez à l'utilisateur de saisir un nom supplémentaire et recherchez s'il existe déjà dans la liste. Affichez un message indiquant si le nom a été trouvé ou non.

5 Map

Créez un programme Java qui permet à l'utilisateur de gérer un répertoire de contacts. Utilisez une Map (par exemple, une HashMap) pour stocker les noms de contacts en tant que clés et les numéros de téléphone en tant que valeurs. Permettez à l'utilisateur d'ajouter de nouveaux contacts, de mettre à jour les numéros de téléphone existants, de supprimer des contacts et d'afficher la liste des contacts avec leurs numéros de téléphone.