source : https://waytolearnx.com/2020/04/exercices-corriges-java-les-classes-et-lheritage-partie-2.html

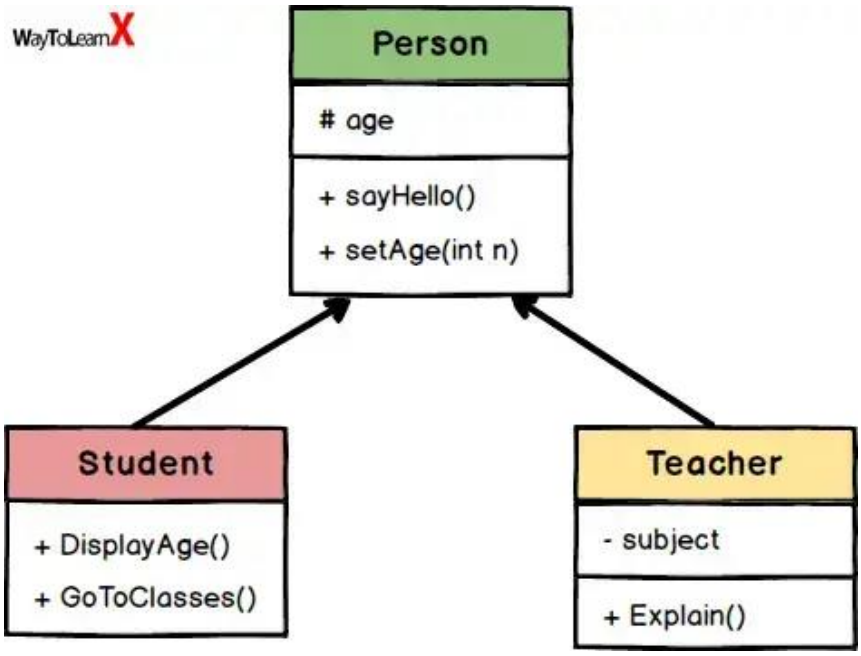
Exercices corrigés Java les classes et l'héritage – Partie 2

avril 27, 2020 1 Commentaire exercice java corrige heritage

Avec des exercices corrigés en Java sur les classes et l'héritage, vous pratiquerez divers concepts du langage Java. Vous commencerez par des exercices Java de base à des exercices plus avancés. La solution est fournie pour chaque exercice. Vous devez essayer de résoudre chaque problème par vous-même avant de vérifier la solution. Si vous avez des questions concernant chaque problème, nous vous encourageons à les poster sur notre forum.

Exercice 2 :

– Créez une classe « Person »
– Créez une classe « Student » et une autre classe « Teacher », les deux héritent de la classe « Person ».
– La classe « Student » aura une méthode publique « GoToClasses », qui affichera à l'écran « I'm going to class. ».
– La classe « Teacher » aura une méthode publique « Explain », qui affichera à l'écran « Explanation begins ». En plus, il aura un attribut privé « subject » de type string.
– La classe « Person » doit avoir une méthode « SetAge(int n) » qui indiquera la valeur de leur âge (par exemple, 15 years old).
– La classe « Student » aura une méthode publique « DisplayAge » qui écrira sur l'écran « My age is: XX years old ».
– Vous devez créer une autre classe de test appelée « Test » qui contiendra « Main » et:
– Créez un objet Person et faites-lui dire « Hello »
– Créer un objet Student, définir son âge à 15 ans, faites-lui dire « Hello », « I'm going to class. » et afficher son âge
– Créez un objet Teacher, 40 ans, demandez-lui de dire « Hello » puis commence l'explication.



Exemple:

```
Hello

I’m going to class.

Hello

My age is: 15 years old

Hello

Explanation begins
```

Exercice 4 :

Créez une classe « House », avec un attribut « surface », un constructeur qui définit sa valeur et une méthode « Display » pour afficher « Je suis une maison, ma surface est de XXX m2 » (XXX: la valeur de surface). Incluez aussi des getters et des setters pour la surface.

La classe « House » contiendra une porte (Door). Chaque porte aura un attribut « color » (de type String), et une méthode « Display » qui affichera « Je suis une porte, ma couleur est bleu » (ou quelle que soit la couleur). Inclure un getter et un setter. Créez également la méthode « GetDoor » dans la classe « House ».

La classe « Apartment » est une sous-classe de la classe « House », avec une surface prédéfinie de 50m2.

Créez également une classe Person, avec un nom (de type String). Chaque personne aura une maison. La méthode « Display » pour une personne affichera son nom, les données de sa maison et les données de la porte de cette maison.

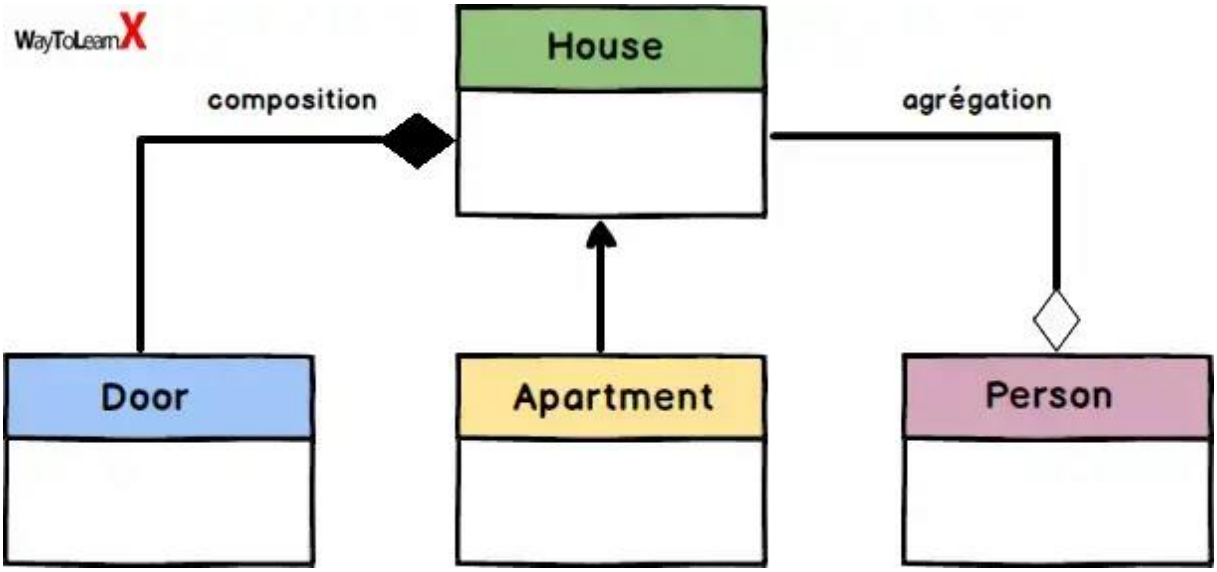
Écrivez un Main pour créer un Apartment, une personne pour y vivre et pour afficher les données de la personne.

Exemple:

```
Je m'appele Thomas.

Je suis un appartement, ma surface est 50 m2

Je suis une porte
```



des notions UML à savoir :

- **La composition** peut être considérer comme une relation "**fait partie de**", c'est à dire que si un objet Y fait partie d'un objet X alors Y ne peut pas exister sans X. Ainsi si X disparaît alors Y également.
- **L'agrégation** peut être considérer comme une relation de type "**a un**", c'est à dire que si un objet X a un objet Y alors Y peut vivre sans X.