Fiche d'exercices : Programmation orientée objet

Exercice 1 : Héritage

Objectif : Comprendre et utiliser l'héritage en programmation orientée objet.

1. Définir une classe `Person` avec les attributs suivants :

- `name` (chaîne de caractères)
- `firstName` (chaîne de caractères)
- `age` (entier)

2. Définir une méthode `introduceSelf` dans la classe `Person` qui affiche les informations de la personne.

3. Définir une classe `Student` qui hérite de la classe `Person` et ajouter l'attribut supplémentaire :

- `school` (chaîne de caractères)

4. Définir une méthode `introduceSelf` dans la classe `Student` qui affiche les informations de l'étudiant, y compris l'établissement.

5. Créer une instance de `Person` et une instance de `Student`, puis appeler leur méthode `introduceSelf`.

Exercice 2 : Composition

Objectif : Utiliser la composition pour créer des relations "partie-tout".

1. Définir une classe `Address` avec les attributs suivants :

- `street` (chaîne de caractères)
- `city` (chaîne de caractères)
- `postalCode` (chaîne de caractères)

2. Définir une classe `Person` (comme précédemment) et ajouter un attribut `address` qui est une instance de la classe `Address`.

3. Modifier la méthode `introduceSelf` de la classe `Person` pour inclure les informations de l'adresse.

4. Créer une instance de `Address` et une instance de `Person`, puis afficher les informations complètes de la personne.

Exercice 3 : Agrégation

Objectif : Comprendre et utiliser l'agrégation pour définir des relations entre classes.

1. Définir une classe `Course` avec les attributs suivants :

- `title` (chaîne de caractères)
- `teacher` (chaîne de caractères)

2. Modifier la classe `Student` pour qu'elle puisse avoir une liste de `courses` (liste d'instances de la classe `Course`).

3. Définir une méthode `addCourse` dans la classe `Student` pour ajouter un cours à la liste des cours de l'étudiant.

4. Définir une méthode `showCourses` dans la classe `Student` pour afficher tous les cours de l'étudiant.

5. Créer une instance de `Student` et ajouter plusieurs cours, puis afficher les cours de l'étudiant.

Exercice 4 : Association

Objectif : Utiliser l'association pour créer des relations entre objets.

1. Définir une classe `Company` avec les attributs suivants :

- `name` (chaîne de caractères)
- `industry` (chaîne de caractères)

2. Ajouter un attribut `company` dans la classe `Person` pour représenter l'association avec une entreprise.

3. Définir une méthode `changeCompany` dans la classe `Person` pour changer l'entreprise associée à une personne.

4. Créer une instance de `Person` et une instance de `Company`, puis associer la personne à l'entreprise et afficher les informations.

Exercice 5 : Diagrammes de classes

Objectif : Créer des diagrammes de classes pour représenter les relations entre les classes.

1. Dessiner un diagramme de classes pour les classes `Person` et `Student` avec la relation d'héritage.

2. Dessiner un diagramme de classes pour les classes `Person` et `Address` avec la relation de composition.

3. Dessiner un diagramme de classes pour les classes `Student` et `Course` avec la relation d'agrégation.

4. Dessiner un diagramme de classes pour les classes `Person` et `Company` avec la relation d'association.