

TD : Introduction à la programmation orientée objet

L'objectif de ce TD est vous permettre de vous faire manipuler les notions de base de la programmation orientée objet.

1) Création de la classe et écriture du constructeur

Question 1. Créer une nouvelle classe nommée `Eleve`, qui servira à représenter un élève de l'école. Tester ce code en créant un objet de type `Eleve`.

Remarque : Dans votre constructeur, créez un premier attribut `prenom`.

Question 2. Modifier le constructeur pour permettre à l'utilisateur de fixer deux attributs pour cette classe : `prenom` et `nom`. Tester ce code en créant plusieurs objets de type `Eleve`.

Question 3. Modifier à nouveau le constructeur pour ajouter deux attributs à cette classe :

- Un attribut `dateDeNaissance`, qui sera un nombre entier fourni par l'utilisateur pour indiquer la date de naissance de l'élève
- Un attribut `notes`, de type `list`, qui contiendra à terme les notes de l'élève, et qui doit être créé automatiquement (en tant que liste vide) lorsqu'on crée un objet de type `Eleve`.

Remarque : La date de naissance sera fournie au format AAAAMMJJ. Par exemple, le 19 octobre 1977 sera représenté par le nombre 19771019

2) Premières méthodes

Question 4. Créer une première méthode pour cette classe, nommée `saluer`, qui affiche "Bonjour, je m'appelle [prenom] [nom]".

Question 5. Créer une méthode `afficherInitiales`, qui affiche les initiales de l'élève, c'est-à-dire la première lettre du prénom (en majuscule et suivie d'un point) puis la première lettre du nom (en majuscule et suivie d'un point).

Indice : Pour obtenir la première lettre d'une chaîne de caractères `s`, il vous suffit d'écrire `s[0]`.

Question 6. Créer une méthode `ajouterNote`, qui prend en argument un nombre entier et ajoute ce nombre à la liste des notes de l'élève.

Question 7. Créer une méthode `calculerMoyenne`, qui renvoie la moyenne des notes de l'élève. Si l'élève n'a aucune note, cette méthode doit renvoyer `-1`.

3) Méthodes supplémentaires

Question 8. Créer une méthode `calculerAge`, qui détermine l'âge de l'élève à partir de sa date de naissance et le renvoie sous forme d'entier.

Remarque : Pour obtenir la date du jour, on pourra utiliser la syntaxe suivante :

```
import datetime

t = datetime.datetime.now()
jour = t.day
mois = t.month
annee = t.year
```

★ **Question 9.** Ecrire la méthode `__repr__` pour permettre l'affichage des informations disponibles sur l'élève lorsqu'on utilise la fonction `print`.

Le résultat devra s'afficher selon l'exemple suivant :

```
Prenom : John
Nom : Doe
Date de naissance : 19771019
Notes : [15, 19, 18]
```

★ **Question 10.** Ecrire la méthode `__eq__` pour permettre un test d'égalité entre deux élèves : on considère que deux élèves qui ont le même nom, le même prénom, et la même date de naissance sont en fait la même personne.

4) Héritage

Question 11. Créer une classe `Delegate` qui hérite de la classe `Eleve`, et vérifier qu'un objet de type `Delegate` peut bien profiter des attributs et des méthodes définies pour la classe `Eleve`.

Question 12. Créer les éléments suivants pour la classe `Delegue` :

- Un attribut `listeMessages` (liste de chaîne de caractères, initialement vide)
- Une méthode `ajouterMessage` (qui ajoute un message dans cette liste)
- Une méthode `transmettreMessages`, qui affiche ces messages à l'écran (en passant à la ligne entre deux messages), et vide cette liste.