

# TD4 : Étudiants et enseignants

---

*Consignes :*

- *Un environnement devra être créé pour votre projet*
- *Toutes vos classes devront être créées dans des fichiers distincts stockés dans un dossier « Models »*
- *Vos fichiers Json seront stockés dans un dossier « data »*
- *Votre programme principal appelant vos classes devra être écrit dans un fichier script.py à la racine de votre projet*
- *N'hésitez pas tester vos méthodes dans le script principal*

## Exercice 1

- Créer une class « Student ».  
A partir de cette classe, l'utilisateur devra saisir le nom et le prénom des élèves de sa promo.
- Ajouter un attribut « rates » de type liste afin que l'utilisateur puisse saisir 5 notes par élève. Attention ses notes devront être contrôlées avant d'être ajoutées à l'attribut « rates ». Ce contrôle devra se faire impérativement dans la classe « Student »  
Contrainte : la note devra être un nombre compris entre 0 et 20
- Ecrire la méthode « average » permettant de calculer la moyenne des notes pour chaque étudiant et l'affecter à l'attribut « average\_student »
- Les enseignants décident d'octroyer des points bonus sur la moyenne de tous les étudiants : à l'aide d'une variable de classe « bonus » que vous viendrez initialiser par le biais d'une méthode de classe, vous demanderez à l'enseignant combien de points bonus il souhaite octroyer et vous l'ajouterez à votre moyenne dans la méthode « average »
- Enregistrez les informations dans un fichier « students.json »

## Exercice 2

- Créer une classe « Person » composée d’ un prénom et d’ un nom
- Apporter les modifications nécessaires à la classe « Student » afin qu’ elle hérite de la classe « Person ».
- De la même manière que la classe « Student », créer une classe « Teacher » qui héritera de la classe « Person ». Vous devrez stocker pour chaque enseignant la liste des matières qu’ il enseigne dans une liste « lst\_lessons ». Une méthode permettra d ’ ajouter une matière à cette liste. Cependant, vous devrez vérifier si cette matière existe parmi la liste « lessons » suivante :

```
lessons = ["HTML", "CSS", "Javascript", "PHP", "Symfony", "Python", "Design UX/UI", "Marketing",  
"PAO"]
```

- Donnez la possibilité de mettre à jour la liste « lessons » via une méthode de classe
- Une fois votre classe créée, permettez à l’ utilisateur de saisir les enseignants dans le script principal ainsi que leurs matières d’ enseignements tant qu’ il n’ a pas saisi « stop »
- Enregistrez les information dans un fichier « teachers.json »

## Exercice 3

- Depuis le fichier Json « students.json » et du fichier « teachers.json », créez une liste « persons » contenant tous les étudiants et tous les enseignants.
- Dans la classe « Person » ajoutez la méthode « work() » dans laquelle vous afficherez la chaine « Je travaille »
- Dans les classes « Student » et « Teacher » vous surchargerez la méthode « work() » en affichant respectivement « J’ étudie » et « J’ enseigne »
- Afin de vérifier les questions précédentes, bouclez sur la liste « persons » et affichez :

```
Bonjour je m'appelle [firstname] [lastname] et [j'étudie|j'enseigne]
```

## Exercice 4

- Créez une classe abstraite « User », cette classe contiendra les éléments suivants :

Attributs d' instance :

- username
- password

Méthodes :

- connect() => mettre « pass » pour l' instant
- disconnect() => méthode abstraite qui affiche « Déconnexion »

- Permettre à l' utilisateur de saisir le « username » et le « password » d' un étudiant ou d' un enseignant.  
Afin d' ajouter les informations « username » et « password » à la bonne personne, l' utilisateur devra indiquer le nom et le prénom de la personne
- A partir de la liste « persons » vous mettrez à jour le fichier Json correspondant (« students.json » ou « teachers.json ») à jour chaque fois que vous saisierez un « username » et un « password »
- Surchargez la fonction connect() dans les classes respectives « Student » et « Teacher »  
Demandez à l' utilisateur un nom et un prénom afin de cibler une personne. Puis vérifiez son username et son password dans la méthode connect(), si les identifiants correspondent à ceux de l' étudiant, affichez le message « étudiant connecté ». Vous ferez de même pour les enseignants en affichant le message « enseignant connecté »
- Appelez la méthode disconnect() sur ce même utilisateur

## Exercice 5

- Typez la classe « User »
- Typez la classe « Person »
- Typez la classe « Student »
- Typez la classe « Teacher »
- Masquez username et password dans la classe « User »