Exercices

Exo 1 : Classe d'objet

Partir de la classe d'objet suivante :

```
class Contact:
    def __init__(self, prenom, nom):
         self.prenom = prenom.capitalize()
        self.nom = nom.capitalize()
```

Lui ajouter les attributs suivants : age, adresse, une méthode `nomcomplet()`.

Puis, instancier des objets de cette classe pour tester.

Exo 2 : Getter et Setter

- Tester le code exemple avec les méthodes accesseur (Getter) et mutateur (Setter) d'un attribut.
- Tester le code exemple utilisant la technique du décorateur `@property`.

Exo 3 : Méthodes spéciales `__len__` et `__add__`

Définition de `__len__()` (`len()`) pour une classe `Groupe` qui contient des "membres".

```
>>> class Groupe:
...     def __init__(self, membres):
...         self.membres = membres
...
...     def __len__(self):
...         return len(self.membres)
...
>>> grp = Groupe(["john", "alain", "paul"])
>>> len(grp)
3
```

Ajouter à la classe `Groupe` la méthode `__add__`.

```
>>> class Groupe:
...     # début du code précédent, à reprendre...
...     ...
...
...
...     def __add__(self, autre):
...         new_membres = self.membres.copy()
...         new_membres.append(autre)
...         return Groupe(new_membres)
...
```

Créer une instance, par exemple avec `grp = Groupe(["olivier", "arthur", "jim"])`, puis voir ce que cela donne lorsqu'on utilise `+` pour lui ajouter un membre.

Exo 4 : Héritage

Créer la classe `Personne`, puis les classes `Etudiant` et `Professeur` qui héritent de `Personne`. Expérimenter.

Exo 5 : Mixin

Expérimenter avec une classe mixin `ExperienceProfessionnelleMixin` qui apporterait les fonctionnalités complémentaires utiles pour construire les classes `Stagiaire` et `Employe`.

Partir du code initial ci-dessous :

```
class Personne:
    def __init__(self, nom):
        self.nom = nom


class ExperienceProfessionnelleMixin:
    def parcours_pro(self):
        pass


class Stagiaire(Personne, ExperienceProfessionnelleMixin):
    def __init__(self, nom, ecole, stage, stages_precedents):
        super().__init__(nom)
        self.ecole = ecole
        self.stage = stage
        self.stages_precedents = stages_precedents


class Employe(Personne, ExperienceProfessionnelleMixin):
    def __init__(self, nom, entreprise, entreprises_precedentes):
        super().__init__(nom)
        self.entreprise = entreprise
        self.entreprises_precedentes = entreprises_precedentes
```