PYTHON № 10: INVITATION AUX CLASSES

1 une Classe, c'est quoi (vision simplifiée)?

1.1 explication

- c'est 1 type d'objet, qui a des attributs et des méthodes
- ce n'est pas obligatoire mais 3 choses sont intéressantes à faire :
 - mettre un constructeur __init__ : c'est 1 méthode particulière qui permet de créer
 1 instance de la classe
 - créer 1 méthode *del* qui permet de supprimer les instances inutiles
 - renseigner la méthode help() qui permet d'obtenir des explications sur la classe
- ensuite, on ajoute des attributs et des méthodes pour travailler
- attribut et méthode peuvent être globale (pour la classe) ou locale (pour l'instance)

1.2 exemple de classe

```
1 # création d'une classe Personne
2 class Personne:
       """Classe définissant une personne caractérisée par :
3
4
       - son nom
5
       - son prénom
6
       - son âge
7
       - son lieu de résidence"""
8
9
       def __init__(self): # Notre méthode constructeur
           """Constructeur de notre classe. Chaque attribut va être instancié
10
           avec une valeur par défaut... original"""
11
12
13
           self.nom = "Smith"
14
           self.prenom = "John" # Quelle originalité
           self.age = 28 # Cela n'engage à rien
15
           self.lieu = "London"
16
17
18 moi = Personne()
   print(moi.nom, moi.prenom, moi.age, moi.lieu)
```

pour l'instant, tous les objets crées comportent les mêmes infos :

```
Smith John 28 London
```

1.3 insertion d'information à la création

```
1 # création d'une classe Personne
2 class Personne:
       """Classe définissant une personne caractérisée par :
3
4
       - son nom
5
       - son prénom
6
       - son âge
7
       - son lieu de résidence"""
8
9
       def __init__(self, nom, age, lieu):
           """Constructeur de notre classe"""
10
11
           self.nom = nom
           self.prenom = "John"
12
           self.age = age
13
           self.lieu = lieu
14
15
16 moi = Personne("Dubert",25,"New-York")
17 print(f"Bonjour, je suis {moi.prenom} {moi.nom} et j'ai {moi.age} ans. J'↔
       habite à {moi.lieu}.")
```

en créant une personne, j'indique aussi des infos sur elle :

```
Donjour, je suis John Dubert et j'ai 25 ans. J'habite à New-York .
```

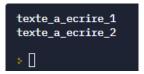
1.4 attribut pour la classe

- pour l'instant, les attributs créés étaient associées à l'instance créée
- on peut aussi créer des attributs pour la classe elle-même
- · voici par un attribut associé à la classe Compteur qui compte le nombre d'objets créés

2 exemple complet

```
class TableauNoir:
     """Classe définissant une surface sur laquelle on peut écrire,
 2
     que l'on peut lire et effacer, par jeu de méthodes. L'attribut modifié
 3
     est 'surface'"""
 4
 5
     def __init__(self):
 6
 7
       """Par défaut, notre surface est vide"""
       self.surface = ""
8
9
10
     def ecrire(self, message_a_ecrire):
       """Méthode permettant d'écrire sur la surface du tableau.
11
       Si la surface n'est pas vide, on saute une ligne avant de rajouter
12
       le message à écrire"""
13
       if self.surface != "":
14
         self.surface += "\n"
15
16
       self.surface += message_a_ecrire
17
18
     def lire(self):
       """Cette méthode se charge d'afficher, grâce à print,
19
       la surface du tableau"""
20
       print(self.surface)
21
22
23
     def effacer(self):
       """Cette méthode permet d'effacer la surface du tableau"""
24
       self.surface = ""
25
26
27 a = TableauNoir()
28 a.ecrire("texte_a_ecrire_1")
29 a.ecrire("texte_a_ecrire_2")
30 a.lire()
31 a.effacer()
32 a.lire()
```

ce qui donne :



3 exercices

- ex 1 : palindrome méthode de classe
- ex 2 : palindrome méthode d'instance
- <u>ex 3</u>: puzzle
- <u>ex 4</u>: logger
- ex 5 : héritage "simple"
- <u>ex 6</u>: puzzle
- ex 7 : héritage multiple cas réel