



FORMATION PYTHON

CLASSES ET OBJETS EN LANGAGE PYTHON

Mustapha HAIN



QU'EST-CE QU'UNE CLASSE ?



Nom: Karim
Age :22 ans
Filière: physique



Nom: Sara
Age :20 ans
Filière: physique



Nom: Laila
Age :22 ans
Filière: chimie

Etudiant



Nom
Age
Filière

Afficher()
Modifier()

1) Vision ensembliste, mathématique :

Une classe est un ensemble d'objets

2) Vision informatique :

Une classe est un type (de données)

Les deux visions se rejoignent :

Une classe est un ensemble et un type de données, dont les éléments sont appelées instances.

Les instances possèdent :

- des caractéristiques dont les types sont semblables : **Attributs**
- des comportements qui leur sont tous identiques : **Méthodes**

Point

-x

-y

void __init__ ()

void deplacer (int, int)

void afficher () ;

Classe en UML

instanc ← Point a()
e

instanc ← Point b()
e

Objets de la classe

Point

-x

-y

void __init__ ()

void deplacer (int, int)

void afficher () ;

Classe en UML

Sans valeurs d'initialisation

```
class Point:
    def __init__(self):
        self.x = 0
        self.y = 0
a = Point()
print("a : x =", a.x, "y =", a.y)

a.x = 1
a.y = 2
print("a : x =", a.x, "y =", a.y)
```

Constructeur

Code de la classe en Python

Point

-x

-y

void __init__ ()

void deplacer (int, int)

void afficher () ;

Classe en UML

Avec des valeurs d'initialisation

```
class Point:  
    def __init__(self, abs, ord):  
        self.x = abs  
        self.y = ord
```

Constructeur

```
a = Point(1, 2)  
print("a : x =", a.x, "y =", a.y)
```

Code de la classe en Python

Du UML au Langage PYTHON

Exemple complet

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y
    def deplace(self, dx, dy):
        self.x = self.x + dx
        self.y = self.y + dy
a = Point(1, 2)
b = Point(3, 4) Rachid
print("a : x =", a.x, "y =", a.y)
print("b : x =", b.x, "y =", b.y)
a.deplace(3, 5)
b.deplace(-1, -2)
print("a : x =", a.x, "y =", a.y)
print("b : x =", b.x, "y =", b.y)
```


voiture

-code

-marque

-puissance

-kilometrage

```
void mod_puiss (int) ;
```

```
void mod_kilo (int) ;
```

```
void afficher () ;
```

Implémenter la classe voiture en langage Python avec trois instances différentes?

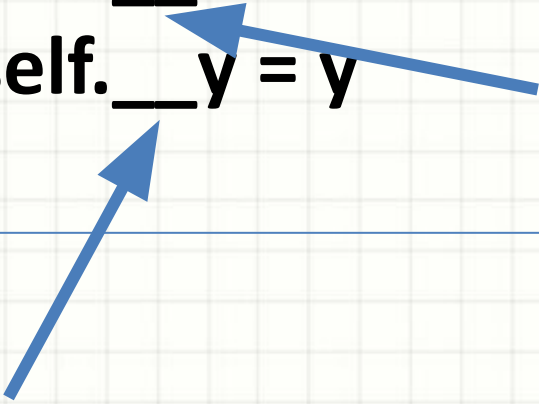
Ecrire un programme pour tester les deux instances?



Définition

On réalise la protection des attributs de notre classe Point grâce à l'utilisation d'attributs privées.

```
class Point:  
    def __init__(self, x, y):  
        self.__x = x  
        self.__y = y
```



```
class Point:
```

```
    def __init__(self, x, y):
```

```
        self.__x = x
```

```
        self.__y = y
```

```
    def deplace(self, dx, dy):
```

```
        self.__x = self.__x + dx
```

```
        self.__y = self.__y + dy
```

```
    def affiche(self):
```

```
        print("abscisse =", self.__x, "ordonnee =", self.__y)
```

```
a = Point(2, 4)
```

```
a.affiche()
```

```
a.deplace(1, 3)
```

```
a.affiche()
```

une classe –souvent comporte trois type des méthodes , à savoir :

- **Les constructeurs ;**
- **Les accesseurs (en anglais accessor) qui fournissent des informations relatives à l'état d'un objet, c'est-à-dire aux valeurs de certains de ses attributs (généralement privés) sans les modifier ;**
`get_x(self)`
`get_nom(self)`
- **Les mutateurs (en anglais mutator) qui modifient l'état d'un objet, donc les valeurs de certains de ses attributs.**
`set_y(self,y)`
`set_nom(self,v_nom)`

class Point:

def __init__(self, x, y):

self.__x = x

self.__y = y

def get_x(self):

return self.__x

def get_y(self):

return self.__y

def set_x(self, x):

self.__x = x

def set_y(self, y):

self.__y = y

a = Point(3, 7)

print("a : abscisse =", a.get_x())

print("a : ordonnee =", a.get_y())

a.set_x(6)

a.set_y(10)

print("a : abscisse =", a.get_x())

print("a : ordonnee =", a.get_y())

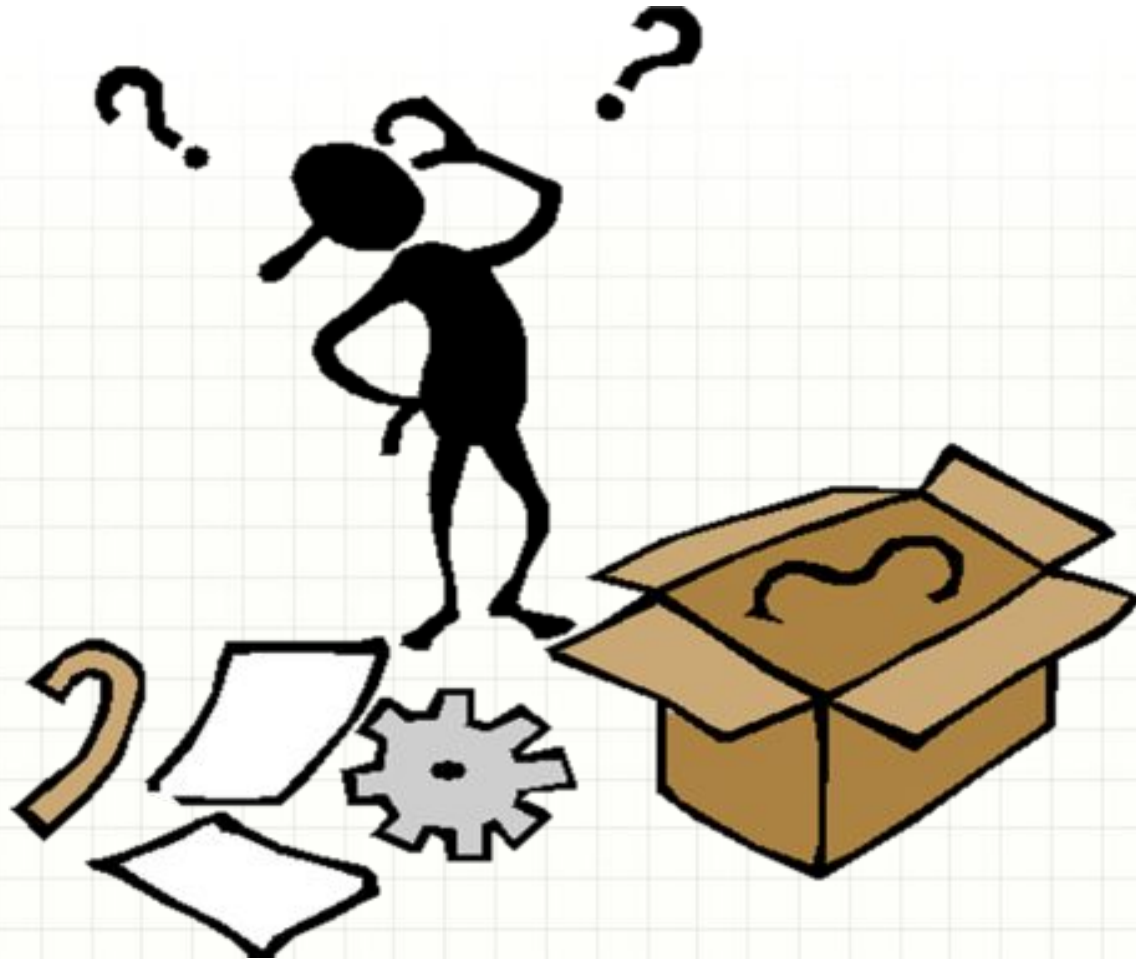
Les variables d'instance stockent des informations relatives à chaque instance alors que les variables de classe servent à stocker les attributs et méthodes communes à toutes les instances de la classe :

```
class etudiant:
    filiere = 'MP'      # class variable shared by all instances
    def __init__(self, nom):
        self.nom = nom  # instance variable unique to each instance
e1 = etudiant('Karim')
e2 = etudiant('Imane')

print(e1.filiere)
print(e2.filiere)
print(e1.nom)
print(e2.nom)
```

Résultat

```
MP
MP
Karim
Imane
```



Travaillons ensemble- Activité 1