Le concept de POO en Python

Définition

La programmation orientée objet est une approche permettant de modéliser des éléments concrets du monde réel tels que les voitures, ainsi que des relations entre des entités telles que les entreprises et les employés, les étudiants et les enseignants

CONSTRUCTEUR

La programmation orienté objet est un type de programmation basée **sur la création des classes et des objets via une méthode appelée instanciation.**

Une classe est un prototype (modèle) codé en un langage de programmation dont le but de créer des objets dotés d'un ensemble de méthodes et attributs qui caractérisent n'importe quel objet de la classe.

#Pour créer une classe en Python, on utilise l'instruction :

```
class nom_de_la_classe:
```

#On crée ensuite une méthode qui permet de construire les objets, appelé constructeur via l'instruction :

```
def __init__(self, paramètres):
```

Définition d'une fonction

initialize

Paramètres de l'objet à créer

Nom du paramètre qui correspond à l'instance depuis laquelle la méthode est appelée. Le nom de ce paramètre est une convention. On pourrait l'appeler autrement.

CONSTRUCTEUR

VARIABLE DE CLASSE

```
class Nom_de_class:
      def __init__(self, variable1, variable2):
      self.variable1 = variable1
      self.variable2 = variable2
```

VARIABLE DE CLASSE

NOMMAGE DES VARIABLES

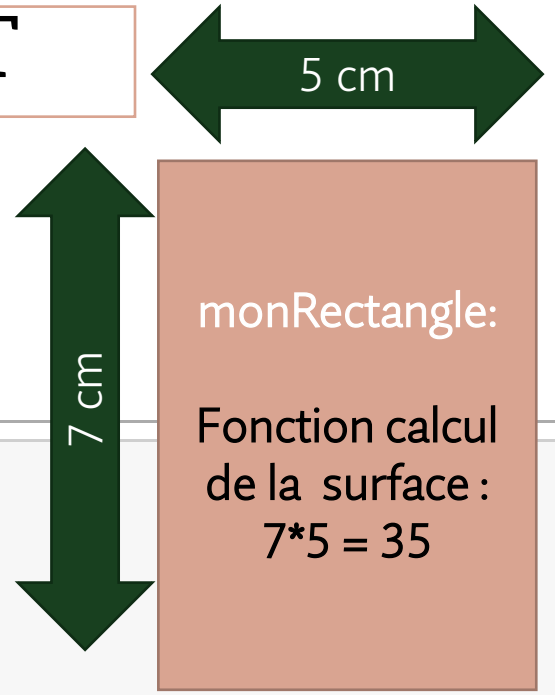
```
s1 = Nom_de_class('variable_x',variable_y)
print(s1.variable1, s1.variable2)
```

DATA

AFFICHAGE

variable_x,variable_y

Exemple



Entrée [10]:

```
class Rectangle:
    def __init__(self,L,l):
        self.Longueur=L
        self.Largeur=l

    # méthode qui calcule la surface
    def surface(self):
        return self.Longueur*self.Largeur

# création d'un rectangle de longueur 7 et de largeur 5
monRectangle = Rectangle(7,5)
print("La surface de mon rectangle est : ", monRectangle.surface())
```

```
La surface de mon rectangle est :  35
```

Rappel de la class

EXEMPLE : POO

```
def __init_ (self, wheel, color, owner=None):
    self.color = color
    self.wheel=wheel
    self.owner = owner
    self.option = False

def paint(self,color):
   self.color = color

@classmethod ➔ décorateur
def create_with_owner(cls,color,owner):
            return cls(color, owner)


def add_option(self):
   self.option = True
```

VARIABLE DE CLASSE

VARIABLE D'INSTANCE OU ARGUMENT