2N6 Programmation 2

avec python™

Classes et instanciations

Qu’est-ce que la programmation objet ?

La programmation orienté objet (OOP)

- Un principe de programmation. On regroupe les données et les fonctions de façon à ce qu'il soit facile à réutiliser et à modifier selon les besoins.
- On créé des Classes qui sont des schémas pour créer des objets.
- Dans les classes on écrit les propriétés qui vont décrire les objets futurs et les méthodes qui décrivent les actions que pourront faire ces objets.
- La OOP permet plus facilement à l'être humain de conceptualiser les programmes et leurs utilisations.

Classes

- Les Classes sont des schémas qui nous permettrons ensuite de créer des objets.

- Elles ont des propriétés et des méthodes
- Cette chaise aurait plusieurs propriétés dont:
  - "largeur", "profondeur", "hauteur"
- Elle aurait aussi des méthodes, par ex :
  - "ajuster_hauteur()"

Classes

```
class Chaise:
    def __init__(self, largeur, profondeur, hauteur):
        self.largeur = largeur
        self.profondeur = profondeur
        self.hauteur = hauteur

    def ajuster_hauteur(self, nouvelle_hauteur):
        self.hauteur = nouvelle_hauteur
```

Le mot-clef **class** signifie que nous allons faire une classe (nécessaire)

Les noms des classes commencent toutes par une majuscule par convention

La méthode "magic" ou dunder **__init__** est toujours appelée lorsqu'on crée un nouvel objet. Il s'agit du constructeur de la **classe**

```
class Chaise:
    def __init__(self, largeur, profondeur, hauteur):
        self.largeur = largeur
        self.profondeur = profondeur
        self.hauteur = hauteur

    def ajuster_hauteur(self, nouvelle_hauteur):
        self.hauteur = nouvelle_hauteur
```

Ici, toutes nos méthodes commencent par **self,** qui référencie l'objet qui est créé.

Instanciation d'objets à partir d'une classe

- On créer de nouveaux objets à partir d'une **classe** en appelant la **classe**, et en lui donnant les valeurs dont son constructeur a besoin.

```
(largeur, profondeur, hauteur) -> None
chaise_1 = Chaise(40, 40,110)
chaise_2 = Chaise(40, 40, 110)

chaise_2.ajuster_hauteur(80)
```

- Chaque objet créé ainsi est appelé une **instance** de la **classe**.

Utiliser les propriétés et méthodes d'un objet

```
chaise_1 = Chaise(40, 40, 110)
chaise_2 = Chaise(40, 40, 110)

print("hauteur 1: ", chaise_1.hauteur)
print("hauteur 2: ", chaise_2.hauteur)

chaise_2.ajuster_hauteur(80)

print("hauteur 1: ", chaise_1.hauteur)
print("hauteur 2: ", chaise_2.hauteur)
```

PROBLÈMES SORTIE CONSOLE DE DÉBOGAGE TERMINAL

```
hauteur 1:  110
hauteur 2:  110
hauteur 1:  110
hauteur 2:  80
```

Création d'objets avec des valeurs par défauts

```python
class Chaise:
    def __init__(self, largeur = 40, profondeur = 40, hauteur = 110):
        self.largeur = largeur
        self.profondeur = profondeur
        self.hauteur = hauteur
```

```python
chaise_3 = Chaise()
chaise_4 = Chaise(35, 45, 95)

print("chaise 3 :")
print(chaise_3.largeur, chaise_3.profondeur, chaise_3.hauteur)

print("chaise 4 :")
print(chaise_4.largeur, chaise_4.profondeur, chaise_4.hauteur)
```

PROBLÈMES SORTIE TERMINAL

```
chaise 3 :
40 40 110
chaise 4 :
35 45 95
```

Modélisation UML

- Le "Unified Modeling Language"(UML) est utilisé pour faire les schémas des classes.