CM7_jupyter

March 15, 2021

Attributs de classe et méthodes de classe

[56]:
```
class A(object):
    nb = 0

    def __init__(self, x):
        print("creation objet de type A")
        self.x = x
        A.nb = A.nb + 1

print("A : nb = ", A.nb)
print("Partie 1")
a = A(3)
print("A : nb = ", A.nb)
print("a : x = ", a.x, " nb = ", a.nb)
print("Partie 2")
b = A(6)
print("A : nb = ", A.nb)
print("a : x = ", a.x, " nb = ", a.nb)
print("b : x = ", b.x, " nb = ", b.nb)
c = A(8)
print("Partie 3")
print("A : nb = ", A.nb)
print("a : x = ", a.x, " nb = ", a.nb)
print("b : x = ", b.x, " nb = ", b.nb)
print("c : x = ", c.x, " nb = ", c.nb)
```

```
A : nb =  0
Partie 1
creation objet de type A
A : nb =  1
a : x =  3  nb =  1
Partie 2
creation objet de type A
A : nb =  2
a : x =  3  nb =  2
b : x =  6  nb =  2
creation objet de type A
Partie 3
```

```
A : nb =  3
a : x =  3  nb =  3
b : x =  6  nb =  3
c : x =  8  nb =  3
```

[57]:
```
class A(object):
    nb = 0

    def __init__(self):
        print("creation objet de type A")
        A.nb = A.nb + 1
        print("il y en a maintenant ", A.nb)

    @classmethod
    def get_nb(cls):
        return A.nb

print("Partie 1 : nb objets = ", A.get_nb())
a = A()
print("Partie 2 : nb objets = ", A.get_nb())
b = A()
print("Partie 3 : nb objets = ", A.get_nb())
```

```
Partie 1 : nb objets =  0
creation objet de type A
il y en a maintenant  1
Partie 2 : nb objets =  1
creation objet de type A
il y en a maintenant  2
Partie 3 : nb objets =  2
```

Les fichiers

[58]:
```
f = open("lesTitres.txt","r")
#lecture
s = f.read()
#affichage
print("** contenu de s **")
print(s)
print("** fin contenu **")
#information sur s
print("type de s : ",type(s))
print("longueur de s : ", len(s))
#fermeture
f.close()
```

```
** contenu de s **
Les fleurs du mal
Les misérables
```

```
Le rouge et le noir
Vipère au poing
Des souris et des hommes

** fin contenu **
type de s :  <class 'str'>
longueur de s :  94
```

[3]:
```
#ouverture en lecture
f = open("lesTitres.txt","r")
#lecture
lst = f.readlines()
#affichage
print("** contenu de lst **")
print(lst)
print("** fin contenu **")
#information sur lst
print("type de s : ",type(lst))
print("longueur de s : ", len(lst))

#fermeture
f.close()
```

```
** contenu de lst **
['Les fleurs du mal\n', 'Les misérables\n', 'Le rouge et le noir\n', 'Vipère aux
poings\n', 'Des souris et des hommes\n']
** fin contenu **
type de s :  <class 'list'>
longueur de s :  5
```

[4]:
```
#ouverture en lecture
f = open("lesTitres.txt","r")
#lecture ligne itérativement
while True:
    s = f.readline()
    if (s != ""):
        print(s)
    else:
        break;
#fermeture
f.close()
```

```
Les fleurs du mal

Les misérables

Le rouge et le noir

Vipère aux poings
```

```
Des souris et des hommes
```

[5]:
```
#ouverture en lecture
f = open("lesTitres.txt","r")
#lecture ligne itérativement
for s in f:
    print(s,len(s))

#fermeture
f.close()
```

```
Les fleurs du mal
 18
Les misérables
 15
Le rouge et le noir
 20
Vipère aux poings
 18
Des souris et des hommes
 25
```

[8]:
```
!more lesTitres.txt  #commande unix
```

```
Les fleurs du mal
Les misérables
Le rouge et le noir
Vipère aux poings
Des souris et des hommes
```

[13]:
```
#ouverture en écriture
f = open("moto.txt","w")
#écriture
f.write("honda\n")
f.write("yamaha\n")
f.write("ducati\n")
#fermeture
f.close()
```

[15]:
```
!more moto.txt
```

```
honda
yamaha
ducati
```

[16]:
```
#ouverture en écriture
f = open("moto.txt","w")
```

```
#liste
lst = ["harley\n","yamaha\n","ducati"]
#écriture
f.writelines(lst)
#fermeture
f.close()
```

[17]:
```
!more moto.txt
```

```
harley
yamaha
ducati
```

[19]:
```
#ouverture en ajout
f = open("moto.txt","a")
#ajouter un saut de ligne
f.write("\n")
#écriture
f.write("laverda")

#fermeture
f.close()
```

[20]:
```
!more moto.txt
```

```
harley
yamaha
ducati
laverda
```

Fichier au format CSV

[59]:
```
!more lesLivres.csv
```

```
titre;auteur;date
Les fleurs du mal;Charles Baudelaire;1857
Les misérables;Victor Hugo;1862
Le rouge et le noir; Stendahl; 1830
Vipère au poing; Hervé Bazin;1948
Des souris et des hommes; John Steinbeck; 1937
```

[30]:
```
#ouverture en lecture
f = open("lesLivres.csv","r")

#importation du module csv
import csv
#lecture - utilisation du parseur csv
lecteur = csv.reader(f,delimiter=";")
#affichage - itération sur chaque ligne
```

```
for ligne in lecteur:
    print(ligne)

#fermeture du fichier
f.close()
```

```
['titre', 'auteur', 'date']
['Les fleurs du mal', 'Charles Baudelaire', '1857']
['Les misérables', 'Victor Hugo', '1862']
['Le rouge et le noir', ' Stendahl', ' 1830']
['Vipère au poing', ' Hervé Bazin', '1948']
['Des souris et des hommes', ' John Steinbeck', ' 1937']
```

[34]:
```
#ouverture en lecture
f = open("lesLivres.csv","r")
#importation du module csv
import csv
#lecture
lecteur = csv.DictReader(f,delimiter=";")
#affichage
for ligne in lecteur:
    print(ligne['titre'])
    print(ligne['auteur'])
    print(ligne['date'])
    print(type(ligne))
#fermeture
f.close()
```

```
Les fleurs du mal
Charles Baudelaire
1857
<class 'collections.OrderedDict'>
Les misérables
Victor Hugo
1862
<class 'collections.OrderedDict'>
Le rouge et le noir
 Stendahl
 1830
<class 'collections.OrderedDict'>
Vipère au poing
 Hervé Bazin
1948
<class 'collections.OrderedDict'>
Des souris et des hommes
 John Steinbeck
 1937
<class 'collections.OrderedDict'>
```

[ ]:
```
#format json et stockage d'instance
```

[36]:
```
class Livre(object):
    def __init__(self, titre, auteur,date):
        self.titre=titre
        self.auteur=auteur
        self.date=date
    def gettitre(self):
        return self.titre
    def getdate(self):
        return self.date
    def settitre(self, titre):
        self.titre=titre
    def setdate(self, date):
        self.date=date
    def __str__(self):
         return self.titre + "["+ self.auteur + ',' +str(self.date)+"]"

leLivre=Livre('title1','auteur1', 2019)
print(leLivre)
```

```
title1[auteur1,2019]
```

[38]:
```
import json

#sauvegarde
f = open("unLivre.json","w")
#dictionnaire
l=leLivre
d = {"Titre":l.titre,"Auteur":l.auteur,"Date":l.date}

#sauver au format json

json.dump(d,f)

#fermer le fichier

f.close();
```

[40]:
```
!more unLivre.json
```

```
{"Titre": "title1", "Auteur": "auteur1", "Date": 2019}
```

[41]:
```
import json

#sauvegarde
f = open("unLivre.json","w")
#dictionnaire
```

```
l=leLivre

#sauver au format json
json.dump(l.__dict__,f) #!!!
#fermer le fichier
f.close();
```

[42]:
```
!more unLivre.json
```

```
{"titre": "title1", "auteur": "auteur1", "date": 2019}
```

[43]:
```
import json
#ouverture fichier
f = open("unLivre.json","r")
#chargement
d = json.load(f)
print(d)
print(type(d))
#transf. en livre
lnew=Livre(d["titre"],d["auteur"], d["date"])
#affichage
print(lnew)
#fermeture
f.close();
```

```
{'titre': 'title1', 'auteur': 'auteur1', 'date': 2019}
<class 'dict'>
title1[auteur1,2019]
```

Rajouter en méthode dans la classe Livre

[73]:
```
import json
class Livre(object):
    def __init__(self, titre, auteur,date):
        self.titre=titre
        self.auteur=auteur
        self.date=date
    def toFile(self, fich):
        f = open(fich,"w")
        l=self
        json.dump(l.__dict__,f)
        f.close()

    @classmethod
    def fromFile(cls, fich):
        f = open(fich,"r")
        d = json.load(f)
        lnew=Livre(d["titre"],d["auteur"], d["date"])
        f.close()
```

```
        return lnew

    def __str__(self):
         return self.titre + "["+ self.auteur + ',' +str(self.date)+"]"


leLivre=Livre('title1','auteur1', 2019)
leLivre.toFile("unLivre.json")
print(leLivre)
l=Livre.fromFile("unLivre.json")
print(l)
l.date=2020
l.toFile("unLivre.json")
print(l)
```

```
title1[auteur1,2019]
title1[auteur1,2019]
title1[auteur1,2020]
```

fichier json et plusieurs instances de classe

[50]:
```
import json
#liste vide
liste = []
#nb. de pers ?
n =5
#saisie liste
for i in range(0,n):
    l=Livre('title1','auteur1', 2019)
    liste.append(l)
#sauvegarde
f = open("unsLivres.json","w")
#créer une liste temporaire
tmp = []
#pour chaque livre
for p in liste:
#créer un dictionnaire
    d = {}
    d["titre"] = p.titre
    d["auteur"] = p.auteur
    d["date"] = p.date
#ajouter dans liste tmp
    tmp.append(d)
    #sauvegarde de la liste tmp
json.dump(tmp,f)
#fermer le fichier
f.close();
```

[55]:
```
import json

#ouverture fichier
f = open("lesLivres.json","r")
#chargement
tmp = json.load(f)
#conv. en liste de livres
liste = []
for d in tmp:
#créer un livre
    l=Livre(d["titre"],d["auteur"], d["date"])
    #l'ajouter dans la liste
    liste.append(l)

print("Nb livre : ",len(liste))

#fermeture
f.close();
for l in liste:
    print(l)
```

```
Nb livre :  5
Les fleurs du mal[ Charles Baudelaire,1857]
Les misérables[Victor Hugo,1862]
Le rouge et le noir[Stendahl,1830]
Vipère au poing[Hervé Bazin,1948]
Des souris et des hommes[John Steinbeck,1937]
```

[ ]:
```
#Rajouter en méthode dans la classe Librarie
```

[78]:
```
import json
class Librairie(object):
    def __init__(self):
        self.lesLivres=[]
    def ajout(self, livre):
        self.lesLivres.append(livre)
    def __str__(self):
        chaine=str(self.lesLivres[0])
        for e in self.lesLivres[1:]:
            chaine=chaine+ "," + str(e)
        return chaine

    @classmethod
    def fromFile(cls,fich):
        f = open(fich,"r")
        #chargement
        tmp = json.load(f)
```

```
        liste = []
        for d in tmp:
            #créer un livre
            l=Livre(d["titre"],d["auteur"], d["date"])
            #l'ajouter dans la liste
            liste.append(l)
        lib=Librairie()
        lib.lesLivres=liste
        f.close();
        return lib

    def toFile(self,fich):
        f = open(fich,"w")
        tmp = []
        for l in self.lesLivres:
        #créer un dictionnaire
            d = {}
            d["titre"] = l.titre
            d["auteur"] = l.auteur
            d["date"] = l.date
            tmp.append(d)
        json.dump(tmp,f)
        f.close();

libN=Librairie.fromFile("lesLivres.json")
print(libN)
lnew=Livre('titre0','auteur0',2020)
libN.ajout(lnew)
libN.toFile("lesLivresBis.json")
```

```
Les fleurs du mal[ Charles Baudelaire,1857],Les misérables[Victor Hugo,1862],Le rouge et le noir[Stendahl,1830],Vipère au poing[Hervé Bazin,1948],Des souris et des hommes[John Steinbeck,1937]
```

Outillage: gestion des exceptions

[79]:
```
try:
    a = int(input("Tell me one number: "))
    b = int(input("Tell me another number: "))
    print("a/b = ", a/b)
except:
    print("Bug in user input.")
```

```
Tell me one number: 3
Tell me another number: 0
Bug in user input.
```

[80]:
```
try:
    a = int(input("Tell me one number: "))
    b = int(input("Tell me another number: "))
    print("a/b = ", a/b)
    print("a+b = ", a+b)
except ValueError:
    print("Could not convert to a number.")
except ZeroDivisionError:
    print("Can't divide by zero")
except:
    print("Something went very wrong.")
```

```
Tell me one number: t
Could not convert to a number.
```

[81]:
```
def get_ratios(L1, L2):
    """ Assumes: L1 and L2 are lists of equal length of numbers
        Returns: a list containing L1[i]/L2[i] """
    ratios = []
    for index in range(len(L1)):
        try:
            ratios.append(L1[index]/L2[index])
        except ZeroDivisionError:
            ratios.append(float('nan')) #nan = Not a Number
        except:
            raise ValueError('get_ratios called with bad arg')
        else:
            print("success")
        finally:
            print("executed no matter what!")
    return ratios

print(get_ratios([1, 4], [2, 4]))
```

```
success
executed no matter what!
success
executed no matter what!
[0.5, 1.0]
```

[ ]:
```

```