Programmation PYTHON
Cours 2

Nassim ZELLAL

2020/2021

Types d’objets en Python - suite

- Listes - classe « list »
- Tuples - classe « tuple »
- Dictionnaires - classe « dict »

Les dictionnaires

Pour instancier un dictionnaire :

- dico = dict()#création d'un dictionnaire vide
- print(type(dico))
- ><class 'dict'>
- #----------#
- dico = {}#création d'un dictionnaire vide
- print(type(dico))
- ><class 'dict'>

Les dictionnaires

- Ajout de clés et de valeurs au dictionnaire :
- dico = {}
- dico["arbre"] = "vert" # [clé] = valeur
- dico["plâtre"] = "blanc"
- dico["pneu"] = 5
- Pour accéder à un élément dans un dictionnaire :
- print(dico["pneu"])
- > 5
- Pour imprimer tout le dictionnaire :
- print(dico)
- > {'plâtre': 'blanc', 'pneu': 5, 'arbre': 'vert'}
- On peut aussi écrire :
- a={'plâtre': 'blanc', 'pneu': 5, 4: 'vert'}
- print(a['plâtre'])
- print(a[4])
- > blanc
- > vert

Les dictionnaires

- dic={"cours2":18,"cours1":15,"cours3":11}
- Pour imprimer les clés du dictionnaire :

- print(dic.keys(),"sont des clés")
- #ou bien print(str(dic.keys())+" sont des clés")
- **> dict_keys(['cours2', 'cours1', 'cours3']) sont des clés**

- Pour imprimer les valeurs du dictionnaire :

- print(dic.values(),"sont des valeurs")
- #ou bien print(str(dic.values())+" sont des valeurs")
- **> dict_values([18, 15,11]) sont des valeurs**

L’opérateur « del »

- Del : supprime un élément d’un dictionnaire.

```
dic={"cours2":18,"cours1":15,"cours3":11}
del dic["cours2"]
print(dic)
>{"cours1":15, "cours3":11}
```

Opérateurs arithmétiques

- **a = (1 + 2) * 3 - 4 # retourne 5 dans a (règle de priorité)**
- **a = 1 + 2 * 3 - 4 # retourne 3 dans a (règle de priorité)**
- **a= 8 - 4 - 2 #retourne 2 et non 6 (règle d'associativité)**
- **a = 7 / 8 # Divise 7 par 8 et met 0.875 dans a**
- **a = 9 ** 2 # 9 à la puissance 10**
- **a = 5 % 2 # le reste de 5 divisé par 2 (modulo)**

Opérateurs de chaînes

- a="clavier"
- print(a[2:7])# opérateur **: (slice/tranche)**
- > avier
- alpha = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
- print(alpha[4:16])
- > EFGHIJKLMNOP
- print("my"**+**" "**+**"dog") #concaténation
- > my dog
- print("table"*****5) #répétition (multiplication de chaîne de caractères)
- > tabletabletabletabletable
- print("ta" in "table")# renvoie True si la sous-chaîne "ta" est contenue dans la chaîne "table"
- > True

Opérateurs de comparaison - nombres

- a == b # égalité
- a != b # différence
- a > b # supériorité stricte
- a < b # infériorité stricte
- a >= b # supériorité
- a <= b # infériorité

Opérateurs de comparaison - chaînes de caractères

- a == b # égalité
- a != b # différence
- a > b # supériorité stricte
- a < b # infériorité stricte
- a >= b # supériorité
- a <= b # infériorité

Opérateurs d'incrémentation et de décrémentation

- Incrémentation :
- a=6
- **a+=1 # ou bien a=a+1**
- **> 7**
- Décrémentation :
- a=6
- **a-=1 #ou bien a=a-1**
- **> 5**

Opérateurs logiques - OU - or

OR: **or**

```
x=3
y=5
if(x==3)or(y==5):
    print("ok")
if(x==3 or y==5):
    print("ok")
if x==3 or y==5:
    print("ok")
```

Opérateurs logiques - ET - and

AND: **and**

```
x=3
if(x==3)and(x<4):
    print("ok")
if(x==3 and x<4):
    print("ok")
if x==3 and x<4:
    print("ok")
```

Opérateurs logiques - PAS - not

NOT

```
x=2
if not(x==3):
    print("ok")
```

Structures de contrôle - conditions - if/else

```
x=2
y=8
if x == y :
    print ("x and y are equals\n")
else:
    print("x and y are different\n")
```

Remarque : Les deux points : terminent une condition.

Structures de contrôle - conditions - elif

```
x=3
y=4
if x == y :
    print("x and y are equals\n")
elif x == 3:
    print("yes")
else:
    print("x and y are different\n")
```

Structures de contrôle - conditions – if/else

```
x="dog"
y="cat"
if x == y :
    print("x and y are equals\n")
else:
    print("x and y are different\n")
```

Structures de contrôle - conditions – if/else

```
x="d"
y="c"
if x > y :
    print("x comes alphabetically after y")
else:
    print("x comes alphabetically before y")
```

Structures de contrôle - boucles - while

```
i=1
nb=7
while i <= 10:
    print(i, "*", nb, "=",i * nb)
    i += 1 #ou bien i=i+1
```

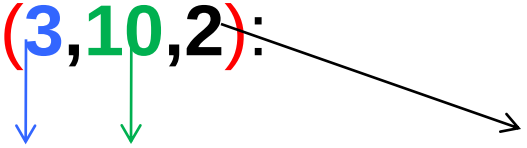
Que va afficher ce script ?

Structures de contrôle - boucles - for

- #Dans l'exemple ci-dessous, la fonction range() génère une liste [ ] de nombres, allant de 0 à 9 (10 itérations). Par défaut, on commence à 0.

```
for x in range(10): # ou bien range(0,10)
    print(x)
#Si on veut que la liste aille jusqu'à 10, faire
    range(10+1) ou bien range(11)
#---------range() avec trois paramètres----------#

for x in range(3,10,2):
    print(x)
#Début de liste, fin de liste, incrément.
```



Structures de contrôle - boucles - for - parcourir une liste

```
tab=[2,6,1]
for x in tab: #ou for x in range(len(tab)):
    print(x) #ou       print(tab[x])
#-----------------------------------------------------#
tab=['yellow','black','blue']
for x in range(len(tab)): #ou for x in tab:
    print(tab[x]) #ou                  print(x)
#-----------------------------------------------------#
tab=[('yellow',3),('black',8),('blue',7)] # 3 tuples
for x,y in tab:
    print(x,y)
```

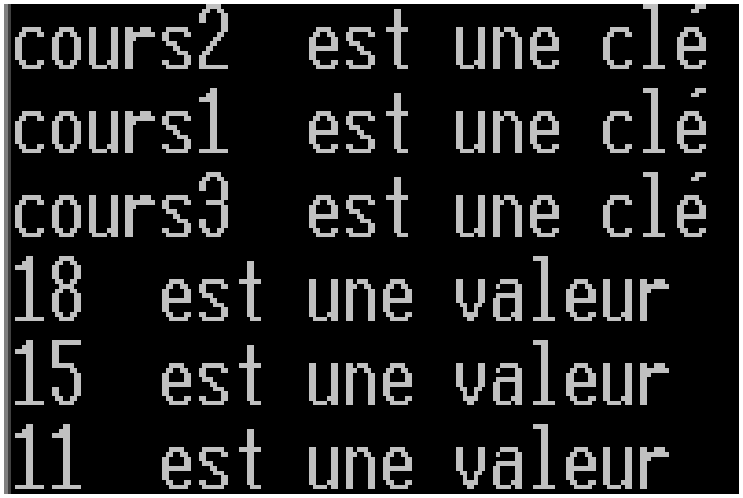
Structures de contrôle - boucles - for - parcourir une liste

```
a="Bonjour j'aime l'USTHB"
mots=a.split()
for x in mots:
    print(x)
#-----------------------Ou bien-----------------------------#
a="Bonjour j'aime l'USTHB"
mots=a.split()
for x in range(len(mots)):
    print(mots[x])
    #la fonction range() crée une suite de 0 à 2, en s'appuyant sur la longueur de « mots » retournée par la fonction len()
```

Structures de contrôle - boucles - for - parcourir un dictionnaire

- #-------Afficher les clés----------#
- dic={"cours2":18,"cours1":15,"cours3":11}
- for i in dic.keys(): # ou for i in dic:
- print(i," est une clé") # ou **+** au lieu de **,**
- #-------Afficher les valeurs----------#
- dic={"cours2":18,"cours1":15,"cours3":11}
- for i in dic.values():
- print(i," est une valeur") # ou **+** avec la fonction str() au lieu de **,**

Structures de contrôle - boucles - for - parcourir un dictionnaire



Structures de contrôle imbriquées

```
animal="cat"
num=['i','ii','iii','iv','v']
if animal == "cat":
    for x in num:
        print("We hate cat","     ",x,"\n")
else:
    print("We prefer dog")
#Ou bien + au lieu de , dans le print()
```

Exercise 1

- tab=['yellow','black','blue']

- #Imprimez toutes les couleurs, à l’exception du noir, en utilisant une boucle et une condition dans une structure imbriquée.

Exercise 2

- Écrire un script qui imprime tous les chiffres romains, à l'exception du **v** et du **i**, en utilisant une boucle et une condition dans une structure imbriquée.

- tab=['i','ii','iii','iv','v']

Exercise 3

- Afficher le type de chaque clé du dictionnaire suivant :
- a={'plâtre': 'blanc', 'pneu': 5, 4: 'vert'}

Exercise 4

- Écrire différemment la tranche (slice) suivante, en proposant deux solutions :
- a="clavier"
- print(a**[2:7]**)

Mon courriel

zellal.nassim@gmail.com