Découvrir et approfondir python

Tuples, tables de hash et dictionnaires :

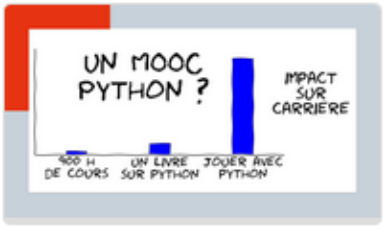
V W4-01 Fonction et portée des variables.mp4

V W4-04 Portée des variables règle LEGB.mp4

V W4-06 Passage d'arguments et appel de fonctions.mp4

Vidéos proposées par l'INRIA sur FUN MOOC

Python 3 : des fondamentaux aux concepts avancés du langage



Thierry PARMENTELAT et Arnaud Legoud

Pour bien comprendre et approfondir :

a) Indiquer à quel numéro correspond :

O Le paramètre :

O Le doc string :

O L'argument :

```
def maFonction(a):
    '''
    Commentaire expliquant les spécifications
    de la fonction
    '''
    return a**2

resultat = maFonction(10)
```

(1) (2) (3)

b) On a le code ci-contre, qu'elle est le résultat sur la console ?

O
```
2
4
16
```

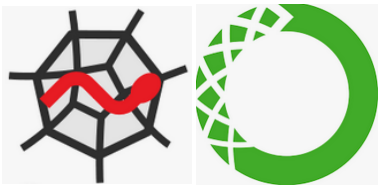
```
def maFonction(a):
    '''
    Commentaire expliquant les spécifications
    de la fonction
    '''
    print(a)
    return a**2

valeur = 2
resultat = maFonction(valeur)
print(maFonction(resultat))
```

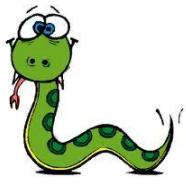
c) Une fonction Python :

O doit contenir une instruction *return*

O peut contenir 0 ou 1 instruction *return*

O peut contenir 0 ou n instructions *return*





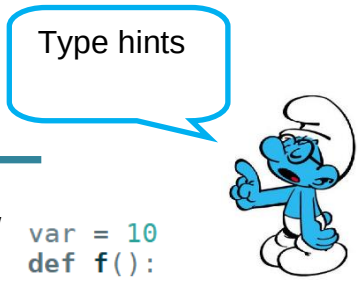

d) Est-il possible pour une fonction de modifier l'objet qui lui est passé en argument ?

O Oui, on peut modifier tous les objets **O** Oui, si l'objet est mutable

O Non, une fonction travaille sur une copie et elle ne peut modifier l'objet dans l'espace de l'appelant.

e) Cocher les déclarations de fonction correctes en python 3.6 :

O def fact(n) : **O** def fact(n : int) **O** def fact(n : int) -> int :



Règle LEGB : variable définie : Localement / Englobante / Globalement / Builtin

```
var = 10
def f():
    var = 20
    def g():
        return var
    return g()
print(f())
```

f) Que va afficher ce code ?

O 10 **O** None **O** 20 **O** Une exception

---

g) Cochez la bonne réponse :

```
x = 0
def f():
    print(x)
```

Ici x est :
O local
O global
O exception

```
x = 0
def f():
    x = 3
    print(x)
```

Ici x est :
O local
O global
O exception

```
x = 0
def f():
    print(x)
    x = 3
```

Ici x est :
O local
O global
O exception

```
x = 0
def f():
    x = 3
    print(x)
    x = 4
```

Ici x est :
O local
O global
O exception

---

h) Quelles sont les manières correctes de déclarer un argument var par défaut dans une fonction.

O
```
def f(a, b, var == 10):
    print(a, b, var)
```

O
```
def f(a, b, var=10):
    print(a, b, var)
```

O
```
def f(a, var=10, b):
    print(a, b, var)
```

O
```
def f(a, var=10, b=30):
    print(a, b, var)
```

O
```
def f(a, b, var, var=10):
    print(a, b, var)
```