

Exercices sur Python n°1

EXERCICE 1 : Dans chacun des cas, dire quelle est la valeur des variables après ces suites d'instructions.

```
a = 5
a = 7
a = a + 2
```

```
b = 3
a = b + 1
b = 6
```

```
b = 3
a = b + 1
b = a + 1
```

```
a = 7
b = 2
a = a % b
```

```
b = 29
a = 37*b + 2
a = a // b
```

EXERCICE 2 : Laquelle des deux fonctions ci-dessous permet d'exécuter l'instruction suivante : `f(f(2))` ?

```
def f(x):
    return 3*x - 1
```

```
def f(x):
    print(3*x - 1)
```

EXERCICE 3 : Indiquer parmi les instructions ci-dessous celle qui correspond à celle ci-contre, quelle que soit la fonction `f`.

```
>>> a = 2
>>> f(a+4)
```

```
>>> f(2)
```

```
>>> f(4)
```

```
>>> f(6)
```

```
>>> f(2) + f(4)
```

EXERCICE 4 : Pour chacune de ces trois fonctions, indiquer le résultat obtenu en donnant **True** et **False** en argument.

```
def oui_ou_non1(b):
    if b:
        return "oui"
    else:
        return "non"
```

```
def oui_ou_non2(b):
    if b:
        reponse = "oui"
        reponse = "non"
    return reponse
```

```
def oui_ou_non3(b):
    if b:
        reponse = "oui"
    else:
        reponse = "non"
    return reponse
```

EXERCICE 5 : Donner le résultat d'un appel avec `20` comme argument pour chacune des fonctions suivantes.

```
def g1(n):
    if n < 100:
        n = 200
    else:
        n = 50
    return n
```

```
def g2(n):
    if n < 100:
        n = 200
    if n >= 100:
        n = 50
    return n
```

```
def g3(n):
    if n < 100:
        n = 200
    elif n >= 100:
        n = 50
    return n
```

EXERCICE 6 : Associer chaque fonction au résultat obtenu avec l'instruction `triangle(3)`

```
def triangle(n):
    lgn = ""
    for i in range(n):
        lgn = lgn + "*"
    print(lgn)
```

```
***
```

```
def triangle(n):
    lgn = ""
    for i in range(n):
        print(lgn)
        lgn = lgn + "*"
```

```
*
**
***
```

```
def triangle(n):
    lgn = ""
    for i in range(n):
        lgn = lgn + "*"
    print(lgn)
```

```
*
**
```

EXERCICE 7 : Associer chacune des fonctions au résultat obtenu avec l'instruction `compter(3)`. Il y a plusieurs résultats qui ne correspondent à aucune des deux fonctions.

```
def compter(n):
    for i in range(n):
        print(i)
```

```
def compter(n):
    for i in range(n):
        return i
```

0

0
1
2

2

1
2
3

3

EXERCICE 8 : Pour chacune des fonctions ci-dessous, indiquer ce qui sera affiché lors d'un appel avec l'argument "avion".

```
def mystere1(mot):
    res = ""
    for lettre in mot:
        res = res + lettre
    print(res)
```

```
def mystere2(mot):
    res = ""
    for lettre in mot:
        res = lettre + res
    print(res)
```

EXERCICE 9 : On considère la fonction ci-dessous.

```
1 def mystere3(n):
2     res = 0
3     for i in range(n):
4         if i%2 == 0:
5             res = 2 * res
6         else:
7             res = res + i
8     return res
```

i	i%2	i%2==0	res
			0

- 1) On considère l'appel `mystere3(6)`. Remplir le tableau ci-contre. La première ligne correspond à l'état de la mémoire après l'exécution de la ligne 2. Les lignes suivantes correspondent à l'état de la mémoire après la ligne 7.
- 2) Quelle est la valeur renvoyée après cet appel?

EXERCICE 10 : On considère la fonction ci-dessous.

```
1 def mystere4(mot):
2     res = ""
3     c = 0
4     for lettre in mot:
5         if c == 0:
6             res = res + lettre
7             c = 1 - c
8     return res
```

ligne	lettre	c==0	res	c
2			""	0
7	"a"	True		
7	"v"			
7	"i"			
7	"o"			
7	"n"			

- 1) On appelle cette fonction avec l'argument "avion". Compléter le tableau ci-dessus en donnant les valeurs des variables après l'exécution de la ligne indiquée. Par contre, la colonne `c==0` correspond à la valeur de cette expression avant l'exécution de la ligne 7.
- 2) Déterminer la valeur renvoyée par cet appel.
- 3) Expliquer ce que fait cette fonction.
- 4) Quelle est la valeur de `mystere4("bonjour")`?