

# Python, 2nde, Exercices, Fiche 1

---

## Rappels calculs élémentaires en Python

```
1 >>> 4*5 # multiplication
2 20
3 >>> 8/2 # division décimale
4 4.0
5 >>> 9//2 # quotient de la division euclidienne
6 4
7 >>> 9%2 # reste de la division euclidienne
8 1
9 >>> 5**2 # puissance
10 25
```

## Exercice 1

On considère les fonctions python suivantes :

```
1 def fonction1(nombre):
2     return nombre*3
3
4 def fonction2(nombre):
5     return nombre**2
6
7 def fonction3(nombre):
8     return nombre/2
9
10 def fonction4(nombre):
11     return nombre//2
12
13 def fonction5(nombre):
14     return nombre%2
```

Que renvoient les commandes suivantes ? Vérifier vos réponses en exécutant les commandes après avoir enregistré les fonctions.

```
1 >>> fonction1(7)
2 ...
3 >>> fonction2(7)
4 ...
5 >>> fonction3(7)
6 ...
7 >>> fonction4(7)
8 ...
9 >>> fonction5(7)
10 ...
```

## Les boucles for ... in range .....

On considère les fonctions suivantes :

```
1 def boucle1():
2     for i in range(5):
3         print(i)
4
5 def boucle2():
6     for i in range(3):
7         print(i)
8
9 def boucle3():
10    for i in range(1,3):
11        print(i)
12
13 def boucle4():
14    for i in range(2,8):
15        print(i)
16
17 def boucle5():
18    for i in range(3,10,2):
19        print(i)
20
21 def boucle6():
22    for i in range(2,7,3):
23        print(i)
```

Taper et exécuter vos fonctions. Tester les.

Ecrire les fonctions boucle7(), boucle8() et boucle9() de façon à avoir les résultats suivants après leur exécution.

```
1 >>> boucle7()
2 0
3 1
```

```
4 | 2
5 | 3
6 | 4
7 | 5
8 | 6
9 | >>> boucle8()
10 | 10
11 | 11
12 | 12
13 | 13
14 | 14
15 | >>> boucle9()
16 | 20
17 | 23
18 | 26
```

## Les tests : If ..... then .....else

On considère les fonctions suivantes :

```
1 | def test1(x):
2 |     if x==4:
3 |         return True
4 |     else :
5 |         return False
6 |
7 | def test2(x):
8 |     if x==8:
9 |         return True
10 |    else :
11 |        return False
12 |
13 | def test3(a):
14 |     b=7
15 |     if a==b:
16 |         return True
17 |     else :
18 |         return False
19 |
20 | def test4(x):
21 |     if x!=8:
22 |         return True
23 |     else :
24 |         return False
25 |
26 | def test5(x):
27 |     if x*2==8:
28 |         return True
29 |     else :
```

que vont retourner les commandes suivantes ?

```
1 >>> test1(4)
2 ...
3 >>> test1(8)
4 ...
5 >>> test2(4)
6 ...
7 >>> test2(8)
8 ...
9 >>> test3(2)
10 ...
11 >>> test3("b")
12 ...
13 >>> test3(7)
14 ...
15 >>> test4(8)
16 ...
17 >>> test4(9)
18 ...
19 >>> test5(8)
20 ...
21 >>> test5(4)
22 ...
```

Ecrire une fonction `pair()` et une fonction `impair()` qui permettent de déterminer si un nombre est pair ou impair.

Exemple d'exécution :

```
1 >>> pair(8)
2 True
3 >>> pair(5)
4 False
5 >>> impair(8)
6 False
7 >>> impair(5)
8 True
```

Ecrire une fonction `multiple_de_3()` qui permet de déterminer si un nombre est un multiple de 3:

```
1 >>> multiple_de_3(15)
2 True
3 >>> multiple_de_3(10)
4 False
```

