
Exercice 1 :

On se donne la fonction suivante :

```
def echanger(a,b):
    z=a
    a=b
    b=z
a=5
b=3
echanger(a,b)
print ("a vaut : "+str(a))
print ("b vaut : "+ str(b))
```

(1) À quoi sert cette fonction ?

(2) Tester la fonction, que remarquez-vous ?

(3) Corriger ce programme pour le faire fonctionner.

Exercice 2 :

On se donne la fonction suivante :

```
def echanger2(l):
    z=l[0]
    l[0]=l[1]
    l[1]=z
l = [1,2]
echanger2(l)
print ("l vaut : " + str(l))
```

(1) À quoi sert cette fonction ?

(2) Tester ce programme.

(3) Quelle est la différence avec le programme précédent ?

Exercice 3 :

Tester la fonction :

```
def echanger3():
    global a,b
    z=a
    a=b
    b=z
a=5
b=3
echanger3()
print ("a vaut : "+str(a))
print ("b vaut : "+ str(b))
```

Expliquer pourquoi ce programme doit fonctionner.

Exercice 4 :

Sans ordinateur

Qu'affiche le programme suivant ? Pourquoi ?

```
def g(x):
    global a
    a=10
    return 2*x
def f(x):
    v=1
    return g(x)+v
a=3
print (f(a)+a)
```

Exercice 5 :

(1) Écrire une fonction qui prend en argument une valeur et renvoie la valeur absolue de celle-ci.

(2) Écrire les commandes dans le programme principal qui demandent un nombre, appellent la fonction puis affichent le résultat.

Exercice 6 :

Écrire en Python une fonction qui prend comme argument un flottant x et renvoie la valeur de $\sqrt{x} + \cos x$. *On utilisera une fonction dans une bibliothèque bien choisie (l'aide est ici)*

Exercice 7 :

On se donne la suite u définie par :

$$\begin{cases} u_{n+1} & = 2u_n + 1 \\ u_0 & = 1 \end{cases}$$

(1) Proposer une fonction qui prend en argument un nombre n et affiche le $n$ème terme.

(2) Proposer une fonction qui prend en argument un nombre A et affiche le plus petit indice n tel que $u_n > A$.

Exercice 8 :

Même exercice que précédemment avec $\begin{cases} v_{n+1} & = 2(n+1)v_n + n^2 \\ v_0 & = 1 \end{cases}$ et $\begin{cases} w_{n+2} & = 2w_{n+1} + nw_n + 1 \\ w_0 & = 1 \\ w_1 & = 2 \end{cases}$

Exercice 9 :

On se donne la fonction suivante :

```
def factorielle(n):
    if n == 0:
        return 1
    else:
        return n * factorielle(n-1)
```

(1) Tester cette fonction pour différente valeur de n.

(2) Pourquoi ce programme fonctionne t'il ?

(3) Écrire la fonction **factorielle_iterative** qui calcule la fonction factorielle à l'aide d'une boucle.

On appelle ce type de fonction des fonctions récursives

Exercice 10 :

Écrire une fonction **chercher** qui prend en argument une liste et un nombre et renvoie l'indice de la première occurrence de ce nombre dans la liste s'il existe et -1 sinon. Par exemple :

```
L = [1,2,5,1]
print (chercher(L,2))
  --> 1
print (chercher(L,3))
  --> -1
print (chercher(L,1))
  --> 0
```