Exercice nº 1

1. Quel est le résultat affiché par :

```
def enigme1(coucou):
    beuh=2*coucou
    return beuh

if enigme1(4)==8:
    print('ok')
else:
    print('bof')
```

2. Quel est le résultat affiché par :

```
def enigme2(coucou):
    beuh=2*coucou
    print(beuh)

if enigme2(4)==6:
    print('ok')
else:
    print('bof')
```

3. Quels sont les résultats affichés par :

```
def enigme3(montou,cheval):
    if mouton < cheval:
        return cheval
    else:
        return mouton

enigme3(5,3)
enigme3(3,5)
```

4. Quels sont les résultats affichés par :

```
def enigme4(a,b,c):
    if a>b and a>c:
     return a
    elif b>a and b>c :
        return b
    else:
        return c

enigme4(2,5,3)
enigme4(1,4,5)
```

Exercice nº 2

Compléter la fonction ci-dessous :

```
def maxi(a,b,c,d):
    '''
    renvoi le maximum des 4 valeurs données en arguments
    param  : a,b,c,d (float)
    return : m (float)
    '''
```

Exercice n° 3

1. On considère la fonction `test` ci-dessous :

```
def test(nombre):
    if nombre%2==0:
        nombre=nombre/2
    else:
        nombre=3*nombre+1
    return nombre
```

   a. Quel est le résultat renvoyé par : `test(42)`
   b. Quel est le résultat renvoyé par : `test(11)`
   c. Quel est le résultat renvoyé par : `test(test(11))`

2. On considère la fonction `suiteV1` ci-dessous :

```
def suiteV1(nombre,repeat):
   for i in range(repeat):
       nombre=test(nombre)
       print(nombre)
```

3. Quel est le résultat affiché par : `suiteV1(11,10)`
4. On considère la fonction `suiteV1` ci-dessous :

```
def suiteV2(nombre):
   while nombre !=0:
       nombre=test(nombre)
       print(nombre)
```

5. Quel est le résultat affiché par : `suiteV2(11)`

Exercice n° 4 DEFIS : Suite du projet "bidon"

Compléter la fonction `bidon()` afin de pouvoir afficher les étapes à suivre pour atteindre l'objectif fixé dans le bidonA ou le bidonB. On s'aidera des fonctions écrites en classe.
La fonction doit également fonctionner lorsque l'on change les valeurs des variables globales.
(Condition d'utilisation : les capacités des bidons ne doivent pas avoir un diviseur commun strictement supérieur à 1)

```
Capacite_bidonA=5
Capacite_bidonB=3
objectif=4
bidonA=0
bidonB=0
#remplir_bidonA()
#remplir_bidonB()
#viderA()
#viderB()
#afficherbidon()
#verser_A_dans_B():
#verser_B_dans_A():

def test():
    global bidonA, Capacite_bidonA, bidonB, Capacite_bidonB,objectif
    ...
```