EXERCICE 52 — Comprendre un programme

On donne ci-dessous trois fonctions calc1, calc2 et calc3 programmées en langage Python.
Les variables a, b et n contiennent des nombres entiers.

```
def calc1(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
        b=3*b
        n=b*n
    return(n)
```

fin de la boucle

```
def calc2(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
        b=3*b
    n=b*n
    return(n)
```

```
def calc3(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
    b=3*b
    n=b*n
    return(n)
```

fin de la boucle

1. Compléter les calculs ci-dessous afin de déterminer ce que retourne calc1(5,1).

Avant le passage dans la boucle : n reçoit 0, a reçoit 5, b reçoit 1.

Lors du 1er passage dans la boucle : n reçoit 6, a reçoit 10, b reçoit 3 et n reçoit : 18.

Lors du 2e passage dans la boucle : n reçoit 13, a reçoit 20, b reçoit 9 et n reçoit : 117.

La condition $n < 20$ n'étant plus vérifiée, on sort de la boucle : calc1(5,1) retourne 117.

2. Que retourne calc2(5,1) ?

Avant le passage dans la boucle : n reçoit 0, a reçoit 5, b reçoit 1.

Lors du 1er passage dans la boucle : n reçoit 6 , a reçoit 10, b reçoit 3

Lors du 2ème passage dans la boucle : n reçoit 13 , a reçoit 20, b reçoit 9

Lors du 3ème passage dans la boucle : n reçoit 29 , a reçoit 40, b reçoit 27

La condition n < 20 n'est plus vérifiée, on sort de la boucle.

n reçoit 27 x 29 = 783 , donc calc2(5,1) retourne 783

3. Que retourne calc3(5,1) ?

```
def calc3(a,b):
    n=0
    while n<20:
        n=a+b
        a=2*a
    b=3*b
    n=b*n
    return(n)
```

fin de la boucle

3. Que retourne calc3(5,1) ?

Avant la boucle : n reçoit 0 , a reçoit 5, b reçoit 1

Lors du 1er passage dans la boucle : n reçoit 6 , a reçoit 10,

Lors du 2ème passage dans la boucle : n reçoit 11 , a reçoit 20,

Lors du 3ème passage dans la boucle : n reçoit 21 , a reçoit 40,

La condition n < 20 n'est plus vérifiée, on sort de la boucle.

b reçoit 3 n reçoit 3 x 21 = 63 ,

donc calc3(5,1) retourne 63

EXERCICE 53

Comprendre un programme

Une petite ville organise chaque année un loto. En 2016, le nombre de participants à ce loto était de 200 et chaque participant devait payer 5 €. On fait l'hypothèse que d'une année sur l'autre, il y a 15 nouveaux participants et que la participation augmente de 0,50 €.

On donne ci-contre un programme écrit en langage Python.

```
def loto(a,j,p):
    s=j*p
    while s<=1500:
        j=j+15
        p=p+0.5
        s=j*p
        a=a+1
    return(a)
```

a	2016	2017	2018	2019
j	200	215	230	
p	5	5,5	6	
s	1000	1182,5	1380	

1. a. **Que retourne loto(2016,200,5) ?**

loto(2016,200,5) retourne 2019

b. **À quoi correspond cette valeur ?**

Il s'agit de la première année où la recette dépassera 1500 euros

2. **On ajoute l'instruction « *a=a-1* » après la boucle while, comme sur la vue d'écran ci-contre.**

Que retourne désormais loto(2016,200,5) ? À quoi correspond cette valeur ?

```
        s=j*p
        a=a+1
    a=a-1
    return(a)
```

loto(2016,200,5) retourne 2018.

Cela correspond à la dernière année où la recette est inférieure ou égale à 1500 euros

EXERCICE 54

Comprendre et compléter un programme

On donne ci-contre une fonction mult1 programmée en langage Python. L'argument *n* de cette fonction contient un entier naturel non nul inférieur à 100.

```
def mult1(n):
    k=n
    c=1
    while k<100:
        c=c+1
        k=c*n
    return(k)
```

1. **Déterminer ce que retourne mult1(21) en complétant le tableau ci-dessous.**

mult1(21) retourne 105

2. **Quel est le rôle de cette fonction ?**

Cette fonction retourne le plus petit multiple de n supérieur ou égal à 100

Étapes	*c*	*k*	Condition vérifiée ?
Avant la boucle	1	21	oui
1er passage dans la boucle	2	2 x 21 = 42	oui
2e passage dans la boucle	3	3 x 21 = 63	oui
3e passage dans la boucle	4	4 x 21 = 84	oui
4e passage dans la boucle	5	5 x 21 = 105	non

3. a. **Compléter le programme de la fonction mult2 ci-contre afin que cette fonction retourne la même valeur que la fonction mult1.**

```
def mult2(n):
    k=n
    while k<100:
        k = k + n
    return(k)
```

b. **Dans la console, on a exécuté la fonction mult1 pour différentes valeurs de *n*. Vérifier que la fonction mult2 renvoie les mêmes résultats.**

```
>>> mult1(1)
100
>>> mult1(2)
100
>>> mult1(3)
102
>>> mult1(7)
105
>>> mult1(23)
115
```

mult2(1) → 100
mult2(2) → 100
mult2(3) → 102
mult2(7) → 105
mult2(23) → 115

A programmer et tester sur ordinateur
Possible sur
repl.it
puis choisir langage Python en bas de page