RÉVISIONS Python : récursivité

Q1. On considère la fonction `mystère` définie ci dessous. Que renvoie `mystere(1001,10)` ?

```
def mystere(x,b):
    if x<b:
        return 0
    else:
        return 1+mystere(x/b,b)
```

Q2. **Tours de Hanoï** Le problème des tours de Hanoï est un jeu de réflexion imaginé par le mathématicien français Édouard Lucas, et consistant à déplacer des disques de diamètres différents d'une tour de « départ » à une tour d'« arrivée » en passant par une tour « intermédiaire », et ceci en un minimum de coups, tout en respectant les règles suivantes :
— on ne peut déplacer plus d'un disque à la fois,
— on ne peut placer un disque que sur un autre disque plus grand que lui ou sur un emplacement vide.
— on suppose que cette dernière règle est également respectée dans la configuration de départ.
Écrire une procédure `def hanoi(n,source,dest,aux)` qui renvoie les instructions pour déplacer les n disques du piquet source vers le piquet dest à l'aide du piquet intermédiaire aux.
Par exemple l'exécution avec 3 disques devra délivrer un message de ce type :

```
Déplacer le disque du piquet  1  vers le piquet  2
Déplacer le disque du piquet  1  vers le piquet  3
Déplacer le disque du piquet  2  vers le piquet  3
Déplacer le disque du piquet  1  vers le piquet  2
Déplacer le disque du piquet  3  vers le piquet  1
Déplacer le disque du piquet  3  vers le piquet  2
Déplacer le disque du piquet  1  vers le piquet  2
```

Q3. On utilise le module `turtle`, qui est une implémentation du langage Logo en python, ce langage comportait un certain nombre d'instructions qui permettaient de commander un robot pouvant tracer des lignes sur une feuille de papier, voici une liste d'instructions possible (cf doc) suivi d'un exemple :
— `forward(n)` avance la tortue de n pas dans la direction qui lui fait face ;
— `backward(n)` recule la tortue de n pas ;
— `left(n)` tourne la tortue d'un angle de n degrés dans le sens trigonométrique ;
— `right(n)` tourne la tortue d'un angle de n degrés dans le sens horaire ;
— `speed(n)` modifie la vitesse de déplacement de la tortue de n = 1 (lent) à n = 10 (rapide) ; n = 0 (le plus rapide) supprime toute animation. Par défaut n = 3 ;
— `penup()` soulève le crayon de sorte que le déplacement de la tortue ne trace pas de trait ;
— `pendown()` baisse le crayon de sorte que le déplacement de la tortue trace un trait ;
— `pensize(n)` modifie l'épaisseur de la ligne que la tortue trace en se déplaçant (par défaut n = 1) ;
— `pencolor(c)` modifie la couleur de la ligne que la tortue trace (c peut être une chaine de caractère, par exemple `'red'`, mais aussi un triplet `(r,g,b)` de trois réels) ;
— `bgcolor(c)` modifie la couleur de la fenêtre où a lieu le tracé ;
— `reset()` efface tous les tracés et replace la tortue à sa position initiale.

```
from turtle import *
reset()

speed(5)
shape("turtle")
pencolor("red")
pensize(2)

def spirale_triangle(n, increment):
    angle = 120
    cote = 0
    for k in range(3 * n):
        cote = cote + increment
        forward(cote)
        left(angle)

spirale_triangle(6, 10)
exitonclick() #pour éviter de faire planter pyzo
```

Après avoir expérimenté le module `turtle`, écrire une fonction `koch(n,a)` qui trace un segment de Koch de longueur a à l'ordre n.