

Algorithmique et programmation Python

Corrigé du contrôle - 1ère

Durée conseillée : 45 minutes. Barème sur 20 points. Calculatrice interdite.

1 Lire un programme / 3 pts

- a. (2 pts) Pour $n = 10$, seuls les $k = 3, 6, 9$ vérifient $k \% 3 == 0$: $s = 3 + 6 + 9 = 18$.
Pour $n = 12$, on ajoute 12 : $s = 30$. Vérification par exécution : **18 30** (1 pt pour chaque valeur).
- b. (1 pt) Elle renvoie la somme des multiples de 3 compris entre 1 et n .

Total : 3 points.

2 Fonctions et listes / 3 pts

a), b) (1,5 pt)

```
def negatifs(L):
    k = 0
    for x in L:
        if x < 0:
            k = k + 1
    return k

def positifs(L):
    R = []
    for x in L:
        if x > 0:
            R.append(x)
    return R
```

Barème : compteur initialisé (0,5), test strict (0,5), bonne valeur renvoyée en dehors de la boucle (0,5) pour a) ; liste initialisée vide et append pour b) comptent dans les 1,5 pt de l'ensemble.

c) (1,5 pt) negatifs renvoie 2 (les éléments -2 et -7), positifs renvoie $[3, 5]$ (0 n'est ni positif ni négatif au sens strict). Sortie :

2
[3, 5]

3 Suite et seuil (calculatrice autorisée) / 4 pts

- a. (1 pt) $u_1 = 0,9 \times 200 + 30 = 210$; $u_2 = 0,9 \times 210 + 30 = 219$.
b. (1,5 pt)

```
def seuil(S):  
    n = 0  
    u = 200  
    while u <= S:  
        u = 0.9 * u + 30  
        n = n + 1  
    return n
```

Barème : initialisation (0,5), condition du while correcte $u \leq S$ (0,5), mise à jour et compteur (0,5).

- c. (1,5 pt) Le programme renvoie 7. Vérification : $u_6 \approx 246,86 \leq 250$ et $u_7 \approx 252,17 > 250$.

4 Dichotomie / 4 pts

- a. (1 pt) $f(1) = -2 < 0$, $f(2) = 5 > 0$, f est continue et strictement croissante sur $[1 ; 2]$: elle s'annule une seule fois (théorème des valeurs intermédiaires pour une fonction strictement monotone).
b. (2 pts)

Étape	Milieu m	f(m)	Nouvel intervalle
1	1,5	0,3750	[1 ; 1,5]
2	1,25	-1,0469	[1,25 ; 1,5]
3	1,375	-0,4004	[1,375 ; 1,5]

Donc $1,375 \leq \alpha \leq 1,5$, d'amplitude 0,125.

- c. (1 pt) Il faut $1/2^n \leq 0,01$, c'est-à-dire $2^n \geq 100$: $2^6 = 64 < 100$ et $2^7 = 128 \geq 100$, donc **7 étapes**. Le programme dichotomie(f, 1, 2, 0.01) donne : **(1.4375, 1.4453125)**.

5 Simulation / 3 pts

- a. (1 pt) **essai** renvoie True dès qu'un 6 est tiré : voir le programme ci-dessous.
b. (1 pt) **frequence** compte les essais réussis et divise par N.

```

from random import randint

def essai():
    for i in range(4):
        if randint(1, 6) == 6:
            return True
    return False

def frequence(N):
    c = 0
    for i in range(N):
        if essai():
            c = c + 1
    return c / N

print(frequence(10000))

```

c) (1 pt) Le contraire de A est « aucun 6 en 4 lancers » de probabilité $(5/6)^4 = 625/1296$. Donc $P(A) = 1 - 625/1296 = 671/1296 \approx 0,518$. Avec $N = 10000$, on attend une fréquence proche de 0,518 (par exemple entre 0,50 et 0,54), différente à chaque exécution.

6 Tri par sélection / 3 pts

- a. (2 pts) Tour 0 : le minimum est 1, échangé avec 6 : [1, 2, 8, 3, 6]. Tour 1 : le minimum de 2, 8, 3, 6 est 2, déjà en place : [1, 2, 8, 3, 6]. Tour 2 : le minimum de 8, 3, 6 est 3, échangé avec 8 : [1, 2, 3, 8, 6]. Tour 3 : le minimum de 8, 6 est 6, échangé avec 8 : [1, 2, 3, 6, 8].
- b. (1 pt) $n(n - 1)/2$ avec $n = 5$: $5 \times 4/2 = 10$ comparaisons (4 + 3 + 2 + 1).