

Fiche 6 : algorithmique et Python

Sept exercices sur les questions de programmation que l'on retrouve régulièrement dans les exercices de suites du bac : calculer un terme, trouver un seuil, calculer une somme, compléter ou interpréter un script.

Bonne nouvelle : ces questions sont presque toujours des points « cadeaux » si tu connais les trois modèles ci-dessous.

Toutes les corrections sont rédigées comme on l'attend au bac : exécution pas à pas, justification de la valeur renvoyée, justification de la terminaison quand elle est demandée.

Les trois modèles à connaître

1. Calculer le terme u_n (boucle **for) :** on part de u_0 , on applique la relation de récurrence n fois.

```
def terme(n):  
    u = 2  
    for i in range(n):  
        u = 3*u - 1  
    return u
```

2. Trouver un seuil (boucle **while) :** on avance tant que la condition n'est pas atteinte, et on compte les étapes.

```
def seuil():  
    n = 0  
    u = 1000  
    while u < 2000:  
        u = 1.05*u  
        n = n + 1  
    return n
```

3. Calculer une somme : on ajoute chaque terme à une variable qui sert de compteur.

```
def somme(n):  
    u = 1  
    s = 1  
    for i in range(n):  
        u = u/2  
        s = s + u  
    return s
```

Les pièges classiques

- **Le seuil renvoie n , pas u_n .** Relis la question : « le plus petit entier n » veut dire qu'on renvoie le compteur.
- L'ordre des instructions dans la boucle : mettre à jour u *avant* ou *après* $n = n + 1$ ne change pas le résultat ici, mais si u et n se servent l'un de l'autre, ça change tout.
- La condition du **while** est **l'inverse** de ce qu'on cherche : on cherche $u_n \geq A$, donc on boucle tant que $u < A$.
- Confondre $<$ et $\leq$ (ou $>$ et $\geq$) : lis l'énoncé au symbole près.
- **range(n)** répète n fois, avec i qui va de 0 à $n - 1$.

Astuce : exécute le script à la main sur les deux ou trois premières étapes, avec un tableau de valeurs. C'est la méthode qui évite 95 % des erreurs. Un script qui « a l'air juste » n'est pas un script vérifié.

Exercice 1 (★) : que renvoie ce script ?

On considère la suite définie par $u_0 = 2$ et $u_{n+1} = 3u_n - 1$. Que renvoie l'appel **terme(3)** du modèle 1 ci-dessus ?

Correction de l'exercice 1

Exécution à la main

On exécute le script à la main avec $n = 3$. La boucle **for i in range(3)** s'exécute 3 fois ($i = 0, 1, 2$).

- Avant la boucle : $u = 2$.
- $i = 0$: $u = 3 \times 2 - 1 = 5$.
- $i = 1$: $u = 3 \times 5 - 1 = 14$.
- $i = 2$: $u = 3 \times 14 - 1 = 41$.

La fonction renvoie 41, c'est-à-dire u_3 .

À chaque passage, la variable u contient le terme suivant de la suite, donc après 3 passages elle contient u_3 .

Exercice 2 (★) : écrire une fonction

On définit $u_0 = 3$ et $u_{n+1} = 0,5 u_n + 2$. Écrire une fonction Python **terme(n)** qui renvoie u_n , puis calculer **terme(3)** à la main.

Correction de l'exercice 2

Le script

```
def terme(n):
    u = 3
    for i in range(n):
        u = 0.5*u + 2
    return u
```

La variable u est initialisée à $u_0 = 3$, puis la relation $u_{n+1} = 0,5 u_n + 2$ est appliquée n fois : la fonction renvoie bien u_n .

Exécution pour $n = 3$

$$u_0 = 3$$

$$u_1 = 0,5 \times 3 + 2 = 3,5$$

$$u_2 = 0,5 \times 3,5 + 2 = 3,75$$

$$u_3 = 0,5 \times 3,75 + 2 = 3,875$$

terme(3) renvoie 3,875.

Exercice 3 (★★) : un seuil à la baisse

Une machine vaut 5 000 € à l'achat et perd 10 % de sa valeur chaque année. On note u_n sa valeur au bout de n années.

1. Exprimer u_n en fonction de n .
2. Écrire une fonction Python qui renvoie le plus petit entier n tel que $u_n < 2000$.
3. Quelle valeur renvoie-t-elle ?

Correction de l'exercice 3

Question 1

Perdre 10 % de sa valeur revient à multiplier par 0,9. La suite (u_n) est donc géométrique de raison 0,9 et de premier terme $u_0 = 5000$.

Donc, pour tout $n \in \mathbb{N}$, $u_n = 5000 \times 0,9^n$.

Question 2

On boucle tant que la valeur est encore supérieure ou égale à 2000, et on compte les années :

```
def seuil():
    n = 0
    u = 5000
    while u >= 2000:
        u = 0.9*u
        n = n + 1
    return n
```

Terminaison : comme $-1 < 0,9 < 1$, $u_n = 5000 \times 0,9^n$ tend vers 0. Il existe donc un rang à partir duquel $u_n < 2000$: la condition $u \geq 2000$ finit par être fausse et la boucle s'arrête.

Question 3

On cherche le plus petit entier n tel que $5000 \times 0,9^n < 2000$, soit $0,9^n < 0,4$. La suite $(0,9^n)$ est strictement décroissante, et :

$$0,9^8 \approx 0,4305 \geq 0,4$$

$$0,9^9 \approx 0,3874 < 0,4$$

La fonction renvoie 9.

Piège évité : la suite décroît, donc la condition du `while` est $u \geq 2000$ (on continue tant qu'on n'est pas passé sous le seuil).

Exercice 4 (★★) : une somme et son seuil

Pour $n \in \mathbb{N}$, on pose $S_n = 1 + \frac{1}{2} + \frac{1}{4} + \dots + \frac{1}{2^n}$.

1. Écrire une fonction Python `somme(n)` qui renvoie S_n . Que renvoie `somme(3)` ?
2. On rappelle que $S_n = 2 - \frac{1}{2^n}$. Écrire une fonction qui renvoie le plus petit entier n tel que $2 - S_n < 0,01$. Quelle valeur renvoie-t-elle ?

Correction de l'exercice 4

Question 1

```
def somme(n):
    u = 1
    s = 1
    for i in range(n):
        u = u/2
        s = s + u
    return s
```

La variable u contient successivement $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \dots$, et s cumule les termes : après n passages, $s = 1 + \frac{1}{2} + \dots + \frac{1}{2^n} = S_n$.

Exécution pour $n = 3$: $s = 1$, puis $u = 0.5$ et $s = 1.5$, puis $u = 0.25$ et $s = 1.75$, puis $u = 0.125$ et $s = 1.875$.

somme(3) renvoie 1,875.

Question 2

Comme $S_n = 2 - \frac{1}{2^n}$, on a $2 - S_n = \frac{1}{2^n}$. On cherche le plus petit entier n tel que $\frac{1}{2^n} < 0,01$.

Voici une fonction possible :

```
def seuil():
    n = 0
    while 1/2**n >= 0.01:
        n = n + 1
    return n
```

Terminaison : comme $-1 < \frac{1}{2} < 1$, $\frac{1}{2^n} \rightarrow 0$, donc $\frac{1}{2^n}$ finit par être inférieur à 0,01 et la boucle s'arrête.

Valeur renvoyée : $\frac{1}{2^n} < 0,01 \iff 2^n > 100$. Comme $2^6 = 64 < 100$ et $2^7 = 128 > 100$, et que 2^n est croissante, le plus petit entier convenable est $n = 7$.

La fonction renvoie 7.

Exercice 5 (★★★) : compléter un script (type bac)

Dans l'exercice du parc animalier (fiche 4, exercice 4), on a $u_0 = 800$ et $u_{n+1} = 0,9 u_n + 60$.

On souhaite savoir à partir de quelle année (n) le parc compte moins de 650 animaux.

Recopier et compléter le script :

```
def annee():
    n = .....
    u = .....
    while .....:
        u = .....
        n = .....
    return n
```

Correction de l'exercice 5

Le script complété

```
def annee():
    n = 0
    u = 800
    while u >= 650:
        u = 0.9*u + 60
        n = n + 1
    return n
```

Justification des compléments : on part de $n = 0$ et $u = u_0 = 800$.

On continue tant que $u \geq 650$, car on cherche le premier moment où u passe **sous** 650.

À chaque passage, u devient $0.9u + 60$, c'est-à-dire le terme suivant, et n compte le rang.

Terminaison : $u_n = 600 + 200 \times 0,9^n$ tend vers $600 < 650$, donc u_n finit par être strictement inférieur à 650.

Valeur renvoyée : $u_n = 600 + 200 \times 0,9^n$ donne :

$$u_{13} \approx 650,8 \geq 650$$

$$u_{14} \approx 645,8 < 650$$

La fonction renvoie 14, ce qui correspond à l'année **2040**.

Exercice 6 (★★★) : conjecturer une limite

Reprenons $u_0 = 0$ et $u_{n+1} = \frac{1}{1 + u_n}$ (fiche 5, exercice 5).

1. Écrire un script Python qui affiche les termes u_1 à u_{10} .
2. Donner les valeurs approchées obtenues et conjecturer la limite.

Correction de l'exercice 6**Question 1**

```
u = 0
for n in range(1, 11):
    u = 1/(1 + u)
    print(n, u)
```

Au n -ième passage, u devient u_n , que l'on affiche avec son rang n .

Question 2

Valeurs approchées :

$$u_1 = 1, u_2 = 0,5, u_3 \approx 0,6667, u_4 = 0,6, u_5 = 0,625$$

$$u_6 \approx 0,6154, u_7 \approx 0,6190, u_8 \approx 0,6176, u_9 \approx 0,6182, u_{10} \approx 0,6180$$

Les termes alternent au-dessus et en dessous d'une valeur voisine de 0,618 en s'en rapprochant.

On conjecture que (u_n) converge vers $\frac{\sqrt{5}-1}{2} \approx 0,618$.

Cette conjecture n'est pas une preuve : elle est cohérente avec la limite calculée dans la fiche 5 (à condition d'admettre la convergence).

Remarque : les termes alternent, ce qui confirme que la suite n'est pas monotone, comme on l'avait vu en fiche 5.

Exercice 7 (★★★) : pourquoi l'algorithme s'arrête-t-il ?

On définit $u_0 = 1$ et $u_{n+1} = 2u_n + 1$. Pour un réel A , on considère la fonction :

```
def seuil(A):
    n = 0
    u = 1
    while u <= A:
        u = 2*u + 1
        n = n + 1
    return n
```

1. Que renvoie `seuil(1000)` ? (On admet que $u_n = 2^{n+1} - 1$.)
2. Pourquoi peut-on affirmer que cet algorithme s'arrête pour tout réel A ?

Correction de l'exercice 7

Question 1

La fonction `seuil(1000)` renvoie le plus petit entier n tel que $u_n > 1000$, c'est-à-dire $2^{n+1} - 1 > 1000$, soit $2^{n+1} > 1001$.

Comme $2^9 = 512 < 1001$ et $2^{10} = 1024 > 1001$, et que 2^k est croissante en k , le plus petit entier convenable vérifie $n + 1 = 10$, donc $n = 9$.

Vérification : $u_8 = 511 \leq 1000$ et $u_9 = 1023 > 1000$.

La fonction renvoie 9.

Question 2

On a $u_n = 2^{n+1} - 1 = 2 \times 2^n - 1$. Comme $2 > 1$, $2^n \rightarrow +\infty$, donc, par produit et par somme, $u_n \rightarrow +\infty$.

Par définition de la limite, pour tout réel A , il existe un rang n_0 tel que $u_n > A$ dès que $n \geq n_0$. Donc la condition $u \leq A$ devient fausse après un nombre fini de passages :

la boucle se termine, quel que soit A .

À retenir : « pourquoi l'algorithme se termine » se justifie toujours par une limite infinie (ou par une limite qui franchit le seuil). C'est un point que les correcteurs adorent.

Ce qui peut tomber au bac

- Un script à **exécuter à la main** ou dont on donne la valeur renvoyée (exercices 1, 2, 5).
- Un **seuil** à programmer avec une boucle **while** (exercices 3, 4, 7).
- Une **justification de la terminaison** par une limite (exercices 3, 4, 5 et 7).
- Un script qui **affiche des termes** pour conjecturer une limite (exercice 6).

QCM : une seule bonne réponse par question

1. Dans `while u < 100:`, la boucle s'arrête quand :

- A. $u \geq 100$ (**bonne réponse**)
- B. $u < 100$
- C. $u = 100$ exactement
- D. après 100 tours

Justification : La boucle continue tant que la condition est vraie et s'arrête dès que $u < 100$ est fausse.

2. Avec `u = 1` puis `for k in range(3): u = 2*u + 1`, que vaut `u` à la fin ?

- A. 31
- B. 9
- C. 7
- D. 15 (**bonne réponse**)

Justification : $1 \rightarrow 3 \rightarrow 7 \rightarrow 15$: trois itérations.

3. Pour $u_0 = 100$ et $u_{n+1} = 0,5 u_n$, quel est le premier rang n tel que $u_n < 1$?

- A. 7 (**bonne réponse**)
- B. 6
- C. 8
- D. 100

Justification : $u_6 = 100 \times 0,5^6 \approx 1,56$ et $u_7 = 100 \times 0,5^7 \approx 0,78 < 1$.

4. ``range(1, n+1)`` parcourt :

- A. **les entiers de 1 à n (bonne réponse)**
- B. les entiers de 0 à n
- C. les entiers de 1 à $n - 1$
- D. $n + 1$ entiers

Justification : La borne finale est exclue : on obtient $1, 2, \dots, n$.

5. Une boucle ``while`` dont la condition ne devient jamais fausse :

- A. **tourne indéfiniment (bonne réponse)**
- B. s'arrête au bout de 100 tours
- C. provoque une erreur de syntaxe
- D. renvoie 0

Justification : C'est un risque classique : il faut vérifier que la variable évolue dans le bon sens.

6. Dans l'exercice 1, avec $u_0 = 2$ et $u_{n+1} = 3u_n - 1$, que renvoie ``terme(3)`` ?

- A. **41 (bonne réponse)**
- B. 23
- C. 17
- D. 53

Justification : $u_1 = 5$, $u_2 = 14$, $u_3 = 41$.

7. Dans l'exercice 2, avec $u_0 = 3$ et $u_{n+1} = 0,5 u_n + 2$, que renvoie ``terme(3)`` ?

- A. 2,875
- B. 4
- C. **3,875 (bonne réponse)**
- D. 3,5

Justification : $u_1 = 3,5$, $u_2 = 3,75$, $u_3 = 3,875$.

8. Dans l'exercice 3 (la machine qui perd 10 % de sa valeur par an, $u_0 = 5\,000$), au bout de combien d'années la valeur passe-t-elle sous 2 000 € ?

- A. 5 ans
- B. 2 000 ans
- C. 10 ans
- D. **9 ans (bonne réponse)**

Justification : $u_n = 5\,000 \times 0,9^n$, et $5\,000 \times 0,9^9 \approx 1\,937 < 2\,000$ alors que $5\,000 \times 0,9^8 \approx 2\,153 \geq 2\,000$.

9. Dans l'exercice 4, avec $S_n = 1 + \frac{1}{2} + \dots + \frac{1}{2^n}$, que renvoie ``somme(3)`` ?

- A. 0,875
- B. 1,75
- C. 2
- D. 1,875 (bonne réponse)

Justification : $S_3 = 1 + 0,5 + 0,25 + 0,125 = 1,875$.

10. Dans l'exercice 7, avec $u_0 = 1$, $u_{n+1} = 2u_n + 1$ et la fonction ``seuil(A)`` du script, que renvoie ``seuil(1000)`` ?

- A. 10
- B. 9 (bonne réponse)
- C. 1 000
- D. 8

Justification : $u_n = 2^{n+1} - 1$; $u_8 = 511 \leq 1\,000$ et $u_9 = 1\,023 > 1\,000$: la boucle s'arrête après 9 tours.