

1G - groupes 8 et 9 - Spécialité mathématiques
Correction du travail à distance n°3

Exercice 1 (n° 6 partie cours)

Considérons la suite (u_n) définie pour tout entier $n \in \mathbb{N}$ par $u_n = n \times \sqrt{n}$.

1. Montrer que (u_n) est croissante.

Pour tout $n \in \mathbb{N}$, $u_{n+1} - u_n = (n+1)\sqrt{n+1} - n\sqrt{n} = n\sqrt{n+1} + \sqrt{n+1} - n\sqrt{n} = n(\sqrt{n+1} - \sqrt{n}) + \sqrt{n+1}$.

Comme la fonction racine carrées est strictement croissante sur $[0; +\infty[$, $\sqrt{n+1} > \sqrt{n}$, et donc $\sqrt{n+1} - \sqrt{n} > 0$.

En multipliant par $n \geq 0$, $n(\sqrt{n+1} - \sqrt{n}) \geq 0$.

De plus, pour tout $n \in \mathbb{N}$, $\sqrt{n+1} > 0$.

Donc, par somme deux nombres positifs, $u_{n+1} - u_n = n(\sqrt{n+1} - \sqrt{n}) + \sqrt{n+1} > 0$.

Pour tout $n \in \mathbb{N}$, $u_{n+1} > u_n$.

La suite (u_n) est strictement croissante.

2. On admet que $\lim u_n = +\infty$.

Écrire un algorithme qui renvoie le plus petit entier n_0 tel que pour tout $n \geq n_0$, $u_n \geq 10^4$.

<pre>n ← 0 U ← n√n Tant que U < 10⁴ n ← n + 1 U ← n√n Fin tant que Afficher n</pre>	<p>Variante un peu plus courte :</p> <pre>n ← 0 Tant que n√n < 10⁴ n ← n + 1 Fin tant que Afficher n</pre>
---	--

3. Programmer l'algorithme à la calculatrice ou Python et donner la valeur de n_0 .

Voici une fonction Python sans argument qui convient

On charge le module math pour la fonction racine carrée (sqrt).

```
from math import *
def seuil():
    n=0
    U=n*sqrt(n)
    while U<10**4:
        n=n+1
        U=n*sqrt(n)
    return(n)
```

On obtient $n_0 = 465$.

$n_0 = 465$ est le plus petit entier tel que $u_n \geq 10^4$.

Et comme la suite est croissante (question 1), pour tout $n \geq 465$, $u_n \geq u_{465} \geq 10^4$.

On peut donc affirmer que pour tout $n \geq 465$, $u_n \geq 10^4$.

Exercice 2 (66 page 88)

$v_0 = 0$ et pour tout $n \in \mathbb{N}$, $v_{n+1} = \frac{v_n}{1 + v_n^2}$.

1. Fonction Python qui renvoie v_n .

```
def v(n):
    V=2
    for k in range(n):
        V=V/(1+V**2)
    return(V)
```

2. Conjecture sur la limite de (v_n) .

$v_1 = 0,4$, $v_{100} \approx 0,0696$; $v_{10000} \approx 0,00706$.

Il semble que (v_n) converge vers 0 (ce n'est qu'une conjecture, on ne l'a pas montré).

Exercice 3 (67 page 88)

$u_0 = 15$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = 3u_n + 7$.

Fonction Python à compléter qui renvoie le plus petit entier p tel que $u_p > 1000$.

```
def seuil67p88():
    A=15
    N=0
    while A<=1000:
        N=N+1
        A=3*A+7
    return(N)
```

On obtient $p = 4$.

Attention à l'inégalité dans le test du while, qui doit être la négation de l'énoncé, soit $A \leq 1000$.

Exercice 4 (73 page 88)

$v_0 = 4$, et pour tout $n \in \mathbb{N}$, $v_{n+1} = v_n + \frac{1}{v_n}$.

1. Algorithme de la somme des termes $v_0 + v_1 + \dots + v_N$

Entrer N

$V \leftarrow 4$

$S \leftarrow 4$

Pour i allant de 1 à N

$V \leftarrow V + \frac{1}{V}$

$S \leftarrow S + V$

Fin Pour

Afficher S

2. Fonction Python

```
def somme(N):
```

```
    V=4
```

```
    S=4
```

```
    for i in range(1,N+1):
```

```
        V=V+1/V
```

```
        S=S+V
```

```
    return(S)
```

Par exemple, `somme(10)` renvoie $v_0 + v_1 + \dots + v_{10} \approx 55,90201$.

Exercice 5 (59 page 87)

On donne $u_0 = 5$ et pour tout $n \in \mathbb{N}$, $u_{n+1} = 7u_n^2 - 5$.

Fonction Python (sans argument) qui retourne la somme $u_0 + u_1 + \dots + u_{100}$.

```
def sommeU():
```

```
    U=5
```

```
    S=5
```

```
    for i in range(1,101):
```

```
        U=7*U**2-5
```

```
        S=S+U
```

```
    return(S)
```

Exercice 6 (97 page 91)

$p_0 = 1013$ (pression en hectopascals à l'altitude 0, niveau de la mer).

La pression diminue de 1,25% à chaque élévation de 100 m. On note p_n la pression à l'altitude $(100 \times n)$ m.

1. Calcul de p_1 et p_2 .

Diminuer de 1,25 % revient à multiplier par $1 - 0,0125 = 0,9875$.

En effet, $x - \frac{1,25}{100} \times x = x \times (1 - 0,0125) = x \times 0,9875$.

Donc $p_1 = p_0 \times 0,9875 = 1013 \times 0,9875 = 1000,3375 \approx 1000$.

$p_2 = p_1 \times 0,9875 = 987,5 \approx 988$.

2. Relation p_{n+1} en fonction de p_n , et interprétation.

Comme diminuer de 1,25 % revient à multiplier par 0,9875,

on a pour tout $n \in \mathbb{N}$, $p_{n+1} = p_n \times 0,9875$.

Autre démonstration, reprenant le calcul effectué à la question 1 :

$$p_{n+1} = p_n - \frac{1,25}{100} \times p_n = p_n \times (1 - 0,0125) = p_n \times 0,9875.$$

Donc la suite (p_n) est la suite géométrique de premier terme $p_0 = 1013$, et de raison $q = 0,9875$.

3. Expression de p_n .

Comme (p_n) est la suite géométrique de premier terme $p_0 = 1013$ et de raison $q = 0,9875$, on a :

Pour tout $n \in \mathbb{N}$, $p_n = p_0 \times q^n = 1013 \times 0,9875^n$.

Pour tout $n \in \mathbb{N}$, $p_n = 1013 \times 0,9875^n$.

4. Pression à 3200 m.

$3200 = 32 \times 100$, on cherche donc p_{32} .

$$p_{32} = 1013 \times 0,9875^{32} \approx 677.$$

La pression atmosphérique à 3200 m est de 677 hectopascals.

5. Déterminons à partir de quelle altitude (à 100m près), la pression devient inférieure à 600 hectopascals.

On cherche le plus petit entier n tel que $p_n \leq 600$.

Méthode 1 : Algorithme de seuil (avec une fonction Python).

```
def seuilpression():
```

```
    N=0
```

```
    P=1013
```

```
    while P>600:
```

```
        N=N+1
```

```
        P=P*0.9875
```

```
    return (N) On obtient  $N = 42$ . Méthode 2 : deux termes et le sens de variation
```

Comme la suite p_n est géométrique de premier terme positif et de raison $q = 0,9875$ vérifiant $0 < q < 1$, la suite (p_n) est strictement décroissante (propriété de cours).

Or, on observe que $p_{41} = 1013 \times 0,9875^{41} \approx 605 > 600$.

Et $p_{42} = 1013 \times 0,9875^{42} \approx 597 \leq 600$.

Donc le plus petit entier n tel que $p_n \leq 600$ est 42.

La pression devient inférieure à 600 hectopascals à partir de 4200 m d'altitude.

Exercice 7 (Problème de la balle)

Lucas lâche une balle d'une hauteur de 24 m. Lorsque la balle rebondit, la hauteur de son rebond perd 10% par rapport à la hauteur du rebond précédent.

On pose $u_0 = 24$, et pour tout $n \geq 1$, on note u_n la hauteur du n^e rebond.

1. Calculer u_1 .

On peut considérer que $u_0 = 24$.

$$u_1 = 24 - 24 \times \frac{10}{100} = 24 \times 0.9 = 21.6.$$

Le premier rebond a une hauteur de 21.6 m.

2. Montrer que (u_n) est une suite géométrique dont on précisera la raison. Pour tout $n \in \mathbb{N}$,

$$u_{n+1} = u_n - u_n \times \frac{10}{100} = u_n - 0.1 \times u_n = 0.9 \times u_n.$$

Donc (u_n) est une suite géométrique de raison $q = 0.9$.

3. On estime que la balle est immobile lorsque le rebond est inférieur à 1 cm.

- (a) Écrire une fonction Python sans argument qui renvoie le plus petit entier n tel que $u_n \leq 0,01$.

```
def seuilballe():
```

```
    N=0
```

```
    U=24
```

```
    while U>0.01:
```

```
        N=N+1
```

U=0.9*U
return (N)

- (b) Combien de rebonds a fait la balle ? Justifier. On note p ce nombre.

On obtient $N = 74$.

La plus petite valeur de n pour laquelle $u_n \leq 0.01$ est 74.

La balle a donc fait 73 rebonds ayant des hauteurs supérieures à 1 cm.

On considère qu'elle a fait 73 rebonds puis s'est immobilisée : $p = 73$.

4. Quelle est alors la distance parcourue par la balle ?

Pour chaque rebond, la balle parcourt la longueur u_n une fois en montant, et une fois en descendant.

L'énoncé dit que « Lucas lâche une balle d'une hauteur de 24 m ».

Notons D la distance totale parcourue par la balle.

$$\begin{aligned} D &= 24 + \sum_{n=1}^p 2u_n \\ &= 24 + 2 \sum_{n=1}^p u_n \\ &= 24 + 2 \times (u_1 + u_2 + \dots + u_{73}) \\ &= 24 + 2 \times 21,6 \times \frac{1 - 0.9^{73}}{1 - 0.9} \\ &\approx 455,802 \end{aligned}$$

La balle a parcouru environ 456 m.

Exercice 8 (sujet C p 99)

1. Comme OA_0A_1 est isocèle en A_1 , on a $OA_1 = A_0A_1$.

En appliquant le théorème de Pythagore dans le triangle OA_0A_1 rectangle en A_1 , il vient

$$OA_0^2 = OA_1^2 + A_1A_0^2, \text{ soit } 2OA_1^2 = 1^2, \text{ puis } OA_1 = \sqrt{\frac{1}{2}} = \frac{\sqrt{2}}{2}.$$

2. (a) Le triangle OA_nA_{n+1} est rectangle isocèle en A_{n+1} , d'après le théorème de Pythagore,

$$2OA_{n+1}^2 = OA_n^2, \text{ ou encore } OA_{n+1} = \frac{\sqrt{2}}{2}OA_n.$$

- (b) En posant $u_n = OA_n$, avec la question précédente, on a pour tout $n \in \mathbb{N}$, $u_{n+1} = \frac{\sqrt{2}}{2}u_n$.

Donc la suite (u_n) est la suite géométrique de premier terme $u_0 = OA_0 = 1$ et de raison

$$q = \frac{\sqrt{2}}{2}.$$

$$\text{Donc pour tout } n \in \mathbb{N}, u_n = u_0 \times q^n = \left(\frac{\sqrt{2}}{2}\right)^n.$$

3. $L_n = OA_0 + A_0A_1 + A_1A_2 + \dots + A_n + 1A_n$.

- (a) Expression de L_n .

Avec les triangles isocèles, il vient

$$L_n = OA_0 + OA_1 + OA_2 + \dots + OA_n$$

$$L_n = u_0 + u_1 + \dots + u_n$$

Donc L_n et la somme des $(n+1)$ premiers termes de la suite géométrique (u_n) .

$$L_n = u_0 \times \frac{1 - q^{n+1}}{1 - q} = 1 \times \frac{1 - \left(\frac{\sqrt{2}}{2}\right)^{n+1}}{1 - \frac{\sqrt{2}}{2}} = \frac{2}{2 - \sqrt{2}} \times \left[1 - \left(\frac{\sqrt{2}}{2}\right)^{n+1}\right].$$

$$\text{Or, avec la quantité conjuguée, } \frac{2}{2 - \sqrt{2}} = \frac{2(2 + \sqrt{2})}{(2 - \sqrt{2})(2 + \sqrt{2})} = \frac{2(2 + \sqrt{2})}{4 - 2} = 2 + \sqrt{2}.$$

$$L_n = (2 + \sqrt{2}) \times \left[1 - \left(\frac{\sqrt{2}}{2} \right)^{n+1} \right].$$

(b) Conjecture sur la limite de L_n .

Il semble que L_n tende vers $2 + \sqrt{2} \approx 3,4142$.

La démonstration utilise un résultat vu en terminale : si $0 < q < 1$, alors $\lim q^n = 0$.

$$\frac{\sqrt{2}}{2} \approx 0,7.$$

Comme $0 < \frac{\sqrt{2}}{2} < 1$, on a $\lim \left(\frac{\sqrt{2}}{2} \right)^{n+1} = 0$.

Donc $\lim L_n = 2 + \sqrt{2} \approx 3,4142$.

Exercice 9 (119 page 100)

1. (a) Le nombre de termes double à chaque étape. On pose a_n le nombre d'éléments de la liste L_n .

On a $a_0 = 1$, et pour tout $n \in \mathbb{N}$, $a_{n+1} = 2a_n$, donc la suite est géométrique de raison 2 et $a_n = a_0 \times 2^n = 1 \times 2^n = 2^n$.

On cherche le plus petit entier n tel que $2^n > 2020$.

D'après le cours, $2 > 1$, la suite (2^n) est strictement croissante, $2^{10} = 1024$, et $2^{11} = 2048$.

Le plus petit entier n tel que L_n contient plus de 2020 éléments est $n = 11$.

On pouvait aussi utiliser un algorithme de seuil.

(b) Fonction qui renvoie le 2020^e terme de L_n où $n = 11$, soit L_{11} .

```
def list():
    L=[1]
    for k in range(11):
        U=[i*0.5 for i in L]
        L=L+U
    return L[2019]
```

La fonction renvoie 0.00390625.

2. On pose u_n le dernier élément de la liste L_n .

(a) $u_0 = 1$, et pour tout $n \in \mathbb{N}$, $u_{n+1} = \frac{1}{2}u_n$, donc u_n est géométrique de raison $\frac{1}{2}$, et $u_n = \frac{1}{2^n}$.

Comme $0 < q < 1$, la suite (u_n) est strictement décroissante.

(b) On conjecture que $\lim u_n = 0$.