Suites numériques

Calculatrice & Algorithmique

Évarite Galois
maths-mde.fr

Exercice 1 :

La suite (v_n) est définie par $v_0 = 16$ et pour tout entier naturel n par : $v_{n+1} = \sqrt{v_n} + 5$.

1. $v_1 = \sqrt{v_0} + 5 = \sqrt{16} + 5 = 9$ et $v_2 = \sqrt{v_1} + 5 = \sqrt{9} + 5 = 8$.

2. Ci-après l'algorithme en langage naturel permettant de calculer le rang n de la suite (v_n).

```
V ⟵ 16
Pour i allant de 1 à N
V ⟵ √V + 5
Fin Pour
```

Exercice 2 :

Soit la suite (u_n) définie par $u_0 = 150$ et pour tout entier naturel n : $u_{n+1} = 0,8u_n + 45$.

1. Ci-après les deux termes u_1 et u_2 :
$u_1 = 0,8u_0 + 45 = 0,8 \times 150 + 45 = 165$.
$u_2 = 0,8u_1 + 45 = 0,8 \times 165 + 45 = 177$.

2. Voici deux propositions d'algorithmes :

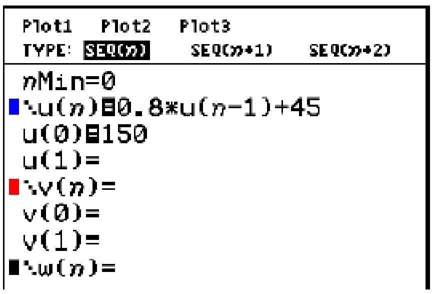
```
U ⟵ 150
N ⟵ 0
Tant que U ⩾ 220
U ⟵ 0,8 × U + 45
N ⟵ N+1
Fin Tant que
```

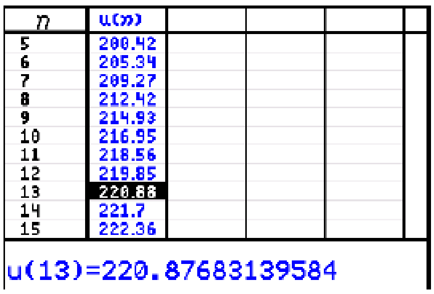
```
U ⟵ 150
N ⟵ 0
Tant que U < 220
U ⟵ 0,8 × U + 45
N ⟵ N+1
Fin Tant que
```

On s'intéresse, à la fin de son exécution, à la valeur de la variable N de l'algorithme.

(a) C'est le deuxième algorithme qui permet d'affecter, en fin d'exécution, à la variable N le plus petit entier naturel n vérifiant $u_n \geqslant 220$.
Le premier algorithme affecte, en fin d'exécution, la valeur 0 à la variable N. En effet, la boucle ne sera jamais exécuté et le rang vaudra 0 puisque la condition $U \geqslant 220$ n'est pas vérifiée pour la valeur initiale 150.

(b) En utilisant la calculatrice, on obtient :





On observe alors que le premier terme de la suite ayant une valeur supérieur ou égale à 220 a pour rang 13.

Exercice 3 :

On considère l'algorithme suivant :

```
a ⟵ 2
Pour i allant de 0 à 5
a ⟵ a × 2
Fin Pour
```

1 Lors de son exécution pas à pas, les différentes valeurs prises par la variable a, sont :

i	a
	2
0	4
1	8
2	16
3	32
4	64
5	128

2 Voici les expressions possibles de la suite (u_n) :

(a) $u_n = 2^{n+1}, \quad \forall n \in \mathbb{N}.$

(b) $\begin{cases} u_0 = 2 \\ u_{n+1} = 2u_n \quad \forall n \in \mathbb{N}. \end{cases}$

(c) $\begin{cases} u_0 = 2 \\ u_n = 2u_{n-1} \quad \forall n \in \mathbb{N}^*. \end{cases}$

Exercice 4 :

On considère l'algorithme suivant :

```
a ⟵ −1
Pour i allant de 0 à 4
a ⟵ a× 2 - i+1
Fin Pour
```

1 Lors de son exécution pas à pas, les différentes valeurs prises par la variable a, sont :

i	a
	-1
0	$-1\times 2+1=-1$
1	$-1\times 2-1+1=-2$
2	$-2\times 2-2+1=-5$
3	$-5\times 2-3+1=-12$
4	$-12\times 2-4+1=-27$

2 La suite (u_n) est définie par la relation de récurrence :

$$\begin{cases} u_0 = -1 \\ u_{n+1} = 2u_n - n + 1, \quad \forall\, n \in \mathbb{N}. \end{cases}$$

Exercice 5 :

On considère la suite (u_n) définie sous Python par :

```
def terme_u(n):
    u=0
    for k in range(1,n+1):
        u=5*u-1
    return u
```

1. Voici les quatre premiers termes de la suite (u_n) :
$u_0 = 0;\ u_1 = 5\times 0 - 1 = -1;\ u_2 = 5\times(-1) - 1 = -6;\ u_3 = 5\times(-6) - 1 = -31.$

2. La suite (u_n) est définie par la relation de récurrence :

$$\begin{cases} u_0 = 0 \\ u_{n+1} = 5u_n - 1, \quad \forall\, n \in \mathbb{N}. \end{cases}$$

Exercice 6 :

On considère la suite arithmétique (u_n) dont le terme de rang n s'obtient sous Python par :

```
def suite(n):
    u=10
    for k in range(1,n+1):
        u=u+4
    return(u)
```

1. $u_0 = 10$ le premier terme u_0 et 4 est la raison de cette suite arithmétique.

2. La formule explicite du n-ème terme de la suite (u_n) est donnée par : $u_n = nr + u_0 = 4n + 10$.

3. Pour tout $n \in \mathbb{N}$, on a :

$$\begin{aligned} u_n \geqslant 1000 \quad &\Leftrightarrow \quad 4n + 10 \geqslant 1000 \\ &\Leftrightarrow \quad 4n \geqslant 1000 - 10 \\ &\Leftrightarrow \quad 4n \geqslant 990 \\ &\Leftrightarrow \quad n \geqslant \frac{990}{4} \\ &\Leftrightarrow \quad n \geqslant 247,5. \end{aligned}$$

Ainsi, $u_n \geqslant 1000$ à partir du rang 248.

4. Voici le programme Python permettant de répondre à la question précédente.

```
n=0
u=10
while u<1000 :
     n=n+1
     u=u+4
print('n=',n)
```

Exercice 7 :

On considère la suite (u_n) définie sous Python par :

```python
from math import*
def listesuite(n):
    u=1
    L=[u]
    for i in range(1,n+1):
        u=sqrt(2*u)
        L.append(u)
    return L
```

1 La suite (u_n) est définie par la relation de récurrence :

$$\begin{cases} u_0 = 1 \\ u_{n+1} = \sqrt{2u_n}, \quad \forall\, n \in \mathbb{N}. \end{cases}$$

2 Ce programme permet de retourner de liste des n premiers termes de la suite (u_n).

3 La suite (u_n) semble converger vers 2.

Exercice 8 :

On doit à Fibonacci, mathématicien italien du XIII^e siècle le problème suivant. « Un homme met un couple de lapins dans un lieu clos. Combien de couples obtient-on en un an si chaque couple engendre tous les mois un nouveau couple à compter du troisième mois de son existence ? » On considère la suite (u_n) définie pour tout entier naturel n comme le nombre de couples présents le n-ième mois.
On pose $u_0 = 0$, on a donc $u_1 = 1$ et, pour tout entier naturel n, $u_{n+2} = u_{n+1} + u_n$.

1 Voici le programme Python permettant d'afficher la liste des termes de la suite de u_0 à u_n.

```python
def fibonacci(n):
    a=0;b=1;L=[a]
    for i in range(1,n+1):
        c=a+b
        a=b
        b=c
        L.append(a)
    return(L)
```

2 Ce programme permet de calculer : fibonacci(12) = 144.

3 Il y aura 144 couples en un an.

Exercice 9 :

1 On considère la suite (v_n) définie par $v_0 = 1$ et $v_{n+1} = \dfrac{2}{3}v_n$ pour tout entier naturel n.

(a) (v_n) est une géométrique de raison $\dfrac{2}{2}$ et de premier terme 1.

(b) Pour tout entier naturel n, $v_n = v_0 \times \left(\dfrac{2}{3}\right)^n = \left(\dfrac{2}{3}\right)^n$.

(c) La somme S des dix premiers termes de la suite (v_n), est donnée par :

$$\begin{aligned}
S &= \sum_{k=0}^{9}\left(\frac{2}{3}\right)^k \\
&= \frac{\left(\frac{2}{3}\right)^{10}-1}{\frac{2}{3}-1} \\
&= \frac{\left(\frac{2}{3}\right)^{10}-1}{-\frac{1}{3}} \\
&= 3\left(1-\left(\frac{2}{3}\right)\right)^{10} \\
&= \frac{58025}{19683}.
\end{aligned}$$

2 On modélise une suite (w_n) sous Python.

```python
def terme(n):
    w=4
    for i in range(n):
        w=2*w-3
    return w
```

(a) Selon ce programme, terme(5) renvoie 35.

(b) Compléter la fonction Sommetermes(n), écrite en langage Python, pour renvoyer la somme des n premiers termes.

```python
def Sommetermes(n):
    w=4
    S=4
    for i in range(1,n):
        w=2*w-3
        S=S+w
    return S
```

Exercice 10 :

Pour chacune des affirmations proposées, indiquer si elle est vraie ou fausse en justifiant votre réponse.

Soit la suite (u_n) définie pour tout entier naturel n par : $\begin{cases} u_0 = 14 \\ u_{n+1} = 2u_n - 5. \end{cases}$

Soit la suite (t_n) définie pour tout entier naturel n par $t_n = u_n - 5$.

1 La suite (t_n) est géométrique. En effet, pour tout $n \in \mathbb{N}$,

$$\begin{aligned}
t_{n+1} &= u_{n+1} - 5 \\
&= 2u_n - 5 - 5 \\
&= 2u_n - 10 \\
&= 2(u_n - 5) \\
&= 2t_n.
\end{aligned}$$

$\boxed{2}$ (t_n) est une suite géométrique de raison 2 et de premier terme $t_0 = u_0 - 5 = 14 - 5 = 9$. Ainsi, pour tout $n \in \mathbb{N}$,
$t_n = t_0 \times 2^n = 9 \times 2^n$ et $u_n = t_n + 5 = 9 \times 2^n + 5$.

$\boxed{3}$ Pour tout $n \in \mathbb{N}^*$, on a :

$$\begin{aligned}(8\times 1+3)+(8\times 2+3)+\cdots+(8\times n+3) &= \sum_{k=1}^{n}(8\times k+3)\\ &= 8\sum_{k=1}^{n}\times k+\sum_{k=1}^{n}3\\ &= 8\frac{n(n+1)}{2}+3n\\ &= 4n(n+1)+3n\\ &= n(4n+4+3)\\ &= n(4n+7).\end{aligned}$$

Exercice 11 :

Une entreprise décide de verser à ses ingénieurs une prime annuelle de 500 euros.
Pour ne pas se dévaluer, il est prévu que chaque année la prime augmente de 2% par rapport à l'année précédente.
On note (u_n) la suite des primes avec $u_1 = 500$.

$\boxed{1}$ $u_2 = 1{,}02u_1 = 510$.
$u_3 = 1{,}02u_2 = 520{,}2$.

$\boxed{2}$ Pour tout $n \in \mathbb{N}^*$, $u_{n+1} = 1{,}02u_n$.
Ainsi, la suite (u_n) est géométrique de raison 1,02 et de premier terme $u_1 = 500$.

$\boxed{3}$ Un ingénieur compte rester 20 ans dans cette entreprise à partir du moment où est versée la prime.

(a) (u_n) étant géométrique, pour tout $n \in \mathbb{N}^*$, $u_{n+1} = 500 \times (1{,}02)^{n-1}$.
Dès lors, $u_{20} = 500 \times (1{,}02)^{19} \approx 728{,}41$. Par conséquent, après 20 ans passés dans l'entreprise, la prime touchée la 20ème année est de 728,41 euros.

(b) Pour tout $n \in \mathbb{N}^*$, on a :

$$\begin{aligned}S &= \sum_{k=1}^{20}u_k\\ &= 500\sum_{k=1}^{20}(1{,}02)^{k-1}\\ &= 500\frac{1-1{,}02^{20}}{1-1{,}02}\\ &= -25000\times\left(1-1{,}02^{20}\right)\\ &\approx 12148{,}68.\end{aligned}$$

Ainsi, les primes touchées sur les 20 années s'élève à environ 12148,68 euros.