# Généralités sur les suites.

***Leonardo Pisano** dit **Fibonacci ( 1175-1250)** mathématicien italien. Dans son ouvrage « **Liber Abaci** » il développe la première approche de la notion de suites numériques.*

## I. Définition.

*Voici un problème posé en 1202 par **Leonardo Pisano** dit **Fibonacci**.*
*Un fermier achète un couple de bébés lapins. Après 2 mois, ce couple commence à se reproduire et donne naissance à un nouveau couple de lapins qui au bout de 2 mois, se reproduira à son tour. Chaque couple donnant naissance à un nouveau couple tous les mois, lesquels commencent à se reproduire au bout de 2 mois.*

| Nombre de mois | Bébés | ados | adultes | total |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 2 | 1 | 0 | 1 | 2 |
| 3 | 1 | 1 | 1 | 3 |
| 4 | 2 | 1 | 2 | 5 |
| 5 | 3 | 2 | 3 | 8 |

*On crée ainsi une suite de nombres :*
{1 ; 1 ; 2 ; 3 ; 5 ; 8 ; 13 ; 21 ; 34 ; ...} *On indexe chacun des nombres de la liste. On Note $u_0$ le nombre de lapins le premier mois, $u_1$ celui le deuxième mois, etc... On a donc :*
$u_0 = 1$ ; $u_1 = 1$ ; $u_2 = 2$ ; $u_3 = 3$
$u_4 = 5$ ; $u_5 = 8$

*On note $(u_n)$ l'ensemble des nombres de cette suite de nombres. On dit que $u_5$ est le terme de la suite $(u_n)$ de rang 5... Attention : $u_1$ n'est pas le premier terme puisqu'on a commencé à $u_0$*
*On a ainsi défini une **suite numérique**.*
*On peut lui associer une **fonction** définie de $\mathbb{N}$ dans $\mathbb{R}$ par : $u : n \to u(n) = u_n$*

**Définition :** Une suite numérique $(u_n)$ est une liste ordonnée de nombres réels telle qu'à tout entier $n$ on associe un nombre réel noté $u_n$. Le nombre $u_n$ est appelé le terme de rang $n$ de cette suite (ou d'indice $n$).

*Attention, ne pas confondre $(u_n)$ qui est la suite, la liste de tous les nombres*
*et $u_n$ qui est un nombre, celui dans la liste qui a pour rang $n$*

## II. Deux différents modes de création d'une suite :

### ☺ Suites définies par une formule explicite : $u_n = f(n)$.

☑ *Savoir-faire : Savoir calculer un terme d'une suite définie en fonction de $n$ :*

*On considère la suite $(u_n)$ définie par : pour tout $n$ de $\mathbb{N}$, $u_n = 2n^2 + 3$.*
*Calcule $u_0$, $u_1$, $u_2$, $u_5$, $u_{100}$*

$u_0 = 2 \times 0^2 + 3 = 3$
$u_1 = 2 \times 1^2 + 3 = 5$
$u_2 = 2 \times 1^2 + 3 = 11$
$u_5 = 2 \times 5^2 + 3 = 53$
$u_{100} = 2 \times 100^2 + 3 = 20\,003$

☑ *Savoir-faire : Savoir jouer avec les indices :*

*Exprime en fonction de $n$ : $u_{n+1}$, $u_{2n}$, $u_{2n-1}$, $2u_n + 1$, $-u_{n+1} + 3$.*

$u_{n+1} = 2 \times (n+1)^2 + 3 = 2n^2 + 4n + 5$
$u_{2n} = 2 \times (2n)^2 + 3 = 8n^2 + 3$
$u_{2n+1} = 2 \times (2n+1)^2 + 3 = 8n^2 + 8n + 1$
$2u_n + 1 = 2(2n^2 + 3) + 1 = 4n^2 + 7$
$-u_{n+1} = -(2 \times (n+1)^2 + 3) + 3$
$= -2n^2 - 4n - 2$

## ☺ Suites définies par récurrence: $u_{n+1} = f(u_n)$.

☑ *Savoir-faire : Savoir calculer un terme d'une suite définie par récurrence :*

*On considère la suite $(u_n)$ définie par : $u_0 = 3$ et pour tout $n$ de $\mathbb{N}$, $u_{n+1} = 2\,u_n + 5$.*

*Calcule $u_1$, $u_2$, $u_3$, $u_4$, $u_{100}$*

$u_1 = 2 \times u_0 + 5 = 2 \times 3 + 5 = 11$ $u_4 = 2 \times u_3 + 5 = 2 \times 59 + 5 = 123$

$u_2 = 2 \times u_1 + 5 = 2 \times 11 + 5 = 27$ $u_{100} =$ impossible

$u_3 = 2 \times u_2 + 5 = 2 \times 27 + 5 = 59$ d'abord calculer $u_{99}$

*Contrairement à une suite définie par une formule explicite, on ne peut pas connaitre $u_{100}$ sans connaître $u_{99}$. Cependant il est possible d'écrire un algorithme.*

☑ *Savoir-faire : Savoir écrire un algorithme pour calculer un terme d'une suite définie par récurrence :*

*On considère la suite $(u_n)$ définie par : $u_0 = 3$ et pour tout $n$ de $\mathbb{N}$, $u_{n+1} = 2\,u_n + 5$. Calcule $u_{100}$*

*En langage naturel*

Initialisation
3 → u
Traitement
Pour i allant de 1 à n, faire
2u + 5 → u
Fin du pour
Sortie
Afficher u.

*En Python*

```
def suite(n):
    u=3
    for i in range (1,n+1):
        u=2*u+5
    return(u)
```

Un algorithme nous permet de calculer rapidement les termes successifs et donc d'obtenir u(100)

*Remarque : On considère la suite $(u_n)$ définie par : $u_0 = 1$, $u_1 = 1$ et pour tout $n$ de $\mathbb{N}$, $u_{n+2} = u_{n+1} + u_n$. Calcule les cinq premiers termes de cette suite.*

C'est la définition par récurrence de la suite de Fibonacci. $u_0 = 1$, $u_1 = 1$, $u_2 = u_1 + u_0 = 1 + 1 = 2$, $u_3 = u_2 + u_1 = 1 + 2 = 3$

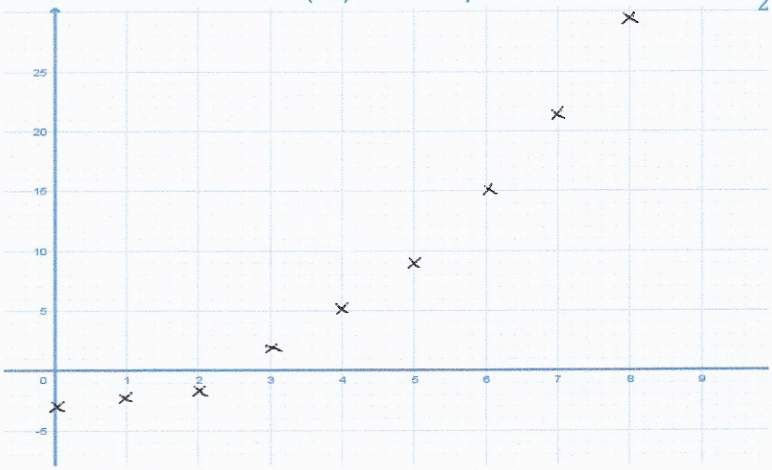
*Lorsqu'on génère une suite par une relation de récurrence, chaque terme de la suite s'obtient à partir d'un ou plusieurs des termes précédents.*

## III. Représentation graphique d'une suite :

**Définition** : Dans un repère du plan, on représente une suite $(u_n)$ par le nuage de points de coordonnées $(n\,;\,u_n)$.

☑ *Savoir-faire : Savoir représenter graphiquement une suite numérique :*

*On considère la suite $(u_n)$ définie par : $\forall n \in \mathbb{N}$, $u_n = \frac{n^2}{2} - 3$. Représenter la suite $(u_n)$.*



| | A | B |
|---|---|---|
| 1 | n | $u_n$ |
| 2 | 0 | -3 |
| 3 | 1 | -2,5 |
| 4 | 2 | -1 |
| 5 | 3 | 1,5 |
| 6 | 4 | 5 |
| 7 | 5 | 9,5 |
| 8 | 6 | 15 |
| 9 | 7 | 21,5 |
| 10 | 8 | 29 |
| 11 | 9 | 37,5 |
| 12 | 10 | 47 |
| 13 | | |

$\frac{A^2}{2} - 3$

## IV. Sens de variation d'une suite numérique.

*On considère la suite $(u_n)$ définie par : $\forall n \in N$, $un = \frac{n^2}{2} - 3$. En observant sa représentation graphique, on remarque que $u_0 \leq u_1$ ; $u_1 \leq u_2$ ; $u_2 \leq u_3$ ; $u_3 \leq u_4$. On a l'impression que $\forall n \in N$, $u_{n+1} \geq \dots u_n$. Peut-on le prouver ?*

Etudions le signe de $U_{n+1} - U_n$ / $U_{n+1} - U_n = \left[\frac{(n+1)^2}{2} - 3\right] - \left(\frac{n^2}{2} - 3\right)$

$U_{n+1} - U_n = \frac{n^2 + 2n + 1}{2} - 3 - \frac{n^2}{2} + 3 = \frac{2n+1}{2}$ donc $U_{n+1} > U_n$

$\forall n \in \mathbb{N}$ donc $(U_n)$ est croissante

**Définition** : On dit qu'une suite $(u_n)$ définie sur N est :

- ♦ <u>croissante</u> si et seulement si pour tout entier naturel n, $u_{n+1} \geq \dots u_n$.
- ♦ <u>décroissante</u> si et seulement si pour tout entier naturel n, $u_{n+1} \leq \dots u_n$.
- ♦ <u>constante</u> si et seulement si pour tout entier naturel n, $u_{n+1} = \dots u_n$.

### ☑ *Savoir-faire : Savoir étudier les variations d'une suite :*

*Pour tout $n$ de $\mathbb{N}$, on donne la suite $(u_n)$ définie par : $un = \frac{1}{n+1}$. Étudie les variations de $(u_n)$.*

*Démontrer que la suite $(u_n)$ est croissante à partir d'un certain rang.*

Conjecture sur les premiers termes : $U_0 = 1$, $U_1 = \frac{1}{2}$, $U_2 = \frac{1}{3}$, $U_3 = \frac{1}{4}$, $U_4 = \frac{1}{5}$, $U_5 = \frac{1}{6}$

Démonstration : $U_{n+1} - U_n = \frac{1}{n+2} - \frac{1}{n+1}$ $U_{n+1} - U_n = \frac{n+1}{(n+2)(n+1)} - \frac{n+2}{(n+2)(n+1)}$

$U_{n+1} - U_n = \frac{n+1-(n+2)}{(n+2)(n+2)} = \frac{-1}{(n+2)(n+1)}$ donc $U_{n+1} \leq 0$ donc

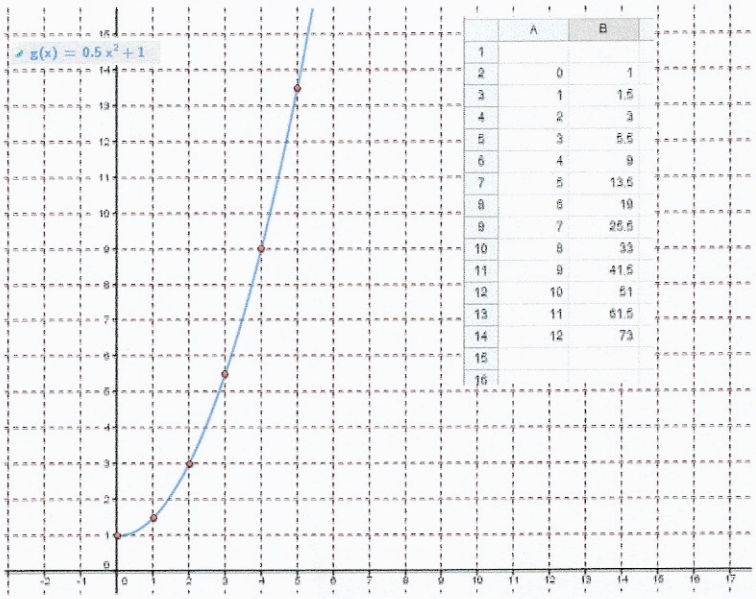
$U_{n+1} \leq U_n$ donc $(U_n)$ est décroissante

Remarque : Pour certaine suites l'inégalité $u_{n+1} > u_n$, n'est vraie que pour n≥p, on dit que la suite $U_n$ est croissante à partir du rang p

### ☺ Suites et fonctions : $u_n = f(n)$.

**Définition** : Soit une fonction $f$ définie sur [ 0 ; +∞ [ et une suite numérique $(u_n)$ définie sur $N$ par $u_n = f(n)$.

- ♦ Si la fonction $f$ est croissante sur [ 0 ; +∞ [ alors la suite $(u_n)$ est croissante.
- ♦ Si la fonction $f$ est décroissante sur [ 0 ; +∞ [ alors la suite $(u_n)$ est décroissante.



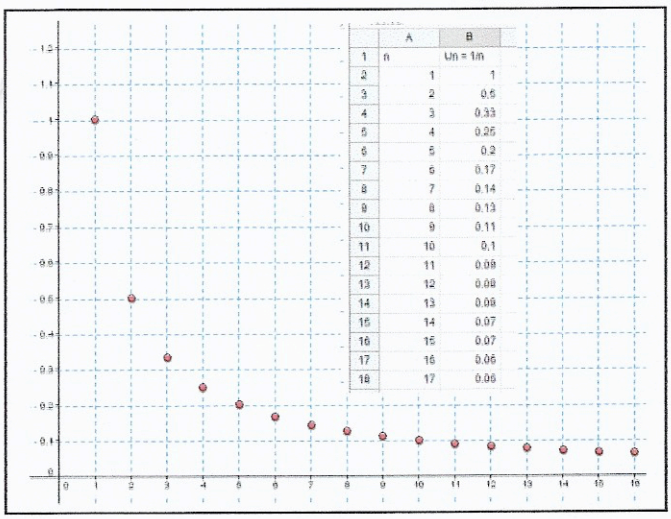

| | A | B |
|---|---|---|
| 1 | | |
| 2 | 0 | 1 |
| 3 | 1 | 1.5 |
| 4 | 2 | 3 |
| 5 | 3 | 5.5 |
| 6 | 4 | 9 |
| 7 | 5 | 13.5 |
| 8 | 6 | 19 |
| 9 | 7 | 25.5 |
| 10 | 8 | 33 |
| 11 | 9 | 41.5 |
| 12 | 10 | 51 |
| 13 | 11 | 61.5 |
| 14 | 12 | 73 |
| 15 | | |
| 16 | | |

Pour une suite définie en fonction de n $U_n = f(n)$ le nuage de points est sur la courbe de la fonction f donc il est normal que la suite et la fonction aient les mêmes variations. Attention cela ne fonctionne pas pour les suites définies par récurrence.

## IV. Notion de limite d'une suite.

*Etudier la limite d'une suite $(u_n)$, c'est se demander ce que deviennent les nombres $u_n$ lorsque n devient de plus en plus grand.*

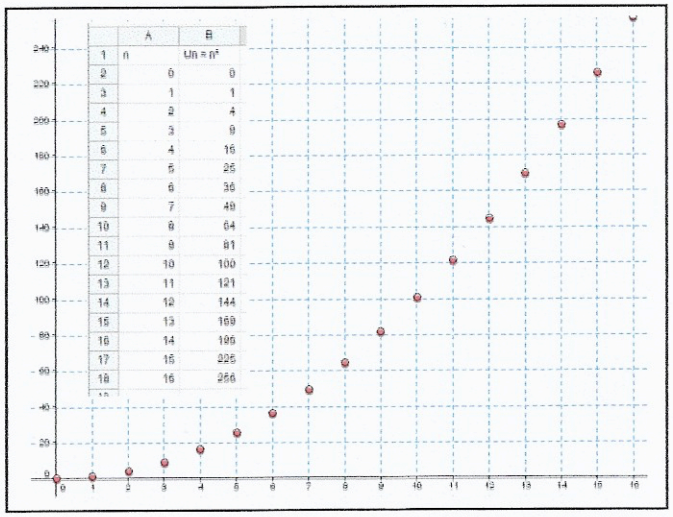
*On considère la suite $(u_n)$ définie par $u_n = \frac{1}{n}$. $\forall n \in N^*$.*



| | A | B |
|---|---|---|
| 1 | n | Un = 1/n |
| 2 | 1 | 1 |
| 3 | 2 | 0,5 |
| 4 | 3 | 0,33 |
| 5 | 4 | 0,25 |
| 6 | 5 | 0,2 |
| 7 | 6 | 0,17 |
| 8 | 7 | 0,14 |
| 9 | 8 | 0,13 |
| 10 | 9 | 0,11 |
| 11 | 10 | 0,1 |
| 12 | 11 | 0,09 |
| 13 | 12 | 0,08 |
| 14 | 13 | 0,08 |
| 15 | 14 | 0,07 |
| 16 | 15 | 0,07 |
| 17 | 16 | 0,06 |
| 18 | 17 | 0,06 |

On conjecture que plus n grandit plus $U_n$ se rapproche de 0. Et on peut être aussi proche que l'on veut de 0. $\lim_{n \to +\infty} U_n = 0$

*On considère la suite $(u_n)$ définie par $u_n = n^2$.*



| | A | B |
|---|---|---|
| 1 | n | Un = n² |
| 2 | 0 | 0 |
| 3 | 1 | 1 |
| 4 | 2 | 4 |
| 5 | 3 | 9 |
| 6 | 4 | 16 |
| 7 | 5 | 25 |
| 8 | 6 | 36 |
| 9 | 7 | 49 |
| 10 | 8 | 64 |
| 11 | 9 | 81 |
| 12 | 10 | 100 |
| 13 | 11 | 121 |
| 14 | 12 | 144 |
| 15 | 13 | 169 |
| 16 | 14 | 196 |
| 17 | 15 | 225 |
| 18 | 16 | 256 |

On a l'impression que $U_n$ peut être aussi grand que l'on veut ($\forall A, \exists n_0 / U_n \geq n_0 \Rightarrow U_n \geq A$). Donc $\lim_{n \to +\infty} U_n = +\infty$

**Définition :** On dit qu'une suite $(u_n)$ converge vers un nombre L si, à partir d'un certain rang, les termes sont aussi proches qu'on le souhaite du nombre L. On le note $\lim\limits_{n \to +\infty} u_n = L$.

**Définition :** On dit qu'une suite $(u_n)$ à pour limite $+\infty$, si, à partir d'un certain rang, les termes sont plus grands que n'importe quel nombre qu'on a choisi. On le note $\lim\limits_{n \to +\infty} u_n = +\infty$.

**Remarque :** Certaines suites n'ont pas de limites : elles ne se rapprochent de rien. $(U_n) = (-1)^n$, $U_n = \cos(n)$, $U_n = (-2)^n$

☑ *Savoir-faire : Savoir utiliser un algorithme Seuil :*

*On considère la suite $(u_n)$ définie par $u_0 = 2$ et $\forall n \in N$, $u_{n+1} = 4\,u_n$. Cette suite est croissante et* admet pour limite.....Ecrire un algorithme permettant de préciser le rang á partir duquel $u_n$ est supérieure à S.

```
Initialisation
S ← ?
N ← 0
U ← 2
Traitement
    Tant que U < S
        N ← 1
        U ← 4n
    Fin du Tant que
Sortie
    Afficher N.
```

```
def seuil(s):
    n=0
    u=2
    while u<s:
        n=n+1
        u=4*n
    return(n)
```