Suites numériques, partie 1

1. Définitions

Définition. *Une suite numérique est une liste ordonnée et numérotée de nombres.*
Couramment, elle est numérotée à partir de 0 ou 1.
Les éléments de cette liste sont appelés termes.
Le numéro de chaque élément est appelé son indice.

Exemple

Voici les 6 notes (sur 10) du trimestre d'un élève :

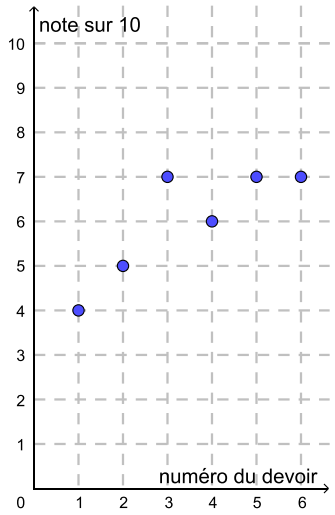
$$4\ ;\ 5\ ;\ 7\ ;\ 6\ ;\ 7\ ;\ 7.$$

Ces notes ont été présentées dans l'ordre chronologique du trimestre.
Si l'on appelle u la suite des notes obtenues, on a donc

$$u = (4; 5; 7; 6; 7; 7).$$

En convenant que la première note porte le numéro 1, le terme d'indice 4 de cette suite est 6, ce que l'on note $u_4 = 6$. Concrètement cela veut dire que la 4e note du trimestre est 6.

La représentation graphique de la suite, c'est-à-dire l'ensemble des points de coordonnées $(n; u_n)$ où n est un entier vérifiant $1 \leq n \leq 6$, est donnée ci-contre.



2. Modes de génération d'une suite

➢ *Par une formule explicite*

La suite est définie directement par une formule en fonction de n.

Exemple

- Soit la suite $(u_n)_{n\geq 1}$ définie par $u_n = \frac{1}{n}$.
  Cette suite est définie pour $n \geq 1$. Le terme d'indice 1 est $u_1 = \frac{1}{1} = 1$, le terme d'indice 10 est $u_{10} = \frac{1}{10}$.
- Soit la suite $(v_n)_{n\geq 0}$ définie par $u_n = 2n$. C'est la suite des entiers naturels pairs. On a $u_0 = 0$, $u_1 = 2$, etc.

➢ *Par une relation de récurrence*

Une suite est définie par récurrence quand elle définie par la donnée :

- de son premier terme ;
- d'une relation qui permet de calcul un terme à partir du précédent. Cette relation est appelée relation de récurrence.

Exemple

Une association caritative a constaté que, chaque année, 20 % des donateurs de l'année précédente ne renouvelaient pas leur don mais que, chaque année, 300 nouveaux donateurs effectuaient un don.
On note u_n le nombre de donateurs lors de la $n^{\text{ième}}$ année.
On suppose que $u_1 = 1000$.

Le nombre de donateurs de la deuxième année est constitué de 80 % des donateurs de la première année qui renouvellent leur don, soit $0{,}8 \times 1000 = 800$ personnes, auxquels s'ajoutent les 300 nouveaux donateurs. Ainsi

$$u_2 = 0{,}8 \times u_1 + 300 = 0{,}8 \times 1000 + 300 = 1100.$$

Avec le même raisonnement, on démontre que pour tout entier $n \geq 1$, on a

$$u_{n+1} = 0{,}8u_n + 300.$$

Ainsi

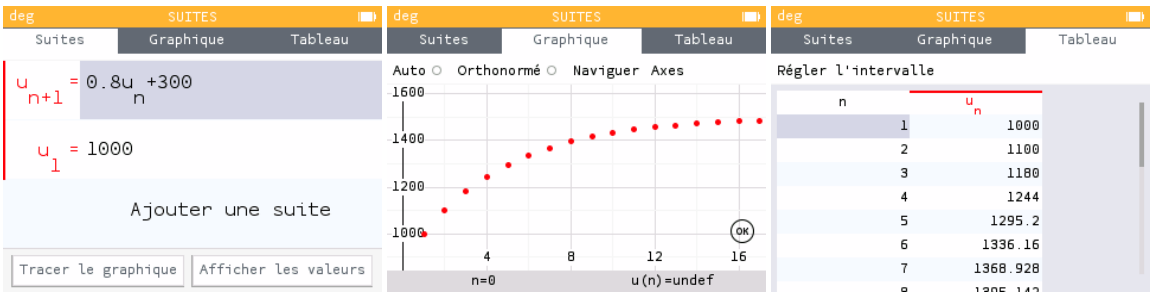
$$u_3 = 0{,}8 \times u_2 + 300 = 0{,}8 \times 1100 + 300 = 1180,$$

etc.

❖ Utilisation de la calculatrice

On a utilisé ci-dessous la calculatrice NumWorks pour calculer les premiers termes de la suite. En descendant dans le tableau de valeurs, on constate que les valeurs sont de plus en plus grandes, on dit que la suite est croissante.

La représentation graphique de la suite, c'est-à-dire l'ensemble des points de coordonnées $(n; u_n)$ avec $n \geq 1$, permet de visualiser ce qui a été observé dans le tableau.



❖ Utilisation du tableur

Le tableur est particulièrement adapté au calcul de suites.

Après avoir saisi le numéro des années dans la colonne A ainsi que `1000` dans la cellule B2, il suffit d'étendre la formule `=0,8*B2+300`, saisie en B3, dans la colonne B pour que les termes de la suite soient calculés.

	A	B
1	**Année**	**Nombre de donateurs**
2	1	1000
3	2	1100
4	3	1180
5	…	…

❖ Algorithme pour calculer u_n et Python

Ci-dessous un algorithme permettant de calculer u_n pour $n \geq 1$ et sa traduction en Python sous forme d'une fonction.

```
d ← 1000
Pour k de 2 à n
    d ← 0,8 × d + 300
Fin Pour
Retourner d
```

```
def donateurs(n):
    d = 1000
    for k in range(2,n+1):
        d = 0.8*d + 300
    return d
```

Exemple d'appel en Python :

```
>>> donateurs(2)
1100.0
```

```
>>> donateurs(100)
1499.9910797019209
```

En Python, une liste est une variable qui contient plusieurs variables, elle se note entre crochets : […]. Par exemple, la liste des deux premiers termes de la suite s'obtient ainsi :

```
>>> L=[donateurs(1), donateur(2)]

>>> L
[1000, 1100.0]
```

On peut « concaténer » deux listes en une seule en utilisant l'opérateur +, cela permet par exemple d'ajouter un élément à la fin d'une liste :

```
>>> L=L+[donateurs(3)]

>>> L
[1000, 1100.0, 1180.0]
```

En utilisant cette idée, voici un algorithme et sa traduction en Python (sous forme d'une fonction) qui génère la liste des n premiers termes de la suite.

```
L ← liste vide
Pour k de 1 à n
    Ajouter donateurs(k) dans L
Fin Pour
```

```
def premierstermes(n):
    L = []
    for k in range (1,n+1):
        L = L + [donateurs(k)]
    return L
```

Exemple d'appel :

```
>>> premierstermes(4)
[1000, 1100.0, 1180.0, 1244.0]
```

La fonction `premierstermes(n)` n'est pas optimisée car les calculs de `donateurs(k)`

« repartent de 1000 » au lieu d'utiliser le terme d'avant dans la suite. Voici une version bien plus efficace.

```
def premierstermes(n):
    L = [1000]
    d = 1000
    for k in range (2,n+1):
        d = 0.8*d + 300
        L = L + [d]
    return L
```

3. Sens de variation d'une suite

Définition. *La suite u est dite <u>décroissante</u> si pour tout n,* $u_{n+1} \leq u_n$.
La suite u est dite <u>croissante</u> si pour tout n, $u_{n+1} \geq u_n$.

Exemple

En se référant aux deux exemples du paragraphe 1 :

- la suite des notes de l'élève n'est ni croissante, ni décroissante ;
- la suite du nombre de donateurs est croissante.