Composition de taux

Le problème

Lorsqu'une quantité varie de t_1% puis de t_2%, de quel taux T a-t-elle varié ?

Beaucoup seraient tentés d'avancer que l'évolution globale est de $(t_1 + t_2)$ %.
Mais il n'en est rien, ce n'est pas aussi simple.

Résolution mathématique

Notons q_0 la quantité de départ.
Après la première variation, la quantité devient : $q_1 = (1 + t_1 / 100) \times q_0$.
Après la deuxième variation, la quantité devient :
$q_2 = (1 + t_2 / 100) \times q_1$, d'où $q_2 = (1 + t_1 / 100) \times (1 + t_2 / 100) \times q_0$.
On note CM = $(1 + t_1 / 100) \times (1 + t_2 / 100)$ le coefficient multiplicateur qui permet de passer directement de q_0 à q_2. CM = $1 + t_1 / 100 + t_2 / 100 + t_1 \times t_2 / 10\,000$.
Si l'on note T l'évolution globale en pourcentage, T est donc le nombre tel que
$1 + T / 100 = CM$, et donc $T = 100 (CM - 1) = (t_1 + t_2 + t_1 \times t_2 / 100)$ %.

Dans le cas d'un taux d'évolution négatif (ce qui correspond à une baisse), le résultat, bien entendu, est le même.

Les programmes

Python

Le programme « evolutions », ci-dessous à gauche, très court, demande d'introduire les variables t1 et t2 pour obtenir le résultat.
Voici (ci-dessous à droite) les résultats obtenus en introduisant les taux 10 et 20, puis les taux –10 et 20.

le programme

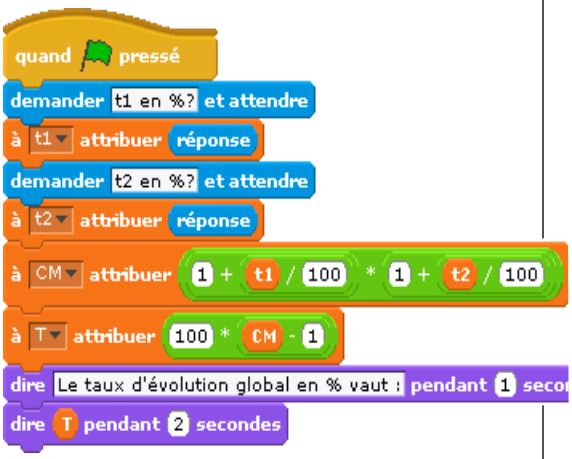
```
>>> def evolutions(t1,t2):
        CM=(1+t1/100)*(1+t2/100)
        T=100*(CM - 1)
        print('Taux global = ',T)
```

le résultat

```
>>> evolutions(10,20)
Taux global =  32.0
>>> evolutions(-10,20)
Taux global =  8.0
```

Scratch

Le fait de mettre le résultat du produit $(1 + t_1 / 100) \times (1 + t_2 / 100)$ dans une variable nommée CM permet d'alléger la syntaxe.



AlgoBox

Au début du programme, on définit les quatre variables $t1$, $t2$, CM et T. Les valeurs respectives de $t1$ et $t2$ seront introduites par l'utilisateur au cours de l'exécution du programme. La variable intermédiaire CM correspond au coefficient multiplicateur. On pourrait court-circuiter le calcul de cette valeur pour passer directement au calcul de T, mais, pour faciliter la compréhension de la construction du programme, il a été choisi de la laisser.

```
VARIABLES
    t1 EST_DU_TYPE NOMBRE
    t2 EST_DU_TYPE NOMBRE
    CM EST_DU_TYPE NOMBRE
    T EST_DU_TYPE NOMBRE
DEBUT_ALGORITHME
    LIRE t1
    LIRE t2
    CM PREND_LA_VALEUR (1+t1/100)*(1+t2/100)
    T PREND_LA_VALEUR 100*(CM-1)
    AFFICHER "Le taux global en % est : "
    AFFICHER T
FIN_ALGORITHME
```

Prolongement : taux d'évolution moyen

Lorsqu'une quantité varie de t_1 % puis de t_2 %, le taux d'évolution moyen est une évolution constante qui, appliquée deux fois, fournirait le même résultat.
Autrement dit, si t désigne ce taux moyen exprimé en %, on a :
$(1 + t / 100)^2 = (1 + t_1 / 100) \times (1 + t_2 / 100)$.

En conservant la même notation, on impose dans la définition du taux moyen :
$1 + t / 100 \geqslant 0$, et donc : $(1 + t / 100)^2 = CM$, d'où $t = 100\left(\sqrt{CM} - 1\right)$.

Avec Python, la fonction racine carrée (en anglais : *square root*) doit être activée avant de taper le programme. La syntaxe pour ce faire est la suivante : **>>> from math import sqrt.**

le programme

```
>>> from math import sqrt
>>> def tauxmoyen(t1,t2):
        CM=(1+t1/100)*(1+t2/100)
        t=100*(sqrt(CM)-1)
        print('Taux moyen=',t)
```