FONCTIONS RÉCURSIVES

L'essentiel

- Une fonction peut dans certains cas s'appeler elle-même, on dit alors que c'est une **fonction récursive**.
- Elle respecte alors les conditions suivantes :
  - elle contient un cas d'arrêt (cas de base) ;
  - elle doit pouvoir se ramener au cas de base en modifiant ses paramètres à chaque appel ;
  - elle doit s'appeler elle-même.
- Un exemple : Fonction récursive, basée sur le fait que : $\begin{cases} a^0 = 1 \\ \forall n \in \mathbb{N}^*,\ a^n = a \times a^{n-1} \end{cases}$

```
def puissance(a,n) :
    """
    Entrée a réel non nul et n entier naturel
    Sortie a^n
    """
    if n == 0 :      # Cas d'arrêt
        return 1
    else :           # Appel récursif : a^n = a * a^(n-1)
        return a * puissance(a, n-1)
```

- Attention aux appels infinis : par exemple, que se passe-t-il si on utilise ici un exposant négatif?
- Remarque : les fonctions récursives sont souvent courtes à programmer puisqu'il suffit d'écrire les définitions de ce que vous voulez faire, par contre elle masque parfois des calculs longs et complexes (voir Exercices en autonomie).

Exercices du jour

Exercice 1: Créer une fonction récursive `suite` qui prend en entrée un entier naturel `n` et renvoie la valeur de u_n où (u_n) est la suite définie par :

$$\begin{cases} u_0 = \dfrac{1}{2} \\ \forall n \in \mathbb{N}, u_{n+1} = 3u_n + 2 \end{cases}$$

Exercice 2: Créer une fonction récursive `fibonaccirec` qui prend en entrée un entier n et calcule le n-ième terme de la suite de Fibonacci définie par :

$$\begin{cases} u_0 = u_1 = 1 \\ \forall n \in \mathbb{N},\ u_{n+2} = u_{n+1} + u_n \end{cases}$$

Proposer et effectuer des tests.
Essayez notamment `fibonaccirec(30)` et `fibonaccirec(50)`. Cela vous paraît-il long? Pourquoi?

Exercices en autonomie

Exercice 3: On convient que la longueur d'un nombre entier est son nombre de chiffres dans son écriture en base 10. Par convention, la longueur de 0 vaut 0.

1. Écrire une fonction récursive `long(n)` qui donne la longueur d'un nombre.
2. En déduire une fonction `decoupe(n)` qui retire à un entier naturel d'au moins 3 chiffres le premier et le dernier. Par exemple : `decoupe(1234)` donne `23`.

Exercice 4: Un entier naturel n est un palindrome s'il peut se lire dans les 2 sens (exemple : 12321). Écrire une fonction récursive `palindrome(n)` qui indique par un booléen si un entier est un palindrome. On pourra utiliser les fonctions de l'exercice 2.

Exercice 5: La touche $\boxed{*}$ du clavier est cassée... vous ne pouvez pas utiliser ce symbole dans cet exercice...

1. Écrire une fonction récursive `produit` qui reçoit deux entiers naturels et renvoie le produit de ces 2 entiers.
2. En déduire une fonction récursive `puissanceN` qui reçoit deux entiers naturels non simultanément nuls a et b et renvoie a^b.
3. En déduire enfin une fonction `puissance` qui réalise la même chose pour deux entiers a et b relatifs nun simultanément nuls.

Aide pour les exercices

Indication 3 1. On pourra se baser sur l'observation suivante :

$$\begin{aligned}\texttt{long(576)} &= 1 + \texttt{long(57)}\\ &= 1 + 1 + \texttt{long(5)}\\ &= 1 + 1 + 1 + \texttt{long(0)}\\ &= 1 + 1 + 1 + 0 = 3\end{aligned}$$

Indication 4 Pour s'aider, on peut se baser sur la remarque suivante :

$$n \in \mathbb{N} \text{ est un palindrome ssi}$$

$$\left|\begin{array}{l} n \text{ n'est composé que d'un chiffre} \\ \text{ou} \\ n \text{ est composé de 2 chiffres identiques} \\ \text{ou} \\ n \text{ est composé d'au moins 3 chiffres :} \begin{cases} \text{le premier et le dernier sont égaux} \\ \text{le nombre formé en retirant ces 2 chiffres est un palindrome} \end{cases} \end{array}\right.$$

Solutions des exercices