Exercice 1 (La fonction puissance)

Il s'agit de la multiplication répétée n fois de x avec lui-même, que l'on écrit habituellement ainsi :
$x_n = x.x.x.x...x.x$
Le facteur x est contenu n fois avec, par convention, $x_0 = 1$.
Pour écrire une version récursive du calcul de x_n, on va définir une fonction 'puissance(x,n)' ainsi :
on commence par déterminer le cas de base à cette opération : il s'agit du cas pour $n = 0$; pour définir ensuite la valeur de puissance(x, n) pour un entier n strictement positif, on utilise le résultat suivant :

si $n > 0$, $x_n = x \times x_{n-1}$

Ainsi, x_n s'obtient en multipliant x par x_{n-1}.
Une définition récursive de l'opération de puissance n-ième d'un nombre x est alors la suivante :

1. Écrire la fonction récursive *puissance*(x, n) qui calcule le nombre x_n pour tout entier naturel n.
2. Dessiner l'arbre d'appels de cette fonction lorsque $x = 3$ et $n = 5$.
3. Parcourir cet arbre d'appels « à rebours » pour illustrer le fonctionnement de la fonction puissance

```
def puissance(x,n):
    #A compléter
    return  #A completer
assert palindrome('antilope') == False
assert palindrome('radar') == True
```

Exercice 2 (Palindrome)

Les palindromes
On appelle palindrome un mot qui se lit dans les deux sens comme « été » ou « radar ».Écrire une fonction récursive palindrome qui teste si un mot est un palindrome.
• Entrée : Un mot (type str).
• Sortie : Un booléen égal à
True si le mot est un palindrome
False sinon Cas de base :
• si le mot est la chaîne vide, c'est un palindrome ;
• si le mot ne contient qu'une seule lettre, c'est un palindrome.

Dans les autres cas

Le mot est un palindrome si et seulement si la première et la dernière lettre sont égales et le mot tronqué de ses première et dernière lettres est un palindrome:

```
def palindrome(mot):
    #A compléter
    return  #A completer
#assert palindrome('antilope') == False
#assert palindrome('radar') == True
```

Exercice 3 (Anagramme)

L'anagramme d'un mot est un mot s'écrivant avec les mêmes lettres que le mot initial.Par exemple :
ironique et onirique ;
baignade et badinage ;
estival et vitales.

Ici, nous ne demandons pas que les mots envisagés soient des mots du dictionnaire. Ainsi 'abc' et 'bca' sont deux anagrammes.

Écrire une fonction python récursive anagramme qui renvoie la liste des anagrammes d'un mot.

- Entrée : Un mot (type str).
- Sortie : La liste des anagrammes du mot.

Principe .Cas de base :

- si le mot est une chaîne vide, la liste des anagrammes est alors constituée de l'unique chaîne vide ;
- si le mot n'a qu'une seule lettre, la liste de ses anagrammes ne contient qu'un seul élément : le mot lui-même.
  Dans les autres cas :

- on peut définir la liste des anagrammes d'un mot à partir de la liste des anagrammes du mot obtenu en enlevant la première lettre, en plaçant cette première lettre successivement dans toutes les positions dans chaque élément de la liste.

- Un exemple : prenons le cas d'un mot de trois lettres : 'abc'.
  • La première lettre est 'a'.
  • Le mot obtenu en enlevant la première lettre est 'bc' et la liste des anagrammes de 'bc' est ['bc','cb'].
  • Pour constituer la liste des anagrammes de 'abc', il suffit de placer la lettre 'a' dans toutes les positions possibles des anagrammes de 'bc'. On obtient la liste ['abc', 'bac', 'bca', 'acb', 'cab', 'cba'].
  • De la même façon, on obtiendra la liste de toutes les anagrammes du mot 'dabc' en glissant la lettre 'd ' dans toutes les positions possibles de toutes les anagrammes de 'abc'