TP 14 - Recursivité 2

Exercice 1 *Calcul approché de* tan
Donner une version récursive du calcul de tan, pour $x \in]-\frac{\pi}{2},\frac{\pi}{2}[$: `tan(x)`
On rappelle que :

$$\tan(x)=\frac{2\tan\left(\frac{x}{2}\right)}{1-\tan^2\left(\frac{x}{2}\right)} \qquad \text{et} \qquad x\approx\tan(x) \text{ si } |x|<10^{-6}$$

⇛ Intégrer une généralisation du domaine de définition de tan.

Exercice 2 Considérons l'algorithme suivant :

```
def puiss2(a):
    i,b=0,a
    while b%2==0:
        i=i+1
        b=b//2
    return i,b
```

La fonction `puiss2` a pour paramètre d'entrée un entier a, et retourne les entiers i et b tels que $a=2^i b$ et b impair.
1. Donner un invariant de boucle de `puiss2`.
2. Donner une version récursive `puiss2_R` de la fonction `puiss2`.

➜ Il s'agit de traiter une étape :
- si a est impair, alors le résultat est $(\dots,\dots)$
- si a est pair, alors on note `(i,b)` le résultat de l'instance `a//2` et on en déduit le résultat pour l'instance a : $(\dots,\dots)$

3. Remplir le tableau des appels et des résultats :

Appels	Résultats
`puiss2_R(224)`	

⇛ on remplit la première colonne de haut en bas puis la deuxième de bas en haut !

Exercice 3
Identifier la fonction suivante où `L` est une liste :

```
def ens(L,M=[]):
    if len(L)==0:
        return(M)
    else:
        a=L.pop(0)
        if not(a in M):
            M.append(a)
        return ens(L,M)
```

Exercice 4 *Suites récurrentes*
Dans chacun des cas suivants, écrire deux fonctions (une itérative et une récursive) calculant la valeur du terme de rang n de la suite. Afficher u_{100}.
1. $u_0=0$, $u_1=-1$, $u_2=0$ et $\forall n\geq 0$

$$u_{n+3}=\sqrt{|u_{n+1}u_n|}-n+u_{n+2}$$

2. $u_0=2$, $v_0=10$ et $\forall n\geq 1$

$$u_n=\frac{u_{n-1}+v_{n-1}}{2} \qquad v_n=\sqrt{u_{n-1}v_{n-1}}$$

Exercice 5 *Analyse sur la récursivité*
Soit $u_0=1$, $u_1=-6$ et $\forall n\geq 2$, $u_n=u_{n-1}-u_{n-2}$
1. Écrire une fonction itérative, `ite_u(n)`, qui calcule u_n.
2. Écrire une fonction récursive `rec1_u(n)`, qui calcule u_n.
3. Afin d'éviter les appels redondants de la fonction récursive, nous choisissons de créer une table gérant tous les appels :

```
T=[[ liste des rangs appelé ],
   [ liste des valeurs associée ]]
```

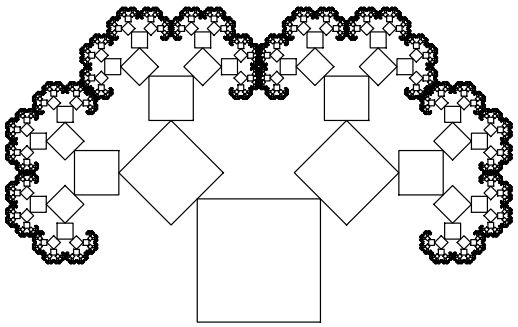
- à chaque appel, avant de faire le calcul, la fonction consulte la table ;
- si le rang n'est pas dans la liste elle lance le calcul et ajoute son rang et sa valeur.
- si le rang est dans la liste des rangs, alors on prend la valeur associée (utiliser `index`).

Définir une nouvelle fonction récursive, `rec2_u(n)`, utilisant la variable `T`. Comme T est une liste, il est inutile de la déclarer en variable globale, ses modifications seront prises en comptes dans les différents appels de la fonction.
4. Créer un tableau mesurant le temps d'exécution des trois fonctions pour les valeurs de $n\in\{15,20,25,30,35\}$.
Pour cela, utiliser la fonction `clock()` du module `time`.

```
from time import clock
t=clock()
... \\ instructions
temps=clock()-t
```


Lycée Saint-Exupery – maffre.saintexupery@gmail.com – MPSI

Exercice 6 *Fractale de carrés*
On considère un carré et on trace deux autres carrés reposant sur ses sommets supérieurs inclinés à 45 degré par rapport au carré de départ.
Ensuite, on réitère le travail sur les nouveaux carrés.



Votre objectif est de concevoir les objets PYTHON permettant de gérer ces figures et de les tracer.

La taille de réduction du côté des carrés, pour la figure ci-dessous, est `k=0.6`