Informatique commune, 11-2021

On donnera les programmes et les explications nécessaires. Les résultats seront fournis avec 10 chiffres après la virgule, et **soulignés.**

1 3 fonctions

Expliquer ce que font les 3 fonctions que_fais_je ; étudier leur complexité en temps.

2 Une suite récurrente

On étudie les suites définies par $u_0 = a \geq 0$, $u_1 = b \geq 0$ et

$$\forall n \geq 0, u_{n+2} = \sqrt{u_{n+1}} + \sqrt{u_n}$$

2.1

Ecrire une fonction $suite(a, b, n)$ qui renvoie u_n.

On pourra utiliser $math.sqrt$.

2.2

Calculer $suite(1, 5, 40)$, $suite(4, 7, 40)$.

Donner les résultats avec 10 chiffres après la virgule, et **soulignés.**

3 Une autre suite récurrente

On étudie les suites définies par $u_0 = x$, $u_1 = 0$ et

$$\forall n \geq 0, u_{n+2} = \frac{1}{2}\left(u_n^2 + u_{n+1}^2\right)$$

On admet que

- ces suites tendent vers une limite L qui vaut 0,1 ou $+\infty$.
- s'il existe n tel que $u_n > 1$ et $u_{n+1} > 1$, $L = +\infty$.
- s'il existe n tel que $u_n < 1$ et $u_{n+1} < 1$, $L = 0$.
- il existe un unique $u_0 = \alpha$ pour lequel $L = 1$.

3.1

Ecrire une fonction *suite(x, n)* qui calcule u_n.

3.2

Proposer une méthode pour calculer α.

Donner cette valeur avec 10 chiffres après la virgule, **soulignée.**

4 Mots de Lukasiewicz

Un mot de Lukasiewicz de longueur $n \geq 1$ est une liste $[u_0, ..., u_{n-1}]$ qui vérifie les 3 propriétés suivantes :

- $\forall i$, $u_i = \pm 1$.
- Pour tout k tel que $0 \leq k \leq n-2$,
$$\sum_{i=0}^{k} u_i \geq 0$$
-
$$\sum_{i=0}^{n-1} u_i = -1$$

Exemples : $[-1]$, $[1, -1, -1]$...

4.1 $n \leq 3$

Trouver tous les mots de Lukasiewicz de longueur n pour $1 \leq n \leq 3$.

4.2 n pair

Trouver tous les mots de Lukasiewicz de longueur n paire.

4.3 Vérification

Ecrire une fonction Python *verifie* qui teste si une liste donnée est un mot de Lukasiewicz et qui renvoie True ou False ; de préférence une fonction de complexité optimale.

Complexité de la fonction *verifie* ?

4.4 Concaténation

Ici, $+$ désigne la concaténation des listes.

Montrer que si u et v sont des mots de Lukasiewicz, $[1] + u + v$ en est un.

Propriété admise

On admet désormais la propriété suivante :

Tout mot de Lukasiewicz w de longueur $n \geq 3$ admet une décomposition unique

$$w = [1] + u + v$$

où u et v sont des mots de Lukasiewicz.

4.5 Comptage

Ecrire une fonction *nombreMots(n)* qui renvoie le nombre de mots de Lukasiewicz de longueur n.

Complexité ?

4.6 Exemples

Calculer nombreMots(25), nombreMots(39).

4.7 S'il vous reste du temps

Démontrer la propriété admise.

```python
def que_fais_je_1(t):
    n = len(t)
    for k in range(0, n-1):
        indice = k
        for j in range(k+1, n):
            if t[j] < t[indice]:
                indice = j
        t[k], t[indice] = t[indice], t[k]

def que_fais_je_2(t):
    n = len(t)
    for k in range(n-1, 0, -1):
        indice = 0
        for j in range(k+1):
            if t[j] > t[indice]:
                indice = j
        t[k], t[indice] = t[indice], t[k]

def que_fais_je_3(t):
    for indice in range(1, len(t)):
        a = t[indice]
        j = indice - 1
        while j >= 0 and t[j] > a:
            t[j+1] = t[j]
            j -= 1
        t[j+1] = a
```