Récursivité

Master 2 I2L, 2019/2020

Exercice 1 : Algorithmes récursifs ?

Questions :
a- Est-ce que les algorithmes ci-dessous sont des algorithmes récursifs ?
b- Est-ce qu'ils se terminent ? Modifier les algorithmes de manière à ce qu'ils se terminent.
c- Que calculent chacun des algorithmes ?

Algorithme log(x, n : entier) : entier
début
  si $x \leq 0$ **alors**

    retourner n
  sinon

    retourner log(x/2, n+1)
  fin si
fin

Algorithme puissance(x, n : entier) : entier
début
  si $n \leq 0$ **alors**

    retourner 1
  sinon

    retourner x * puis(x, n-1)
  fin si
fin

Algorithme somme(n : entier) : entier
début
  si $n = 0$ **alors**

    retourner 0
  sinon

    retourner $|n|$+somme(n+1)
  fin si
fin

Exercice 2 : Suite récurrente

Ecrire une fonction récursive qui calcule le n^{ieme} terme de la suite u définie par :

$$\begin{cases} u_0 &= 0.8 \\ u_{n+1} &= 0.6u_n(u_n - 1) \end{cases}$$

Exercice 3 : Fibonacci

a- Ecrire une fonction récursive qui calcule le n^{eme} terme de la suite de Fibonacci :

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

b - Représenter sous forme graphique le calcul du terme u_5.

c - Calculer la complexité temporelle de votre algorithme.

Exercice 4 : Ackermann

a- Ecrire une fonction récursive qui calcule $A(m, n)$ défini comme ceci :

$$\begin{cases} A(0, n) &= n + 1 \\ A(m, 0) &= A(m - 1, 1), \text{ pour } m > 0 \\ A(m, n) &= A(m - 1, A(m, n - 1)), \text{ pour } m > 0 \text{ et } n > 0 \end{cases}$$

b - Représenter sous forme graphique le calcul du terme $A(2, 3)$.