TD 2 - La récursivité - Master 1 Bio-informatique

Novembre 2015

Exercice 1

Écrire une fonction `facoriel(n)` récursive qui calcule la factorielle de n. Donnez l'arbre de récursion pour `factorielle(3)`.

Exercice 2

Écrire une fonction `suite(n)` récursive qui calcule le terme d'indice n de la suite définie récursivement par :

$$\begin{cases} u_0 = 3 \\ u_n = 2 \times u_{n-1} + 1 \quad \text{si } n > 1 \end{cases}$$

Cette suite est la suite : $(3, 7, 15, 31, \ldots)$ et l'appel de `suite(2)` devra renvoyer 15.

Donnez l'arbre de récursion pour `suite(3)`.

Exercice 3

Écrire une fonction `suite(n)` récursive qui calcule le terme d'indice n de la suite définie récursivement par :

$$\begin{cases} u_0 = 0 \\ u_0 = 1 \\ u_{n+2} = 2 \times u_{n+1} - u_n + 1 \quad \text{si } n \geq 0 \end{cases}$$

Cette suite est la suite : $(0, 1, 3, 6, 10, \ldots)$ et l'appel de `suite(3)` devra renvoyer 6.

Donnez l'arbre de récursion pour `suite(4)`.

Exercice 4

Écrire une fonction `binomial(n,k)`, qui calcule le binomial : $\binom{n}{k} = \frac{n!}{k! \times (n-k)!}$ en utilisant la relation de récurrence suivante :

$$\begin{cases} \binom{n+1}{p+1} = \binom{n}{p} + \binom{n}{p+1} & \text{si } n > p \geq 0 \\ \binom{n}{0} = 1 & \text{si } n \geq 0 \\ \binom{n}{n} = 1 & \text{si } n \geq 0 \end{cases}$$

Donnez l'arbre de récursion pour `binomial(5, 2)`.

Exercice 5

Proposer une fonction récursive `recherche(T,e)` qui prend en paramètre un tableau T et un élément e et qui renvoie la première position de e dans le tableau T.

Par exemple, `recherche([1,1,3,4,4,5,6,8], 4)` renvoie la valeur 3.

Donnez l'arbre de récursion pour `recherche([1,1,3,4,4,5,6,8], 4)`.

Exercice 6

Commencez par découvrir le module turtle de Python :

```
1 from turtle import *
2 forward(100)
3 right(120)
4 forward(50)
5 left(100)
6 forward(100)
```

Que fait ce programme ?

Exercice 7

Utilisez les primitives du module `turtle` pour proposer un programme qui dessine le flocon de von Koch.

Le flocon de von koch est un dessin qui s'obtient récursivement de la manière suivante :

— on commence par dessiner la première image, qui est constituée d'un segment qui relie le point $[0, 0]$ à $[1, 0]$;

— à partir d'une image, on construit une nouvelle image en remplaçant chaque segment s par le dessin $f(s)$ defini de la façon suivante :

1. on divise le segment s en trois segments de longueurs égales,
2. on construit un triangle équilatéral ayant pour base le segment du mileu,
3. on supprime la base du triangle de la deuxième étape.

— on réitère autant de fois que possible la transformation précédente.

Vous pouvez aller voir la page de wikipedia pour plus d'informations.

Exercice 8

Utilisez les primitives du module `turtle` pour proposer un programme qui dessine la courbe du dragon.

La courbe du dragon est un dessin qui s'obtient récursivement de la manière suivante :

— on commence par dessiner la première image, qui est une courbe constituée d'un segment qui relie le point $[0, 0]$ au point $[1, 0]$;

— à partir d'une image, on construit une nouvelle image en parcourant la courbe segment par segment du point $[0, 0]$ au point $[1, 0]$ et en remplaçant, pendant le parcours, chaque segment par deux segments à angles droits (qui forment avec le segment retiré un triangle rectangle isocèle) l'angle droit des deux segments est placé alternativement à gauche, puis à droite de la courbe, tout au long du parcours.

— on réitère autant de fois que possible la transformation précédente.

Vous pouvez aller voir la page de wikipedia pour plus d'informations.

Exercice 9

On appelle composition de n une liste d'entiers strictement positifs telle que la somme des entiers fassent n.

Par exemple, les compositions de 4 sont : $[1, 1, 1, 1]$, $[1, 1, 2]$, $[1, 2, 1]$, $[1, 3]$, $[2, 1, 1]$, $[2, 2]$, $[3, 1]$ et $[4]$.

Donner un algorithme qui à un entier n donné donne toutes les compositions de n.

Exercice 10

On appelle partition de n une liste décroissante d'entiers strictement positifs telle que la somme des entiers fassent n.

Par exemple, les partitions de 4 sont : $[1,1,1,1]$, $[2,1,1]$, $[2,2]$, $[3,1]$ et $[4]$.

Donner un algorithme qui à un entoer n donné donne toutes les partitions de n.

Exercice 11

On appelle permutation de n une liste contenant n entiers tous deux à deux différents et dans les entiers vont de 1 à n.

Par exemple, les permutations de 3 sont : $[1,2,3]$, $[1,3,2]$, $[2,1,3]$, $[2,3,1]$, $[3,1,2]$ et $[3,2,1]$.

Donner un algorithme qui à un entier n donné donne toutes les permutations de n.