TD 2 - Recursivity - Master 1 Bio-informatique

November 2017

Exercise 1

Write a recursive function `facorial(n)` which computes the factorial of n. Give the recursion tree for `factorial(3)`.

Exercise 2

Write a recursive function `serie(n)` which computes the term u_n of the series defined by :

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

This series is the series :$(3, 7, 15, 31, \ldots)$ and the call of `serie(2)` has to return 15.

Give the recursion tree for `suite(3)`.

Exercise 3

Write a recursive function `serie(n)` which computes the term u_n of the series defined by :

$$\begin{cases} u_0 = 0 \\ u_1 = 1 \\ u_{n+2} = 2 \times u_{n+1} - u_n + 1 \quad \text{si } n \geq 0 \end{cases}$$

This series is the series : $(0, 1, 3, 6, 10, \ldots)$ and the call of `suite(3)` has to return 6.

Give the recursion tree for `suite(4)`.

Exercise 4

Write a binomial function `binomial(n,k)`, which computes the binomial : $\binom{n}{k} = \frac{n!}{k! \times (n-k)!}$ by using the following recurrent relation :

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

Give the recursion tree for `binomial(5, 2)`.

Exercise 5

Suggest a recursive function `search(T,e)` which takes as parameter an array (i.e. table) T and an element e and returns the first position of e in the array T.

For example, `search([1,1,3,4,4,5,6,8], 4)` return the value 3.

Give the recursion tree for `search([1,1,3,4,4,5,6,8], 4)`.

Exercise 6

Begin by taking a look at the `turtle` module of Python :

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

What does this program do ?

Exercise 7

By using the primitive functions of `turtle`, suggest a program to draw the Von Koch flake.
The Von Koch flake is a drawing that you can recursively obtain as follows :

— Begin by drawing the first image, which is made of a segment linking the points $[0, 0]$ and $[1, 0]$;

— From one image, we build the new image by replacing each segment s by the drawing $f(s)$ defined as follows :

1. Divide the segment s in 3 segments of equal length.
2. Build a equilateral triangle having as basis the segment in the middle.
3. Suppress the basis of the second step.

— Iterate as often as possible the last transformation.

Wikipedia page can help you for more information.

Exercise 8

Use the primitives of the module `turtle` to write a program to draw a dragon curve.
The dragon curve is a drawing which can be obtained recursively as follow :

— Begin to draw the first image which is a curve constitued by a segment which links the point $[0, 0]$ to the point $[1, 0]$;

— from one image, we build a new image by scanning the segment curve from the point $[0, 0]$ to the point $[1, 0]$ and by replacing, during the scan, each segment by two segments forming a right angle. The two new segments form an isocele right-angle triangle with the deleted segment. The right angle of the 2 segments has to be placed on the left and the right of the curve, alternatively.

— Iterate as often as possible the previous transformation.

Wikipedia page can help you for more information.

Exercise 9

We call a composition of n, a list of strictly positive integers as the sum of the integers is equal to n.
For example, the compositions of 4 are : $[1, 1, 1, 1]$, $[1, 1, 2]$, $[1, 2, 1]$, $[1, 3]$, $[2, 1, 1]$, $[2, 2]$, $[3, 1]$ and $[4]$.
Give an algorithm which provides all the compositions of n for a given n.