Trees — Part 3

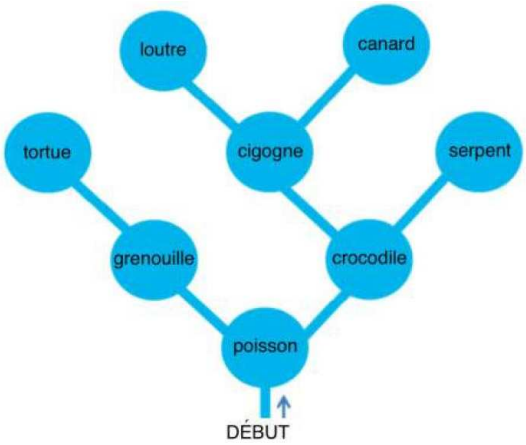
This week, we use the same Tree class as before, see Listing 1.

```
1  class Tree:
2      def __init__(self, data, left=None, right=None):
3          self.data = data
4          self.left = left
5          self.right = right
6
7      def __str__(self):
8          return str(self.data)
```

Listing 1: The Tree data structure.

Exercice 1

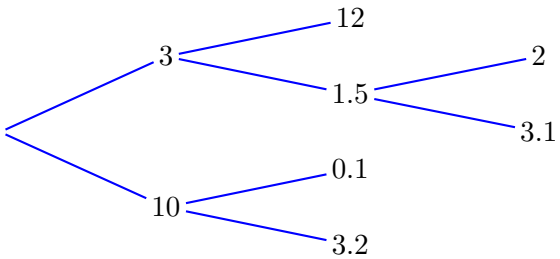
Let's consider the following image:



1. Define, in Python, a tree corresponding to this image (We'll suppose that we look from down to up, so that, for example, "grenouille" is the left child of "poisson").
2. This tree has 8 nodes (we do not count the Null trees that appear in the previous question). Write a function that takes as parameter a tree, and gives back the number of nodes.
3. This tree has height 4. The height of a tree is 0 for the Null tree, 1 for a tree who has no children, and so on. Write a function that takes as parameter a tree, and gives back its height.

Exercice 2

This time, we consider trees that contain only numerical values. For instance:



1. Define, in Python, a tree corresponding to this figure.
2. The cost of a tree is the sum of elements on a branch whose sum is maximal. It can be defined recursively: (a) the cost of an empty tree is 0; (b) for a non-empty tree, you take the cost of the left child, the cost of the right child, and the cost of the resulting tree is the maximum of the two, plus the data stored on this node. Write a function that takes as parameter a tree, and gives back its cost.