Correction

NSI - 2021 Sujet Zéro (21-sujet-zero)

Exercice 3 - Arbre binaire de recherche

Attention : La convention choisie dans l'exercice dit que la hauteur d'un arbre binaire ne comportant qu'un nœud (c'est-à-dire seulement sa racine) est 1. Ici la hauteur correspond au nombre de niveau de l'arbre.

1. La taille correspond au nombre de nœud. Cet arbre est de taille 9.
Sa hauteur (nombre de niveaux) est de 4.

2. 1. Le numéro binaire associé au nœud G est 101.

2. 2. $13_{(10)} = 1101_{(2)}$ ce qui correspond au nœud I

2. 3. Les nœuds de hauteur 2 sont numérotés sur 2 bits, ceux de hauteur 3 sur 3 bits, … ceux de hauteur h, sur h bits.

2. 4. Au minimum, pour un arbre complètement dégénéré (linéaire), il n'y a qu'un nœud par niveau soit $n = h$. Tandis que pour un arbre binaire complet, il y a $n = 2^h - 1$ nœuds car sur h bits, il y a 2^h valeurs possibles. Le -1 provient du fait qu'il n'y a pas de nœud portant la valeur 0.
Ainsi, n est borné par :

$$h \leq n \leq 2^h - 1$$

3. 1. Tableau représentant l'arbre binaire complet :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
15	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O

3. 2. Le père du nœud d'indice i avec $i \geq 2$ a pour indice la partie entière de $i/2$.

4. Les arbres binaires de recherche permette d'implémenter l'algorithme de recherche par dichotomie.

```
def recherche(arbre, element):
    """ Recherche par dichotomie """
    i = 1
    while i < len(arbre):  # Tant que l'on est dans le tableau
        if element == arbre[i]:
            return True
        elif element < arbre[i]:  # Recherche dans le sous-arbre gauche
            i = 2*i
        else:  # Recherche dans le sous-arbre droit
            i = 2*i + 1
    return False
```