TD : PARCOURIR LES ARBRES BINAIRES

RAPPEL

À chaque nœud d'un arbre binaire, on associe :

- une **clé** (on peut aussi utiliser le terme « **valeur** » ou le terme « étiquette »),
- un « **sous-arbre gauche** »
- un « **sous-arbre droit** »

Un sous-arbre, droite ou gauche, est un arbre, même s'il contient un seul nœud ou pas de nœud de tout.

Un sous-arbre vide est noté **NIL** (du latin nihil qui signifie « rien »).

Soit un arbre **`T`** :

- **`T.racine`** correspond au nœud racine de l'arbre **`T`**.

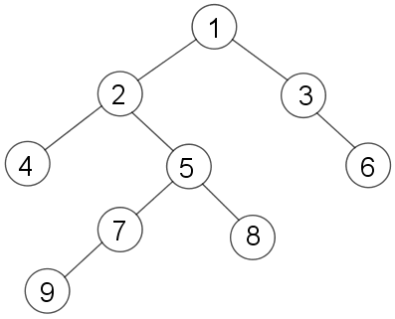
Soit un nœud **`x`** :

- **`x.gauche`** correspond au sous-arbre gauche du nœud **`x`**
- **`x.droit`** correspond au sous-arbre droit du nœud **`x`**
- **`x.cle`** correspond à la clé du nœud **`x`**

Si le nœud **`x`** est une feuille, alors **`x.gauche`** et **`x.droite`** sont des arbres vides (NIL)

EXEMPLE : CALCULER LA HAUTEUR D'UN ARBRE

Le nombre de niveaux total est appelé **hauteur de l'arbre**.



Hauteur de l'arbre = 5

```
class Noeud:
    def init__(self,key):
        self.gauche = None
        self.droit = None
        self.val = key

def Hauteur(noeud):
    if noeud is None:
        return 0
    else:
        lheight = Hauteur(noeud.gauche)
        rheight = Hauteur(noeud.droit)
        return 1 + max(lheight, rheight)
```

```
# test de la fonction Hauteur

racine = Noeud(1)
racine.gauche = Noeud(2)
racine.droit = Noeud(3)
racine.gauche.gauche = Noeud(4)
racine.gauche.droit = Noeud(5)
racine.droit.droit = Noeud(6)
racine.gauche.droit.gauche = Noeud(7)
racine.gauche.droit.droit = Noeud(8)
racine.gauche.droit.gauche.gauche = Noeud(9)

print(Hauteur(racine))
```

TRAVAIL A FAIRE : CALCUL DE LA TAILLE D'UN ARBRE

La **taille d'un arbre** est le nombre de nœuds présents dans l'arbre. Ecrire une fonction qui permet de calculer la taille d'un arbre et tester avec l'exemple ci-dessus