

Exercice 7

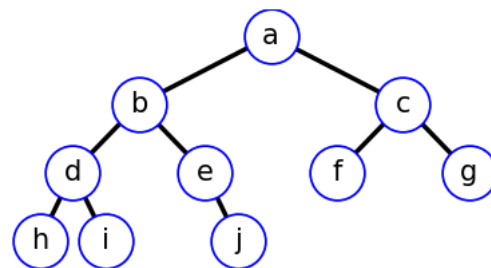
Arbres binaires - hauteur - parcours en largeur

On dispose d'une classe Noeud qui permet de représenter un arbre binaire :

```
class Noeud:  
    def __init__(self, valeur, gauche = None, droit = None, parent = None):  
        self.valeur = valeur  
        self.gauche = gauche  
        self.droit = droit  
        self.parent = parent
```

Par exemple dans l'arbre ci-contre

- Le noeud e a pour valeur 'e'
- Son fils gauche est **None**
- Son fils droit est le noeud j
- Son parent est le noeud b



? QUESTION 1:

On considère l'arbre de l'exemple ci-dessus.

1. Quelle est sa taille?
2. Combien de feuilles a-t-il?
3. On rappelle que la hauteur (ou profondeur) d'un noeud est le nombre d'arêtes qu'il faut pour l'atteindre depuis la racine et que la hauteur d'un arbre est celle de son noeud le plus profond.  
Quelle est la hauteur de l'arbre de l'exemple?
4. Si on effectue un parcours en largeur de cet arbre depuis la racine, donner l'ordre dans lequel seront visités les noeuds.

On donne la méthode ancetres de la classe Noeud sensée renvoyer le nombre d'ancêtres d'un noeud: par exemple l'exécution de `noeud_f.ancetres()` affichera 2 en console.

```
def ancetres(???):  
    s=0  
    while self.parent is not None:  
        ???  
        ???  
    return ???
```

? QUESTION 2:

1. Recopier et compléter la méthode ancetres en remplaçant les ??? par les bonnes instructions.
2. À quoi correspond le nombre d'ancêtres d'un noeud?

On donne ci-contre un algorithme de parcours en largeur d'un arbre binaire.

**? QUESTION 3:**

Recopier cet algorithme en rajoutant ce qu'il faut pour qu'il renvoie une liste contenant les noeuds rencontrés lors de ce parcours.

**Données :** arbre binaire

f une file vide

On met l'arbre dans la file

**Tant que** la file n'est pas vide **faire**

    On défile la file dans une variable tmp

**Si** tmp.gauche n'est pas vide **alors**

        On enfile tmp.gauche

**Si** tmp.droit n'est pas vide **alors**

        On enfile tmp.droit

**renvoyer**

**? QUESTION 4:**

1. Décrire une méthode permettant de déterminer la hauteur d'un arbre en utilisant les questions 2 et 3.
2. Écrire en Python un programme qui réalise votre méthode.