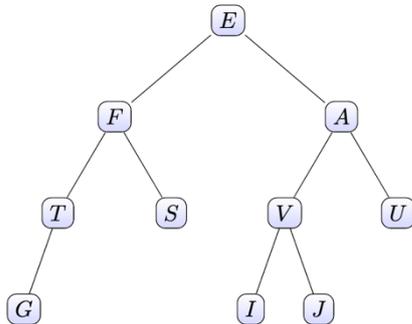


# Exercices de révisions : arbres binaires

## 1. GENERALITES



On considère l'arbre représenté ci-contre.

- Quelle est la taille de cet arbre ? **10**
- Quelle est la valeur de sa racine ? **E**
- Quelle est la hauteur de cet arbre ? **4**
- Quel est le nombre de feuilles cet arbre ? **5**

e) Quel est le père du nœud  $T$  ? Donnez un frère du nœud  $U$ .

**F est le père du nœud de valeur T.**

**V est la valeur du nœud frère du nœud de valeur U.**

## 2. UN ALGORITHME

On considère l'algorithme suivant :

```
Algorithme Mystere( $T, x$ )  
Entrée :  $T$  : un arbre et  $x$  : un noeud de cet arbre  
Sortie : Un entier  
1 si  $x = x.racine$  alors  
2   | retourner 0  
3 sinon  
4   | retourner 1 + Mystere( $T, x.pere$ )
```

a) Quelle sera la valeur retournée par l'appel `Mystere (arbre, 'V')` ?  
*arbre désigne l'arbre de l'exercice précédent.*

**Cet appel retournera la valeur 2 car l'appel récursif suivant sera  $1 + \text{mystere}(\text{arbre}, A)$  puis ensuite  $1 + 1 + \text{mystere}(\text{arbre}, E)$ . Or  $\text{mystere}(\text{arbre}, E)$  renvoie 0 (c'est le cas de base) donc le retour obtenu est  $1 + 1 + 0 = 2$ .**

b) Expliquez ce que ce que calcule cet algorithme.

**La fonction `Mystere` calcule la profondeur du noeud passé en paramètre.**

### 3. PARCOURS

Donnez l'affichage obtenu lors des différents parcours de l'arbre de l'exercice 1 :

- a) Préfixe: **EFTGSAVIJU**
- b) Postfixe (ou suffixe): **GTSFIJVUAE**
- c) Infixe: **GTFSEIVJAU**
- d) Largeur: **EFATSVUGIJ**

### 4. ARBRES BINAIRES DE RECHERCHE

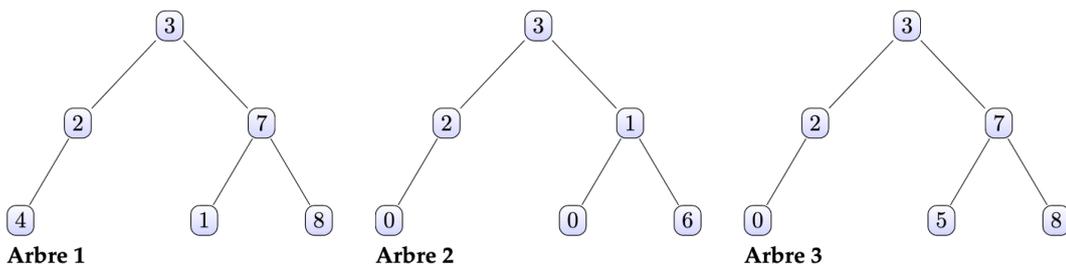
- a) Complétez cette définition d'un arbre binaire de recherche :

Un *arbre binaire de recherche* est un arbre binaire tel que :

- Les clés (ou valeurs) du sous-arbre gauche sont **inférieures ou égales à celle de la racine**.
- Les clés\* du sous-arbre droit sont **supérieures strictement à celle de la racine**.
- Les deux sous-arbres sont **eux même des arbres binaires de recherche**.

\* Les mots *clé, valeur ou étiquette* sont équivalents.

- b) Parmi les arbres ci-dessous, lesquels ne sont pas des *arbres binaires de recherche* ?



## 5) PARCOURS EN LARGEUR

Complétez le code de parcours en largeur d'un arbre binaire :

```
def BFS(arbre):
    if arbre == None:
        return
    # On crée une file vide
    q = File()
    # On enfile l'arbre
    q.enfiler(arbre)
    while not q.estVide():
        # On affiche la valeur de l'arbre en tête de file
        print(q.tete().val, end=' - ')

        node = .....

        if node.gauche is not None:
            .....

        if node.droit is not None:
            .....
```

Ce code a aussi été vu dans le cours sur le parcours des arbres binaires:

```
def BFS(arbre):
    # Si l'arbre est vide, on sort
    if arbre == None:
        return

    # On crée une file vide
    q = File()
    # On enfile l'arbre
    q.enfiler(arbre)

    while not q.estVide():
        # print(q) # pour avoir une trace de la pile
        # On affiche la valeur de l'arbre en tête de file
        print(q.tete().val, end=' - ')
        # on la défile
        node = q.defiler()

        # On enfile le sous-arbre gauche si il existe
        if node.gauche is not None:
            q.enfiler(node.gauche)

        # On enfile le sous-arbre droit si il existe
        if node.droit is not None:
            q.enfiler(node.droit)
```