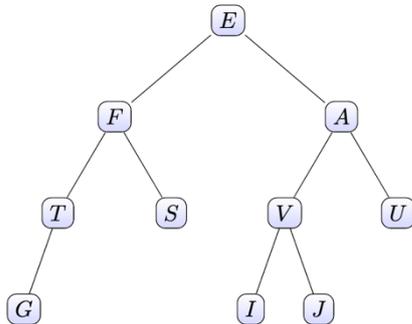


Exercices de révisions : arbres binaires

1. GENERALITES



On considère l'arbre représenté ci-contre.

- a) Quelle est la taille de cet arbre ?
- b) Quelle est la valeur de sa racine ?
- c) Quelle est la hauteur de cet arbre ?
- d) Quel est le nombre de feuilles cet arbre?

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

.....
.....

b) Expliquez ce que ce que calcule cet algorithme.

.....

3. PARCOURS

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

- a) Préfixe:
- b) Postfixe (ou suffixe):
- c) Infixe:
- d) Largeur:

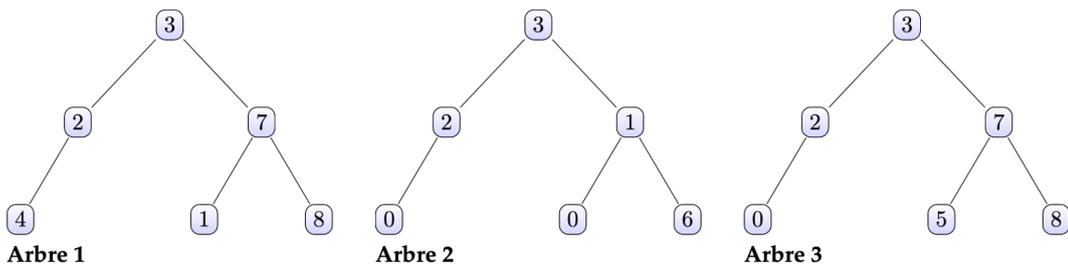
4. ARBRES BINAIRES DE RECHERCHE

- 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
- Les clés du sous-arbre droit sont
- Les deux sous-arbres sont

- 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:  
            .....
```