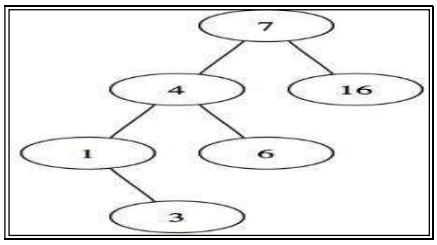
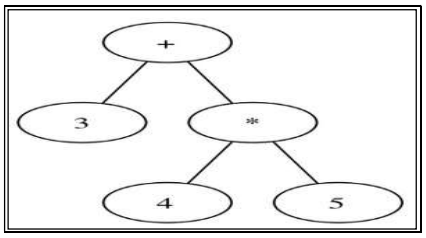
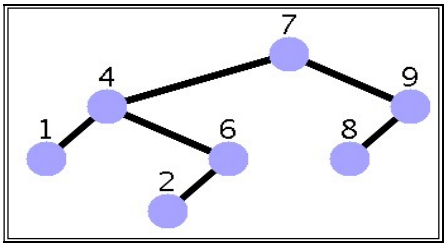
SÉRIE N° 7 : ARBRES BINAIRES

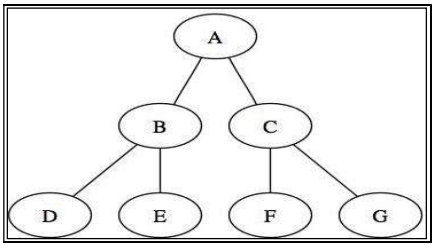
Exercices d'application

Exercice 1 : Avec Python, créer les arbres suivants :









Exercice 2 :

1. Ecrire la fonction **Vide(A)** qui prend en paramètre un arbre A et qui détermine si l'arbre est vide ou non
2. Ecrire la fonction **Racine(A)** qui prend en paramètre un arbre A et qui retourne la valeur de la racine
3. Écrire une fonction booléenne **Feuille(A)** qui teste qu'un arbre est une feuille, c'est-à-dire un nœud qui n'a aucun fils.
4. Ecrire la fonction **NbrNoueds(A)** qui prend en paramètre un arbre A et qui retourne le nombre de nœuds de l'arbre
5. Ecrire la fonction **SommeArbre(A)** qui prend en paramètre un arbre A et qui retourne la somme de valeurs de l'arbre
6. Ecrire la fonction booléenne **Recherche(A,x)** qui prend en paramètre un arbre A et qui teste si une valeur x appartient à un arbre.
7. Ecrire la fonction **Minimum(A)** qui prend en paramètre un arbre A et qui retourne la valeur minimale de l'arbre
8. Écrire la fonction **Hauteur(A)** qui calcule la hauteur d'un arbre, définie ainsi :
   - la hauteur d'un arbre est 1 + le maximum des hauteurs des sous-arbres (gauche, droit),
   - la hauteur d'une feuille étant 0.
9. Écrire une fonction booléenne **ARB(A)** qui teste qu'une liste représente bien un arbre binaire.
10. Écrire la fonction **infixe (A)** qui retourne la liste résultant du parcours infixe d'un arbre :
    - en chaque noeud, on parcourt le fils gauche, puis on note la valeur du noeud, puis on parcourt le fils droit.
11. Écrire la fonction **prefixe (A)** qui retourne la liste résultant du parcours préfixe d'un arbre :
    - en chaque noeud, on note la valeur du noeud ,puis on parcourt le fils gauche, puis on parcourt le fils droit.
12. Écrire la fonction **postfixe (A)** qui retourne la liste résultant du parcours postfixe d'un arbre :
    - en chaque noeud, on parcourt le fils gauche, puis on parcourt le fils droit, puis on note la valeur du noeud.