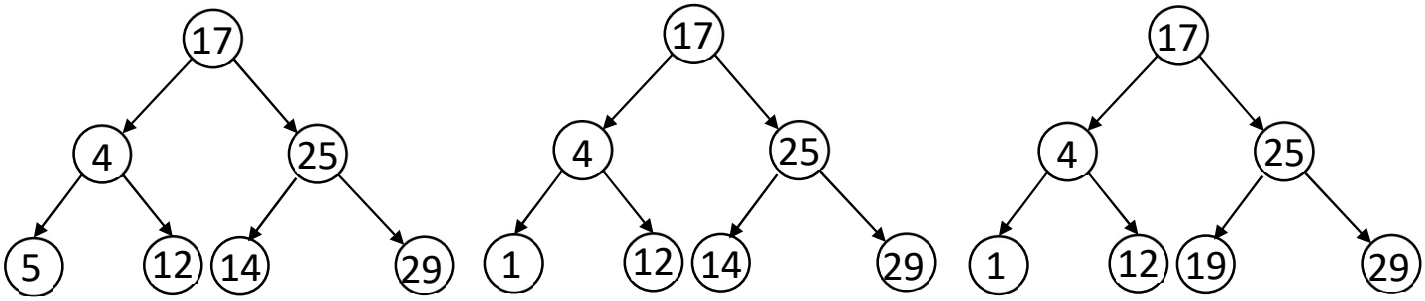
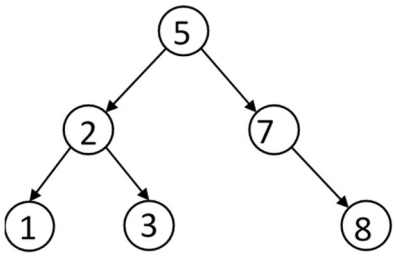
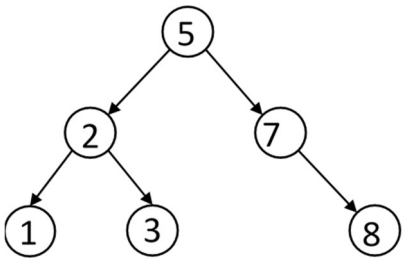
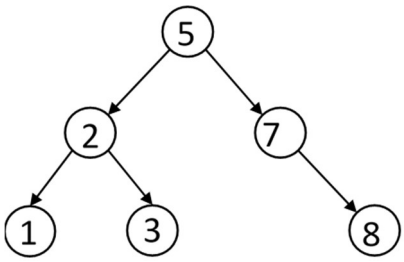
Exercice 1 : Reconnaitre un ABR

1- Parmi les arbres binaires suivants, entourer ceux qui sont des arbres binaires de recherche :



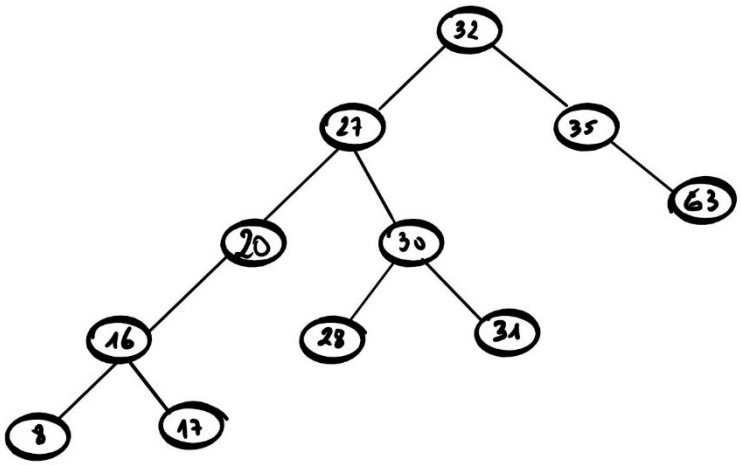
2- Donner les clés du premier arbre présenté ci-dessus :

Exercice 2 : Insérer le nœud indiqué dans les arbres suivants :

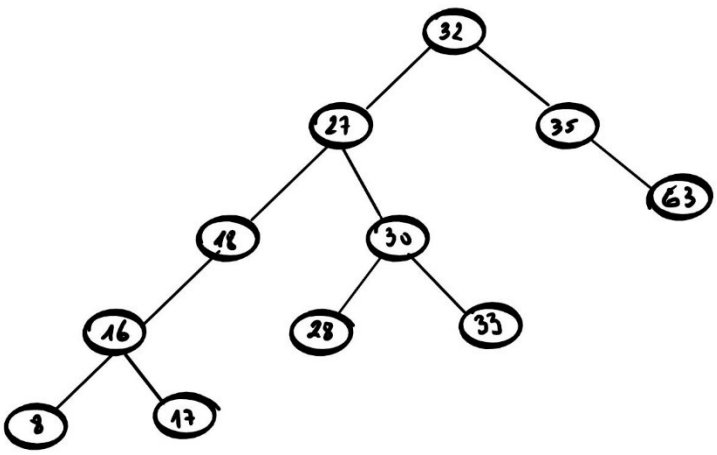
Insérer 6	Insérer 4	Insérer 9
		

Exercice 3 : Les arbres ci-dessous sont-ils des ABR ?

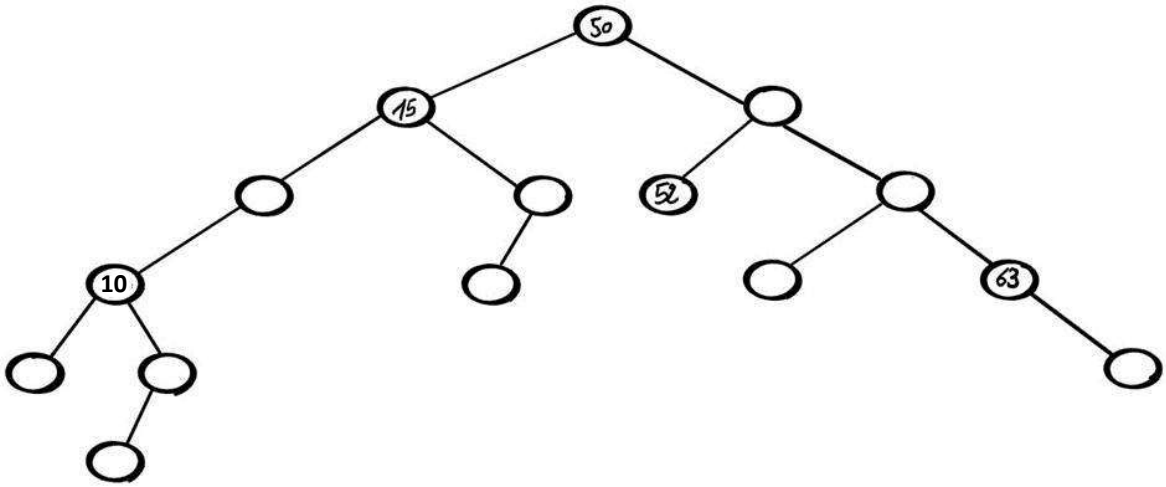
arbre 1 :



arbre 2 :



Exercice 4 : Compléter l'arbre binaire ci-dessous afin qu'il soit un ABR :



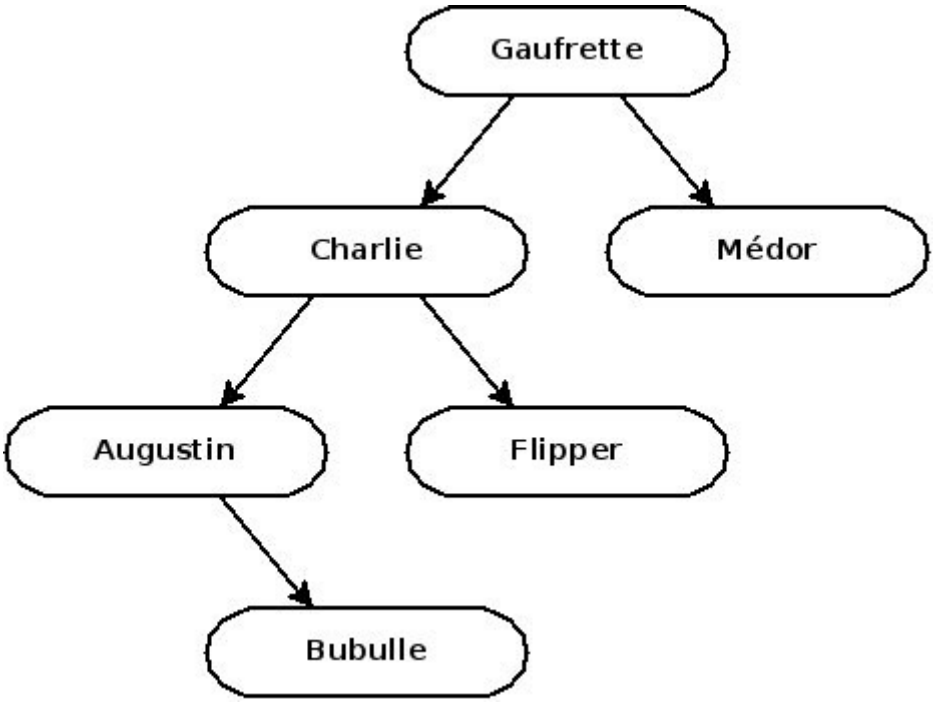
Exercice 5 : Un arbre avec des clés qui sont des chaines de caractères, le rangement se faisant par ordre alphabétique.

L'arbre ci-dessous reprend le stockage des contenus dans un dictionnaire python ou tableau associatif dans les autres langages.

```
dict = {}
dict['Gaufrette'] = 'mes_meilleurs_chansons.mp3'
dict['Charlie'] = 'vidéo_bestOff.mp4'
dict['Augustin'] = [i for i in range(10)]
dict['bubulle'] = 'https://www.mon_site.fr'
dict['Fliper'] = ['delphin']*10
dict['Médor'] = 'OsARonger.mp4'
```

1- Donner les clés de cet arbre.

2- Cet arbre est-il un ABR ?



Exercice 6 : Créer un arbre binaire de recherche.

1- Dessiner ci-dessous un ABR en insérant à partir d'un arbre vide, dans l'ordre, les clés :

2 ⇨1 ⇨3	*1 ⇨2 ⇨3*	*3 ⇨1 ⇨2*

2- Dessiner ci-dessous un ABR en insérant à partir d'un arbre vide, dans l'ordre, les clés :

255 ⇨50 ⇨300 ⇨310 ⇨260 ⇨270 ⇨280 ⇨1 ⇨7

Exercice 7 : Le code python donné dans la suite, permet d'implémenter des ABR.

1- Ecrire le code de la méthode *insererElement()*

2- Compléter le programme principal afin de pouvoir créer l'ABR de l'exercice 6 précédent.

3- Ecrire le code de la méthode rechercherElement*()*

4- Ecrire le code de la méthode max*()*

5- Ecrire le code de la méthode min*()*

6- Ecrire le code de la méthode estABR*()*

```python
from trace_arbre import Graph , Pile , File

class Arbre(Graph) :
    def __init__(self, info , fg = None , fd = None) :
        self.info = str(info)
        self.fg = fg
        self.fd = fd

    def insererElement(self,val):
        '''
            Insère dans l'arbre, le noeud dont la clé a la valeur val
        '''
        pass

    def rechercherElement(self,val) :
        '''
            Retourne True si la clé de valeur val est dans l'arbre, False sinon
        '''
        pass


    def max(self) :
        '''
            Retourne la valeur maximale des clés de l'arbre
        '''
        pass

    def min(self) :
        '''
            Retourne la valeur minimale des clés de l'arbre
        '''
        pass

    def estABR(self):
        '''
            Retourne True si l'arbre est un ABR, False sinon
        '''
        pass
# Main ---------------------------------------------------
A = Arbre(255)
```