

La classe ABR ci-dessous permet d'implémenter une structure d'arbre binaire de recherche.

```
class Noeud:
    ''' Classe implémentant un noeud d'arbre binaire
    disposant de 3 attributs :
    - valeur : la valeur de l'étiquette,
    - gauche : le sous-arbre gauche.
    - droit : le sous-arbre droit. '''

    def __init__(self, v, g, d):
        self.valeur = v
        self.gauche = g
        self.droite = d

class ABR:
    ''' Classe implémentant une structure
    d'arbre binaire de recherche. '''

    def __init__(self):
        '''Crée un arbre binaire de recherche vide'''
        self.racine = None

    def est_vide(self):
        '''Renvoie True si l'ABR est vide et False sinon.'''
        return self.racine is None

    def parcours(self, tab = []):
        ''' Renvoie la liste tab complétée avec tous les
        éléments de l'ABR triés par ordre croissant. '''

        if self.est_vide():
            return tab
        else:
            self.racine.gauche.parcours(tab)
            tab.append(...)
            ...
            return tab
```

```
def insere(self, element):
    '''Insère un élément dans l'arbre binaire de recherche.'''
    if self.est_vide():
        self.racine = Noeud(element, ABR(), ABR())
    else:
        if element < self.racine.valeur:
            self.racine.gauche.insere(element)
        else :
            self.racine.droite.insere(element)

def recherche(self, element):
    '''
    Renvoie True si element est présent dans l'arbre
    binaire et False sinon.
    '''
    if self.est_vide():
        return ...
    else:
        if element < self.racine.valeur:
            return ...
        elif element > self.racine.valeur:
            return ...
        else:
            return ...
```

Compléter les fonctions récursives parcours et recherche afin qu'elles respectent leurs spécifications.

Voici un exemple d'utilisation :

```
>>> a = ABR()
>>> a.insere(7)
>>> a.insere(3)
>>> a.insere(9)
>>> a.insere(1)
>>> a.insere(9)
>>> a.parcours()
[1, 3, 7, 9, 9]

>>> a.recherche(4)
False

>>> a.recherche(3)
True
```