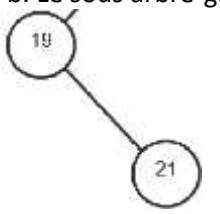
Correction

NSI - 2021 Métropole Jour 1 (--)

Exercice 1 - Arbres binaires de recherche

1. a. Cet arbre contient 4 feuilles qui ont respectivement pour valeur 12, val, 21 et 32.

1. b. Le sous arbre-gauche du nœud 23 est :



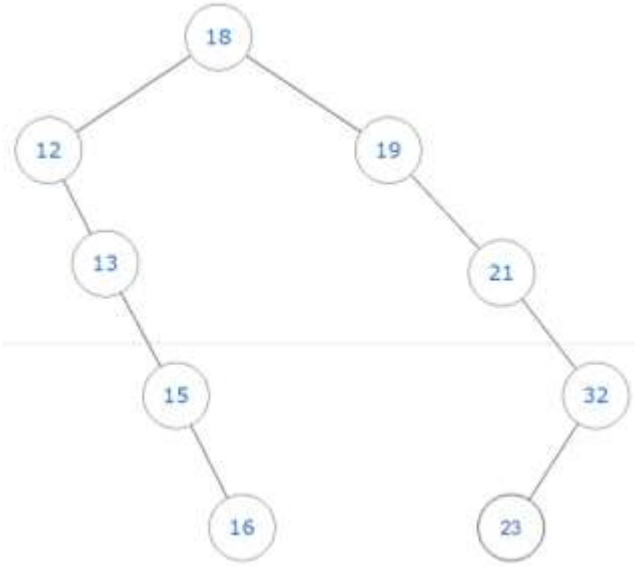
1. c. La hauteur de cet arbre est de 4 et sa taille (nombre de nœud) est de 9.
Remarque : L'énoncé explique qu'un arbre limité à un nœud a une hauteur de 1 mais ici nous devrions parler de niveau plutôt que de hauteur. Cet arbre possède 4 niveaux et a une hauteur de 3.

1. d. Les valeurs possibles pour val sont 16 et 17.

2. a. Parcours infixe : 12, 13, 15, val = 16, 18, 19, 21, 23 et 32.

2. b. Parcours suffixe : 12, 13, val = 16, 15, 21, 19, 32, 23 et 18.

3. a.



3. b. Les instructions sont :

```
racine = Noeud(18)
racine.insere_tout([15, 13, 12, 16, 23, 19, 21, 32])
```

Remarque : Il y a plusieurs possibilités pour l'ordre des valeurs à insérer :
Autre possibilité : `[23, 15, 32, 19, 16, 13, 21, 12]`
Impossible : `[13, 15, 12, 16, 23, 19, 21, 32]` car la valeur 15 doit être insérée avant la valeur 13.

Correction

NSI - 2021 Métropole Jour 1 (--)

3. c. L'ordre d'exécution est : bloc 3 (car 19 > 18), bloc 2 (car 19 < 23) et bloc 1 (car 19 est déjà dans l'arbre).

4.

Solution itérative :

```
def recherche2(self, v):
    n = self
    while n :
        if v == n.v:
            return True
        elif v < n.v:
            n = n.ag
        else:
            n = n.ad
    return False
```

Solution récursive :

```
def recherche(self, v):
    if v == self.v:
        return True
    elif v < self.v and self.ag:
        return self.ag.recherche(v)
    elif self.ad:
        return self.ad.recherche(v)
    return False
```