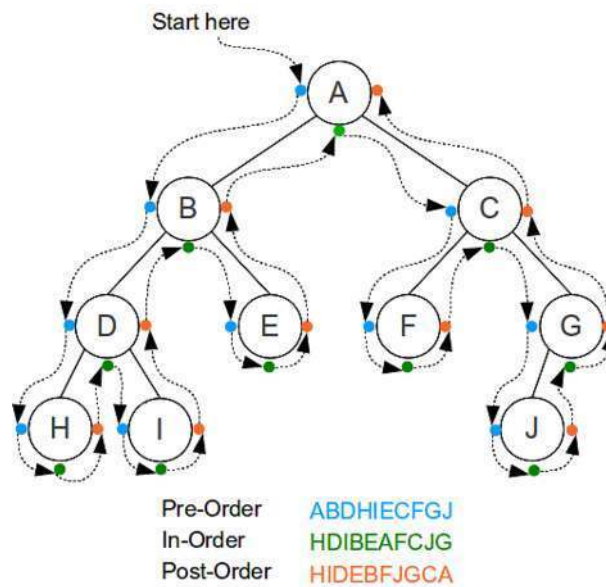


1 (Arbre Binaire : Construction - 2 points)



2 (Arbre Binaire de Recherche - 4 points)

PRECONDITIONS

aucune

AXIOMES

recherche(x, arbre-vide) = faux

$x = \text{contenu}(\text{racine}(B)) \Rightarrow \text{recherche}(x, B) = \text{vrai}$

$x < \text{contenu}(\text{racine}(B)) \Rightarrow \text{recherche}(x, B) = \text{recherche}(x, g(B))$

$x > \text{contenu}(\text{racine}(B)) \Rightarrow \text{recherche}(x, B) = \text{recherche}(x, d(B))$

([88, 65, 64, 11, 59, 54, 13, 33, 51, 34, 46, 39, 40, 45, 44, 42], True)

([17, 89, 19, 57, 54, 26, 32, 36, 41, 46, 47, 93, 48, 60, 74, 88], False)

([94, 76, 74, 17, 63, 57, 52, 41, 39, 19, 35, 22, 31, 27, 26, 23], True)

([92, 32, 91, 36, 55, 56, 59, 79, 76, 73, 61, 10, 44, 11, 22, 31], False)

3 (Matrices : Symétrique - 4 points)

```

1 def isSymmetric(A):
2     l, c = len(A), len(A[0])
3     if l != c:
4         return False
5     i, sym = 0, True
6     while i < l and sym:
7         j = 0
8         while j < l and sym:
9             sym = A[i][j] == A[j][i]
10            j += 1
11        i += 1
12    return sym

```

4 (Arbre Binaire : Similarités - 5 points)

```

1 def postorder(B,l):
2     if B != None:
3         postorder(B.left, l)
4         postorder(B.right, l)
5         l.append(B.key)
6
7 def checkPostOrder(A, B):
8     lA,lB = [],[]
9     postorder(A,lA)
10    postorder(B,lB)
11    sA,sB = len(lA),len(lB)
12    if sA != sB:
13        return False
14    i,c = 0,True
15    while i < sA and c:
16        c = sA[i] == sB[i]
17        i += 1
18    return c

```

5 (Arbre Binaire : PME - 6 points)

```

1 def pme(B):
2     q = newQueue()
3     enqueue(B, q)
4     enqueue(None, q)
5     (h, lce, nbe) = (0, 0, 0)
6     while not isEmpty(q):
7         B = dequeue(q)
8         if B == None:
9             h += 1
10            if not isEmpty(q):
11                enqueue(None, q)
12        else:
13            if B.left == B.right:
14                lce += h
15                nbe += 1
16            else:
17                if B.left != None:
18                    enqueue(B.left, q)
19                if B.right != None:
20                    enqueue(B.right, q)
21    return (lce/nbe)

```