```python
from queuelinked import LinkedQueue

class BinaryTree:
    class _Node:
        __slots__ = '_element', '_left', '_right'

        def __init__(self,element, left=None, right=None):
            self._element = element
            self._left = left
            self._right = right

    def __init__(self):
        self._root = None
        self._size = 0

    def maketree(self, e, left, right):
        self._root = self._Node(e,left._root,right._root)
        left._root = None
        right._root = None

    def levelorder(self):
        Q = LinkedQueue()
        t = self._root
        print(t._element,end='--')
        Q.enqueue(t)

        while not Q.is_empty():
            t = Q.dequeue()
            if t._left:
                print(t._left._element, end='--')
                Q.enqueue(t._left)
            if t._right:
                print(t._right._element, end='--')
                Q.enqueue(t._right)

    def inorder(self, troot):
        if troot:
            self.inorder(troot._left)
            print(troot._element, end='--')
            self.inorder(troot._right)

    def preorder(self,troot):
        if troot:
            print(troot._element,end='--')
            self.preorder(troot._left)
            self.preorder(troot._right)

    def postorder(self, troot):
        if troot:
            self.postorder(troot._left)
            self.postorder(troot._right)
            print(troot._element, end='--')

a = BinaryTree()
x = BinaryTree()
y = BinaryTree()
z = BinaryTree()
r = BinaryTree()
s = BinaryTree()
t = BinaryTree()

x.maketree(40,a,a)
y.maketree(60,a,a)
z.maketree(20,x,a)
r.maketree(50,a,y)
s.maketree(30,r,a)
t.maketree(10,z,s)

t.levelorder()
print()
t.preorder(t._root)
print()
t.inorder(t._root)
print()
t.postorder(t._root)
print()


```