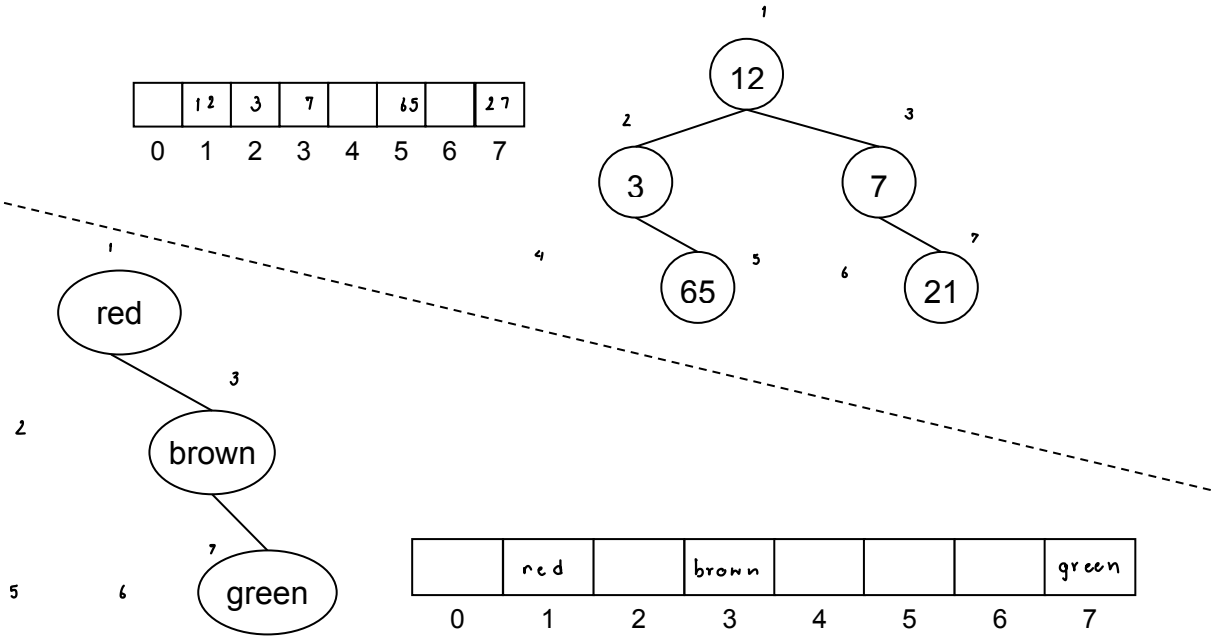
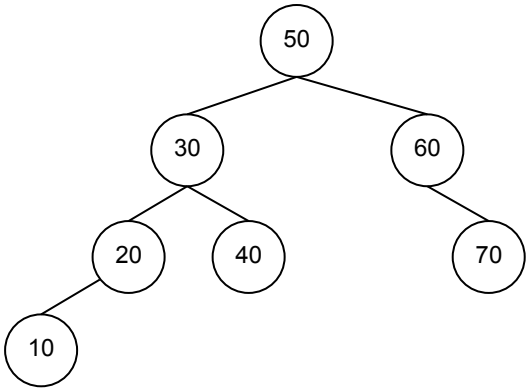
Binary Tree

1. Fill in the arrays to represent binary trees below



2. For the binary tree below



2.1. Which node is the root ? 50
2.2. Is this a binary tree ? why ? no parent class than have more than 2 children
2.3. List all the leaves 10, 40, 70
2.4. Is this binary tree complete ? why ? no , not every node have children
2.5. What is a relationship between 20 and 30 ? children - parent
2.6. What is a relationship between 20 and 40 ? siblings
2.7. What is a relationship between 20 and 70 ? same level
2.8. List all nodes in level 2 20 , 40 , 70
2.9. What is the height of the tree ? 3
2.10. What is the height of node 70 ? 1
2.11. Write a path from root to 10 50, 30, 20 , 10
2.12. How many nodes are there in a subtree rooting at node 30 ? 3
2.13. Pretorder traversal gives 50, 30, 20, 10 , 40, 60, 70
2.14. Inorder traversal gives 10, 20, 30, 40, 50, 60, 70
2.15. Postorder traversal gives 10, 20, 40, 30, 70, 60, 50

3. Write an algorithm to find height of a binary tree
4. Write an algorithm to find number of nodes in a binary tree
5. Write an algorithm to list all leave nodes in a binary tree

```python
class Node:
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

def size(node):
    if node is None:
        return 0
    else:
        return (size(node.left)+ 1 + size(node.right))

def height(node):
    if node is None:
        return 0
    else:
        lDepth = height(node.left)
        rDepth = height(node.right)
        if (lDepth > rDepth):
            return lDepth+1
        else:
            return rDepth+1

def LeafNodes(root:Node):
    if (not root):
        return

    if (not root.left and not root.right):
        print(root.data, end = " ")
        return

    if root.left:
        LeafNodes(root.left)

    if root.right:
        LeafNodes(root.right)
```