1. Mark the following statements as true or false.

a. A binary tree must be nonempty.

b. The level of the root node is 0.

c. If a tree has only one node, the height of this tree is 0 because the number of levels is 0.

d. The inorder traversal of a binary tree always outputs the data in ascending order.

1. The binary tree of the following Figure is to be used for Exercises 1 through 6.



1. Find LA, the node in the left subtree of A.

2. Find RA, the node in the right subtree of A.

3. Find RB, the node in the right subtree of B.

4. List the nodes of this binary tree in an inorder sequence.

5. List the nodes of this binary tree in a preorder sequence.

6. List the nodes of this binary tree in a postorder sequence.

1. The binary search tree of the following Figure is to be used for Exercises 1through 4.



1. List the path from the node with info 80 to the node

with info 79.

1. A node with info 35 is to be inserted in the tree.

List the nodes that are visited by the function

insert to insert 35. Redraw the tree after inserting 35.

1. Delete node 52 and redraw the binary tree.
2. Delete node 40 and redraw the binary tree.
3. Delete nodes 80 and 58 in that order. Redraw the

binary tree after each deletion.

1. Write the definition of the function, **nodeCount**, that returns the number of nodes in a binary tree.
2. Write the definition of the function, **leavesCount**, that takes as a parameter a reference to the root node of a binary tree and returns the number of leaves in a binary tree.
3. Draw the binary tree representation of the following arithmetic expression:

“(((5+2) ∗ (2−1))/((2+9)+((7−2)−1)) ∗8)”.

1. Insert, into an empty binary search tree, entries with keys 30, 40, 24, 58, 48, 26, 11, 13 (in this order). Draw the tree after each insertion.

يمكن حل أسئلة من الكتاب , بالإضافة الى المحاضرات

**The answer:**

**1. Mark the following statements as true or false.**

**a. True**

**b. True**

**c. True**

**d. False**

**2. The binary tree of the following Figure is to be used for Exercises 1 through 6.**

**1. L\_(A): D**

**2. R\_(A): E**

**3. R\_(B): F**

**4. Inorder sequence: D, A, E, B, F, C**

**5. Preorder sequence: A, D, B, E, C, F**

**6. Postorder sequence: D, E, B, F, C, A**

**3. The binary search tree of the following Figure is to be used for Exercises 1 through 4.**

**1. Path from node with info 80 to node with info 79: 80, 65, 70, 79**

**2. Nodes visited during insertion of 35: 80, 65, 70, 50, 35**

**3. Binary tree after deleting node 52: (tree diagram not provided)**

**4. Binary tree after deleting node 40: (tree diagram not provided)**

**5. Binary tree after deleting nodes 80 and 58: (tree diagrams not provided)**

**4. Definition of the function nodeCount:**

**```python**

**def nodeCount(root):**

**if root is None:**

**return 0**

**return 1 + nodeCount(root.left) + nodeCount(root.right)**

**```**

**5. Definition of the function leavesCount:**

**```python**

**def leavesCount(root):**

**if root is None:**

**return 0**

**if root.left is None and root.right is None:**

**return 1**

**return leavesCount(root.left) + leavesCount(root.right)**

**```**

**6. Binary tree representation of the arithmetic expression:**

**```**

**\***

**/ \**

**/ \**

**/ \**

**/ \**

**+ \***

**/ \ / \**

**5 2 - 8**

**/ \**

**+ -**

**/ \ / \**

**2 9 1**

**/ \**

**7 2**

**```**

**7. Binary search tree after inserting entries with keys 30, 40, 24, 58, 48, 26, 11, 13:**

**```**

**Inserted 30:**

**30**

**Inserted 40:**

**30**

**\**

**40**

**Inserted 24:**

**30**

**/ \**

**24 40**

**Inserted 58:**

**30**

**/ \**

**24 40**

**\**

**58**

**Inserted 48:**

**30**

**/ \**

**24 40**

**\**

**58**

**/**

**48**

**Inserted 26:**

**30**

**/ \**

**24 40**

**\ \**

**26 58**

**/**

**48**

**Inserted 11:**

**30**

**/ \**

**24 40**

**/ / \**

**11 26 58**

**/**

**48**

**Inserted 13:**

**30**

**/ \**

**24 40**

**/ / \**

**11 26 58**

**/**

**48**

**\**

**13**

**```**