ALYA FOUAD AZIZ

1. Mark the following statements as true or false.

a. A binary tree must be nonempty. **خاطئ:** يمكن أن تكون الشجرة الثنائية فارغة، مما يعني أنها لا تحتوي على أي عقد. تسمى هذه الشجرة الثنائية الفارغة

b. The level of the root node is 0.

* صواب: يُعرف مستوى عقدة الجذر دائمًا بأنه 0 في شجرة ثنائية.

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

* صواب: إذا كانت الشجرة تحتوي على عقدة واحدة فقط، فهي تعتبر عقدة الجذر نفسها. وبالتالي، فإن ارتفاع الشجرة يكون 0 لأن هناك مستوى واحد فقط (مستوى الجذر).

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

* خطأ: يعتمد ترتيب البيانات التي يتم إخراجها بواسطة التجوال الوسطي على بنية الشجرة الثنائية. على سبيل المثال، لن تؤدي الشجرة الثنائية غير المتوازنة إلى إخراج البيانات بترتيب تصاعدي.

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.

In the given binary tree, the node in the left subtree of A is **B**

* 1. Find RA, the node in the right subtree of A.

In the given binary tree, the node in the right subtree of A is **C**.

* 1. Find RB, the node in the right subtree of B.

In the given binary tree, the node in the right subtree of B is **E**.

* 1. List the nodes of this binary tree in an inorder sequence.
* Inorder traversal visits the left subtree, then the root, and then the right subtree. Following this order, the nodes are listed as:
* **D B E A F C G**

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

Preorder traversal visits the root, then the left subtree, and then the right subtree. Following this order, the nodes are listed as:

**A B D E C F G**

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

Postorder traversal visits the left subtree, then the right subtree, and then the root. Following this order, the nodes are listed as:

**D E B F G C A**

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.

public static int nodeCount(Node root) {

if (root == null) {

return 0; // Empty tree has 0 nodes

}

// Recursively count nodes in the left and right subtrees

int leftCount = nodeCount(root.left);

int rightCount = nodeCount(root.right);

// Combine counts from left, right, and root itself

return leftCount + rightCount + 1;

}

1. 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.

public static int leavesCount(Node root) {

if (root == null) {

return 0; // Empty tree has no leaves

}

// Check if the current node is a leaf (both left and right children are null)

if (root.left == null && root.right == null) {

return 1; // Leaf node contributes 1 to the count

}

// Recursively count leaves in the left and right subtrees

int leftLeaves = leavesCount(root.left);

int rightLeaves = leavesCount(root.right);

// Combine counts from left and right subtrees

return leftLeaves + rightLeaves;

}

1. Draw the binary tree representation of the following arithmetic expression:

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

\*

/ \

/ \

/ \

/ \

/ \

/ \

/ \

+ 8

/ \ / \

5 2 + -

/ \ / \

2 9 7 2

/ \

- 1

Explanation:

1. The root node is \*, representing the multiplication operation.
2. The left child of the root is /, representing the division operation.
3. The right child of the root is 8, representing the constant value 8.
4. The left child of the left child (/) is +, representing the addition operation.
5. The right child of the left child (/) is -, representing the subtraction operation.
6. Each leaf node represents a constant value (5, 2, 2, 9, 7, 1).
7. 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.
8. 30
9. / \

24 40

1. / \ / \

11 26 48 58

1. /
2. 13

يمكن حل أسئلة من الكتاب , بالإضافة الى المحاضرات

**Good Luck**