Practical 12

Aim : Write a Program to determine the depth of a given Tree by Implementing MAXDEPTH

MAXDEPTH:-maxDepth function is a fundamental operation in tree data structures, allowing you to determine how deep a tree is. It is often used in various algorithms and applications, such as balancing trees, traversing trees, and analyzing tree structures.

Algorithm :

1. Define the Node Structure:
    - Create a class Node with attributes:
        - int data (to store the value of the node)
        - Node left (pointer to the left child)
        - Node right (pointer to the right child)
2. Define the Method maxDepth(Node root):
    - Input: A node root (the root of the binary tree).
    - Output: An integer representing the maximum depth of the tree.
    - Steps:

1. If root is null, return 0 (base case: empty tree).

2. Recursively calculate the maximum depth of the left subtree: leftDepth = maxDepth(root.left).

3. Recursively calculate the maximum depth of the right subtree: rightDepth = maxDepth(root.right).

4. Return 1 + max(leftDepth, rightDepth) (add 1 for the current node).

3. Define the Main Method:

- Create a sample binary tree by instantiating Node objects and linking them.
- Call the maxDepth method with the root of the tree.
- Print the result

Program :-

```
public class Main {
  // Node class representing a node in the tree
  static class Node {
    int data;
    Node left, right;

    Node(int data) {
      this.data = data;
      this.left = null;
      this.right = null;
    }
```

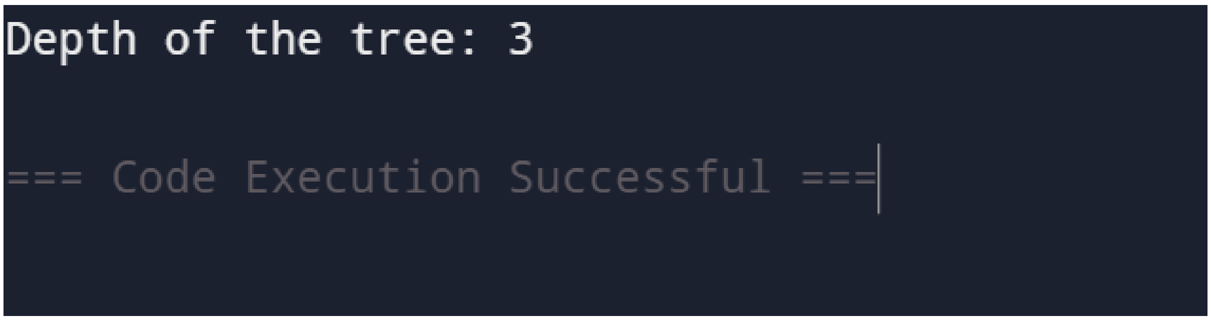
```
}

// Method to calculate the maximum depth of a tree
public static int maxDepth(Node root) {
   if (root == null) {
      return 0; // Empty tree has depth 0
   }
   // Recursively find depths of left and right subtrees
   int leftDepth = maxDepth(root.left);
   int rightDepth = maxDepth(root.right);
   // Return the larger depth + 1 (for the current node)
   return Math.max(leftDepth, rightDepth) + 1;
}

// Main method for creating a sample tree and calculating its depth
public static void main(String[] args) {
   // Create a sample binary tree
   Node root = new Node(1);
   root.left = new Node(2);
   root.right = new Node(3);
   root.left.left = new Node(4);
   root.left.right = new Node(5);

   // Calculate and print the depth of the tree
   int depth = maxDepth(root);
   System.out.println("Depth of the tree: " + depth);
```

```
  }
}
```



```
Depth of the tree: 3

=== Code Execution Successful ===
```