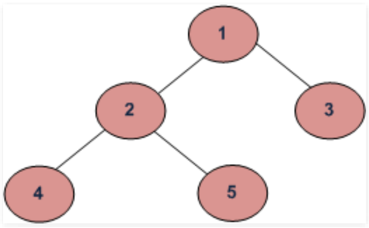
Write a Program to Find the Maximum Depth or Height of a Tree

Given a binary tree, find height of it. Height of empty tree is 0 and height of below tree is 3.



Example Tree

Recommended: Please solve it on “*PRACTICE*” first, before moving on to the solution.

Recursively calculate height of left and right subtrees of a node and assign height to the node as max of the heights of two children plus 1. See below pseudo code and program for details.

Algorithm:

```
 maxDepth()
1. If tree is empty then return 0
2. Else
     (a) Get the max depth of left subtree recursively  i.e.,
          call maxDepth( tree->left-subtree)
     (a) Get the max depth of right subtree recursively  i.e.,
          call maxDepth( tree->right-subtree)
     (c) Get the max of max depths of left and right
          subtrees and add 1 to it for the current node.
         max_depth = max(max dept of left subtree,
                             max depth of right subtree)
                             + 1
     (d) Return max_depth
```

See the below diagram for more clarity about execution of the recursive function maxDepth() for above example tree.

```
             maxDepth('1') = max(maxDepth('2'), maxDepth('3')) + 1
                                = 2 + 1
                                  /    \
                                /        \
                              /            \
                            /                \
                          /                    \
                maxDepth('1')                  maxDepth('3') = 1
= max(maxDepth('4'), maxDepth('5')) + 1
= 1 + 1   = 2
                   /    \
                 /        \
               /            \
             /                \
           /                    \
 maxDepth('4') = 1     maxDepth('5') = 1
```

Implementation:

C Java Python

```python
# Python program to find the maximum depth

# A binary tree node
class Node:

    # Constructor to create a new node
    def __init__(self, data):
        self.data = data
        self.left = None
        self.right = None

# Compute the "maxDepth" of a tree -- the
# along the longest path from the root noc
# farthest leaf node
def maxDepth(node):
    if node is None:
        return 0 ;

    else :
        # Compute the depth of each subtre
        lDepth = maxDepth(node.left)
        rDepth = maxDepth(node.right)

        # Use the larger one
        if (lDepth > rDepth):
            return lDepth+1
        else:
            return rDepth+1


# Driver program to test above function
root = Node(1)
root.left = Node(2)
root.right = Node(3)
root.left.left = Node(4)
root.left.right = Node(5)


print "Height of tree is %d" %(maxDepth(rc

# This code is contributed by Nikhil Kumar
```

Run on IDE