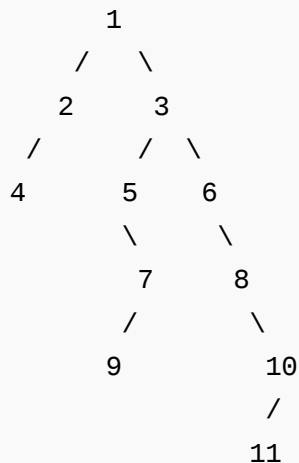


Find depth of the deepest odd level leaf node

Write a C code to get the depth of the deepest odd level leaf node in a binary tree. Consider that level starts with 1. Depth of a leaf node is number of nodes on the path from root to leaf (including both leaf and root).

For example, consider the following tree. The deepest odd level node is the node with value 9 and depth of this node is 5.



We strongly recommend you to minimize the browser and try this yourself first.

The idea is to recursively traverse the given binary tree and while traversing, maintain a variable “level” which will store the current node’s level in the tree. If current node is leaf then check “level” is odd or not. If level is odd then return it. If current node is not leaf, then recursively find maximum depth in left and right subtrees, and return maximum of the two depths.

```
// C program to find depth of the deepest odd level leaf node
```



```

#include <stdio.h>
#include <stdlib.h>

// A utility function to find maximum of two integers
int max(int x, int y) { return (x > y)? x : y; }

// A Binary Tree node
struct Node
{
    int data;
    struct Node *left, *right;
};

// A utility function to allocate a new tree node
struct Node* newNode(int data)
{
    struct Node* node = (struct Node*) malloc(sizeof(struct Node));
    node->data = data;
    node->left = node->right = NULL;
    return node;
}

// A recursive function to find depth of the deepest odd level leaf
int depthOfOddLeafUtil(Node *root, int level)
{
    // Base Case
    if (root == NULL)
        return 0;

    // If this node is a leaf and its level is odd, return its level
    if (root->left==NULL && root->right==NULL && level%2)
        return level;

    // If not leaf, return the maximum value from left and right subtree
    return max(depthOfOddLeafUtil(root->left, level+1),
               depthOfOddLeafUtil(root->right, level+1));
}

/* Main function which calculates the depth of deepest odd level leaf.
   This function mainly uses depthOfOddLeafUtil() */
int depthOfOddLeaf(struct Node *root)
{
    int level = 1, depth = 0;
    return depthOfOddLeafUtil(root, level);
}

// Driver program to test above functions

```

ITT Tech - Official Site

[itt-tech.edu](https://www.it-tech.edu/)

Tech-Oriented Degree Programs.
Education for the Future.



Popular Posts

[All permutations of a given string](#)

[Memory Layout of C Programs](#)

[Understanding "extern" keyword in C](#)

[Median of two sorted arrays](#)

[Tree traversal without recursion and without stack!](#)

[Structure Member Alignment, Padding and Data Packing](#)

[Intersection point of two Linked Lists](#)

[Lowest Common Ancestor in a BST.](#)

[Check if a binary tree is BST or not](#)

[Sorted Linked List to Balanced BST](#)

```

int main()
{
    struct Node* root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);
    root->left->left = newNode(4);
    root->right->left = newNode(5);
    root->right->right = newNode(6);
    root->right->left->right = newNode(7);
    root->right->right->right = newNode(8);
    root->right->left->right->left = newNode(9);
    root->right->right->right->right = newNode(10);
    root->right->right->right->right->right = newNode(11);

    printf("%d is the required depth\n", depthOfOddLeaf(root));
    getchar();
    return 0;
}

```

Output:

5 is the required depth

Time Complexity: The function does a simple traversal of the tree, so the complexity is $O(n)$.

This article is contributed by **Chandra Prakash**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.



ORACLE **12^c**
DATABASE

Oracle Database 12c Training and Certification

Explore new features, maximize
your skills, and achieve your
potential.

[View Courses](#)



Better Than Hadoop.

HPCC Systems is Big Data Processing and Analytics
Open Source. Proven. Trusted.

 LexisNexis® [Learn More](#) 



 695  [Subscribe](#)

Related Topics:

- [Print a Binary Tree in Vertical Order | Set 2 \(Hashmap based Method\)](#)
- [Print Right View of a Binary Tree](#)
- [Red-Black Tree | Set 3 \(Delete\)](#)
- [Construct a tree from Inorder and Level order traversals](#)
- [Print all nodes at distance k from a given node](#)
- [Print a Binary Tree in Vertical Order | Set 1](#)
- [Interval Tree](#)
- [Check if a given Binary Tree is height balanced like a Red-Black Tree](#)

 22  [Tweet](#) 3  1

Writing code in comment? Please use [ideone.com](https://www.ideone.com) and share the link here.

Recent Comments

[affizerv](#) Your example has two 4s on row 3, that's why it...

[Backtracking | Set 7 \(Sudoku\)](#) · 29 minutes ago

[RVM](#) Can someone please elaborate this Qs from above...

[Flipkart Interview | Set 6](#) · 49 minutes ago

[Vishal Gupta](#) I talked about as an Interviewer in general,...

[Software Engineering Lab, Samsung Interview | Set 2](#) · 49 minutes ago

[@meya](#) Working solution for question 2 of 4f2f round....


[Amazon Interview | Set 53 \(For SDE-1\)](#) · 1 hour ago
sandeep void rearrange(struct node *head)
{...

Given a linked list, reverse alternate nodes and

append at the end · 2 hours ago

Neha I think that is what it should return as,
in...


Find depth of the deepest odd level leaf node · 2
hours ago

AdChoices 

► [Binary Tree](#)

► [Graph C++](#)


► [Java Tree](#)

AdChoices 

► [Java to C++](#)

► [Node](#)

► [Tree Root](#)

AdChoices 

► [In Memory Tree](#)

► [Compare C++ Code](#)

► [Tree 3 Level](#)