Difference between sums of odd level and even level nodes of a Binary Tree

Given a a Binary Tree, find the difference between the sum of nodes at odd level and the sum of nodes at even level. Consider root as level 1, left and right children of root as level 2 and so on.

For example, in the following tree, sum of nodes at odd level is (5 + 1 + 4 + 8) which is 18. And sum of nodes at even level is (2 + 6 + 3 + 7 + 9) which is 27. The output for following tree should be 18 – 27 which is -9.

```
      5
    /   \
   2     6
 /  \     \
1    4     8
    /     / \
   3     7   9
```

A straightforward method is to **use level order traversal**. In the traversal, check level of current node, if it is odd, increment odd sum by data of current node, otherwise increment even sum. Finally return difference between odd sum and even sum. See following for implementation of this approach.

C implementation of level order traversal based approach to find the difference.

The problem can also be solved **using simple recursive traversal**. We can recursively calculate the required difference as, value of root's data subtracted by the difference for subtree under left child and the difference for subtree under right child. Following is C implementation of this

Google™ Custom Search

GeeksforGeeks

52,731 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion

approach.

```c
// A recursive program to find difference between sum of nodes at
// odd level and sum at even level
#include <stdio.h>
#include <stdlib.h>

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

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

// The main function that return difference between odd and even level
// nodes
int getLevelDiff(struct node *root)
{
   // Base case
   if (root == NULL)
         return 0;

   // Difference for root is root's data - difference for left subtree
   // - difference for right subtree
   return root->data - getLevelDiff(root->left) - getLevelDiff(root->r
}

// Driver program to test above functions
int main()
{
    struct node *root = newNode(5);
    root->left = newNode(2);
    root->right = newNode(6);
    root->left->left  = newNode(1);
    root->left->right = newNode(4);
    root->left->right->left = newNode(3);
    root->right->right = newNode(8);
    root->right->right->right = newNode(9);
```

Geometric Algorithms

Dynamic DevOps Suite
Fast, flexible and hands-free DevOps from build through Deploy
OM Software
Click to Learn More >

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

```
    root->right->right->left = newNode(7);
    printf("%d is the required difference\n", getLevelDiff(root));
    getchar();
    return 0;
}
```

Output:

```
-9 is the required difference
```

Time complexity of both methods is O(n), but the second method is simple and easy to implement.

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.

Related Tpoics:

- 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

29 Tweet 3 1

Writing code in comment? Please use **ideone.com** and share the link here.

15 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

Arun · 3 months ago

Hi I have implemented it in java.

Here is my piece of code.

public class OddEvenLevelDiff {

public static void main(String[] args) {

// Node root = testcase1();

Node root = testcase2();

int oddLevelSum = oddLevelSum(root);

int evenLevelSum = evenLevelSum(root);

int diff = oddLevelSum - evenLevelSum;

System.out.println("odd sum: "+ oddLevelSum);

695 Subscribe

Recent Comments

affiszerv 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

► Java Tree

► Java to C++

System.out.println("even sum: "+ evenLevelSum);

see more

^ | ⌄ • Reply • Share ›

anonymous • 5 months ago

This also works. Please review it.

```
void diff_sum_odd_even_lev(Node root,int* sum,int k)
{
        if(!root)
                return;
        if(k%2==1)
                *sum+=root->data;
        else
                *sum-=root->data;
        diff_sum_odd_even_lev(root->left,sum,k+1);
        diff_sum_odd_even_lev(root->right,sum,k+1);
}


int main()
{
        Node root=newnode(1);
```

see more

^ | ⌄ • Reply • Share ›

gourav pathak ➦ anonymous • 3 months ago

yes that's correct and works fine

^ | ⌄ • Reply • Share ›

Guest • 5 months ago

```
void diff_sum_odd_even_lev(Node root,int* sum,int k)

{

        if(!root)

                return;

        if(k%2==1)

                *sum+=root->data;

        else

                *sum-=root->data;
```

see more

^ | ⌄ • Reply • Share ›

Raunak Lakhwani • 5 months ago

class Node {

int data;

Node left,right;

}

public class Main {

static Node head;

/**

* @param args

*/

public static void main(String[] args) {

head = new Node();

see more

Reply • Share ›

Zubin • 7 months ago

Can someone please explain the recursion? Thanks in advance!

2 • Reply • Share ›

gourav pathak Zubin • 3 months ago

the nodes which are at odd level for t are at even level for t->left and t->
>data-diff(t->lc)-diff(t->rc)

Reply • Share ›

bhavneet • 8 months ago

int alteranteLevelSum(struct node * root, int level)
{
if(root==NULL)
return 0;

else if(level%2==0)
return 1 + alteranteLevelSum(root->left, level+1) + alteranteLevelSum(root->

else

return alteranteLevelSum(root->left, level+1) + alteranteLevelSum(root->right
}

int DiffOddEvenLevel(struct node * root)
{
return alternateLevelSum (root, 0)- alternateLevelSum(root, 1); // :; :)
}

1 ^ | ⌄ • Reply • Share ›

harshal • 8 months ago

void diffSum(struct tree *root, int level, int *esum, int *osum)
{
if(root==NULL)
return;
if(root->left && root->right)
{
level++;
diffSum(root->left,level,esum,osum);
diffSum(root->right,level,esum,osum);
level--;
}
if(level%2==0)
*esum+=root->data;
else
*osum+=root->data;
}

1 ^ | ⌄ • Reply • Share ›

Shiva • 8 months ago

The sum function returns osum and esum. (osum - esum) gives us the answ

```
void sum(struct node *root,int level,int *esum,int *osum)
```

```
{
    if(root == NULL)
    {
      return;
    }

    if(level%2)
    {
       *esum +=  root->data;
    }
    else
    {
       *osum +=  root->data;
    }

    sum(root->left,level + 1,esum,osum) ;
    sum(root->right,level + 1,esum,osum);
}
```

2 · Reply · Share ›

Ankur Singh Shiva · 6 months ago

This code is ok I did it same way for amazon online test but it was failir
the case :( )

· Reply · Share ›

rahul23 Shiva · 8 months ago

Nobody can beat the logic in the post. Although ur code works fyn but i

1 · Reply · Share ›

Commendable. How did you think of this approach?

```
/* Paste your code here (You may delete these lines if not writing c
```

1 ^ | ˅ • Reply • Share ›

rahul23 ➦ sh • 9 months ago

True that..It is worth to be praised..Hats off to the guy..Awesome logic. think of that approach??This is something exceptional

^ | ˅ • Reply • Share ›

Vipul Verma ➦ rahul23 • 8 months ago

This is power of recursion. How elegant the solution is.

1 ^ | ˅ • Reply • Share ›

Subscribe   Add Disqus to your site