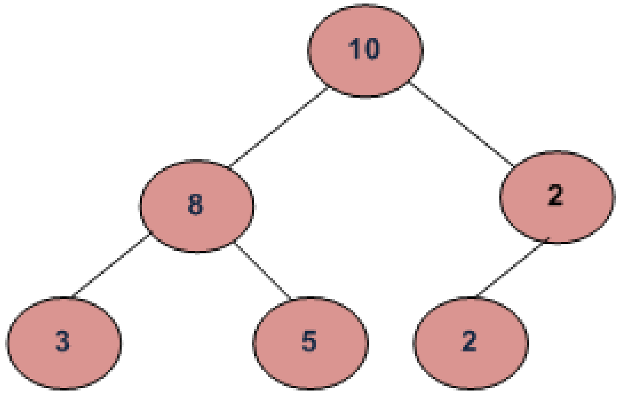
Check for Children Sum Property in a Binary Tree.

Given a binary tree, write a function that returns true if the tree satisfies below property.

For every node, data value must be equal to sum of data values in left and right children. Consider data value as 0 for NULL children. Below tree is an example



Algorithm:
Traverse the given binary tree. For each node check (recursively) if the node and both its children satisfy the Children Sum Property, if so then return true else return false.

Implementation:

```
/* Program to check children sum property */
#include <stdio.h>
#include <stdlib.h>

/* A binary tree node has data, left child and right child */
struct node
{
```

GeeksforGeeks
52,731 people like GeeksforGeeks.

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

```
    int data;
    struct node* left;
    struct node* right;
};

/* returns 1 if children sum property holds for the given
    node and both of its children*/
int isSumProperty(struct node* node)
{
  /* left_data is left child data and right_data is for right child da
  int left_data = 0,  right_data = 0;

  /* If node is NULL or it's a leaf node then
     return true */
  if(node == NULL ||
     (node->left == NULL && node->right == NULL))
    return 1;
  else
  {
    /* If left child is not present then 0 is used
       as data of left child */
    if(node->left != NULL)
      left_data = node->left->data;

    /* If right child is not present then 0 is used
      as data of right child */
    if(node->right != NULL)
      right_data = node->right->data;

    /* if the node and both of its children satisfy the
       property return 1 else 0*/
    if((node->data == left_data + right_data)&&
        isSumProperty(node->left) &&
        isSumProperty(node->right))
      return 1;
    else
      return 0;
  }
}

/*
 Helper function that allocates a new node
 with the given data and NULL left and right
 pointers.
*/
struct node* newNode(int data)
{
```

Geometric Algorithms

Made
For Android™
Optimize 3D game and app performance on the Adreno™ GPU
DOWNLOAD
The Adreno™ SDK
QDevNet
Qualcomm Developer Network

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

```
  struct node* node =
      (struct node*)malloc(sizeof(struct node));
  node->data = data;
  node->left = NULL;
  node->right = NULL;
  return(node);
}

/* Driver program to test above function */
int main()
{
  struct node *root  = newNode(10);
  root->left         = newNode(8);
  root->right        = newNode(2);
  root->left->left   = newNode(3);
  root->left->right  = newNode(5);
  root->right->right = newNode(2);
  if(isSumProperty(root))
    printf("The given tree satisfies the children sum property ");
  else
    printf("The given tree does not satisfy the children sum property

  getchar();
  return 0;
}
```

Time Complexity: O(n), we are doing a complete traversal of the tree.

As an exercise, extend the above question for an n-ary tree.

This question was asked by Shekhar.

Please write comments if you find any bug in the above algorithm or a better way to solve the same problem.

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

1 Tweet 0 0

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

15 Comments **GeeksforGeeks**

Sort by Newest

695 Subscribe

Recent Comments

karthik it should have been max_wrap= max_wrap -...

Maximum circular subarray sum · 1 minute ago

affiszerv Your example has two 4s on row 3, that's why it...

Backtracking | Set 7 (Sudoku) · 45 minutes ago

RVM Can someone please elaborate this Qs from above...

Flipkart Interview | Set 6 · 1 hour ago

Vishal Gupta I talked about as an Interviewer in general,...

Software Engineering Lab, Samsung Interview | Set 2 · 1 hour 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

Join the discussion…

append at the end · 3 hours ago

AdChoices

▶ SUM Function

▶ Binary Tree

▶ Java Tree

AdChoices

▶ Java to C++

▶ Tree Root

▶ Tree View

AdChoices

▶ Check SUM

▶ SUM To

▶ In Memory Tree

AlienOnEarth • 4 days ago

Jave Implementation: Routine returns int value = root.data if the tree satisfies c

static int isChildSumProperty(BTNode root)

{

if(root==null)

return 0;

if(isLeaf(root))

{

return root.data;

}

int l = isChildSumProperty(root.left);

int r = isChildSumProperty(root.right);

see more

Reply • Share ›

neelabhsingh • 2 months ago

GeeksforGeeks.

if((node->data == left_data + right_data)&&
isSumProperty(node->left) &&
isSumProperty(node->right))

```
else
return 0;
}
```

Please check it. There is no use of else please remove it..

^ | ˅ • Reply • Share ›

AlienOnEarth ➦ neelabhsingh • 4 days ago

Else is needed. What if the condition is false? what should function ret

^ | ˅ • Reply • Share ›

NAVEEN PRAJAPATI • 2 months ago

we can also do this by using post order traversal . below is the function given..

```
void check_sum_property(struct node *root)
{

if(root)
{
if(root->data = (root->left->data) + (root->right->data))
{
printf("node following check sum property %d--",root->data);
}
check_sum_property(root->left);
check_sum_property(root->right);
}
}
```

^ | ˅ • Reply • Share ›

invince_guitar • 7 months ago

Similar method..but done in my way.

//..

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

struct node
{
int data;
struct node *l;
struct node *r;
};

struct node* create_node(int num)
{
struct node *temp=(struct node *)malloc(sizeof(struct node));
temp->data=num;
temp->l=NULL;
temp->r=NULL;
```

see more

Reply • Share ›

Sanjith Sakthivel • 9 months ago

```
int childrensum(node *root).
{
if(root==NULL)
{
return 0;
}
if(root->left==NULL&&root->right==NULL)
{
return root->data;
}
if(root->data==childrensum(root->left)+childrensum(root->right))
{
printf("Parent = Sum of Children ");.
return root->data;
}
else
{
printf("Wrong");
return -1;
}
}
```

˄ | ˅ • Reply • Share ›

ubiquitous • 9 months ago

```
[sourcecode language="java"]
int validateChildrenSumProperty(TreeNode t)
{
if(t.left == null && t.right == null)
{
return t.v;
}
int left = 0, right = 0;
if(t.left != null)
{
left = validateChildrenSumProperty(t.left);
}
if(t.right != null)
{
right = validateChildrenSumProperty(t.right);
}
if(left == -1 || right == -1)
{
```

Reply • Share ›

Nitesh • 10 months ago

```
 int sumProperty(node *tree, int sum)
{
    if(tree == NULL && sum != 0)
        return false;

    /*Mease we are on leaf node*/
    if(tree ->left == NULL && tree->right == NULL )
    {
            if((sum-tree->data) ==0)
               return true;
    }
    int subSum = sum - tree->data;
    if(sumProperty(tree->left,subSum) || sumProperty(tree->right, sub
       return true;

    return false;
}
```

Reply • Share ›

abhishek08aug • a year ago

C++ code:

```
 #include <iostream>
#include <stdlib.h>
using namespace std;

class tree_node {
  private:
    int data;
    tree_node * left;
    tree_node * right;
  public:
    tree_node() {
      left=NULL;
      right=NULL;
    }
    void set_data(int data) {
      this->data=data;
```

see more

^ | ⌄ • Reply • Share ›

devil_001 • a year ago

this is another method :

```
 #include <stdio.h>
#include <stdlib.h>
struct node
{
    int data;
    struct node* left;
    struct node* right;
};
struct node* newNode(int data)
{
  struct node* node =
      (struct node*)malloc(sizeof(struct node));
  node->data = data;
  node->left = NULL;
  node->right = NULL;
  return(node);
```

see more

Reply • Share ›

devil_001 • a year ago

another method :

```

#include <stdio.h>
#include <stdlib.h>
struct node
{
    int data;
    struct node* left;
    struct node* right;
};
struct node* newNode(int data)
{
  struct node* node =
      (struct node*)malloc(sizeof(struct node));
  node->data = data;
  node->left = NULL;
  node->right = NULL;
```

see more

Reply • Share ›

Nikin • a year ago

```
{
if(sr == NULL) return true;
int lData = 0, rData = 0;
if(sr->left)
lData = sr->left->data;
if(sr->right)
rData = sr->right->data;

return ( sr->data == (lData + rData) &&
isSumTree(sr->left) && isSumTree(sr->right));

}
```

• Reply • Share ›

Nirdesh Nikin • a year ago

You are missing one condition in a base case here in the 1st line.Your

```
if(sr == null || (sr->left==null && sr->right==null)){
   return true;
}
```

• Reply • Share ›

Teja • 4 years ago

This qn was asked in Amazon written test.

• Reply • Share ›

AT Teja • a year ago

Java code

```
/* public boolean childSum(Node n) {
        if (n == null || n.left == null && n.right == null)
                return true;
        if (n.left == null || n.right == null)
                return n.left == null? (n.data == n.right.data)
        return (n.data == n.left.data + n.right.data) && childS
} */
```

Reply • Share ›

Subscribe   Add Disqus to your site