GeeksforGeeks

A computer science portal for geeks

Placements Practice GATE CS IDE Q&A
GeeksQuiz

Print Ancestors of a given node in Binary Tree

Given a Binary Tree and a key, write a function that prints all the ancestors of the key in the given binary tree.

For example, if the given tree is following Binary Tree and key is 7, then your function should print 4, 2 and 1.

```
              1
            /   \
          2      3
        /  \
      4     5
     /
    7
```

Thanks to Mike , Sambasiva and wgpshashank for their contribution.

C

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

using namespace std;

/* A binary tree node has data, pointer to left child
   and a pointer to right child */
struct node
{
   int data;
   struct node* left;
   struct node* right;
};

/* If target is present in tree, then prints the ancestors
   and returns true, otherwise returns false. */
bool printAncestors(struct node *root, int target)
{
  /* base cases */
  if (root == NULL)
     return false;

  if (root->data == target)
     return true;

  /* If target is present in either left or right subtree of this node,
     then print this node */
  if ( printAncestors(root->left, target) ||
       printAncestors(root->right, target) )
  {
    cout << root->data << " ";
    return true;
  }

  /* Else return false */
  return false;
}

/* Helper function that allocates a new node with the
   given data and NULL left and right pointers. */
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);
}

/* Driver program to test above functions*/
int main()
{

  /* Construct the following binary tree
              1
            /   \
          2      3
        /  \
      4     5
     /
    7
  */
  struct node *root = newnode(1);
  root->left        = newnode(2);
  root->right       = newnode(3);
  root->left->left  = newnode(4);
  root->left->right = newnode(5);
  root->left->left->left  = newnode(7);

  printAncestors(root, 7);

  getchar();
  return 0;
}
```

Java

```
// Java program to check if Binary tree is sum tree or not

// A binary tree node
class Node {

    int data;
    Node left, right, nextRight;

    Node(int item) {
        data = item;
        left = right = nextRight = null;
    }
}

class BinaryTree {

    static Node root;

    /* If target is present in tree, then prints the ancestors
     and returns true, otherwise returns false. */
    boolean printAncestors(Node node, int target) {

         /* base cases */
        if (node == null) {
            return false;
        }

        if (node.data == target) {
            return true;
        }

        /* If target is present in either left or right subtree of this node,
         then print this node */
        if (printAncestors(node.left, target)
                || printAncestors(node.right, target)) {
            System.out.print(node.data + " ");
            return true;
        }
        /* Else return false */
        return false;
    }

    public static void main(String args[]) {
        BinaryTree tree = new BinaryTree();
        tree.root = new Node(1);
        tree.root.left = new Node(2);
        tree.root.right = new Node(3);
        tree.root.left.left = new Node(4);
        tree.root.left.right = new Node(5);
        tree.root.left.left.left = new Node(7);

        tree.printAncestors(root, 7);

    }
}

// This code has been contributed by Mayank Jaiswal
```

Output:

```
4 2 1
```

Time Complexity: O(n) where n is the number of nodes in the given Binary Tree.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

23 Comments Category: Trees

Related Posts:

- Check if removing an edge can divide a Binary Tree in two halves
- Check sum of Covered and Uncovered nodes of Binary Tree
- Lowest Common Ancestor in a Binary Tree | Set 2 (Using Parent Pointer)
- Construct a Binary Search Tree from given postorder
- BFS vs DFS for Binary Tree
- Maximum difference between node and its ancestor in Binary Tree
- Inorder Non-threaded Binary Tree Traversal without Recursion or Stack
- Check if leaf traversal of two Binary Trees is same?

(Login to Rate and Mark)

2.5 Average Difficulty : **2.5/5.0**
Based on **22** vote(s)

□ Add to TODO List
□ Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.