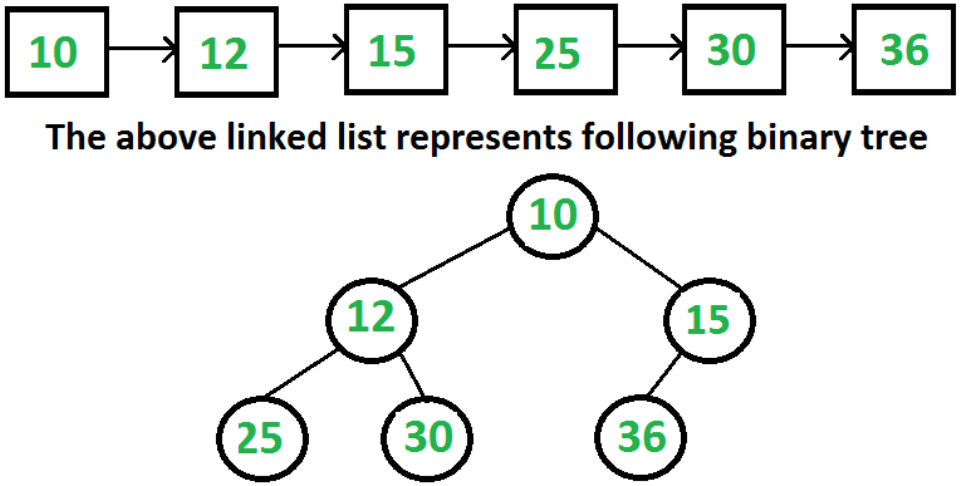
GeeksforGeeks
A computer science portal for geeks

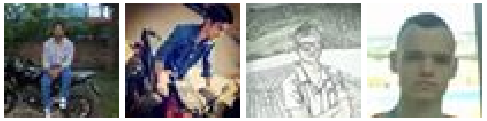
Construct Complete Binary Tree from its Linked List Representation

Given Linked List Representation of Complete Binary Tree, construct the Binary tree. A complete binary tree can be represented in an array in the following approach.

If root node is stored at index i, its left, and right children are stored at indices 2*i+1, 2*i+2 respectively.
Suppose tree is represented by a linked list in same way, how do we convert this into normal linked representation of binary tree where every node has data, left and right pointers? In the linked list representation, we cannot directly access the children of the current node unless we traverse the list.



We are mainly given level order traversal in sequential access form. We know head of linked list is always is root of the tree. We take the first node as root and we also know that the next two



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

nodes are left and right children of root. So we know partial Binary Tree. The idea is to do Level order traversal of the partially built Binary Tree using queue and traverse the linked list at the same time. At every step, we take the parent node from queue, make next two nodes of linked list as children of the parent node, and enqueue the next two nodes to queue.

1. Create an empty queue.
2. Make the first node of the list as root, and enqueue it to the queue.
3. Until we reach the end of the list, do the following.
………**a.** Dequeue one node from the queue. This is the current parent.
………**b.** Traverse two nodes in the list, add them as children of the current parent.
………**c.** Enqueue the two nodes into the queue.

Below is the code which implements the same in C++.

```
// C++ program to create a Complete Binary tree from its Linked List
// Representation
#include <iostream>
#include <string>
#include <queue>
using namespace std;

// Linked list node
struct ListNode
{
    int data;
    ListNode* next;
};

// Binary tree node structure
struct BinaryTreeNode
{
    int data;
    BinaryTreeNode *left, *right;
};

// Function to insert a node at the beginning of the Linked List
void push(struct ListNode** head_ref, int new_data)
{
    // allocate node and assign data
    struct ListNode* new_node = new ListNode;
    new_node->data = new_data;

    // link the old list off the new node
```

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

```
    new_node->next = (*head_ref);

    // move the head to point to the new node
    (*head_ref)     = new_node;
}

// method to create a new binary tree node from the given data
BinaryTreeNode* newBinaryTreeNode(int data)
{
    BinaryTreeNode *temp = new BinaryTreeNode;
    temp->data = data;
    temp->left = temp->right = NULL;
    return temp;
}
```

```
// converts a given linked list representing a complete binary tree in
// linked representation of binary tree.
void convertList2Binary(ListNode *head, BinaryTreeNode* &root)
{
    // queue to store the parent nodes
    queue<BinaryTreeNode *> q;

    // Base Case
    if (head == NULL)
    {
        root = NULL; // Note that root is passed by reference
        return;
    }

    // 1.) The first node is always the root node, and add it to the q
    root = newBinaryTreeNode(head->data);
    q.push(root);

    // advance the pointer to the next node
    head = head->next;

    // until the end of linked list is reached, do the following steps
    while (head)
    {
        // 2.a) take the parent node from the q and remove it from q
        BinaryTreeNode* parent = q.front();
        q.pop();

        // 2.c) take next two nodes from the linked list. We will add
        // them as children of the current parent node in step 2.b. Pu
        // into the queue so that they will be parents to the future n
        BinaryTreeNode *leftChild = NULL, *rightChild = NULL;
        leftChild = newBinaryTreeNode(head->data);
        q.push(leftChild);
        head = head->next;
        if (head)
        {
            rightChild = newBinaryTreeNode(head->data);
            q.push(rightChild);
            head = head->next;
        }

        // 2.b) assign the left and right children of parent
        parent->left = leftChild;
        parent->right = rightChild;
    }
}

// Utility function to traverse the binary tree after conversion
void inorderTraversal(BinaryTreeNode* root)
{
    if (root)
    {
        inorderTraversal( root->left );
        cout << root->data << " ";
        inorderTraversal( root->right );
    }
}

// Driver program to test above functions
int main()
{
    // create a linked list shown in above diagram
    struct ListNode* head = NULL;
    push(&head, 36);  /* Last node of Linked List */
    push(&head, 30);
    push(&head, 25);
    push(&head, 15);
    push(&head, 12);
    push(&head, 10); /* First node of Linked List */

    BinaryTreeNode *root;
    convertList2Binary(head, root);

    cout << "Inorder Traversal of the constructed Binary Tree is: \n";
    inorderTraversal(root);
    return 0;
}
```

Recent Comments

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

Backtracking | Set 7 (Sudoku) · 32 minutes ago

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

Flipkart Interview | Set 6 · 52 minutes ago

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

Software Engineering Lab, Samsung Interview | Set 2 · 52 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

Output:

```
Inorder Traversal of the constructed Binary Tree is:
25 12 30 10 36 15
```

Time Complexity: Time complexity of the above solution is O(n) where n is the number of nodes.

This article is compiled by **Ravi Chandra Enaganti**. 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

49 Tweet 3 1

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

21 Comments **GeeksforGeeks**

Sort by Newest ▾

Join the discussion…

monu • 12 days ago

hi

^ | ˅ • Reply • Share ›

sunil • a month ago

Another way would be to count the number of nodes > send half the nodes to l
number send the extra one to left)
recursively build the left subtree (termination condition would be when there ar
(while building the tree keep sending the next node up the tree for building the

^ | ˅ • Reply • Share ›

Dave • 2 months ago

Call me crazy but isn't a binary tree supposed to be sorted?

^ | ˅ • Reply • Share ›

Monin Anicet Sia • 9 months ago

tu est ou?

^ | ˅ • Reply • Share ›

Celeste Bah • 9 months ago

very strong man!

Reply • Share ›

SYED IMRAN • 9 months ago

```
#include
#include
#include
struct tnode
{
int data;
struct tnode *left,*right;
};
struct node
{
int data;
struct node *next;
};
typedef struct tnode tnode;
typedef struct node node;
node * head=NULL;
void push(node **h,int t)
{
```

see more

Reply • Share ›

sandeep kothapally • 10 months ago

i have a code table like for example

2 assigned as "10"

3 assigned as "010"

4 assigned as "0110"

5 assigned as "0111"

now i wanted to write it as a binary tree so that if i input
10010 i could traverse the tree and decode it as 2 3
my query is to how to design such tree so that i can use it at the decoder

Reply • Share ›

nilesh • 11 months ago

consider this:

```
struct node *createtree(List *head,int count)
{
List *ptr;
while(i++next;
}
if(!ptr)
return;
struct node *root=newnode(ptr->data);
root->left=createtree(head,2*count);
root->right=createtree(head,2*count+1);
return root;
}
```

Reply • Share ›

AMIT • 11 months ago

your solution uses o(n) time and o(n) space
a similar but much easier solution, I think--
1. copy all the elements of list into an array
2. build the binary tree
this will also take o(n) time and o(n)space

Reply • Share ›

EOF ➦ AMIT • 10 months ago

I was going to comment the same, but saw yours ... :D

Reply Share ›

abhishek08aug • 11 months ago

Intelligent :D

Reply • Share ›

Prabhakaran Dhandapani • a year ago

```
void convert( NODE *ptr, NODE *left, NODE *right ) {
if( left == NULL ) right = NULL;
if( ptr ) {
if( ptr == start ) root = ptr;
if( left == NULL || right == NULL )
convert( ptr -> right, NULL, NULL );
else convert( ptr -> right, right -> right, right -> right -> right );
ptr -> left = left;
ptr -> right = right;
}
}
```

function call should be like :

```
convert(start, start -> right, start -> right -> right);
```

Explanation will be uploaded tomorrow on.
http://www.facebook.com/groups...

Reply • Share ›

Vimal • a year ago

There is one bug in the above code.
Queue implementation is not correct.
After one iteration of while loop above , queue would be
{15,12} but it should be { 12,15}
Correct me if i am wrong.

Reply • Share ›

neon007 · a year ago

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

typedef struct list {
int data;
struct list * next;
} list;

typedef struct tree {
int data;
struct tree * left;
struct tree * right;
struct tree * next;
} tree;

typedef struct next {
struct next * next;
} next;
```

see more

Reply · Share ›

Aadarsh · a year ago

Can we simply do this?

```
 struct node* createTree(int m)
{
    if(m<=size)
    {
        struct node *q = createNode(arr[m]);
        q->left = createTree(m*2);
        q->right = createTree(2*m + 1);
        return q;
    }
    else return NULL;
```

^ | ⌄ • Reply • Share ›

Meghasyam • a year ago

Hi Ravi

why so complex solution ,
we can simply do in 2 steps

1. traverse the list and get one node at a time
2. send this value to complete binary tree functions

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

^ | ⌄ • Reply • Share ›

Ravi Chandra ➦ Meghasyam • a year ago

Hi Meghasyam,

You have simplified the second step to one sentence. When it actually need the steps that I have given.

Can you elaborate step 2 on how are you going to implement?

^ | ⌄ • Reply • Share ›

Uddhav ➦ Meghasyam • a year ago

What is the time complexity in case of your solution? I think it will be nl
complexity to be n.

| • Reply • Share ›

ceiling fans with lights at wa • a year ago

You can easily get ceiling fan as market is full of this
item but it becomes important to trust only the renowned name.
Investing in this type of lighting unit is something that you should do for
you can be sitting in your living room, dining room or bedroom and
marveling at the lights and beautiful blades that they have plus you get to expe
cool breezes at the same time. Enhancing with decorative table lamps not onl
to your room, but also a soft glow that is guaranteed to improve mood.

^ | ˅ • Reply • Share ›

Uddhav • a year ago

This is nice, much like a BFS traversal for tree or say it is reversing BFS trave

^ | ˅ • Reply • Share ›

John Parkins • a year ago

nice solution..
404-894-9877

^ | ˅ • Reply • Share ›

✉ Subscribe   Ⓓ Add Disqus to your site