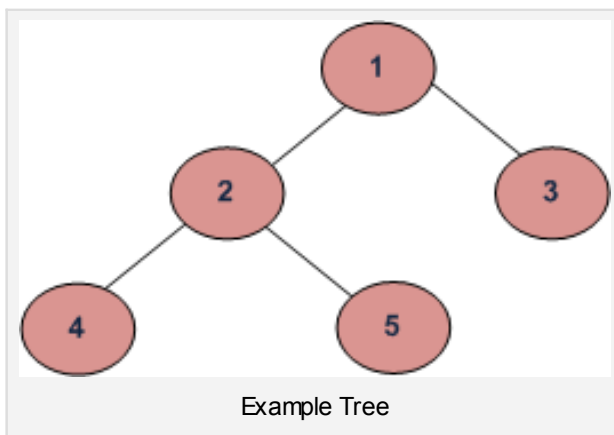


Write a C program to Delete a Tree.

To delete a tree we must traverse all the nodes of the tree and delete them one by one. So which traversal we should use – Inorder or Preorder or Postorder. Answer is simple – Postorder, because before deleting the parent node we should delete its children nodes first

We can delete tree with other traversals also with extra space complexity but why should we go for other traversals if we have Postorder available which does the work without storing anything in same time complexity.

For the following tree nodes are deleted in order – 4, 5, 2, 3, 1



Program

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

/* A binary tree node has data, pointer to left child
and a pointer to right child */
```

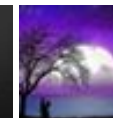
Google™ Custom Search



GeeksforGeeks



52,731 people like [GeeksforGeeks](#).



[Interview Experiences](#)

[Advanced Data Structures](#)

[Dynamic Programming](#)

[Greedy Algorithms](#)

[Backtracking](#)

[Pattern Searching](#)

[Divide & Conquer](#)

[Mathematical Algorithms](#)

[Recursion](#)

[Geometric Algorithms](#)

```

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

/* 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);
}

```

```

/* This function traverses tree in post order to
   to delete each and every node of the tree */
void deleteTree(struct node* node)
{
    if (node == NULL) return;

    /* first delete both subtrees */
    deleteTree(node->left);
    deleteTree(node->right);

    /* then delete the node */
    printf("\\n Deleting node: %d", node->data);
    free(node);
}

```

```

/* Driver program to test deleteTree function*/
int main()
{
    struct node *root = newNode(1);
    root->left      = newNode(2);
    root->right     = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    deleteTree(root);
    root = NULL;
}

```



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

```

printf("\n Tree deleted ");

getchar();
return 0;
}

```

The above deleteTree() function deletes the tree, but doesn't change root to NULL which may cause problems if the user of deleteTree() doesn't change root to NULL and tries to access values using root pointer. We can modify the deleteTree() function to take reference to the root node so that this problem doesn't occur. See the following code.

```

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

/* 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;
};

/* 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);
}

/* This function is same as deleteTree() in the previous program */
void _deleteTree(struct node* node)
{
    if (node == NULL) return;

    /* first delete both subtrees */
    _deleteTree(node->left);
    _deleteTree(node->right);
}

```

Custom market
research at scale.

Get \$75 off

 Google consumer surveys





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)
,

```

/* then delete the node */
printf("\n Deleting node: %d", node->data);
free(node);
}

/* Deletes a tree and sets the root as NULL */
void deleteTree(struct node** node_ref)
{
    _deleteTree(*node_ref);
    *node_ref = NULL;
}

/* Driver program to test deleteTree function*/
int main()
{
    struct node *root = newNode(1);
    root->left      = newNode(2);
    root->right     = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    // Note that we pass the address of root here
    deleteTree(&root);
    printf("\n Tree deleted ");

    getchar();
    return 0;
}

```

Time Complexity: $O(n)$

Space Complexity: If we don't consider size of stack for function calls then $O(1)$ otherwise $O(n)$



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](#)



3



Tweet

0



1

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

{...

Given a linked list, reverse alternate nodes and
append at the end · 3 hours ago

AdChoices

► [Binary Tree](#)

► [Java Tree](#)

► [Java to C++](#)

AdChoices

► [XML Tree Viewer](#)

► [Red Black Tree](#)

► [Tree Structure](#)

AdChoices

► [Root Cause Tree](#)

► [Tree Trees](#)

► [In the Tree](#)

@geeksforgeeks, **Some rights reserved**

Contact Us!

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team