GeeksforGeeks

A computer science portal for geeks

Login

Home	Algorithms	DS	GATE	Interv	view Corner	Q&A	С	C++	Java	Books	Contribute	Ask a Q	About
Array	Bit Magic	C/C+	+ Arti	cles	GFacts	Linked L	ist	MCQ	Misc	Outpu	t String	Tree	Graph

Iterative Preorder Traversal

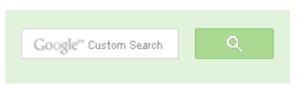
Given a Binary Tree, write an iterative function to print Preorder traversal of the given binary tree.

Refer this for recursive preorder traversal of Binary Tree. To convert an inherently recursive procedures to iterative, we need an explicit stack. Following is a simple stack based iterative process to print Preorder traversal.

- 1) Create an empty stack *nodeStack* and push root node to stack.
- 2) Do following while *nodeStack* is not empty.
-a) Pop an item from stack and print it.
-b) Push right child of popped item to stack
-c) Push left child of popped item to stack

Right child is pushed before left child to make sure that left subtree is processed first.

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include <stack>
using namespace std;
/* A binary tree node has data, left child and 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.*/
```





52,731 people like GeeksforGeeks.











Interview Experiences

Advanced Data Structures

Dynamic Programming

Greedy Algorithms

Backtracking

Pattern Searching

Divide & Conquer

Mathematical Algorithms

Recursion

```
struct node* newNode(int data)
    struct node* node = new struct node;
    node->data = data;
    node->left = NULL;
    node->right = NULL;
    return (node);
// An iterative process to print preorder traversal of Binary tree
void iterativePreorder(node *root)
    // Base Case
    if (root == NULL)
       return;
    // Create an empty stack and push root to it
    stack<node *> nodeStack;
    nodeStack.push(root);
    /* Pop all items one by one. Do following for every popped item
       a) print it
      b) push its right child
       c) push its left child
    Note that right child is pushed first so that left is processed fi
    while (nodeStack.empty() == false)
        // Pop the top item from stack and print it
        struct node *node = nodeStack.top();
        printf ("%d ", node->data);
        nodeStack.pop();
        // Push right and left children of the popped node to stack
        if (node->right)
           nodeStack.push(node->right);
        if (node->left)
            nodeStack.push(node->left);
// Driver program to test above functions
int main()
    /* Constructed binary tree is
```



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 *root = newNode(10);
root->left = newNode(8);
root->right = newNode(2);
root->left->left = newNode(3);
root->left->right = newNode(5);
root->right->left = newNode(2);
iterativePreorder(root);
return 0;
```

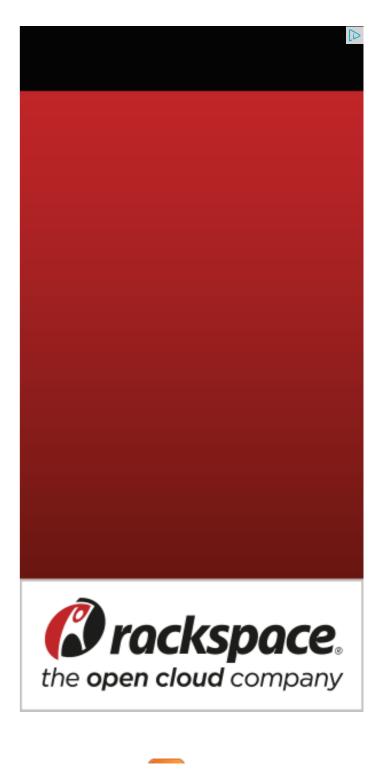
Output:

10 8 3 5 2 2

This article is compiled by Saurabh Sharma and reviewed by GeeksforGeeks team. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above







- 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









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

19 Comments

GeeksforGeeks

Sort by Newest ▼



Join the discussion...



carmen cojocaru · 3 months ago

Can you post an implementation for the post-order also? I've seen some versi complicated. Yours is so clean. Thank you.



Vivek • 6 months ago

no need to push the right child into the stack

void preOrderIterative(struct node *root)





Recent Comments

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

Backtracking | Set 7 (Sudoku) · 35 minutes ago

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

Flipkart Interview | Set 6 55 minutes ago

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

Software Engineering Lab, Samsung Interview | Set

2 · 55 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 · 3 hours ago

```
struct stknode *st=NULL;

while(!empty(st) || root)

{
    while(root)
    {
        printf("%d ",root->data);
```

see more

```
Amit Bgl • 9 months ago

wow code :D

^ | V • Reply • Share >

dex • 9 months ago

/*

iterative preorder of bst using explicit stack

*/

#include<stdio.h>

#include<stdlib.h>

struct tree

{
```

- AdChoices [>
- ► Graph Java
- ► Java to C++
- ▶ Preorder
- AdChoices [>
- ▶ Java Array
- ► Tree Root
- ▶ Node
- AdChoices [>
- ► Tree Block
- ► Stack
- ▶ Null Pointer

```
ιπτ σατα;
          struct tree *left;
          struct tree *right;
  };
  typedef struct tree node;
  void addnode(node *,int);
  struct llist
                                                  see more
2 ^ Reply · Share >
       dex → dex · 9 months ago
      debugged!,
      I constructed the tree itself wrong while checking p->data and n.
      dex ⋅ 9 months ago
iterative preorder of bst using explicit stack
*/
#include
#include
struct tree
int data;
struct tree *left;
struct tree *right;
```

```
typedef struct tree node;
void addnode(node *,int);
struct llist
struct tree *to:
                                                 see more
zyzz • 10 months ago
i think this one is easy
   /void preorder(struct node *temp){
  int top=0;
  struct node *s[20];
  s[0]=NULL;
  printf("preorder : \n");
  while(temp!=NULL){
      printf("%d \t", temp->data);
      if(temp->right!=NULL){
          s[++top]=temp->right;
      if(temp->left!=NULL){
```

see more





```
SHASHI KUMAR • a year ago
#include
#include
struct node
int data;
struct node *right;
struct node *left;
}*root,*S[10];
int i=0;
void Push(struct node *p)
S[i++]=p;
struct node *Pop()
return S[--i];
void preorder()
```

see more

∧ | ∨ • Reply • Share ›



Veer Verma • a year ago

Is modified version of Morris Traversal possible for PreOrder??



Ashok • 2 years ago



Using stack is as good as using recursion. Modify the morris algorithm from in order to get the intended answer.

```
2 ^ Reply · Share >
```



Ashok • 2 years ago

Using stack is as good as using recursion. Modify the morris algorithm from in order to get the intended answer.

```
/* Paste your code here (You may delete these lines if not writing code | V • Reply • Share >
```



Venki • 2 years ago

I guess the code can be refactored. Preorder relatively consumes less stack s the nodes to stack, which is not necessary. See sample code (not tested),

```
етге (
              pMove = s.top();
              s.pop();
      ReplyShare
       Palash → Venki • a year ago
      You'd at best be saving one node space in the stack, that you are anyw
      beginning. No point of this.
      Venki → Palash • a year ago
             It is nothing but do-undo beheviour. It consumes processing po
             NULL doesn't cost much when compared to repeated do-undo
             Suman • 2 years ago
[sourcecode language="JAVA"]
public static void traverselterative(TreeNode root){
Stack<TreeNode> stack = new Stack<TreeNode>();
stack.push(root);
TreeNode currentNode = root;
while(!stack.isEmpty()){
while (currentNode != null){
System.out.println(currentNode.data);
currentNode = currentNode.left:
if (currentNode != null){
stack.push(currentNode);
```

```
currentNode = stack.pop();
currentNode = currentNode.right;
if (currentNode != null){
stack.push(currentNode);
Leet • 2 years ago
```



Can we do preorder traversal without recursion and without stack?



Vikas → Leet • 2 years ago

That would be using Morris Traversal. Look it up on wiki.



atul → Vikas • 2 years ago

morris traversal does inorder traversal

```
/* Paste your code here (You may delete these lines if r
```





Add Disgus to your site