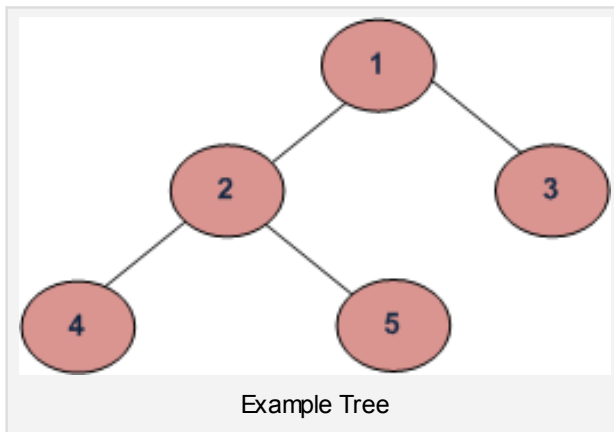


Level Order Tree Traversal

Level order traversal of a tree is **breadth first traversal** for the tree.



Level order traversal of the above tree is 1 2 3 4 5

METHOD 1 (Use function to print a given level)

Algorithm:

There are basically two functions in this method. One is to print all nodes at a given level (printGivenLevel), and other is to print level order traversal of the tree (printLevelorder). printLevelorder makes use of printGivenLevel to print nodes at all levels one by one starting from root.

```

/*Function to print level order traversal of tree*/
printLevelorder(tree)
for d = 1 to height(tree)
    printGivenLevel(tree, d);
  
```

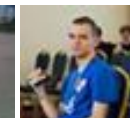
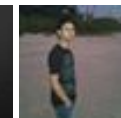
Google™ Custom Search



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

```

/*Function to print all nodes at a given level*/
printGivenLevel(tree, level)
if tree is NULL then return;
if level is 1, then
    print(tree->data);
else if level greater than 1, then
    printGivenLevel(tree->left, level-1);
    printGivenLevel(tree->right, level-1);

```

Implementation:

```

#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;
};

/*Function prototypes*/
void printGivenLevel(struct node* root, int level);
int height(struct node* node);
struct node* newNode(int data);

/* Function to print level order traversal a tree*/
void printLevelOrder(struct node* root)
{
    int h = height(root);
    int i;
    for(i=1; i<=h; i++)
        printGivenLevel(root, i);
}

/* Print nodes at a given level */
void printGivenLevel(struct node* root, int level)
{
    if(root == NULL)
        return;
    if(level == 1)

```

Build Web Apps
in Minutes

NEBULON SYSTEMS

The ACME FOOD COMPANY

Free Download!

IRON SPEED

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("%d ", root->data);
else if (level > 1)
{
    printGivenLevel(root->left, level-1);
    printGivenLevel(root->right, level-1);
}
}

/* Compute the "height" of a tree -- the number of
   nodes along the longest path from the root node
   down to the farthest leaf node.*/
int height(struct node* node)
{
    if (node==NULL)
        return 0;
    else
    {
        /* compute the height of each subtree */
        int lheight = height(node->left);
        int rheight = height(node->right);

        /* use the larger one */
        if (lheight > rheight)
            return(lheight+1);
        else return(rheight+1);
    }
}

/* 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()
{
    struct node *root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);

```

Market research
that's fast and
accurate.

Get \$75 off

 Google consumer surveys



```

root->left->left = newNode(4);
root->left->right = newNode(5);

printf("Level Order traversal of binary tree is \n");
printLevelOrder(root);

getchar();
return 0;
}

```

Time Complexity: $O(n^2)$ in worst case. For a skewed tree, printGivenLevel() takes $O(n)$ time where n is the number of nodes in the skewed tree. So time complexity of printLevelOrder() is $O(n) + O(n-1) + O(n-2) + \dots + O(1)$ which is $O(n^2)$.

METHOD 2 (Use Queue)

Algorithm:

For each node, first the node is visited and then its child nodes are put in a FIFO queue.

```

printLevelorder(tree)
1) Create an empty queue q
2) temp_node = root /*start from root*/
3) Loop while temp_node is not NULL
    a) print temp_node->data.
    b) Enqueue temp_node's children (first left then right children) to q
    c) Dequeue a node from q and assign its value to temp_node

```

Implementation:

Here is a simple implementation of the above algorithm. Queue is implemented using an array with maximum size of 500. We can implement queue as linked list also.

```

#include <stdio.h>
#include <stdlib.h>
#define MAX_Q_SIZE 500

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

```

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
append at the end · 3 hours ago

AdChoices 

[▶ Binary Tree](#)

[▶ Java Tree](#)


[▶ Graph C++](#)

AdChoices 

► [Java to C++](#)

► [Graph Java](#)

► [Root Tree](#)

AdChoices 

► [Red Black Tree](#)

► [Tree Structure](#)

► [Tree Trees](#)

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

/* function prototypes */
struct node** createQueue(int *, int *);
void enqueue(struct node **, int *, struct node *);
struct node *deQueue(struct node **, int *);

/* Given a binary tree, print its nodes in level order
   using array for implementing queue */
void printLevelOrder(struct node* root)
{
    int rear, front;
    struct node **queue = createQueue(&front, &rear);
    struct node *temp_node = root;

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

        /*Enqueue left child */
        if(temp_node->left)
            enqueue(queue, &rear, temp_node->left);

        /*Enqueue right child */
        if(temp_node->right)
            enqueue(queue, &rear, temp_node->right);

        /*Dequeue node and make it temp_node*/
        temp_node = deQueue(queue, &front);
    }
}

/*UTILITY FUNCTIONS*/
struct node** createQueue(int *front, int *rear)
{
    struct node **queue =
        (struct node **)malloc(sizeof(struct node*) *MAX_Q_SIZE);

    *front = *rear = 0;
    return queue;
}

void enqueue(struct node **queue, int *rear, struct node *new node)
```

```

{
    queue[*rear] = new_node;
    (*rear)++;
}

struct node *deQueue(struct node **queue, int *front)
{
    (*front)++;
    return queue[*front - 1];
}

/* 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()
{
    struct node *root = newNode(1);
    root->left = newNode(2);
    root->right = newNode(3);
    root->left->left = newNode(4);
    root->left->right = newNode(5);

    printf("Level Order traversal of binary tree is \n");
    printLevelOrder(root);

    getchar();
    return 0;
}

```

Time Complexity: $O(n)$ where n is number of nodes in the binary tree

References:

http://en.wikipedia.org/wiki/Breadth-first_traversal

Please write comments if you find any bug in the above programs/algorithms or other ways to

solve the same problem.



Better Than Hadoop.

HPCC Systems is Big Data Processing and Analytics
Open Source. Proven. Trusted.

 LexisNexis® [Learn More](#) 

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](#)



10



Tweet

0



2

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