```cpp

////// Insert a Node in binary tree//////////////////////////////////////////////////
/////// insert the node where we find a node whose left or right child is null///////
#include<stdlib.h>
#include<stdio.h>
#include<iostream>
#include<queue>

using namespace std;

struct BinaryNode
{
    struct BinaryNode *left;
    struct BinaryNode *right;
    int data;

};

struct BinaryNode * createBinaryNode(int value)
{
    struct BinaryNode *B=(struct BinaryNode *)malloc(sizeof(struct BinaryNode));
    B->right=NULL;
    B->left=NULL;
    B->data=value;
    return B;
};

queue<BinaryNode *> q;

void insertInBinaryNode(struct BinaryNode *root,struct BinaryNode *newNode)
{
    if(root==NULL)
    {
        root=newNode;
        return;
    }
    else
        q.push(root);

    while(!q.empty())
    {
        struct BinaryNode *temp=q.front();
        q.pop();
        if(temp->left)
            q.push(temp->left);
        else
        {
            temp->left=newNode;
            return;
        }


        if(temp->right)
            q.push(temp->right);
        else
        {
            temp->right=newNode;
            return;
        }
    }

}

queue<BinaryNode *> q1;
```

```c

void levelOrderTraversal(struct BinaryNode *root)
{
    if(root==NULL)
        return;
    else
        q1.push(root);

    while(!q1.empty())
    {
        struct BinaryNode *temp=q1.front();
        printf("%2d\t",temp->data);
        q1.pop();
        if(temp->left)
            q1.push(temp->left);
        if(temp->right)
            q1.push(temp->right);
    }

}

int main()
{
    struct BinaryNode *root=createBinaryNode(20);
    struct BinaryNode *newNode=createBinaryNode(200);
    root->left=createBinaryNode(30);
    root->right=createBinaryNode(40);
    root->left->left=createBinaryNode(50);
    //root->left->right=createBinaryNode(60);
    root->right->left=createBinaryNode(70);
    root->right->right=createBinaryNode(80);
    printf("Level Order Traversal Before Insering node\n");
    levelOrderTraversal(root);
    insertInBinaryNode(root,newNode);
    printf("\nLevel Order Traversal after Insering node\n");
    levelOrderTraversal(root);
}
```