```
#include <iostream>
#include <queue>
using namespace std;
struct Node
{
    int key;
    Node *left, *right;
    bool isThreaded;
};
void setQ(Node *root, std::queue<Node *> *q)
{
    if (root == NULL)
        return;
    if (root->left)
        setQ(root->left, q);
    q->push(root);
    if (root->right)
        setQ(root->right, q);
}
void generate_thrdtre(Node *root, std::queue<Node *> *q)
{
    if (root == NULL)
        return;
    if (root->left)
        generate_thrdtre(root->left, q);
    q->pop();
    if (root->right)
        generate_thrdtre(root->right, q);
    else
    {
        root->right = q->front();
        root->isThreaded = true;
    }
}
void createThreaded(Node *root)
{
    std::queue<Node *> q;
    setQ(root, &q);
    generate_thrdtre(root, &q);
}
Node *leftMost(Node *root)
{
    while (root != NULL && root->left != NULL)
```

```
        root = root->left;
    return root;
}
void inOrder(Node *root)
{
    if (root == NULL)
        return;
    Node *cur = leftMost(root);
    while (cur != NULL)
    {
        cout << cur->key << "";
        if (cur->isThreaded)
            cur = cur->right;
        else
            cur = leftMost(cur->right);
    }
}
Node *createND(int key)
{
    Node *temp = new Node;
    temp->left = temp->right = NULL;
    temp->key = key;
    return temp;
}
int main()
{
    Node *root = createND(1);
    root->left = createND(2);
    root->right = createND(3);
    root->left->left = createND(4);
    root->left->right = createND(5);
    createThreaded(root);
    cout << "Threaded tree : ";
    inOrder(root);
    return 0;
}
```