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

// Definition for a binary tree node
struct TreeNode {
    int val;
    struct TreeNode *left;
    struct TreeNode *right;
};

// Function to create a new node
struct TreeNode* createNode(int val) {
    struct TreeNode* newNode = (struct TreeNode*)malloc(sizeof(struct TreeNode));
    newNode->val = val;
    newNode->left = NULL;
    newNode->right = NULL;
    return newNode;
}

// Function to invert the binary tree
struct TreeNode* invertTree(struct TreeNode* root) {
    if (root == NULL) {
        return NULL;
    }

    // Swap left and right children of the current node
    struct TreeNode* temp = root->left;
    root->left = invertTree(root->right);
    root->right = invertTree(temp);

    return root;
```

```
}

// Function to print the inorder traversal of the binary tree
void printInorder(struct TreeNode* root) {
    if (root == NULL) {
        return;
    }

    printInorder(root->left);
    printf("%d ", root->val);
    printInorder(root->right);
}

int main() {
    // Create the binary tree
    struct TreeNode* root = createNode(4);
    root->left = createNode(2);
    root->right = createNode(7);
    root->left->left = createNode(1);
    root->left->right = createNode(3);
    root->right->left = createNode(6);
    root->right->right = createNode(9);

    printf("Original Tree (Inorder Traversal): ");
    printInorder(root);
    printf("\n");

    // Invert the binary tree
    struct TreeNode* invertedRoot = invertTree(root);

    printf("Inverted Tree (Inorder Traversal): ");
```

```
    printInorder(invertedRoot);
    printf("\n");

    return 0;
}
```