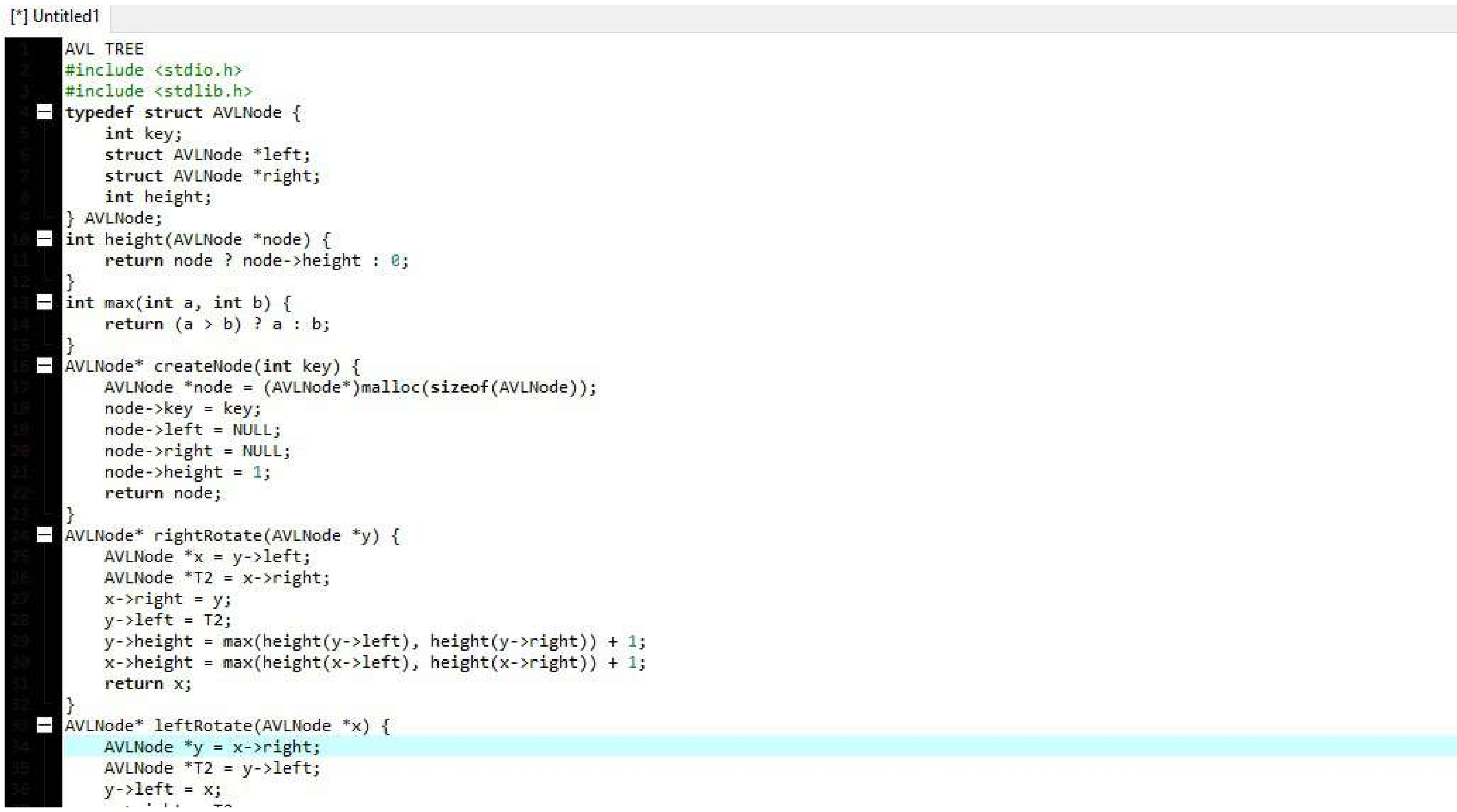
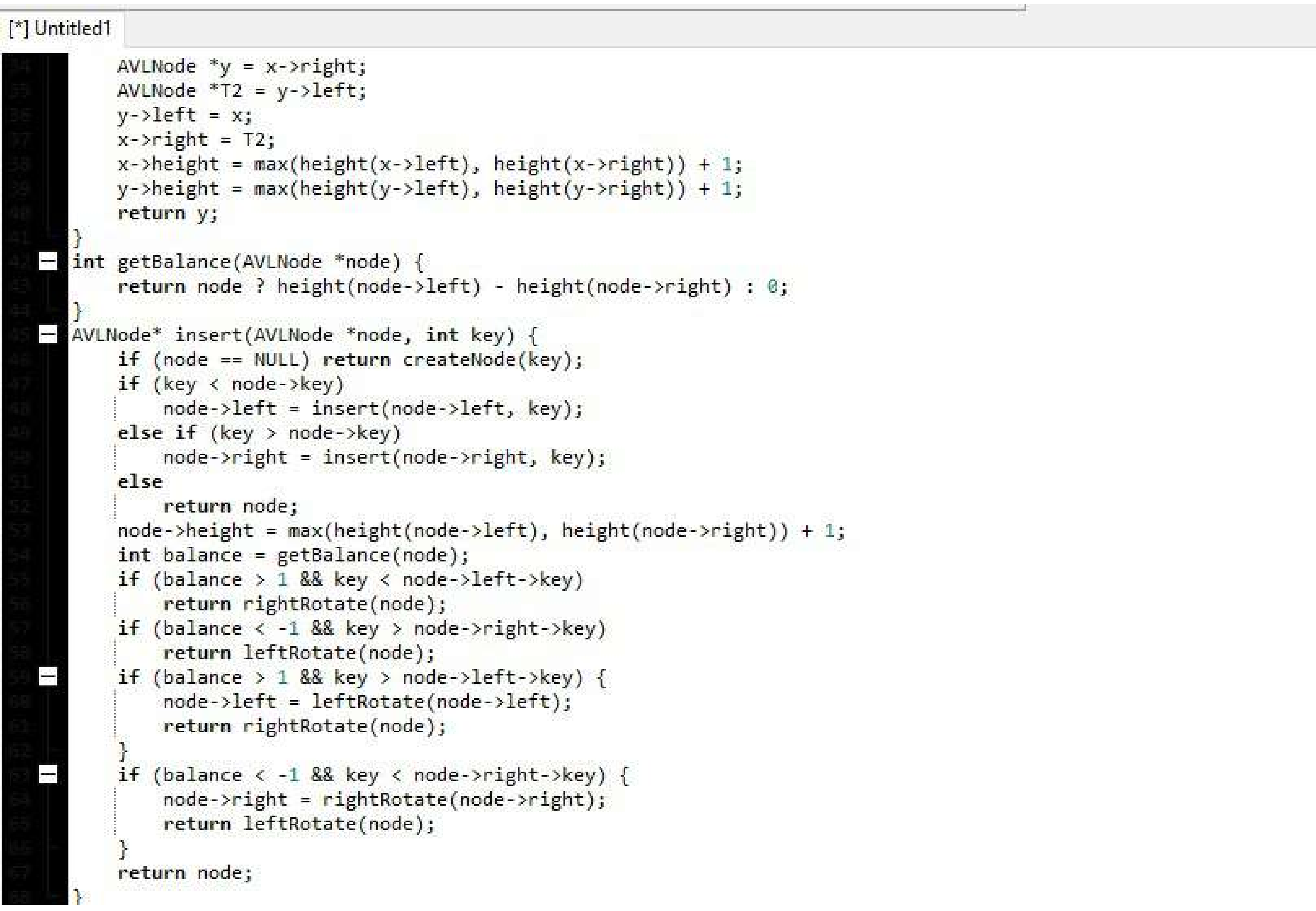
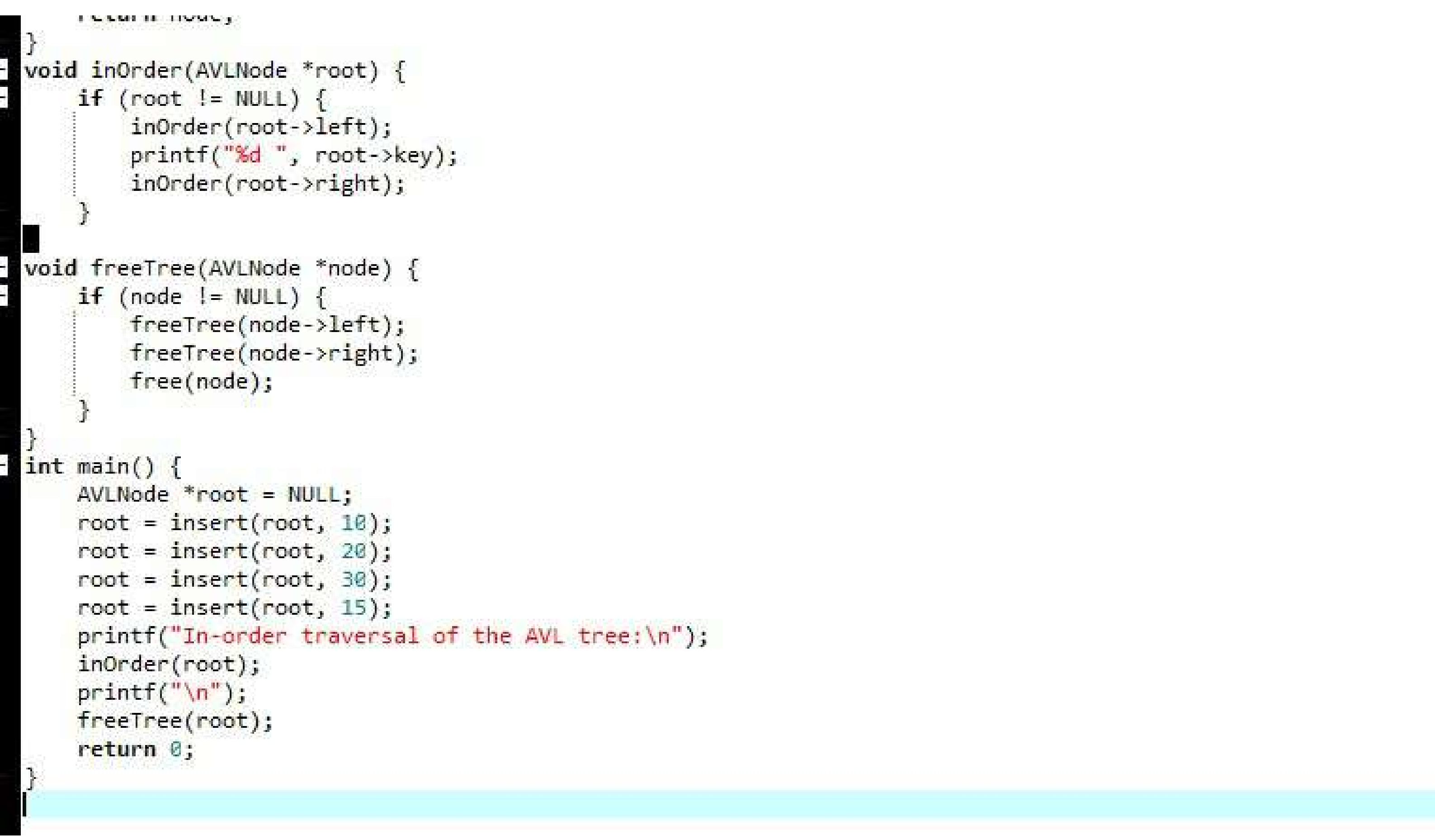


```c
AVL TREE
#include <stdio.h>
#include <stdlib.h>
typedef struct AVLNode {
    int key;
    struct AVLNode *left;
    struct AVLNode *right;
    int height;
} AVLNode;
int height(AVLNode *node) {
    return node ? node->height : 0;
}
int max(int a, int b) {
    return (a > b) ? a : b;
}
AVLNode* createNode(int key) {
    AVLNode *node = (AVLNode*)malloc(sizeof(AVLNode));
    node->key = key;
    node->left = NULL;
    node->right = NULL;
    node->height = 1;
    return node;
}
AVLNode* rightRotate(AVLNode *y) {
    AVLNode *x = y->left;
    AVLNode *T2 = x->right;
    x->right = y;
    y->left = T2;
    y->height = max(height(y->left), height(y->right)) + 1;
    x->height = max(height(x->left), height(x->right)) + 1;
    return x;
}
AVLNode* leftRotate(AVLNode *x) {
    AVLNode *y = x->right;
    AVLNode *T2 = y->left;
    y->left = x;
```



```
    AVLNode *y = x->right;
    AVLNode *T2 = y->left;
    y->left = x;
    x->right = T2;
    x->height = max(height(x->left), height(x->right)) + 1;
    y->height = max(height(y->left), height(y->right)) + 1;
    return y;
}
int getBalance(AVLNode *node) {
    return node ? height(node->left) - height(node->right) : 0;
}
AVLNode* insert(AVLNode *node, int key) {
    if (node == NULL) return createNode(key);
    if (key < node->key)
        node->left = insert(node->left, key);
    else if (key > node->key)
        node->right = insert(node->right, key);
    else
        return node;
    node->height = max(height(node->left), height(node->right)) + 1;
    int balance = getBalance(node);
    if (balance > 1 && key < node->left->key)
        return rightRotate(node);
    if (balance < -1 && key > node->right->key)
        return leftRotate(node);
    if (balance > 1 && key > node->left->key) {
        node->left = leftRotate(node->left);
        return rightRotate(node);
    }
    if (balance < -1 && key < node->right->key) {
        node->right = rightRotate(node->right);
        return leftRotate(node);
    }
    return node;
}
```



```c
    return node;
}
void inOrder(AVLNode *root) {
    if (root != NULL) {
        inOrder(root->left);
        printf("%d ", root->key);
        inOrder(root->right);
    }

void freeTree(AVLNode *node) {
    if (node != NULL) {
        freeTree(node->left);
        freeTree(node->right);
        free(node);
    }
}
int main() {
    AVLNode *root = NULL;
    root = insert(root, 10);
    root = insert(root, 20);
    root = insert(root, 30);
    root = insert(root, 15);
    printf("In-order traversal of the AVL tree:\n");
    inOrder(root);
    printf("\n");
    freeTree(root);
    return 0;
}
```