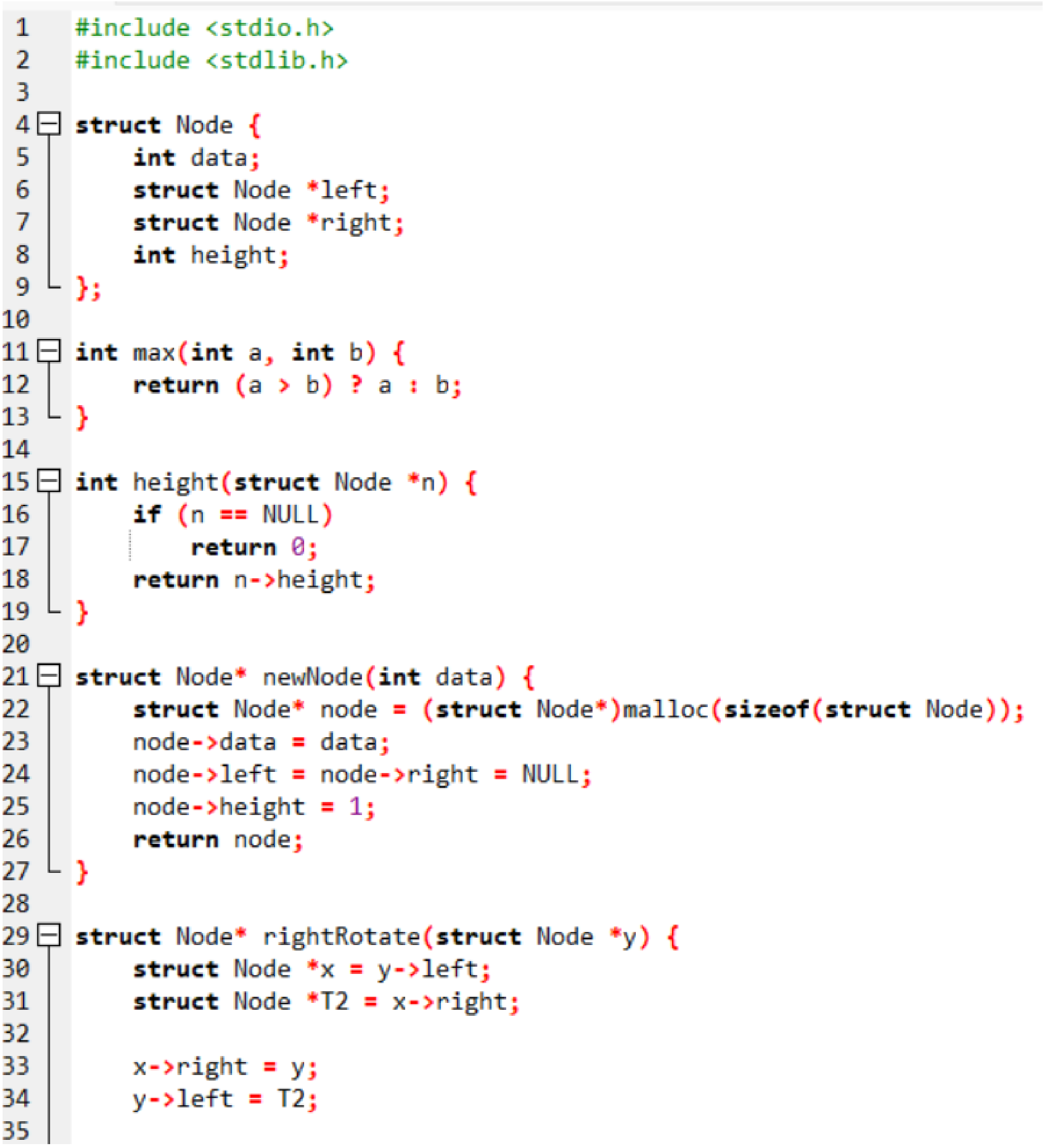
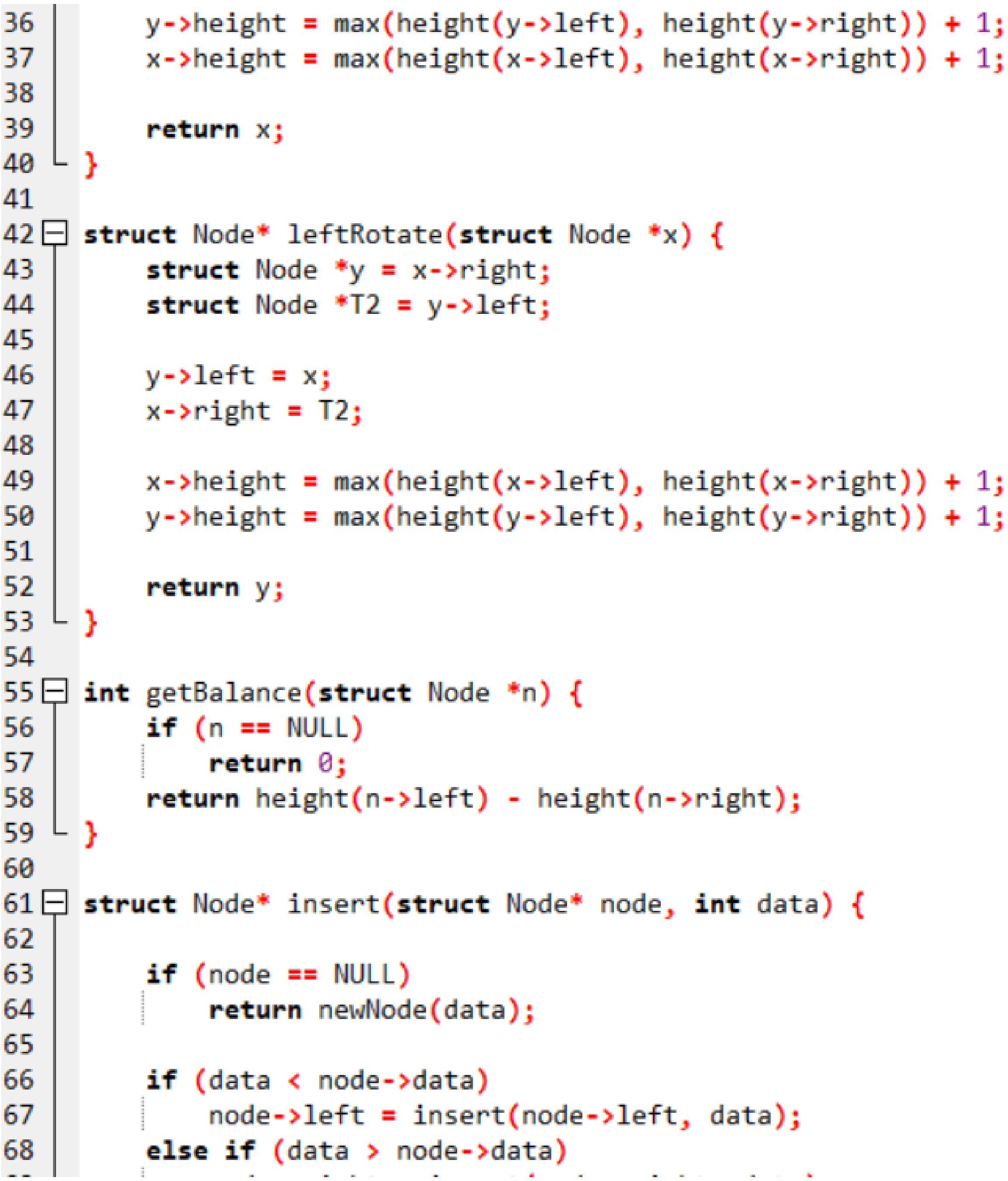
NAME:M.SHESHANK
ROLL NO: CH.SC.U4CSE24125

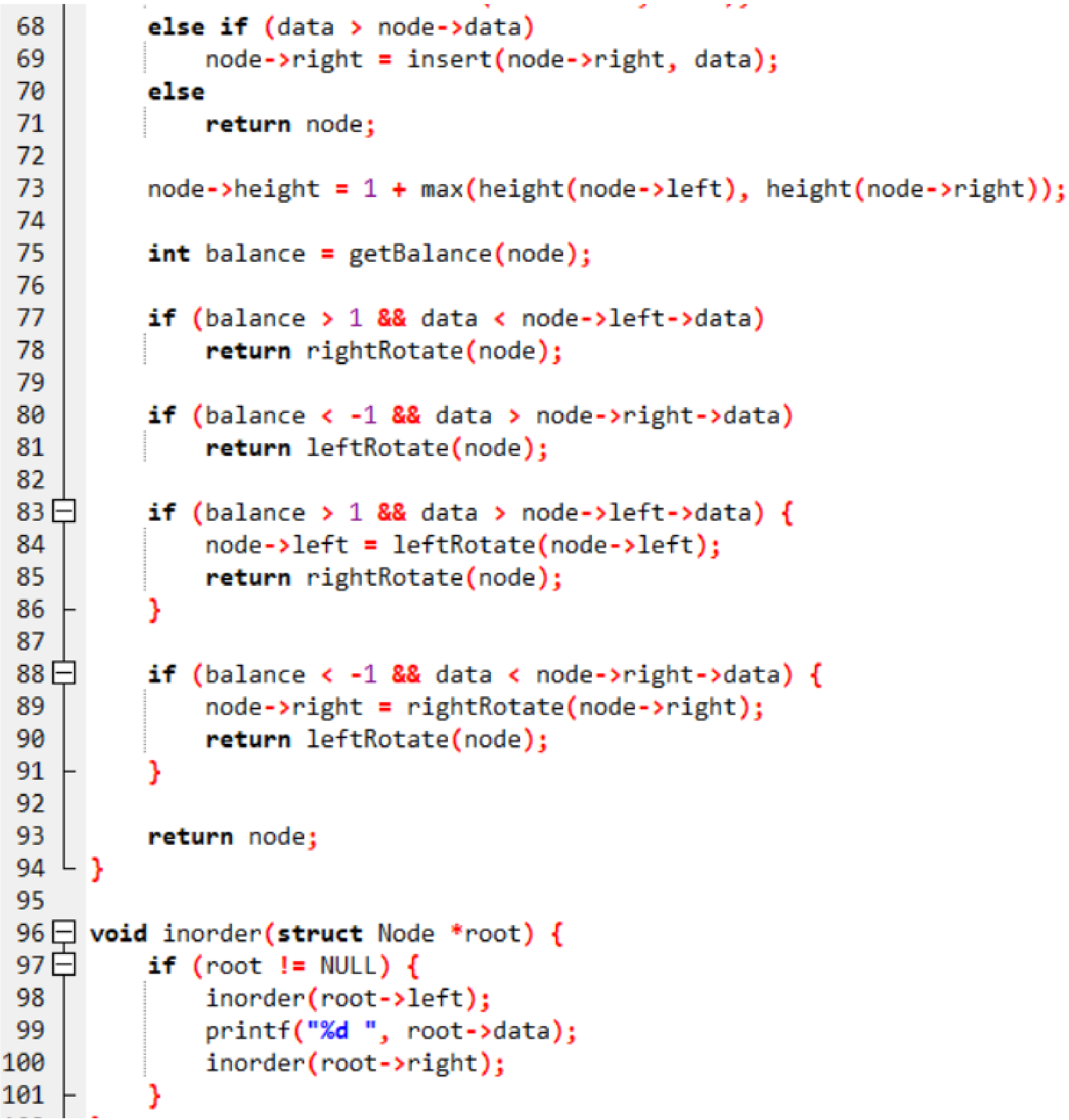
AVL TREE:
CODE:



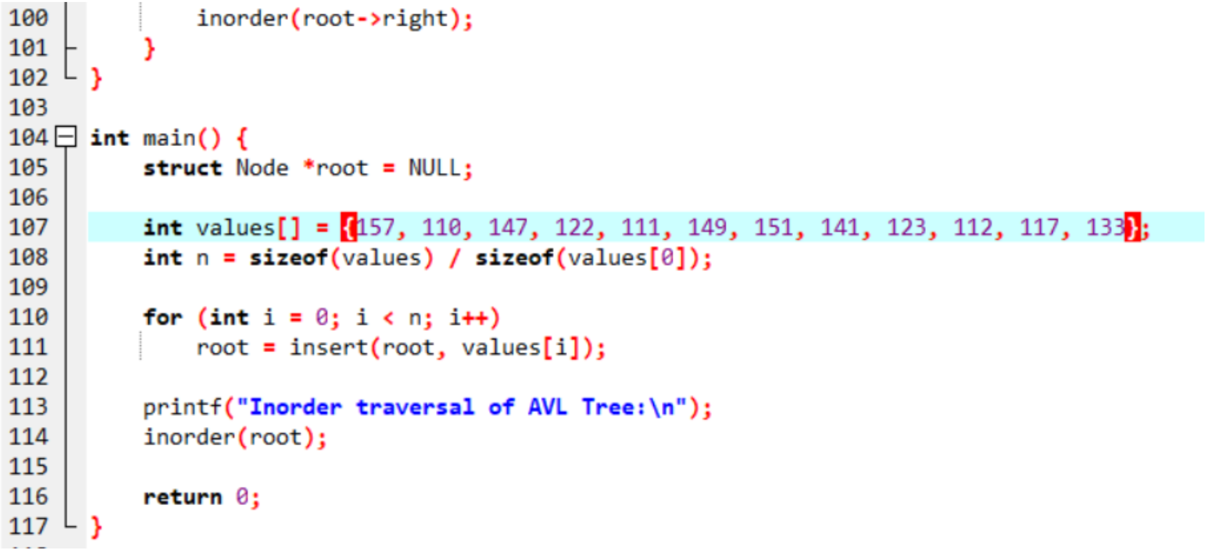
```c
#include <stdio.h>
#include <stdlib.h>

struct Node {
    int data;
    struct Node *left;
    struct Node *right;
    int height;
};

int max(int a, int b) {
    return (a > b) ? a : b;
}

int height(struct Node *n) {
    if (n == NULL)
        return 0;
    return n->height;
}

struct Node* newNode(int data) {
    struct Node* node = (struct Node*)malloc(sizeof(struct Node));
    node->data = data;
    node->left = node->right = NULL;
    node->height = 1;
    return node;
}

struct Node* rightRotate(struct Node *y) {
    struct Node *x = y->left;
    struct Node *T2 = x->right;

    x->right = y;
    y->left = T2;

```



```c
    y->height = max(height(y->left), height(y->right)) + 1;
    x->height = max(height(x->left), height(x->right)) + 1;

    return x;
}

struct Node* leftRotate(struct Node *x) {
    struct Node *y = x->right;
    struct Node *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(struct Node *n) {
    if (n == NULL)
        return 0;
    return height(n->left) - height(n->right);
}

struct Node* insert(struct Node* node, int data) {

    if (node == NULL)
        return newNode(data);

    if (data < node->data)
        node->left = insert(node->left, data);
    else if (data > node->data)
```



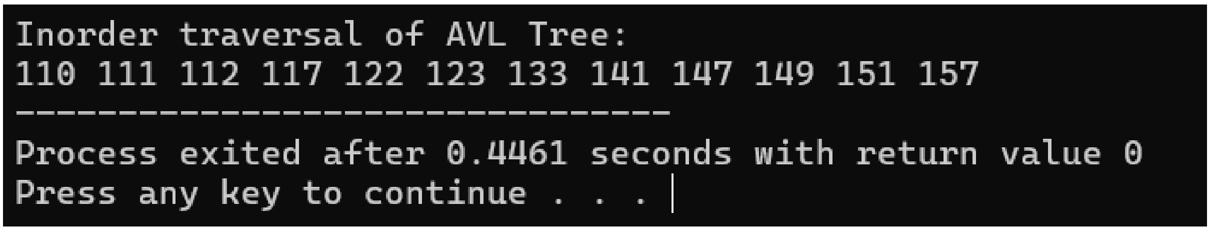
```c
    else if (data > node->data)
        node->right = insert(node->right, data);
    else
        return node;

    node->height = 1 + max(height(node->left), height(node->right));

    int balance = getBalance(node);

    if (balance > 1 && data < node->left->data)
        return rightRotate(node);

    if (balance < -1 && data > node->right->data)
        return leftRotate(node);

    if (balance > 1 && data > node->left->data) {
        node->left = leftRotate(node->left);
        return rightRotate(node);
    }

    if (balance < -1 && data < node->right->data) {
        node->right = rightRotate(node->right);
        return leftRotate(node);
    }

    return node;
}

void inorder(struct Node *root) {
    if (root != NULL) {
        inorder(root->left);
        printf("%d ", root->data);
        inorder(root->right);
    }
```



```c
        inorder(root->right);
    }
```

```c
}

int main() {
    struct Node *root = NULL;

    int values[] = {157, 110, 147, 122, 111, 149, 151, 141, 123, 112, 117, 133};
    int n = sizeof(values) / sizeof(values[0]);

    for (int i = 0; i < n; i++)
        root = insert(root, values[i]);

    printf("Inorder traversal of AVL Tree:\n");
    inorder(root);

    return 0;
}
```

OUTPUT:



```
Inorder traversal of AVL Tree:
110 111 112 117 122 123 133 141 147 149 151 157
--------------------------------
Process exited after 0.4461 seconds with return value 0
Press any key to continue . . .
```