Data Structures

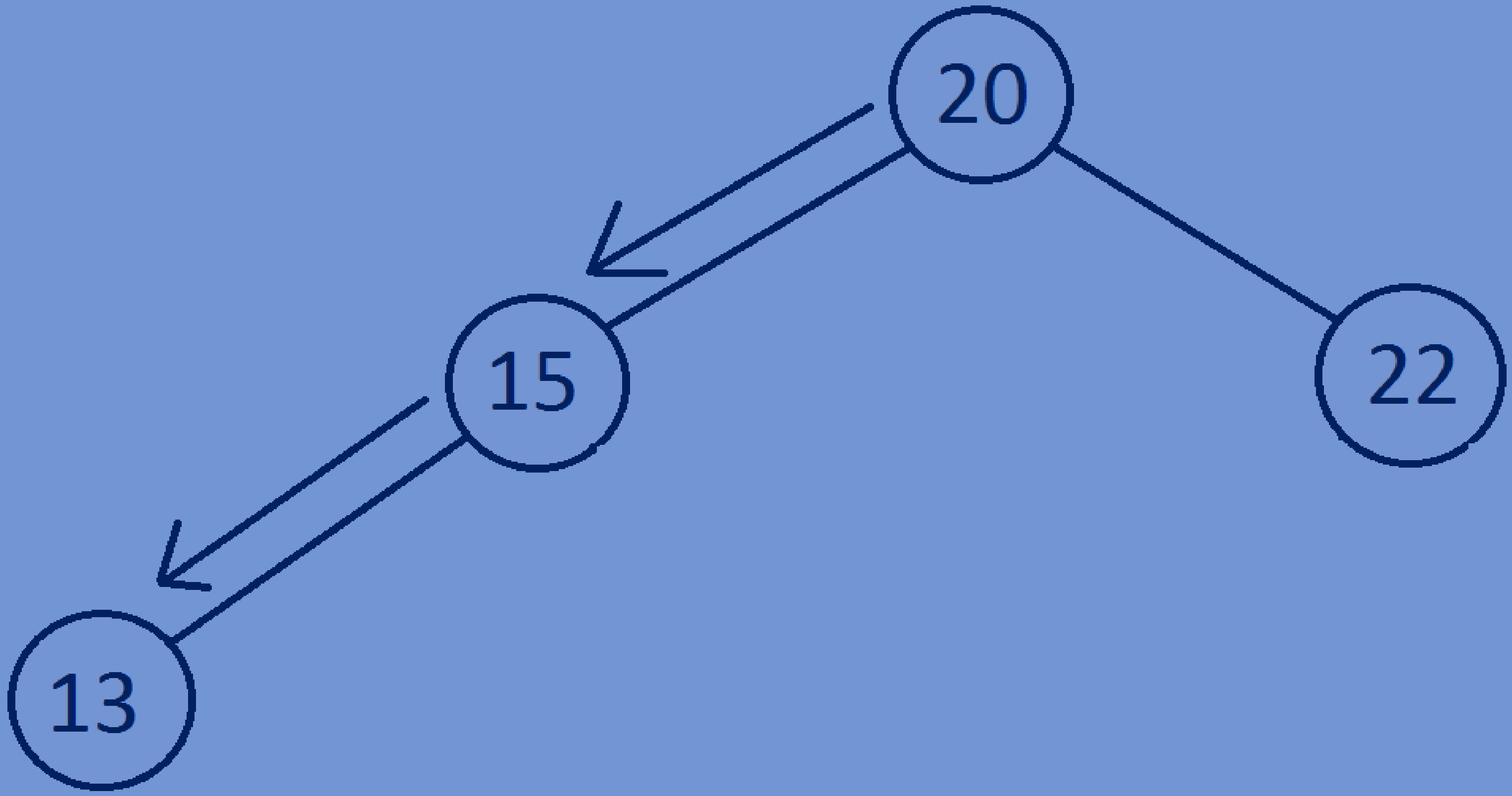
Finding height of a given Binary Tree.

Height of a binary Tree:

→ height of a node is equal to the total no of edges in the longest path from node to a leaf(furthest leaf) ,height of a tree is equal to height of the root node.

→ Height of a tree is equal to the total no of edges in the longest path from Root node to the leaf node(furthest leaf).

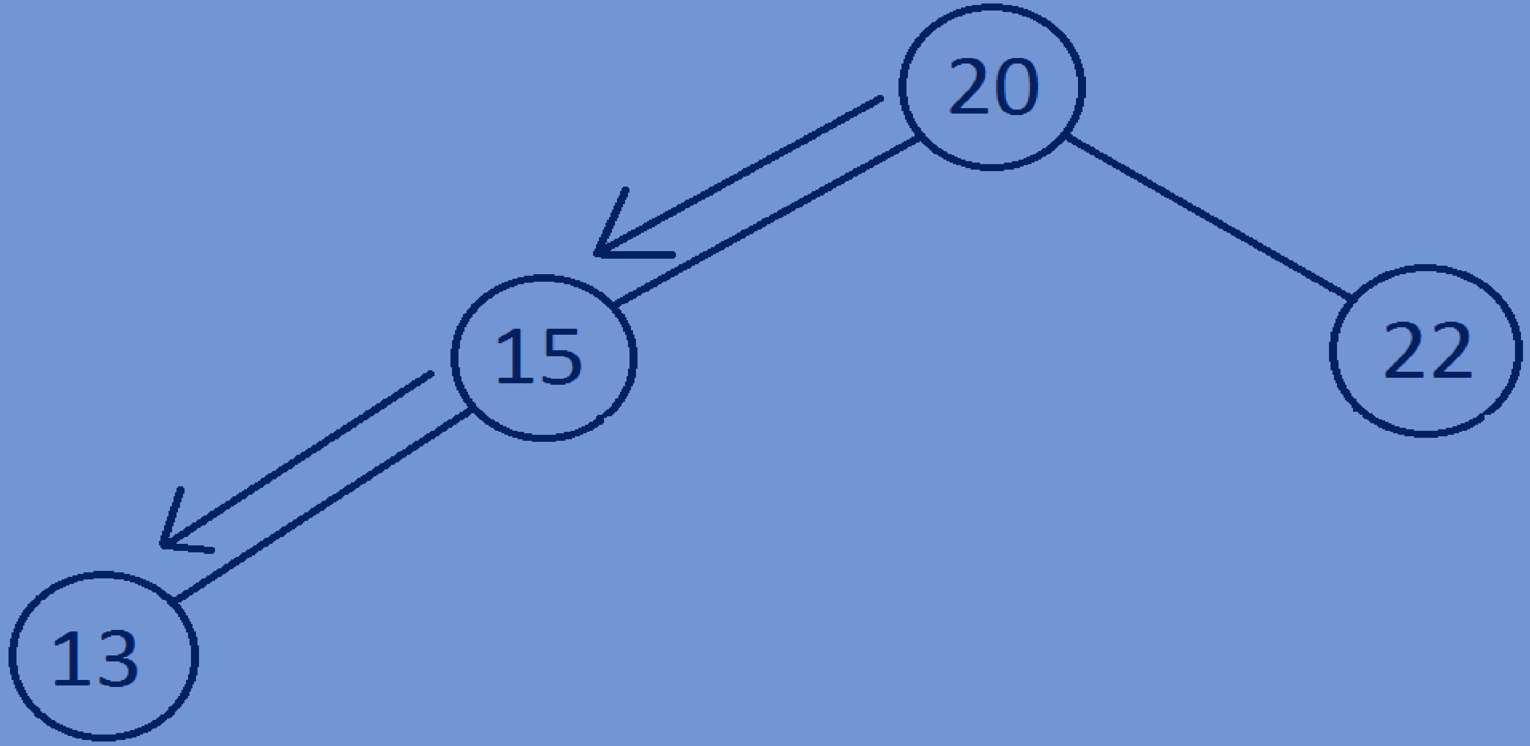
→ Height of a leaf node is equal to zero and height of empty tree is equal to -1.



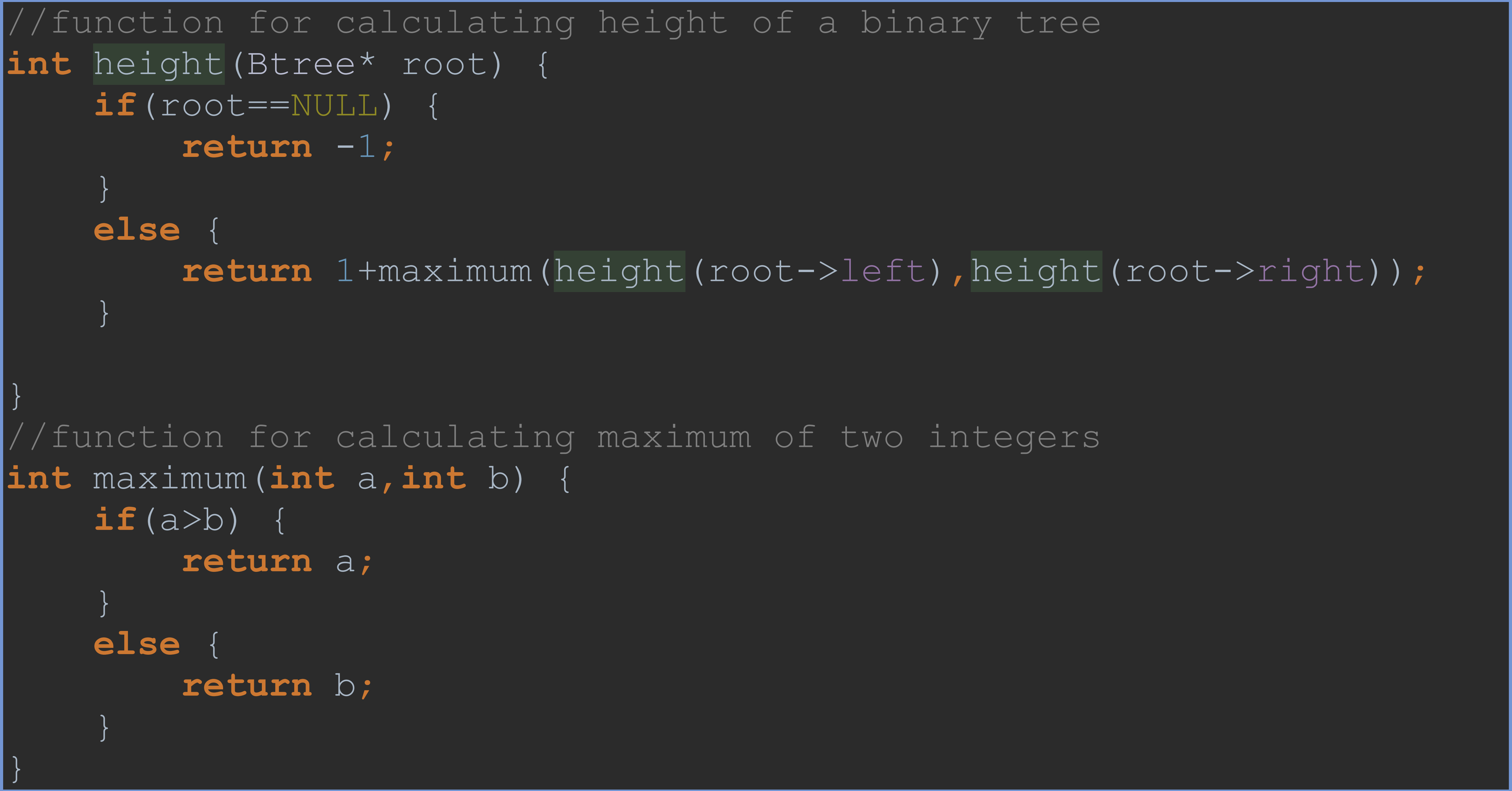
Finding Height of a binary tree:

- If root is equal to NULL (if tree is empty) then return -1.
- Calculate the height of left subtree and right subtree of a node and the Height of the node will be equal to the maximum of height of left subtree And height of right subtree +1.

1+maximum(height(root->left),height(root->right))



Function for calculating the height of a Binary Tree:

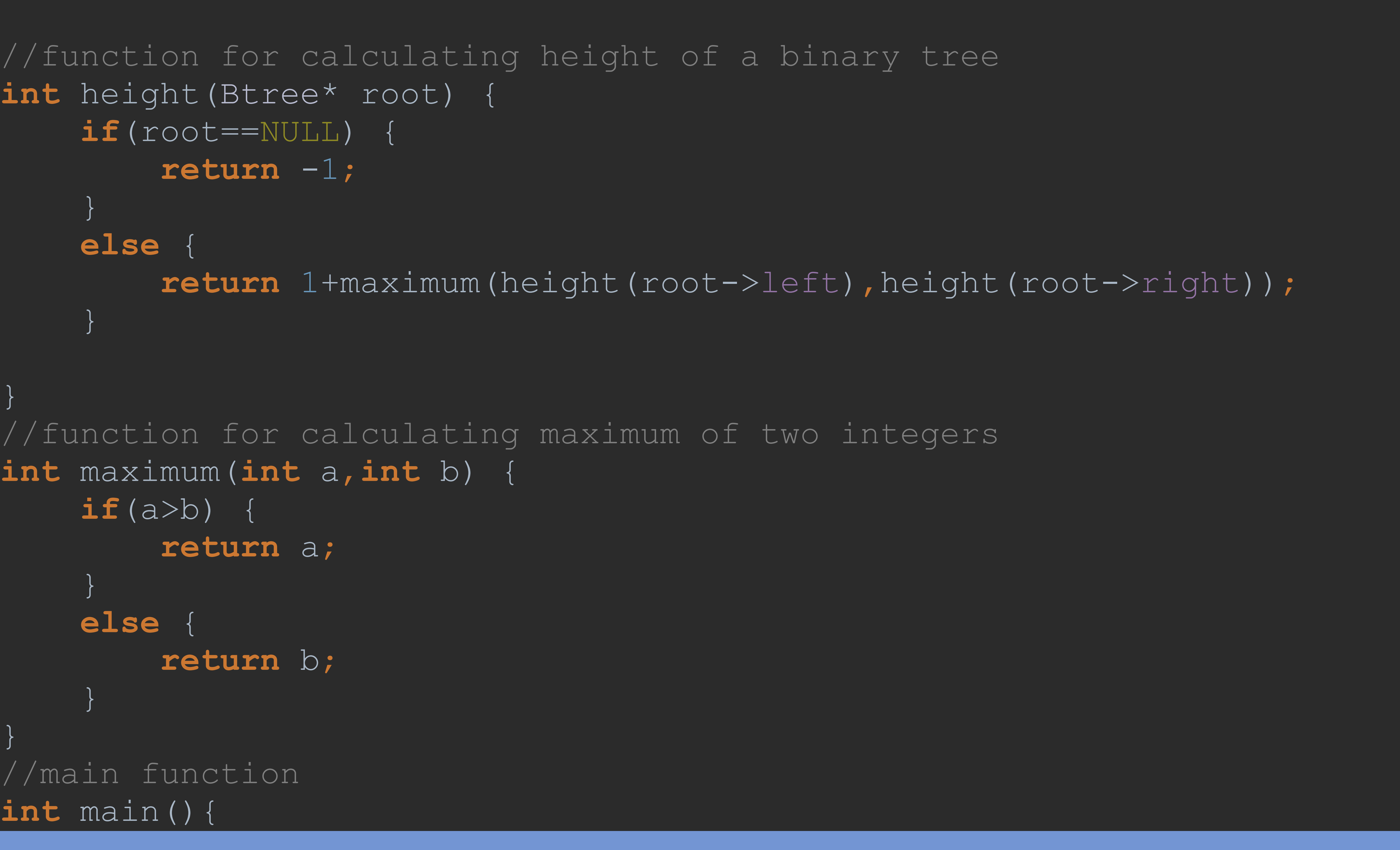


```
//function for calculating height of a binary tree
int height(Btree* root) {
    if(root==NULL) {
        return -1;
    }
    else {
        return 1+maximum(height(root->left),height(root->right));
    }

}
//function for calculating maximum of two integers
int maximum(int a,int b) {
    if(a>b) {
        return a;
    }
    else {
        return b;
    }
}
```

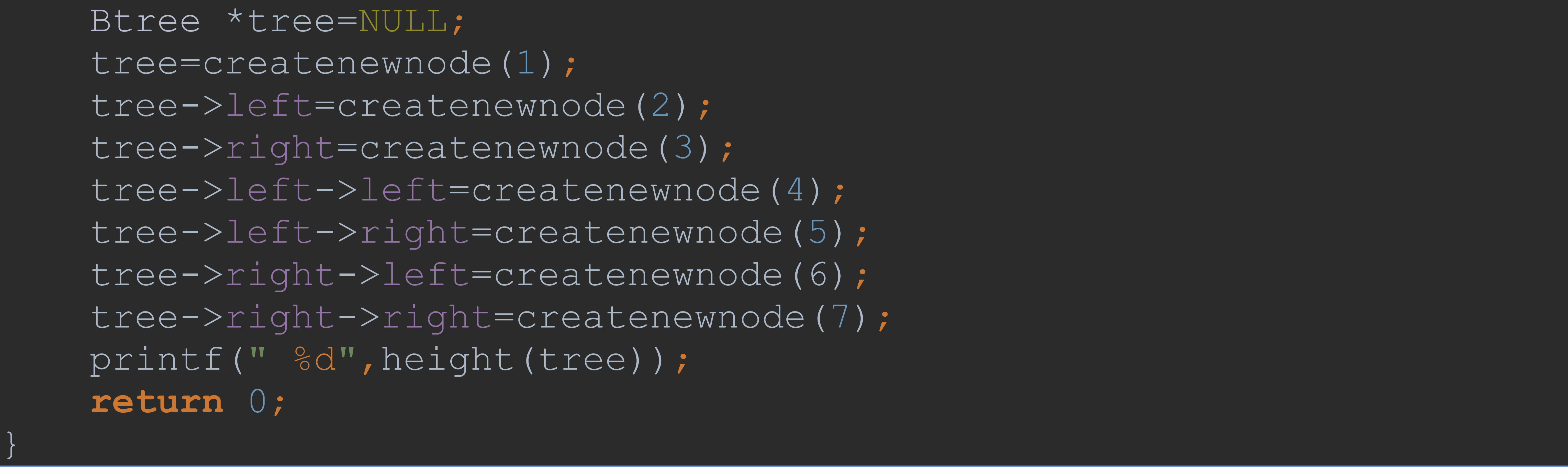
Whole program:

```c
#include<stdio.h>
#include<stdlib.h>
//creating a node
typedef  struct Btree{
    int data;
    struct Btree *left;
    struct Btree *right;
}Btree;
//creating new nodes
Btree *createnewnode(int data){
    Btree *newnode=(Btree*)malloc(sizeof(Btree));
    newnode->data=data;
    newnode->left=NULL;
    newnode->right=NULL;
    return newnode;
}
```



```
//function for calculating height of a binary tree
int height(Btree* root) {
    if(root==NULL) {
        return -1;
    }
    else {
        return 1+maximum(height(root->left),height(root->right));
    }

}
//function for calculating maximum of two integers
int maximum(int a,int b) {
    if(a>b) {
        return a;
    }
    else {
        return b;
    }
}
//main function
int main(){
```



```
    Btree *tree=NULL;
    tree=createnewnode(1);
    tree->left=createnewnode(2);
    tree->right=createnewnode(3);
    tree->left->left=createnewnode(4);
    tree->left->right=createnewnode(5);
    tree->right->left=createnewnode(6);
    tree->right->right=createnewnode(7);
    printf(" %d",height(tree));
    return 0;
}
```

Output:

```
2
```