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

/* A binary tree node has data, pointer to left child
and a pointer to right child */
struct node {
    int data;
    struct node* left;
    struct node* right;
};

/* Compute the "maxDepth" of a tree -- the number of
    nodes along the longest path from the root node
    down to the farthest leaf node.*/
int maxDepth(struct node* node)
{
    if (node == NULL)
        return -1;
    else {
        /* compute the depth of each subtree */
        int lDepth = maxDepth(node->left);
        int rDepth = maxDepth(node->right);

        /* use the larger one */
        if (lDepth > rDepth)
            return (lDepth + 1);
        else
            return (rDepth + 1);
    }
}

/* Helper function that allocates a new node with the
given data and NULL left and right pointers. */
struct node* newNode(int data)
{
    struct node* node
        = (struct node*)malloc(sizeof(struct node));
    node->data = data;
    node->left = NULL;
```

```
40:     node->right = NULL;
41:
42:     return (node);
43: }
44:
45: int main()
46: {
47:     struct node* root = newNode(1);
48:
49:     root->left = newNode(2);
50:     root->right = newNode(3);
51:     root->left->left = newNode(4);
52:     root->left->right = newNode(5);
53:
54:     printf("Height of tree is %d", maxDepth(root));
55:
56:     getchar();
57:     return 0;
58: }
59:
```