```cpp
#include<iostream>
using namespace std;

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

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

		// use the larger one
		if (lDepth > rDepth)
			return(lDepth + 1);
		else
			return(rDepth + 1);
		// OR just
		// return 1 + max(maxDepth(n->left), maxDepth(n->right));
	}
}

/* Helper function that allocates a new node with the
given data and NULL left and right pointers. */
 Node* newNode(int data)
{
	Node* n = new Node; // dynamically allocate new objects of type Node

	n->data = data;
	n->left = nullptr;
	n->right = nullptr;

	return(n);
}

int main()
{
	Node *root = newNode(1);

	root->left = newNode(2);
	root->right = newNode(3);
	root->left->left = newNode(4);
	root->left->right = newNode(5);

	cout << "The maximum depth is " << maxDepth(root);
	return 0;
}
```