#pragma once

#include <stdio.h>

#include <stdlib.h>

#include <limits.h>

class MirrorBST

{

public:

MirrorBST();

~MirrorBST();

//Node structure with two pointers as normal for a binary tree

struct node {

int data;

struct node\* left;

struct node\* right;

};

struct node\* createNewNode (int value){

//Allocate space needed for new node

struct node\* node = (struct node\*)

malloc(sizeof(struct node));

node->data = value;

node->left = NULL;

node->right = NULL;

return(node);

};

void swapPointers(node\* node) {

//Create a temp pointer, then swap pointers

struct node\* temp = node->left;

node->left = node->right;

node->right = temp;

//Recursively swap all nodes in the subtree

swapPointers(node->right);

swapPointers(node->left);

};

int main() {

struct node\* root = createNewNode(4);

//Manually assign left and right pointers

root->left = createNewNode(2);

root->left->left = createNewNode(1);

root->left->right = createNewNode(3);

root->right = createNewNode(5);

root->right->left = createNewNode(6);

root->right->right = createNewNode(7);

swapPointers(root);

};

};