**#include<iostream>**

**using namespace std;**

**class bstnode {**

**public:**

**int data;**

**bstnode \*left, \*right;**

**bstnode(int x) {**

**data = x;**

**left = right = NULL;**

**}**

**};**

**class bst {**

**bstnode \*root;**

**public:**

**bst() {**

**root = NULL;**

**}**

**void insert(int x);**

**bstnode\* find(int x);**

**bstnode\* find\_min(bstnode\* root);**

**int longest\_path(bstnode\* T);**

**void display(bstnode\* t);**

**void swapper(bstnode\* &t);**

**};**

**void bst::insert(int x) {**

**bstnode \*p, \*q, \*r;**

**r = new bstnode(x);**

**if (root == NULL) {**

**root = r;**

**return;**

**}**

**p = root;**

**while (p != NULL) {**

**q = p;**

**if (x > p->data) p = p->right;**

**else p = p->left;**

**}**

**if (x > q->data) q->right = r;**

**else q->left = r;**

**}**

**bstnode\* bst::find(int x) {**

**bstnode\* temp = root;**

**while (temp != NULL) {**

**if (x == temp->data) return temp;**

**if (x > temp->data) temp = temp->right;**

**else temp = temp->left;**

**}**

**return NULL;**

**}**

**bstnode\* bst::find\_min(bstnode\* root) {**

**while (root->left != NULL) {**

**root = root->left;**

**}**

**return root;**

**}**

**int bst::longest\_path(bstnode\* T) {**

**if (T == NULL || (T->left == NULL && T->right == NULL)) return 0;**

**int hl = longest\_path(T->left);**

**int hr = longest\_path(T->right);**

**return max(hl, hr) + 1;**

**}**

**void bst::display(bstnode\* t) {**

**if (t != NULL) {**

**display(t->left);**

**cout << "\t" << t->data;**

**display(t->right);**

**}**

**}**

**void bst::swapper(bstnode\* &t) {**

**if (t != NULL) {**

**swap(t->left, t->right);**

**swapper(t->left);**

**swapper(t->right);**

**}**

**}**

**int main() {**

**int ch, x;**

**bst b;**

**bstnode \*p, \*q, \*root = NULL;**

**do {**

**cout << "\n1. Create\n2. Find\n3. Find Min\n4. Longest Path\n5. Display\n6. Swap\nEnter your choice: ";**

**cin >> ch;**

**switch (ch) {**

**case 1:**

**root = NULL;**

**cout << "Enter total number of nodes: ";**

**int n;**

**cin >> n;**

**cout << "Enter tree values: ";**

**for (int i = 0; i < n; i++) {**

**cin >> x;**

**b.insert(x);**

**}**

**break;**

**case 2:**

**cout << "Enter node to be searched: ";**

**cin >> x;**

**p = b.find(x);**

**if (p == NULL) cout << "\nNode not found";**

**else cout << "Node found: " << p->data;**

**break;**

**case 3:**

**q = b.find\_min(root);**

**cout << "Minimum value in tree: " << q->data;**

**break;**

**case 4:**

**cout << "Longest path in tree: " << b.longest\_path(root) + 1;**

**break;**

**case 5:**

**b.display(root);**

**break;**

**case 6:**

**b.swapper(root);**

**break;**

**}**

**} while (ch != 7);**

**return 0;**

**}**