Jacob Ciurca

Dr. Lu

Lab 3

Cosc 320

9/22/2020

**PreLab:** reviewed notes on tree and binary tree as well as reviewed provided source code

**Lab:**

**CountOneChild.h:**

#ifndef COUNTONECHILD\_H\_INCLUDED

#define COUNTONECHILD\_H\_INCLUDED

#include <iostream>

#include <sstream>

#include <iomanip>

#include <string>

#include <queue>

#ifndef NULL

#include <cstddef>

#endif // NULL

#include "d\_tnode.h"

using namespace std;

tnode<char> \*buildTreeOne(){

tnode<char> \*root, \*b, \*c, \*d, \*e, \*f, \*g;

d = new tnode<char> ('D');

f = new tnode<char> ('F');

g = new tnode<char> ('F');

e = new tnode<char> ('E', (tnode<char> \*) NULL, g);

b = new tnode<char> ('B', d, (tnode<char> \*)NULL);

c = new tnode<char> ('C',e, f);

root = new tnode<char> ('A',b, c);

return root;

}

template <typename T>

void countOneChild(tnode<T> \*t, int& count){

if(t != NULL){

if (t->left == NULL && t->right != NULL){

count++;

}

if (t->left != NULL && t->right == NULL){

count++;

}

countOneChild(t->left, count); // descend left

countOneChild(t->right, count); // descend right

}

}

#endif // COUNTONECHILD\_H\_INCLUDED

**Lab03\_01.cpp:**

#include <iostream>

#include "d\_tnode.h"

#include "CountOneChild.h"

using namespace std;

int main()

{

int count=0;

tnode<char> \*t = buildTreeOne();

countOneChild(t, count);

cout << "There are " << count << " one child interior nodes in tree one.\n";

return 0;

}

**Max.h:**

#ifndef MAX\_H\_INCLUDED

#define MAX\_H\_INCLUDED

#include <iostream>

#include <sstream>

#include <iomanip>

#include <string>

#include <queue>

#ifndef NULL

#include <cstddef>

#endif // NULL

#include "d\_tnode.h"

tnode<int> \*buildIntTree(){

tnode<int> \*root, \*b, \*c, \*d, \*e, \*f;

d = new tnode<int> (5);

e = new tnode<int> (48);

f = new tnode<int> (12);

b = new tnode<int> (40, d, (tnode<int> \*) NULL);

c = new tnode<int> (15, e, f);

root = new tnode<int> (35, b, c);

return root;

}

template <typename T>

void FindMaxValue(tnode<T> \*t, int& max){

if(t != NULL){

if(t->nodeValue > max){

max = t->nodeValue;

}

FindMaxValue(t->left, max); // descend left

FindMaxValue(t->right, max); // descend right

}

}

#endif // MAX\_H\_INCLUDED

**Lab03\_02.cpp:**

#include <iostream>

#include "max.h"

#include "d\_tnode.h"

using namespace std;

int main()

{

int max=0;

tnode<int> \*t = buildIntTree();

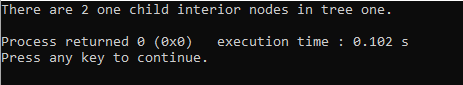
FindMaxValue(t, max);

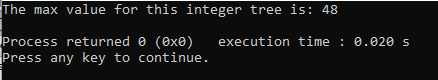
cout << "The max value for this integer tree is: " << max << endl;

return 0;

}

**Sample Outputs:**





**PostLab:** This lab was good at testing our knowledge on trees and binary trees as well as showing us all of the different things you can do with them. This lab took me about 1 hour to complete and I completed all of the work by myself with no help.