

1 Introduction

This lab is a series of exercises to get you back in the swing of C++ and Data Structures programming. You should have seen all (or nearly all) of the material in COSC 220.

2 Instructions

When you are finished submit all your work through the MyClasses page for this class. Create a directory called Lab03, put each programming exercise into its own subdirectory of this directory, zip the entire Lab03 directory up into the file Lab03.zip, and then submit this zip file to Lab #3.

Make sure that you:

- Follow the coding and documentation standards for the course as published in the MyClasses page for the class.
- Make sure that each project includes a makefile that will compile the program on the Linux HPCL lab machines.
- Check the contents of the zip file before uploading it. Make sure all the files are included.
- Make sure that the file was submitted correctly to MyCLasses.

3 Exercise

Take the IntBinaryTree class from the example code and add in the following functions to the class.

1. numNodes(): This function returns the total number of nodes in the tree.
2. numLeaves(): This function returns the total number of leaf nodes in the tree.
3. height(): This function returns the height of the tree.
4. PrintHeightPaths(): This function will print the node values of each path from the root to a leaf that is the same length as the height of the tree.

As is usual in tree functions, a non-recursive version usually calls a recursive version. If this is how you set your functions up, make the non-recursive version public and your recursive versions private or protected. If you run the following main,

```
#include <iostream>
#include <cstdlib>
#include <ctime>

#include "IntBinaryTree.h"

using namespace std;

int main() {
    srand(time(0));
    int sz = 0;

    cout << "Input Array Size: ";
    cin >> sz;

    int *A = new int[sz];
    IntBinaryTree B;
```

```
for (int i = 0; i < sz; i++)
    A[i] = rand() % 100 + 1;

for (int i = 0; i < sz; i++)
    cout << A[i] << " ";
cout << endl;

for (int i = 0; i < sz; i++)
    B.insertNode(A[i]);
cout << endl;

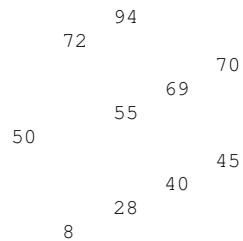
B.PrintTree();

cout << "Number of Nodes: " << B.numNodes() << endl;
cout << "Number of Leaves: " << B.numLeaves() << endl;
cout << "Height: " << B.height() << endl;
cout << "Height Paths" << endl;
B.PrintHeightPaths();

delete[] A;
return 0;
}
```

you should get output similar to the following.

```
Input Array Size: 10
50 72 8 28 55 40 45 69 94 70
```



```
Number of Nodes: 10
Number of Leaves: 3
Height: 5
Height Paths
50 8 28 40 45
50 72 55 69 70
```