Lab 11

Objectives

The purpose of this lab is to reinforce tree, and BST implementations and recursive thinking in C++.

Requirements

You need download the BinaryTree project from our course website. Create a new .cpp file. In that file implement the following methods:

(1). depth function

1. If root is NULL, return -1.
2. Otherwise, determine the maximum of the depths of the left and right subtrees, add 1 and return.

```
template <class T>
int depth(binary_tree_node<T> *root)
```

(2). max function

1. Assert that root is not NULL
2. Get the max in the left subtree, the max in the right subtree, and the data value in root. Ignore a subtree if it is empty.
3. Return the largest of those three values.

```
template <class T>
T max(binary_tree_node<T> *root)
```

Testing the above two functions. Here are the results from the first four sample trees:

```
size of s1: 3
depth of s1: 1
max of s1: 3
size of s2: 12
depth of s2: 3
max of s2: 12
size of s3: 57
depth of s3: 10
max of s3: 0.462529
size of s4: 67
```

```
depth of s4: 11
max of s4: 0.488894
```

(3). tree_sum function

Returns the sum of the values in all the nodes in the tree.

```
double tree_sum(binary_tree_node<double> *root)
```

(4). tree_average function

Returns the average of the values in all the nodes in the tree.

```
double tree_average(binary_tree_node<double> *root)
```

(5). tree_is_balanced function

Returns true if and only if the tree is balanced. This means that the absolute value of the difference in depth between the two subtrees of root is no more than 1. It also means that both the left and the right subtree are also balanced. An empty tree is considered to be balanced.

1. If root is empty, the tree is balanced.
2. Otherwise, if either the right or the left subtree is not balanced, then the tree is not balanced.
3. Otherwise, if the difference in depths between the left and right subtrees is greater than one then the tree is not balanced.
4. Otherwise, the tree is balanced.

```
template <class T>
bool tree_is_balanced(binary_tree_node<T> *root)
```

Testing the above three functions. Here are the results from the seven sample trees:

```
sum of s3 12.9191
average of s3 0.226651
size of s3 57
sum of b1 11018.4
average of b1 47.493
size of b1 232
sum of b2 29775
average of b2 48.8917
size of b2 609
s1 is balanced? 1
s2 is balanced? 1
s3 is balanced? 0
s4 is balanced? 0
s5 is balanced? 1
b1 is balanced? 1
b2 is balanced? 1
```