lab_avl Awful AVL Trees

Due: Sunday, November 1 at 11:59 PM
Doxygen for lab_avl

Assignment Description

In this lab we'll practice AVL tree rotations and insertions, and see some silly test cases.

Checking Out The Code

As usual, just run

```
svn up
```
TERMINAL

from your `cs225` directory. This will create a new folder in your working directory called `lab_avl`.

As usual, check out the Doxygen for lab_avl.

Implement Rotation Functions

You must implement `rotateLeft()`, `rotateRight()`, and `rotateRightLeft()`. We have implemented `rotateLeftRight()` for you as an example for implementing `rotateRightLeft()`.

Implement the `insert()` Function

You must implement the `insert()` function. `insert()` should add a node with a key and value at the correct location in the tree, then rotate the tree appropriately (while returning from each recursive function) to fix the tree's balance.

Testing Your Code

To test your code, compile using `make`:

```
make
```
TERMINAL

Then run it with:

```
./testavl color
```
TERMINAL

You will see that the output is colored — green means correct output, red means incorrect output, and underlined red means expected output that was not present. This mode is a bit experimental, and it might cause problems with your own debugging output (or other problems in general). To turn it off, simply leave off the “color” argument:

```
./testavl
```
TERMINAL

You may also `diff` your solution with our expected output:

```
./testavl > out
vimdiff out soln_testavl.out
```
TERMINAL

Type `[Escape] [:] [q] [a] [ENTER]` to exit `vimdiff`.

Committing Your Code

Commit your code the usual way:

```
svn ci -m "lab_avl submission"
```
TERMINAL

Grading Information:

The following files are used in grading:

- `avltree.cpp`
- `avltree.h`

All other files including any testing files you have added will not be used for grading.