About Lab 6

Start Lab 6 by

- Unzipping the starter code file into your Lab 6 directory, starting up Eclipse as usual, and adding the jsoup jar file to your build path. When you do this the errors in HTMLScanner.java file should go away.

- Writing the MyTreeMap<K, V> class. A lot of the code for this is given to you.

  Remember that AVL trees are also binary search trees so every node satisfies the binary search tree property.

  In this implementation an empty tree has both children null. A leaf has both children empty trees.

Clicker Question: What is the BST property?

A. The value stored at any node is greater than the values of its kids.
B. The value stored at any node is less than the value of its kids.
C. The value at any node is greater than its left child and less than its right child.
D. Every node except the leaves must have exactly two children.

You need to write a get(key) method -- the BST property should tell you how to search a Binary Search Tree.

Another clicker question:
Remember that an <u>empty</u> tree has both children null. A <u>leaf</u> has both children empty trees.

When writing get(key) how do you know if *key* is not in the tree?

A. If *key* is the value of the current node's key.
B. if the current node is empty.
C. If the current node is a leaf and *key* is not its key.
D. If get(key) returns null.

You also need a put(key, value) method that inserts into the search tree. This has two parts -- the usual recursive BST insert( ) and then rotations to re-balance the tree. We talked about the BST insert algorithm, and in our March 15 discussion of AVL trees we looked at code for it. For this implementation put(key, value) returns the previous value that was stored for the given key, or null if this key is new.

You should be able to start with an empty tree, then add data to it one step at a time.

- The tricky thing you need to implement is the restructure( ) method that does tree rotations to fix problems with inserts and deletes from the AVL tree. Look carefully at the notes from Wednesday, March 15 before you start.

- Implement MyTreeSet<T> on top of MyTreeMap<T, Boolean>. This should be easy. Since AVL trees only allow one node to have a given key, they are an easy way to represent sets of keys, as long as the keys can be compared (i.e., they implement the Comparable<T> interface).

- Implement the WebPageIndex class. This looks scary but is actually quite straightforward. The given HTMLScanner class, based on the given jsoup code, does most of the work. The only method here that is longer than 2 or 3 lines is the constructor. This uses the HTMLScanner to build a MyTreeMap of the words on a web page and a MyTreeSet of the links on the page.

We will make use of the WebPageIndex class in Lab 7 when we build a search engine for web pages.