CS 340 Programming Quiz 4
Due 11:59 PM Friday October 15
• Implement the inner class InorderIterator in the BinaryTree class. Your implementation must use the parent reference. You cannot use recursion or an additional data structure such as a stack or a linked list.
• The string representation for the tree is an in order representation. The constructor that is passed 4 strings expects the string representation of the tree and strings that represent open, close and empty. In the examples discussed in class open, close and empty were (, ( and ! respectively. Tokens in the string representation of the tree are separated by white space.
String Representation of the Tree shown Below

$$(((D)B)A(((F(H))C(G))))$$
import java.util.*;
import java.io.*;

public class BinaryTree {
    // Implements a Binary Tree of Strings
    private class Node {
        private Node left;
        private String data;
        private Node right;
        private Node parent; // reference to the parent node; parent is null for the root node

        private Node(Node L, String d, Node r, Node p) {
            left = L;
            data = d;
            right = r;
            parent = p;
        }
    }

    private Node root;

    public BinaryTree() {
        /create an empty tree
        root = null;
    }
}
```java
public BinaryTree(String d) {
    //create a tree with a single node
    root = new Node(null, d, null, null);
}

public BinaryTree(String t, String open, String close, String empty) {
    //create a binary tree from the in order format discussed in class
    //t is a syntactically correct string representation of the tree
    //open and close are the strings which represent the beginning and end markers of a tree
    //empty represents an empty tree
    //the example in class used ( ) and ! for open, close and empty respectively
    //the data in the tree will not include strings matching open, close or empty
    //most of the work should be done in a recursive private method
    Scanner s = new Scanner(t);
    root = (buildTree(s, open, close, empty)).root;
}
```
private BinaryTree buildTree(Scanner s, String open, String close, String empty) {
    String nxt = s.next();
    if (nxt.equals(empty)) return new BinaryTree();
    // else nxt should match open
    BinaryTree b1 = buildTree(s, open, close, empty); // the left child
    String d = s.next(); // the data
    BinaryTree b2 = buildTree(s, open, close, empty); // the right child
    s.next(); // consume the close string
    return new BinaryTree(b1, d, b2);
}

public BinaryTree(BinaryTree b1, String d, BinaryTree b2) {
    // merge the trees b1 AND b2 with a common root with data d
    root = new Node(b1.root, d, b2.root, null);
    if (b1.root != null) b1.root.parent = root;
    if (b2.root != null) b2.root.parent = root;
}
public class InorderIterator implements Iterator<String> {
// An iterator that returns data in the tree in an in order pattern
// this implementation must use the parent pointer and must not use an additional data structure
// You and add instance methods and variables

    public InorderIterator() {
    }

    public boolean hasNext() {
    }

    public String next() {
    }

    public void remove() {
        // optional method not implemented
    }
}
public Iterator<String> inorder() {
    //return a new iterator object
    return new InorderIterator();
}

public static void main(String args[]) {
    BinaryTree b = new BinaryTree(args[0], "(" , "" , "!");
    Iterator<String> itr = b.inorder();
    while (itr.hasNext()) {
        System.out.print(itr.next() + " ");
    }
    System.out.println();
}
}
Programming Quiz 4 Submission

• Upload one zip file called pq4.zip to Canvas. The zip file must contain **only one file called BinaryTree.java**. Do not upload your whole Eclipse project!

• I gave you a simple main to test your program but I will use a different test driver