```java
/****
 *
 * TreeNodeList extends TreeNode by adding a node component and a siblings
 * component.  The node is a reference to a single TreeNode.  The siblings
 * component is a reference to a TreeNodeList, which may contain zero or more
 * additional TreeNodes.  Hence, TreeNodeList is used to represent list
 * constructs in a parse tree.  It can equivalently be viewed as a way to
 * represent n-ary constructs.
 *
 */
public class TreeNodeList extends TreeNode {

    /**
     * Construct this with the given id and child TreeNode references.
     */
    public TreeNodeList(TreeNode node, TreeNodeList siblings) {
        this.node = node;
        this.siblings = siblings;
    }

    /**
     * Return the String representation of this subtree, which is the recursive
     * toString of each of its nodes, separated by a ';' on a new line plus
     * another blank line.  See the documentation for <a href=
     * "TreeNode.html#toString()"> TreeNode.toString() </a> for a general
     * description the way trees are represented as strings.
     */
    public String toString(int level) {
        String indent = "";
        for (int i = 0; i < level; i++) {
            indent += "  ";
        }
        if (siblings == null) {
            return node == null ? " " : node.toString(level);
        }
        else {
            return (node == null ? "" : node.toString(level)) + "\n" +
                indent + "  ;\n" + indent + siblings.toString(level);
        }
    }

    /** Reference to the first node of this (sub)list. */
    public TreeNode node;

    /** Reference to the rest of this (sub)list */
    public TreeNodeList siblings;

}
```