

```

1  /****
2  *
3  * TreeNode4 extends TreeNode by adding four child components, which are
4  * references other TreeNodes. Hence, TreeNode4 is used to represent
5  * quaternary syntactic constructs in a parse tree.
6  *
7  */
8  public class TreeNode4 extends TreeNode {
9
10     /**
11     * Construct this with the given id and child TreeNode references.
12     */
13     public TreeNode4(int id, TreeNode child1, TreeNode child2,
14                     TreeNode child3, TreeNode child4) {
15         super(id);
16         this.child1 = child1;
17         this.child2 = child2;
18         this.child3 = child3;
19         this.child4 = child4;
20     }
21
22     /**
23     * A la the other constructor, but with line and column numbers.
24     */
25     public TreeNode4(int id, TreeNode child1, TreeNode child2,
26                     TreeNode child3, TreeNode child4, int line, int column) {
27         super(id, line, column);
28         this.child1 = child1;
29         this.child2 = child2;
30         this.child3 = child3;
31         this.child4 = child4;
32     }
33
34     /**
35     * Return the String representation of this subtree, which is the String
36     * value of its ID, followed on the next four indented lines by the
37     * recursive toString of its four children. See the documentation for <a
38     * href= "TreeNode.html#toString()"> TreeNode.toString() </a> for a general
39     * description the way trees are represented as strings.
40     */
41     public String toString(int level) {
42         String indent = "";
43         for (int i = 0; i < level; i++) {
44             indent += " ";
45         }
46         return symPrint(id) + toStringLineAndColumn(" ") + "\n" +
47             indent + " " + (child1 == null ? "null" : child1.toString(level+1)) + "\n" +
48             indent + " " + (child2 == null ? "null" : child2.toString(level+1)) + "\n" +
49             indent + " " + (child3 == null ? "null" : child3.toString(level+1)) + "\n" +
50             indent + " " + (child4 == null ? "null" : child4.toString(level+1));
51     }
52
53     /** Reference to the left child of this node. */
54     public TreeNode child1;
55
56     /** Reference to the first middle (or second, or third-from-the-last) child
57
58     * of this node. */
59     public TreeNode child2;
60
61     /** Reference to the second middle (or third, or next-to-the-last) child of
62     * this node. */
63     public TreeNode child3;
64
65     /** Reference to the right (or fourth, or last, or rightmost, or
66     * whatever-the-heck-you-want-to-call-it) child of this node. */
67     public TreeNode child4;
68
69 }

```