```java
import java.io.PrintStream;
import java.util.Scanner;

public class QuestionTree {

    private QuestionNode root;
    private Scanner userIn;
    private PrintStream userOut;
    private int totalGames;
    private int gamesWon;

    public QuestionTree Scanner input, PrintStream output) {
        if (input == null || output==null) {
            throw new IllegalArgumentException("Error");
        }
        root = new QuestionNode("Student");
        userIn=input;
        userOut=output;
        gamesWon=0;
        totalGames=0;
    }

    public void play() {
        root = playHelper(root);
        totalGames++;
    }

    private QuestionNode playHelper(QuestionNode node) {
        //is a leaf node (Ans)
        if(node.left ==null && node.right==null) {
            userOut.println("Were you guessing " + node.data);
            //check for user response
            if(UserInterface.nextAnswer(userIn)) {
                userOut.println("Computer wins! Better luck next time!");
                gamesWon++;
            }else {
                //create a new question answer node
                node = newQA(node);
            }
        }else { // the node is a question so we'll ask it
            userOut.println(node.data);
            //if yes go left
            if(UserInterface.nextAnswer(userIn)) {
                node.left= playHelper(node.left);
            }else {
                node.right=playHelper(node.right);
            }
        }
        return node;
    }

    public void load Scanner in) {
        while(in.hasNextLine()) {
            root = loadHelper(in,root);
        }
    }

    private QuestionNode loadHelper(Scanner in, QuestionNode node) {
        if (in == null || root==null) {
```

```
60              throw new IllegalArgumentException("Error");
61          }
62          String next = in.nextLine();
63          String [] parse = next.split(":");
64          node = new QuestionNode(parse[1]);
65          if(next.contains("Q:")){
66              node.left = loadHelper(in, node.left);
67              node.right = loadHelper(in, node.right);
68          }
69          return node;
70      }
71
72      public void save PrintStream out) {
73          save(root, out);
74      }
75
76      private void save QuestionNode node, PrintStream output) {
77          if (root == null || output==null) {
78              throw new IllegalArgumentException("Error");
79          }
80          if(node.left!= null && node.right!=null) {
81              // save with a Q: in front then call recursively
82              output.println("Q:"+node.data);
83              save(node.left, output);
84              save(node.right,output);
85          }else {
86              output.println("A:"+node.data);
87          }
88
89      }
90
91      public int totalGames() {
92           return totalGames;
93      }
94
95      public int gamesWon() {
96           return gamesWon;
97      }
98
99      //helper methods
100     private QuestionNode newQA(QuestionNode node) {
101         if (root==null) {
102             throw new IllegalArgumentException("Error");
103         }
104         userOut.println("Computer Loses! What was your object?");
105         QuestionNode ans = new QuestionNode(userIn.nextLine());
106         userOut.println "Write a yes/no question to distinguish your item \n from " +
    node.data);
107         QuestionNode question = new QuestionNode(userIn.nextLine());
108         userOut.println("What is the answer to your question (Yes/No)?");
109         if(UserInterface.nextAnswer(userIn)) {
110             question.left = ans;
111             question.right = node;
112         }else {
113             question.right = ans;
114             question.left = node;
115         }
116         return question;
117     }
```

```
118 }
119
```