```java
package edu.caltech.cs2.project02.choosers;

import edu.caltech.cs2.project02.interfaces.IHangmanChooser;

import java.io.File;
import java.io.FileNotFoundException;
import java.util.*;

public class RandomHangmanChooser implements IHangmanChooser {

    private static final Random rand = new Random();

    private final String secretWord;
    private int maxGuesses;
    private SortedSet<Character> guesses;

    public RandomHangmanChooser(int wordLength, int maxGuesses) throws FileNotFoundException {
        this.maxGuesses = maxGuesses;
        if (wordLength < 1 || maxGuesses < 1) {
            throw new IllegalArgumentException();
        }
        Scanner scanner = new Scanner(new File("data/scrabble.txt"));

        SortedSet<String> set = new TreeSet<>();
        guesses = new TreeSet<>();

        while (scanner.hasNext()) {
            String word = scanner.nextLine();

            if (word.length() == wordLength) {
                set.add(scanner.nextLine());
            }
        }

        if (set.isEmpty()) {
            throw new IllegalStateException();
        }
```

```
38
39          int which = rand.nextInt(set.size());
40          secretWord = (String)set.toArray()[which];
41      }
42
43      @Override
44      public int makeGuess(char letter) {
45          if (guesses.size() < 1) {
46              throw new IllegalArgumentException();
47          }
48          if (guesses.contains(letter)) {
49              throw new IllegalArgumentException();
50          }
51
52          if (!Character.isLowerCase(letter)) {
53              throw new IllegalArgumentException();
54          }
55
56          guesses.add(letter);
57
58          int occurence = 0;
59          for (int i = 0; i < secretWord.length(); i
    ++) {
60              if (secretWord.charAt(i) == letter) {
61                  occurence++;
62              }
63          }
64          maxGuesses--;
65          return occurence;
66      }
67
68      @Override
69    public boolean isGameOver() {
70          if (!getPattern().contains("-") || maxGuesses
     == 0) {
71              return true;
72          }
73          else {
74              return false;
75          }
76    }
```

```
 77
 78   @Override
 79   public String getPattern() {
 80       String printedWord = "";
 81       for (int i = 0; i < secretWord.length(); i
    ++) {
 82           if (guesses.contains(secretWord.charAt(i
    ))) {
 83               printedWord += secretWord.charAt(i);
 84           }
 85           else {
 86               printedWord += "-";
 87           }
 88       }
 89       return printedWord;
 90   }
 91
 92   @Override
 93   public SortedSet<Character> getGuesses() {
 94         return guesses;
 95   }
 96
 97   @Override
 98   public int getGuessesRemaining() {
 99         return maxGuesses;
100   }
101
102   @Override
103   public String getWord() {
104         maxGuesses = 0;
105         return secretWord;
106   }
107 }
```