```java
import java.util.Comparator;

import components.map.Map;
import components.map.Map1L;
import components.queue.Queue;
import components.queue.Queue1L;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;

/**
 * Simple Word Counting program (clear of Checkstyle and FindBugs warnings).
 *
 * @author Shafin Alam
 */
public final class WordCounter {

    /**
     * Default constructor--private to prevent instantiation.
     */
    private WordCounter() {
        // no code needed here
    }

    /**
     * This method should create and display the HTML page with the words and
     * their counts.
     *
     * @param file
     *            the HTML page being written to
     * @param outputFile
     *            the name of the output file
     */
    private static void outputHTMLPage(SimpleWriter file, String outputFile,
            Map<String, Integer> map, Queue<String> wordList) {
```

```
37          file.println("<html><head><title>Word
      Counter</title></head><body>");
38          file.println(
39                  "<style>table, th, td { border: 1px solid black;}
      </style><body>");
40          file.println("<h2>Words Counted in " + outputFile +
      "</h2>");
41          file.println("<table style= \" width: 25%\">");
42          file.println("<tr><th>Words</th><th>Counts</th></tr>");
43          wordList.dequeue();
44          wordList.dequeue();
45          for (int i = 0; i < map.size(); i++) {
46              String word = wordList.front();
47              int c = map.value(word);
48              file.println("<tr><th>" + word + "</th><th>" + c +
      "</th></tr>");
49              wordList.rotate(1);
50          }
51          file.println("</body></table></html>");
52      }
53
54      /**
55       * This is method is meant to get all of the words in a given
      file, and then
56       * count how many times they appear in that file.
57       *
58       * @param in
59       *             reads the file that the user input
60       * @return returns the map of words and their counts
61       */
62      public static Map<String, Integer> table(SimpleReader in) {
63          Map<String, Integer> wordTable = new Map1L<>();
64          String word = "";
65          while (!in.atEOS()) {
66              word = in.nextLine();
67              StringBuilder term = new StringBuilder("");
68              for (int i = 0; i < word.length(); i++) {
69                  char letter = word.charAt(i);
70                  term.append(word.charAt(i));
```

```
 71                 if (letter == ' ' || letter == ',' || i ==
    word.length() - 1) {
 72                     String termToWord = term.toString();
 73                     termToWord.replaceAll("[a-zA-Z0-9]",
    "").toLowerCase();
 74                     if (!wordTable.hasKey(termToWord)) {
 75                         wordTable.add(termToWord, 1);
 76 
 77                     } else {
 78 
 79                         int c = wordTable.value(termToWord);
 80                         wordTable.remove(termToWord);
 81                         wordTable.add(termToWord, c + 1);
 82                     }
 83                     term.delete(0, term.length() - 1);
 84                 }
 85             }
 86         }
 87         return wordTable;
 88     }
 89 
 90     /**
 91      * Comparator used to order the words in alphabetic order.
 92      *
 93      */
 94     private static class StringLT implements Comparator<String> {
 95         @Override
 96         public int compare(String s1, String s2) {
 97             return s1.compareTo(s2);
 98         }
 99     }
100 
101     /**
102      * Main method.
103      *
104      * @param args
105      *             the command line arguments; unused here
106      */
107     public static void main(String[] args) {
```

```
108         SimpleReader in = new SimpleReader1L();
109         SimpleWriter out = new SimpleWriter1L();
110         out.println("Enter an input file: ");
111         String inputFileName = in.nextLine();
112         out.println("Enter the name for an output file: ");
113         String outputFileName = in.nextLine();
114         SimpleReader file = new SimpleReader1L(inputFileName);
115         SimpleWriter output = new SimpleWriter1L(outputFileName +
    ".html");
116         Map<String, Integer> wordTable = table(file);
117
118         Comparator<String> sort = new StringLT();
119         Queue<String> wordList = new Queue1L<String>();
120         Map<String, Integer> map = new Map1L<String, Integer>();
121         map.transferFrom(wordTable);
122         while (map.size() > 0) {
123             Map.Pair<String, Integer> pair = map.removeAny();
124             wordList.enqueue(pair.key());
125             wordTable.add(pair.key(), pair.value());
126         }
127         wordList.sort(sort);
128         outputHTMLPage(output, inputFileName, wordTable, wordList);
129         file.close();
130         in.close();
131         out.close();
132     }
133
134 }
135
```