```java
import components.map.Map;
import components.map.Map1L;
import components.queue.Queue;
import components.queue.Queue1L;
import components.set.Set;
import components.set.Set1L;
import components.simplereader.SimpleReader;
import components.simplereader.SimpleReader1L;
import components.simplewriter.SimpleWriter;
import components.simplewriter.SimpleWriter1L;

/**
 * a Java program that counts word occurrences in a given input file and outputs
 * an HTML document with a table of the words and counts listed in alphabetical
 * order.
 *
 * @author Charan
 */
public final class WordCount {

    /**
     * Template Method.
     */
    private WordCount() {
    }

    /**
     * Returns a word from the input string, each separated by a separator.
     *
     * @param inputText
     *            the String that words will be extracted from
     * @param z
     *            the initial positon to iterate from
     * @param separator
     *            the separation characters
     * @param k
     *            final position.
     * @return word from string.
     */
    private static String separateWordfromString(String inputText, int z,
            Set<Character> separator, int[] k) {
```

```
42          int i = z; // starting position
43          int j = z; // ending position
44          String word = "";
45
46          while (i < inputText.length()
47                  && separator.contains(inputText.charAt(i))) {
48              i++;
49          }
50
51          j = i; //set start = end after incrementing start
52
53          // Find the end of the word
54          while (j < inputText.length()
55                  && !separator.contains(inputText.charAt(j))) {
56              j++;
57          }
58
59          // Update final position
60          k[0] = j;
61
62          // Check if a word was found
63          if (i < inputText.length()) {
64              // If a word was found, return the word
65              word = inputText.substring(i, j);
66          }
67          return word;
68      }
69
70      /**
71       * Creates a map to store words and their occurence count.
72       *
73       * @param in
74       *             SimpleReader input text file.
75       * @param separator
76       *             defined separators.
77       * @return returns the map.
78       */
79      private static Map<String, Integer>
    createMapofOccurences(SimpleReader in,
80              Set<Character> separator) {
81          Map<String, Integer> map = new Map1L<>();
82          String line = "", word = "";
83          int i = 0; // index
84          int[] arr = new int[1];
85          while (!in.atEOS()) {
```

```
 86              line = in.nextLine();
 87              while (i < line.length()) {
 88                  word = separateWordfromString(line, i, separator,
    arr);
 89                  if (map.hasKey(word)) {
 90                      map.replaceValue(word, map.value(word) + 1);
 91                  } else {
 92                      map.add(word, 1);//
 93                  }
 94                  i = arr[0] + 1; //move to the next word in the line
 95              }
 96              i = 0;
 97          }
 98          return map;
 99      }
100
101      /**
102       * puts each character of a given string into a set.
103       *
104       * @param s
105       *              the given {@code String}
106       * @param set
107       *              the {@code Set} to be replace
108       * @replaces set
109       */
110      static void makeSetfromString(String s, Set<Character> set) {
111
112          for (int i = 0; i < s.length(); i++) {
113              if (!set.contains(s.charAt(i))) {
114                  set.add(s.charAt(i));
115              }
116          }
117      }
118
119      //private static void
120
121      /**
122       * Method to generate the HTML page with table of word counts.
123       *
124       * @param name
125       *              the output file name
126       * @param in
127       *              the input file name
128       * @param map
129       *              stores words and the counts
```

```java
     */
    private static void makeHTMLPage(String name, String in,
            Map<String, Integer> map) {
        SimpleWriter out = new SimpleWriter1L(name); //output file
        Queue<String> list = new Queue1L<>(); // list with the
    words
        Map<String, Integer> temp = map.newInstance(); // map woth
    the words and the couts
        temp.transferFrom(map); // transfer from map to temp, to
    manipulate in method.

        while (temp.size() > 0) { //iterate while there are items
    left in the map.
            Map.Pair<String, Integer> pair = temp.removeAny();
            // removes a random pair from the map
            list.enqueue(pair.key());
            map.add(pair.key(), pair.value());
        }
        out.println("<html>" + "<head>" + "<title>");
        out.println("Words Counted: " + in);
        out.println("</title>" + "</head>" + "<body>");
        out.println("<h2>" + "Words Counted: " + in + "</h2>");
        out.println("<hr />" + "<table border =\"2\">");
        out.println("<tr>" + "<th>" + "Words" + "</th>" + "<th>" +
    "Count"
                + "</th>" + "</tr>");

        while (list.length() > 0) {
            out.println("<tr>" + "<td>");
            out.print(list.dequeue());
            out.println("</td>" + "<td>");
            out.print(map.value(list.dequeue()));
            out.println("</td>" + "</tr>");
        }
        out.println("</table> + </body> + </html>");
        out.close();
    }

    /**
     * Main.
     *
     * @param args
     */
    public static void main(String[] args) {

```

```
170         SimpleReader in = new SimpleReader1L();
171         SimpleWriter out = new SimpleWriter1L();
172
173         /*
174          * Ask for the name for input and output file name
175          */
176         out.println("input file name? include file type,  ex:
    file.txt");
177         String input = in.nextLine();
178         out.println("output file name? include file type,  ex:
    output.txt");
179         String output = in.nextLine();
180         SimpleReader readIn = new SimpleReader1L(input);
181         Set<Character> characters = new Set1L<>();
182         makeSetfromString("! ,.-\";_?<>", characters); //defined
    separators
183
184         Map<String, Integer> wordMap =
    createMapofOccurences(readIn,
185                 characters);
186
187         makeHTMLPage(output, input, wordMap);
188
189         /*
190          * Close streams
191          */
192         in.close();
193         out.close();
194     }
195 }
196
```