```
1   /*-------------------------------------------------------------------------------------------------------
2    * Author:      Dan Cassidy
3    * Date:        2015-07-20
4    * Assignment:  HW4-3
5    * Source File: BlankTrimmer.java
6    * Language:    Java
7    * Course:      CSCI-C 490, Android Programming, MoWe 08:00
8   -------------------------------------------------------------------------------------------------------*/
9   import static java.nio.file.StandardCopyOption.*;
10  import static java.nio.file.StandardOpenOption.*;
11
12  import java.io.*;
13  import java.nio.file.*;
14  import java.util.*;
15
16  /**
17   * Utility class to trim extra blanks from a file.
18   *
19   * @author Dan Cassidy
20   */
21  public class BlankTrimmer
22  {
23      /**
24       * Trims any extra blanks from the passed file.
25       *
26       * @param filePath The file to be trimmed.
27       * @throws InvalidPathException if the file path is not a file or the path cannot be parsed.
28       * @throws IOException if there is some other I/O exception.
29       * @throws NoSuchFileException
30       * @throws SecurityException
31       */
32      public static void trim(String filePath) throws IOException, NoSuchFileException
33      {
34          Path mainFile;
35          Path tempFile;
36
37          // Check main file.
38          mainFile = Paths.get(filePath);
39          if (Files.notExists(mainFile))
40              throw new NoSuchFileException(mainFile.toString());
41          if (Files.isDirectory(mainFile))
42              throw new InvalidPathException(mainFile.toString(), "Not a file.");
43          if (!Files.isReadable(mainFile))
44              throw new SecurityException("Main file could not be read.");
45          mainFile = mainFile.toRealPath();
46
47          // Generate and check temp file name.
48          do
49          {
50              tempFile = Paths.get(mainFile.getParent().toString(), createRandomFileName());
51          } while (Files.exists(tempFile));
52
53          // Open main and temp files.
54          try (   InputStream in = Files.newInputStream(mainFile);
55                  OutputStream out = Files.newOutputStream(tempFile, CREATE_NEW))
56          {
57              BufferedReader reader = new BufferedReader(new InputStreamReader(in));
58              BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(out));
59
60              // Copy text from main file to temp file, trimming blanks along the way.
```

```
61              String line = null;
62              while ((line = reader.readLine()) != null)
63              {
64                  StringTokenizer tokenizer = new StringTokenizer(line, " ");
65                  while (tokenizer.hasMoreTokens())
66                  {
67                      writer.append(tokenizer.nextToken());
68                      writer.append((tokenizer.hasMoreTokens() ? " " : "\n"));
69                  }
70              }
71
72              // Force the writer to write everything to the file.
73              writer.flush();
74
75              // Automatically close main and temp files.
76          }
77
78          // Move the temp file to the main file, overwriting in the process.
79          Files.move(tempFile, mainFile, REPLACE_EXISTING);
80      }
81
82      /**
83       * Generates a random file name with the prefix of "Temp" followed by 16 random characters from
84       * A-Z, with a file extension of ".tmp".
85       *
86       * @return String containing the generated file name.
87       */
88      private static String createRandomFileName()
89      {
90          // Define the prefix, suffix, and extension of the file name, as well as how many random
91          // characters are generated.
92          String fileNamePrefix = "Temp";
93          String fileNameSuffix = "";
94          String fileNameExtension = ".tmp";
95          int numRandomChars = 16;
96
97          Random generator = new Random();
98          String randomFileName = "";
99
100         // Generate the random characters.
101         for (int numChars = 1; numChars <= numRandomChars; numChars++)
102             randomFileName += (char)(generator.nextInt(26) + 'A');
103
104         // Return the amalgam.
105         return fileNamePrefix + randomFileName + fileNameSuffix + fileNameExtension;
106     }
107 }
108
```