WORKED EXAMPLE 19.1 Word Properties

It is fun to find words with interesting properties. To see how streams make this easy, download the code for Worked Example 19.1 from the companion code for this book.

Problem Statement The French word *oiseau* has five distinct vowels, which is pretty nifty. Are there English words like that? Just a few or a lot? Which words are the shortest and longest among them?

Step 1 Get the data.

In this case, we need a list of English words. The companion code for this book contains a copy of the `words.txt` file that is available on computers with a Unix-based operating system.

Step 2 Make a stream.

In this case, it is very easy to get a stream of words because the input file has one word per line. Simply call

```
try (Stream<String> lines = Files.lines(Paths.get("words.txt")))
{
   . . .
}
```

Step 3 Transform the stream.

First, the input contains many words ending with 's, such as Alice's. We don't want them:

```
try (Stream<String> lines = Files.lines(Paths.get("words.txt")))
{
   Stream<String> words = lines.filter(w -> !w.endsWith("'s"));
   . . .
}
```

And we only want words that have all five vowels. This is complex enough that we should write a separate method

```
public static boolean hasFiveVowels(String word)
```

This method needs to check whether the word contains all five vowels. As an aside, this too can be done with streams:

```
return word.toLowerCase().codePoints() // A stream of code points
   .filter(c -> c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u')
   .distinct() // The distinct vowels in word
   .count() == 5;
```

Now that we have this method implemented, we add a second filter to our stream of words:

```
Stream words = lines
   .filter(w -> !w.endsWith("'s"))
   .filter(w -> hasFiveVowels(w));
```

Step 4 Collect the results.

Once we have the words that we want, we can ask questions about them. How many are there?

```
long count = words.count();
```

It turns out that there are 469. Let's see a few.

```
List<String> examples = words
   .limit(20)
   .collect(Collectors.toList());
```

The result is a list of these words: Aurelio, Aureomycin, Australopithecus, Austronesian, Barquisimeto, Beaujolais, Beauvoir, Byelorussia, Carboniferous, Cointreau, Ecuadorian, Ecuadorians, Figueroa, Hermaphroditus, Milquetoast, Mozambique, Teotihuacan, abstemious, accentuation, adulteration.

What is the shortest one?

```
String shortest = words
   .min((s, t) -> s.length() < t.length())
   .orElse("");
```

That's the word Aurelio. Are there others of the same length?

```
List<String> allShortest = words
   .filter(w -> w.length() == 7)
   .collect(Collectors.toList());
```

This yields two words: Aurelio and sequoia. Not bad—just one letter longer than the French *oiseau*. I'll leave it to you to figure out how to get the longest word—it is counterrevolutionaries.