This question involves the process of taking a list of words, called wordList, and producing a formatted string of a specified length. The list wordList contains at least two words, consisting of letters only.

When the formatted string is constructed, spaces are placed in the gaps between words so that as many spaces as possible are evenly distributed to each gap. The equal number of spaces inserted into each gap is referred to as the *basic gap width*. Any *leftover spaces* are inserted one at a time into the gaps from left to right until there are no more leftover spaces.

The following three examples illustrate these concepts. In each example, the list of words is to be placed into a formatted string of length 20.

Example 1: wordList: ["AP", "COMP", "SCI", "ROCKS"]

Total number of letters in words: 14 Number of gaps between words: 3

Basic gap width: 2 Leftover spaces: 0

#### Formatted string:

Example 2: wordList: ["GREEN", "EGGS", "AND", "HAM"]

Total number of letters in words: 15 Number of gaps between words: 3

Basic gap width: 1 Leftover spaces: 2

The leftover spaces are inserted one at a time between the words from left to right until there are no more leftover spaces. In this example, the first two gaps get an extra space.

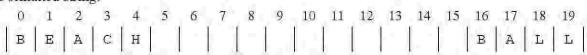
#### Formatted string:

Example 3: wordList: ["BEACH", "BALL"]

Total number of letters in words: 9 Number of gaps between words: 1

Basic gap width: 11 Leftover spaces: 0

#### Formatted string:



You will implement three static methods in a class named StringFormatter that is not shown.

(a) Write the StringFormatter method totalLetters, which returns the total number of letters in the words in its parameter wordList. For example, if the variable List<String> words is ["A", "frog", "is"], then the call StringFormatter.totalLetters(words) returns 7. You may assume that all words in wordList consist of one or more letters.

Complete method totalLetters below.

```
/** Returns the total number of letters in wordList.
```

\* Precondition: wordList contains at least two words, consisting of letters only.

\*/
public static int totalLetters(List<String> wordList)

(b) Write the StringFormatter method basicGapWidth, which returns the basic gap width as defined earlier.

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Assume that totalLetters works as specified regardless of what you wrote in part (a). You must use totalLetters appropriately to receive full credit.

Complete method basicGapWidth below.

- /\*\* Returns the basic gap width when wordList is used to produce
  - \* a formatted string of formattedLen characters.
  - \* Precondition: wordList contains at least two words, consisting of letters only.
  - \* formattedLen is large enough for all the words and gaps.
    \*/

(c) Write the StringFormatter method format, which returns the formatted string as defined earlier.

The StringFormatter class also contains a method called leftoverSpaces, which has already been implemented. This method returns the number of leftover spaces as defined earlier and is shown below.

WRITE YOUR SOLUTION ON THE NEXT PAGE.

Assume that basicGapWidth works as specified, regardless of what you wrote in part (b). You must use basicGapWidth and leftoverSpaces appropriately to receive full credit.

Complete method format below.

- /\*\* Returns a formatted string consisting of the words in wordList separated by spaces.
  - \* Precondition: The wordList contains at least two words, consisting of letters only.
- \* formattedLen is large enough for all the words and gaps.
- \* Postcondition: All words in wordList appear in the formatted string.
- The words appear in the same order as in wordList.
- \* The number of spaces between words is determined by basicGapWidth and the
- \* distribution of leftoverSpaces from left to right, as described in the question.

\*/
public static String format(List<String> wordList, int formattedLen)