PLTW 2.4.6 - Activity 2

Read through PLTW 2.4.6 steps 9-12 as you work through activity 2. Answer the following at the beginning of this Google Doc, then use the Google Doc format of activity 2 (below) to complete the first activity. *The PDF was converted to Google Doc format to make it easy to work with*. Copy this entire document into your notebook when done, and label as PLTW 2.4.6 Activity 2.

Reflection Question: You were asked to describe ways you could improve your `totalSentiment` method. Why is it important to reflect on programs you have already written?

11 Copy the list of positive and negative adjectives you generated at the beginning of this project.

Positive Adjectives:

Negative Adjectives:

PLTW 2.4.6 - Activity 2

12 Use your browser to search for a new review on a topic of your choice. Paste the review in the space provided below. You may choose to use this new review in your `SimpleReview.txt` file.

ACTIVITY 2

Sentiment Value and Star Ratings

Now that you have read several reviews and started exploring the `sentimentVal` method, you will use this method to determine the overall sentiment of an entire review.

- Pick a review of your choice. Copy and paste the content of the review into a new text file that you create using a text editor, making sure to save the file with a `.txt` extension.
- If time permits, do this with multiple reviews. For testing purposes, you may also want to create a `SimpleReview.txt` file like the following:

```
This was a terrible restaurant! The pizza crust was too chewy,
and I disliked the pasta. I would definitely not come back.
```

Tip

Strings are objects and have methods that can access information about them and create new strings. The course framework describes those methods which are part of the course, such as `length`, `indexOf`, and `substring`.

In the `Review.java` file:

1. Write the following method that determines the sentiment value of a review. Utilize the existing `textToString` method to complete the implementation of this method.

```
public static double totalSentiment(String fileName)
```

Tip

When calling the `String` methods within the framework, such as `indexOf` and `substring`, adjustments must be made to account for strings being indexed starting at zero.

2. Test the method by calling `totalSentiment("SimpleReview.txt")` and printing the returned value.

3. Write the following method that determines the *star rating* of a review.

```
public static int starRating(String fileName)
```

Tip

To provide instructions for the computer to process many different input values, selection statements may need to be nested together to form more than two branches and options. Pathways can be broken down into a series of individual selection statements based on the conditions that need to be checked and nesting together the conditions that should only be checked when other conditions fail or succeed.

Sample code:

Method Call	*Return*
`totalSentimentVal("26WestReview.txt")`	`29.05`
`starRating("26WestReview.txt")`	`4`
`totalSentimentVal("SimpleReview.txt")`	`-2.92`
`starRating("SimpleReview.txt")`	`1`

Check Your Understanding

4. Explain how the `totalSentiment` method works, including how you're using **strings** and **iteration** in your solution.

5. Test your `starRating` method for multiple reviews, including your simple sample review.

   a. Do the ratings make sense? Explain why or why not.

b. Describe at least two ways in which you could adjust your `totalSentiment` method so that your program returns even more reasonable ratings.

A student is writing a `starRating` method based on his `totalSentiment` method, which calculates the sum of all of the sentiment values in a review. He comes up with the following:

```
public static int starRating(String fileName)
{
   double totalSentiment = totalSentiment(fileName);
   if(totalSentiment < 15)
   {
      return 4;
   }
   else if(totalSentiment < 10)
   {
      return 3;
   }
   else if(totalSentiment < 5)
   {
      return 2;
   }
   else if(totalSentiment < 0)
   {
      return 1;
   }
   else
   {
      return 0;
   }
}
```

6. Explain to the student, using **specific examples** of (a) what **logical error** he made in writing his code, and (b) how to **fix** it.

PLTW 2.4.6 - Activity 2