Name: _____

Date: _____

ACTIVITY 1

Introduction (Reviews and Sentiment Value)

The persuasive power of words can be seen in a variety of places such as entertainment, news, social media, and even reviews and comments. This activity focuses on the sentiment value of individual words, and you will start by reading a few online reviews of your choice.

1. In this activity you'll be calling the `sentimentVal` method. Find this method in the `Review` class and answer the following questions:

a. Record the method signature in the space below.

b. Does this method require a parameter? If so, what type of parameter is required? What happens if you pass a different type of parameter instead? (Try it!)

c. Does the method return a value? If so, what is the return type?

2. Write code that tests the `sentimentVal` method by calling it with several different `String` parameters and printing out the return value. Any words not in the list will have a return value of zero. Make sure to keep calling the method until you have at least three strings that have a return value other than zero. Record the method calls, including provided strings and return values in the chart below:

Examples:

Method Call	*Return Value*
`sentimentVal("happily");`	`2.32`
`sentimentVal("terrible");`	`-3.38`
`sentimentVal("cold");`	`-0.04`

Method Call	*Return Value*

Open up the `cleanSentiment.csv` file and look up a few of the words that were used in the method call to verify the results.

Check Your Understanding

3. Determine whether or not each statement would compile. Briefly justify your answers.

a. `double num = sentimentVal("warm");`

b. `String word = sentimentVal(0.5);`

c. `double x = sentimentVal("good", "bad");`

Name: _____

Date: _____

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.

Name: _____

Date: _____

ACTIVITY 3

Autogenerated Review

In this activity you will write and test a method that will autogenerate a review based on the review used in Activity 2. The method will autogenerate a review by replacing adjectives in the given review with randomly selected adjectives from a list.

1. Using Notepad or another simple text editor, compile the adjective lists into two separate text files:

- **positiveAdjectives.txt** should contain all the identified positive adjectives.
- **negativeAdjectives.txt** should contain all the identified negative adjectives.

2. Open your existing `SimpleReview.txt` file and annotate each adjective by adding a "`*`" at the beginning of the word. For example, the `SimpleReview.txt` from Activity 2 would be:

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

3. Describe the action of the `indexOf` and `substring` methods. Explain how these methods can be used to divide an existing string into two parts. Why is it necessary to add 1 to the position of the space when copying the remainder of the string?

4. Write the method `fakeReview`. The method signature is given below.

```
public static String fakeReview(String fileName)
```

The parameter `fileName` should be the name of the annotated review created at the start of class (with "`*`" at the beginning of the adjectives). The `fakeReview` method should loop over the string containing the contents of the marked-up review to build and return a fake review.

Example:

A test file containing:

```
The *quick *brown fox jumps over the *lazy dog.
```

may produce as its output:

```
The speedy smoky fox jumps over the fast dog.
```

Tip

Multiple method calls can be joined into a single expression, and the return values from those method calls will be a combination of all the methods executed in order to create a single value.

The values returned from method calls can be stored in local variables and used as actual parameters in other method calls or combined with other local variables to form new expressions.

Check Your Understanding

5. In the `Review` class all methods are static. Explain the difference between calling static methods as opposed to calling non-static methods.

6. Autogenerated reviews may be created that are intentionally positive or negative. Explain how your method `fakeReview` could be modified to create a fake positive or fake negative review.

Name: _____

Date: _____

ACTIVITY 4

Create a Positive or Negative Review

Fake reviews can be slanted positive or negative. To create a positive review the program only need substitute positive adjectives for neutral or negative adjectives. To create a negative review the program would substitute only negative adjectives for neutral or positive adjectives. In this activity you'll modify the `fakeReview` method written in Activity 3. The method will be redesigned so that it will make reviews stronger; that is, either more negative or more positive.

Answer the following:

1. Determine one possible reason to autogenerate a negative review.

2. Determine one possible reason to autogenerate a positive review.

3. Discuss with a partner what the method needs to do to create a stronger review and write down the algorithm using pseudocode.

4. Implement the algorithm by updating the `fakeReview` method.

Check Your Understanding

5. What is the result of the following statement?

```
oneWord = oneWord.substring(1);
```

6. Why was it necessary to remove the * character (or other symbol) used to annotate the adjective before getting its sentiment value?

7. In this activity you wrote a method that created a more positive or more negative review. How could this method be used or modified to create a super positive review (one that is extremely favorable)?

Name: _____

Date: _____

ACTIVITY 5

Open-Ended Activity

This open-ended activity requires you to develop a program on a topic that interests you. As a class, spend a few minutes reviewing the requirements of the open-ended activity.

Requirements:

- Write a program with a `main` method
- Create at least one new method which is called from `main` that takes at least one parameter
- Call at least two distinct methods in the `String` class
- Utilize conditional statements or compound Boolean expressions
- Utilize iteration

In addition, review the provided scoring guidelines so that you understand what you'll be expected to explain once you're done completing your program.

It's strongly recommended that the implementation of the program involve collaboration with another student. Your selected program can be anything that you choose that meets the requirements and allows you to demonstrate your understanding.

Before beginning, make sure that you understand the expectations for the activity.

- Who will you be working with? Are you allowed to work with a partner? In a group of three or four?
- Among the members of your group (or with your partner), how will the implementation be completed?
- If you'll be using pair programming, will your teacher be instructing you when to switch driver and navigator, or is this something that you need to keep track of?
- What should you do if your group/pair is stuck? Does your teacher want you to come straight to them? Are you allowed to ask another group?

Check Your Understanding

Once your program has been implemented and tested, answer the following questions on your own:

1. Why did you choose to implement this program?

2. Describe the development process used in the completion of the project.

3. Provide the method header for a method that you implemented that takes at least one parameter. Explain why you chose the given parameter(s), including type, and why you made the method static or non-static.

4. Provide the code segment(s) where two distinct methods in the `String` class are called. Describe what each method call is doing, and what is being returned (if anything) by the method calls.

5. Copy and paste one code segment that uses nested conditional statements or compound Boolean expressions. What is one other way that this code could be written to achieve the same result? Provide an equivalent code segment to the one included above.

6. Copy and paste one code segment that uses iteration. Describe how the loop you used works and provide an equivalent code segment to the one included above that uses a different type of loop.