CS3354 – Spring 2019 – Assignment 1

Due date: Friday, Feb. 8 at 11:55 p.m.

Project Title: Sentiment Analysis (stage 1)

Goal: The goal of this assignment is to help students familiarize themselves with the following Java programming concepts:
1. Input/Output to and from the terminal.
2. Storing data in a file and reading data from a file.
3. Creating object-oriented classes and methods to handle data.
4. Using data structures to store data in main memory (e.g. HashSet).
5. Working with character strings.
6. Using Javadoc comments and generating and html documentation of the program.

Description:
For this assignment you will create a program to classify a set of movie reviews as positive or negative based on their sentiment. This process is known as Sentiment Analysis, and there are multiple approaches to analyze the data and estimate the sentiment. More information about sentiment analysis can be found on Wikipedia and other sources.
https://en.wikipedia.org/wiki/Sentiment_analysis

In this assignment, you are to write a Java program that will classify a review as positive or negative by counting the number of positive and negative words that appear in that review. Your program will have multiple inputs as command line arguments:

- Paths to two text files: list of positive words (positive-words.txt) and list of negative words (negative-words.txt).
- Paths to two folders: the folder 'pos' contains the positive reviews and the folder 'neg' contains the negative reviews, both manually assigned by humans. The reviews within each folder are given in separate .txt files, one review per file.

All .txt files are provided in Data.zip file

Your Java program will automatically classify the .txt files in the folders and output the total count of how many were correctly classified, compared to the human ground truth. The execution should flow as:

1. The program starts as >>`java SentimentAnalysisApp <path_to_pos_words> <path_to_neg_words> <path_to pos_reviews_folder> <path_to_neg_reviews_folder>`
2. The program loads the positive words and negative words and stores them in two separate lookup tables. The HashSet data structure can be used as a lookup table in Java as it provides a fast way to look if a word exists in it or not.
3. The program iterates over the .txt files and, in each file, it counts the number of positive and negative words that the review contains. If the review contains more positive than negative words it is classified as positive, and vice versa. If the same number of positive and negative words were found on the review, it counts as negative.
4. After each review has been classified, the program prints out in the command line the file name of the review, its real class and its predicted class.
5. At the end the program should also print how many reviews were correctly classified and how many were misclassified – both total number or broken down per positive or negative review are accepted.

Java class file containing the `main()` method needs to be named **`SentimentAnalysisApp,`** and it should accept the following command line arguments:

```
java SentimentAnalysisApp <path_to_pos_words> <path_to_neg_words> <path_to
pos_reviews_folder> <path_to_neg_reviews_folder>
```

Run example (assumes Data directory is under the current directory):

```
>>java SentimentAnalysisApp ./Data/positive-words.txt ./Data/negative-words.txt
./Data/Movie-reviews/pos ./Data/ Movie-reviews/neg
```

Hint: Java provides the method split() which allows us to split a String into multiple tokens by specifying a separator character.
E.g.

```
String animals = "dog cat bear elephant giraffe";

String[] animalsArray = animals.split("\\s+");
```

For each review:
Step 0: Open .txt file.
Step 1: Remove punctuation marks using regex \p{Punct}
https://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html
e.g. `review = review.replaceAll("\\p{Punct}", "");`
Step 2: Convert everything to lower case.
Step 3: Tokenize (split) review using white space as separating character.
Step 4: Count the number of positive and negative words.
Step 5: Close .txt file.

Assignment Submission:

- **This assignment can be done individually or with a partner**
- You may use an IDE (e.g. BlueJ, Netbeans, Eclipse) or just an editor (e.g. Notepad+, Atom) and command line operations (javac, java) in Unix or Windows/DOS to develop your program.
- You don't need to create any GUI for this assignment. Command line operations are enough.
- Use a standard Java coding style to improve your program's visual appearance and make it more readable, e.g. BlueJ coding style: http://www.bluej.org/objects-first/styleguide.html or Google Java style guide: https://google.github.io/styleguide/javaguide.html
- A Git repository will be used to submit this assignment at https://git.txstate.edu/
- Junye Wen (Junye.wen@txstate.edu) will give you an access to the repository https://git.txstate.edu/CS3354/<NetID_>
- If you work with the partner, you need to add the partner to your git repository as a collaborator.
- *Do NOT include executable .class or .jar files in your submission. Only .java files*
    - More information about how to use Git will be provided soon