Object-Oriented Programming: Homework #6

Time

- Assigned on 5/22/2023 (Monday); Due at 09:00 on 6/5/2023 (Monday)

Submission

- Source Code Submit on iLearning.
- Your program code for this homework should be submitted as a single .java file named HW6_StudentID.java (**DO NOT** submit the entire Eclipse project, only the source files).
- In addition, please delete the package before uploading the file.

Objectives

- Work with Java strings. In the supporting information, we list a few string methods that you may find useful.
- Work with reading strings from files.
- Further practice with dynamic-allocated arrays.
- Further practice with accumulating data statistics.
- Further practice with the process of fixing code errors.
- Familiarity with implementation of class and object.

Program Descriptions

We all know companies that dedicate entire data centers with hundreds of thousands of machines to the analysis of social media data. In many cases, these machines are high-cost ones due to their large-memory configuration.

You were hired as an intern by a startup that is working hard to raise money from venture capitalists, but until they have enough cash to buy equipment, they need you to write data analytics code that can run on the computers that their interns bring with them for the internship. One of the company founders was your TA in NCHU OOP and is aware that you have experience in reading and processing strings from files.

The company wants to offer a new service to their customers: low-cost statistics on the usage of hashtags on Twitter data streams. They want you to create a program that customers can use to upload twitter data and get statistics about the most popular hashtags or statistics about a hashtag of particular interest to the customer. Tweet data files can be quite large: users produce 500 million tweets per day and around 200 billion tweets per year. Due to the limited availability of memory in the company machines, your program must analyze tweets, update statistics about hashtag popularity accordingly, and then discard the tweet.

The screenshots below show how users will interact with your program (user input appears in **red**.)

First, the user runs the program and sees the menu of options, selecting option 1 and providing the name of the file containing tweet data:

```
Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option: 1
Enter filename: starwars-small.csv

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option:
```

Now the user asks for option 2 (show stats) and option 3 (show at most 10 most popular tweets so far):

```
--------> Enter your option: 2
Tweets: 3, Retweets: 2, Hashtags: 3

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags
3. show most popular hashtags
9. quit
--------> Enter your option: 3
Tag #starwars - 3 occurrence(s)
Tag #theriseofskywalker - 1 occurrence(s)
Tag #lost - 1 occurrence(s)
```

As you can see, the starwars-small.csv file is a very small one, useful for debugging code.

The user can **KEEP UPLOADING** new files. Now another one.

The user asks to load another file and look at the stats:

```
--------> Enter your option: 1
Enter filename: frozen2-small.csv

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option: 3
Tag #starwars - 3 occurrence(s)
Tag #frozen2 - 2 occurrence(s)
Tag #theriseofskywalker - 1 occurrence(s)
Tag #lost - 1 occurrence(s)
Tag #disney - 1 occurrence(s)
Tag #cupcakes - 1 occurrence(s)
Tag #frozen - 1 occurrence(s)
Tag #family - 1 occurrence(s)

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option: 2
Tweets: 6, Retweets: 3, Hashtags: 8
```

The input files have real data. They were obtained by connecting to a Twitter API (application programming interface) and invoking services that retrieve small collections of tweets. These input files (and many others) are available in the iLearning (folder datasets).

The input files contain one tweet per line. The tweets may contain characters such as emoticons that **do not display well as a text file**. As an example, the file starwars-small.csv contains only three tweets:

```
2019/10/23 01:47,b'RT @v_jvee: #StarWars #TheRiseOfSkywalker \nWhen you see the Haters getting out numbered Seven Times Over https://t.co/NffikRPhAR'

2019/10/23 01:46,b'@StormPodcast Can\xe2\x80\x99t wait for all your coverage. Thanks to all three of you for everything you do! #Lost #StarWars\xe2\x80\xa6 https://t.co/hTUfXm09P9'

2019/10/23 01:46,b'RT @JoanieBrosas: Retweet if you got your #StarWars tickets!\n\xf0\x9f\x93\xb7by @ssnwwc costume by @cleighcreations https://t.co/3gTdl8e0qe'
```

You will need to process each tweet so that you extract the information you need: is it a retweet? What are its hashtags?

Program Specifications

To facilitate the integration of your code with other components of the system, the team leads at the company stated that your solution must use the declarations provided in the picture below. The picture contains the declaration of the three functions that you must implement and use in your solution. These functions manipulate the Hashtag class.

Note that the size of the list storing Hashtag is dynamically allocated, growing as your system processes tweets and finds hashtags. As you add hashtags to the hashtag list, you may need to allocate a larger area for the list by doubling the capacity list (or using the **ArrayList** library). You are asked to keep the list in decreasing order of hashtag popularity.

```
/* class that keeps track of the statistics for a single
 * hashtag.
 */
class Hashtag {
  String name;
  long counter;    // total number of occurrences

    // Students need to implement constructor
};
```

The functions that you are required to implement are:

```
void readTweet(string line);
```

Parameters:

- line: string containing the tweet information received by the function

Return value: none

Functionality: it processes the string in order to identify its hashtags and if it is a retweet. Real-life twitter data feeds present retweets in different ways. For the purpose of this homework, you only need to handle the following formatting for retweets:

- Appear preceded by “RT @”
  `2019/10/23 01:47,b'RT @v_jvee: #StarWars #TheRiseOfSkywalker`
- For hashtags, you can assume that they only contain letters and digits.

```
void insertHashtag(string ht);
```

Parameters:

- ht: string

Return value: none

Functionality: the function searches for the string `ht` in the hashlist's array. If the hashtag is already in the list, its counter is updated and the order of the elements in the array may need to be rearranged if the increment of the popularity counter changes the ordering of the hashtags.

If the hashtag is not in the list, it needs to be inserted. If there is no capacity in the array, a resize to double its capacity must be carried out.

In order to account for #STARWARS and #starwars as the same hashtag, hashtags in the hashlist must be all in lowercase. The **Useful functions** section discusses the library support `toLowerCase()` and other methods that you may find useful when manipulating the string ht.

```
void showMostPopularHashtags(int k);
```

This function simply prints to the console the `k` most popular hashtags, i.e., the first `k` elements of the array in `hashlist`.

If the hashtag array size is less than 10, then `k` should be set to the array size.

Example Program Run

Your program needs to be able to open input files and read tweets lines from it for processing them. We provide starter code for tweets.java that prints the menu of options (function `printMenu`) and reads a user option from the console (function `getOption`)

The following screenshots show the format for a few error scenarios:

```
Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option: 5
Invalid option

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags)
3. show most popular hashtags
9. quit
--------> Enter your option: d
Invalid option
```

```
--------> Enter your option: 1
Enter filename: starwarsssssss.csv
File can't be opened.

Welcome to Twitter Feeds Stats
The options are:
1. load tweet data file and update stats
2. show overall stats (number of tweets, retweets, and hashtags
3. show most popular hashtags
9. quit
--------> Enter your option:
```

Useful functions

- `String.split(String regex)`: The array of strings computed by **splitting** this string around matches of the given regular expression.

  The string `"boo:and:foo"`, for example, yields the following results with these expressions:

Regex	Result
`":"`	`{"boo", "and", "foo"}`
`"o"`	`{"b", "", ":and:f"}`

- `String.replaceAll(String regex, String replacement)`: Replaces each substring of this string that matches the given regular expression with the given replacement. Note that **backslashes (\)** and **dollar signs ($)** in the replacement string may cause the results to be different than if it were being treated as a literal replacement string.

  The string "`boo:and:foo`", for example, yields the following results with these expressions:

Regex	Replacement	Result
`":"`	`""`	`"booandfoo"`
`"o"`	`"a"`	`"baa:and:faa"`

- `String.contains(CharSequence s)`: returns true if this string contains `s`, false otherwise.
- `String.indexOf(String str)`: returns the index of the first occurrence of the specified substring, or -1 if there is no such occurrence.
- `String.toLowerCase()`: returns the String, converted to lowercase.