Lab 03 – Parsing Tweets with the String class

Introduction

Twitter and similar micro-blogging platforms allow users to broadcast short messages to a potentially large audience. With Twitter, the messages are called "tweets" and consist of 140 characters or less. Tweets are often sent from mobile phones, and tweeters can choose whether their tweets are public or not.

The platforms allow people to communicate in ways that were not possible even a few years ago, and they have already had a profound effect on society. They even affect how society reacts to natural and manmade disasters—ordinary citizens can now broadcast timely and location specific information that is of vital importance to both emergency management agencies and members of the public.

Sifting through the tweets for useful information is difficult, however. As a result, there have been proposals to manually add structure to tweets sent during disasters. One proposal is called Tweak the Tweet (TtT).[1] In TtT, information is marked using a "hashtag" such as **`#loc`** (for location):

```
MT @carlseelye: #lovelandfire #Wind just switched and now the smoke is thick
around our house #loc 40.352,-105.2045
```

Adding the structure makes it far easier for a computer to classify the tweets.

In this lab, you will use methods of the **`String`** class to process messages similar to TtT tweets (the data is made from real TtT data but it has been altered to suit the lab). You will use the **`substring`** and other methods to pull out information from the text, manipulate it, and print it out to the screen.

Lab Objectives

By the end of the lab, you should:

- understand how **`String`** constants and String objects are represented in Java;
- be able to declare, initialize, and assign variables of type **`String`**;
- perform string processing using the methods of the **`String`** class;
- concatenate strings (and other values) together using the '**+**' operator.

Prerequisites

The lab deals with material from Chapter 4 (specifically, the discussions of the **`String`** class). It assumes you know how define simple classes in Java, and that you can declare and assign values to variables.

What to Submit

You will create a single source file (**`ParseTheTweet.java`**), and it should be submitted to our course website for grading.

[1] http://epic.cs.colorado.edu/?page_id=11

Exercise – Parsing Tweets

The class **`ParseTheTweet`** will read in a single tweet from the keyboard, and so when writing your program, you will need to use the **`Scanner`** class in addition to the **`String`** class. As usual, almost everything you write will go in the **`main`** method of the **`ParseTheTweet`** class.

The tweets processed by the call all encode the following information using so-called "hashtags": The report *type* (**`#typ`**); some further *detail* (**`#det`**); a location (**`#loc`**) such as a street address; and latitude (**`#lat`**) and longitude (**`#lng`**). The type indicates the meaning of the tweet (i.e., whether it is a request for help or reports factual information), and the report detail provides additional information. Each of the hashtags is followed by some value (such as an actual latitude or longitude value).

When writing your code, you can assume that all of the tweets to process have the following format:

`#typ` *value*`;` **`#det`** *value*`;` **`#loc`** *value*`;` **`#lat`** *value*`;` **`#lng`** *value*`;`

That is, they consist of a series of hashtags, each followed by a value, and each value followed by a semicolon (**`;`**). 9 sample tweets are provided at the end of this document. You should test your code on them.

Instructions

1. At the top of your source file, include a comment stating your Java class name, your name, the date, the program purpose, and containing the statement of academic honesty as you did in previous labs.
2. Add an import statement for the **`Scanner`** class: **`import java.util.Scanner;`**
3. Create an instance of the **`Scanner`** class called **`theScanner`**. It should read from **`System.in.`**
4. Declare a String variable called **`tweet`** and initialize it to the empty string **`""`**. This will hold the tweet entered by the user.
5. Declare variables to store the information from the tweet. The variables should have the names and data types shown.

o **`type`**:	**`String`**
o **`detail`**:	**`String`**
o **`location`**:	**`String`**
o **`latitudeString:`**	**`String`**
o **`longitudeString:`**	**`String`**
o **`latitude`**:	**`double`**
o **`longitude:`**	**`double`**

6. Use the **`theScanner`** object to read in an entire line of text (the whole tweet) and assign the value to **`tweet`**. Since you want the whole line, you should use the **`nextLine()`**method.

   HINT: For testing purposes, it might be better to use a single (fixed) **`String`** constant assigned to **`temp`**, rather than use **`theScanner`** to read in a new value every time. After you've been able to successfully parse this hard coded value, go back and insert the code to read values from the **`theScanner`**.

7. Declare **`int`** values **`start`** and **`finish`** (these will store start and end indexes for substrings).
8. Use the **`indexOf()`**method of **`tweet`** to find the starting index of the substring `"#typ"`. Add 5 to the index and assign the result to **`start`**. This will record the start of the tweet's type value.
9. Again use the **`indexOf()`**method of the **`tweet`** object to find the index of the first semicolon `";"`. Assign the value to **`finish`**. This will give you the ending index of the tweet's type value.
10. Use the **`substring`** method of the **`tweet`** object to find the substring demarcated by indexes **`start`** and **`finish`**. Assign this string to **`type`**.
11. Use the **`trim()`** method to remove leading and trailing white spaces (if any) from the value of **`type`**, and assign the result to **`type`**.
12. Now use the **`substring`** method to extract the substring of **`tweet`** beginning at index **`finish+1`**. Assign the result to **`tweet`**. This is tantamount to deleting the initial part of the tweet (the part encoding the type). The **`+1`** eliminates a leading semicolon that would otherwise be included.
13. Repeat steps 8-12 to obtain values for **`detail`**, **`location`**, **`latitudeString`**, and **`longitudeString`**.

    HINT: It might be sufficient to copy and paste your code, replacing the variable names.
14. To obtain numerical values for the latitude and longitude, we will use **`theScanner`** again. Here, however, we don't want to read from the keyboard. Instead, we want to read directly from the strings **`latitudeString`** and **`longitudeString`**. To do this, include the statement

    ```
    theScanner = new Scanner(latitudeString);
    ```

    in your program (after all the other code you have written).
15. Afterwards, use the **`theScanner`** object's **`nextDouble()`** method to read in a floating-point value from **`latitudeString`**. Assign this value to **`latitude`**.
16. Repeat steps 14-15 to obtain the floating-point value for **`longitude`**.
17. The **`type`** values come from a very small set of values, and we want to make them all caps. To do this, use the **`toUpperCase`** method of the **`String`** class (and assign the result to **`type`**).
18. We also want to ensure that the **`detail`** and **`location`** values are free of commas (which might pose an obstacle to further processing). Use the **`replace`** method of the **`String`** class to do this. Replace each comma with a single hyphen (-).
19. Now use **`System.out`** and **`print`** or **`println`** statements to produce formatted output, as shown in the included examples. **HINT:** Use the escape sequence **`\t`** to include tabs as needed.

Submission and Grading

After you have completed and thoroughly tested **`ParseTheTweet.java`**, submit it to our course website in order to receive credit for the lab (if you have any questions about how to submit this assignment, then ask your lab instructor for help at least 24 hours before this assignment is due). Always double check that your submission was successful on our course website.

The lab will be graded according to the following guidelines.

- A score between 0 and 100 will be assigned.
- If the source file(s) are not submitted before the specified deadline, if they do not compile, or if the submitting student has an unexcused absence for a single lab period devoted to the assignment, then a grade of 0 will be assigned. (Note: students who show up to their first lab period and ***personally*** show their lab TA that they finished and submitted the week's lab assignment.)
- If the required comment for all labs describing the program and the academic honesty statement is not included at the top of the file, then 10 points will be deducted. Note: this required comment can be found in Lab 02.
- If a (single) source file name is incorrect, then 10 points will be deducted.
- The program will be evaluated using the sample tweets below and additional tweets. For each test case, the output must be correct to receive credit for that test case.

Example Tweets

Test your code with these.

1. `"#typ offer; #det free essential supplies 4 evacs pets.; #loc 2323 55th st, boulder; #lat 40.022; #lng -105.226;"`
2. `"#typ structure; #det damaged; #loc 224 left fork road (shed) (house okay); #lat 40.029854; #lng -105.391055;"`
3. `"#typ wind; #det just switched, and now the smoke is thick around our house; #loc 40.352,-105.2045; #lat 40.3523; #lng -105.2045;"`
4. `"#typ fire; #det firefighter sees a glow; #loc sw from west coach rd toward sunshine canyon; #lat 40.0515; #lng -105.332;"`
5. `"#typ structure; #det damaged; #loc unknown number, by 204 gold run road; #lat 40.050904; #lng -105.373941;"`
6. `"#typ evac; #det overflow shltr 4 evacuees at; #loc walt clark middle school, loveland; #lat 40.383; #lng -105.113;"`
7. `"#typ no fire; #det activity; #loc west of the pinewood res dam; #lat 40.367; #lng -105.292;"`
8. `"#typ photo; #det local wild horse; #loc wild horse; #lat 40.052304; #lng -105.319374;"`
9. `"#typ smoke; #det its raining ash; #loc windsor, co; #lat 40.499812; #lng -105.012075;"`

Sample Output

```
Type:           OFFER
Detail:         free essential supplies 4 evacs pets.
Location:       2323 55th st- boulder
Latitude:       40.022
Longitude:      -105.226

Type:           STRUCTURE
Detail:         damaged
Location:       224 left fork road (shed) (house okay)
Latitude:       40.029854
Longitude:      -105.391055

Type:           WIND
Detail:         just switched- and now the smoke is thick around our house
Location:       40.352--105.2045
Latitude:       40.3523
Longitude:      -105.2045

Type:           FIRE
Detail:         firefighter sees a glow
Location:       sw from west coach rd toward sunshine canyon
Latitude:       40.0515
Longitude:      -105.332

Type:           STRUCTURE
Detail:         damaged
Location:       unknown number- by 204 gold run road
Latitude:       40.050904
Longitude:      -105.373941

Type:           EVAC
Detail:         overflow shltr 4 evacuees at
Location:       walt clark middle school- loveland
Latitude:       40.383
Longitude:      -105.113

Type:           NO FIRE
Detail:         activity
Location:       west of the pinewood res dam
Latitude:       40.367
Longitude:      -105.292

Type:           PHOTO
Detail:         local wild horse
Location:       wild horse
Latitude:       40.052304
Longitude:      -105.319374

Type:           SMOKE
Detail:         its raining ash
Location:       windsor- co
Latitude:       40.499812
Longitude:      -105.012075
```